EPICS QT Framework

3.6.2

Generated by Doxygen 1.6.1

Wed Dec 12 11:10:24 2018

Contents

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 Lesser General Public License.

1.3 Platforms

epicsqt might be usable in all environments where you find Qt. It is compatible with Qt >= 4.4.

1.4 Screenshots

- ASgui screen shots
- other applications using epicsqt widgets
- Qt Designer
- Qt Creator

Screenshots are only available in the HTML docs.

1.5 Downloads

Stable releases and development snapshots are available at the epicsqt project page.

For getting a development snapshot from the SVN repository:

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

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

1.6 Installation

Read QE_GettingStarted.pdf in the documentation for setting up an enviroment for building or using the epicsqt framework.

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

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

Any Qt specific queries? start at the Qt Project

1.7 Support

Visit the sourceforge epicsqt support page for assistance.

1.8 Related Projects

Qwt, The core of a Channel Access aware plotting widget.

1.9 Credits:

Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

Project admin:

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

Chapter 2

GNU Lesser General Public License

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

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

See the GNU General Public License for more details.

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

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

Chapter 3

ASgui screen shots

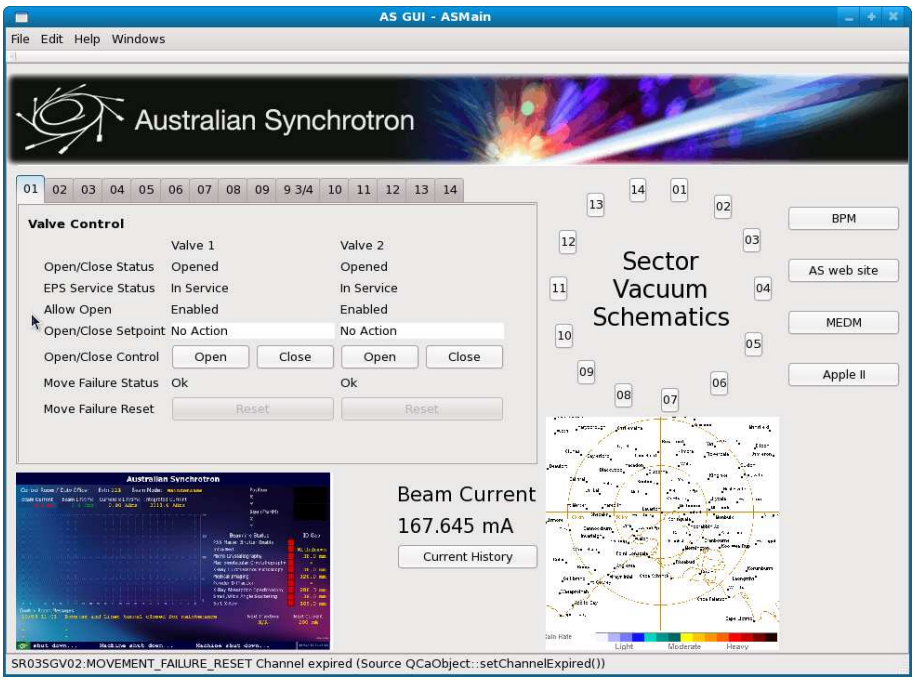


Figure 3.1: Australian Synchrotron mock up

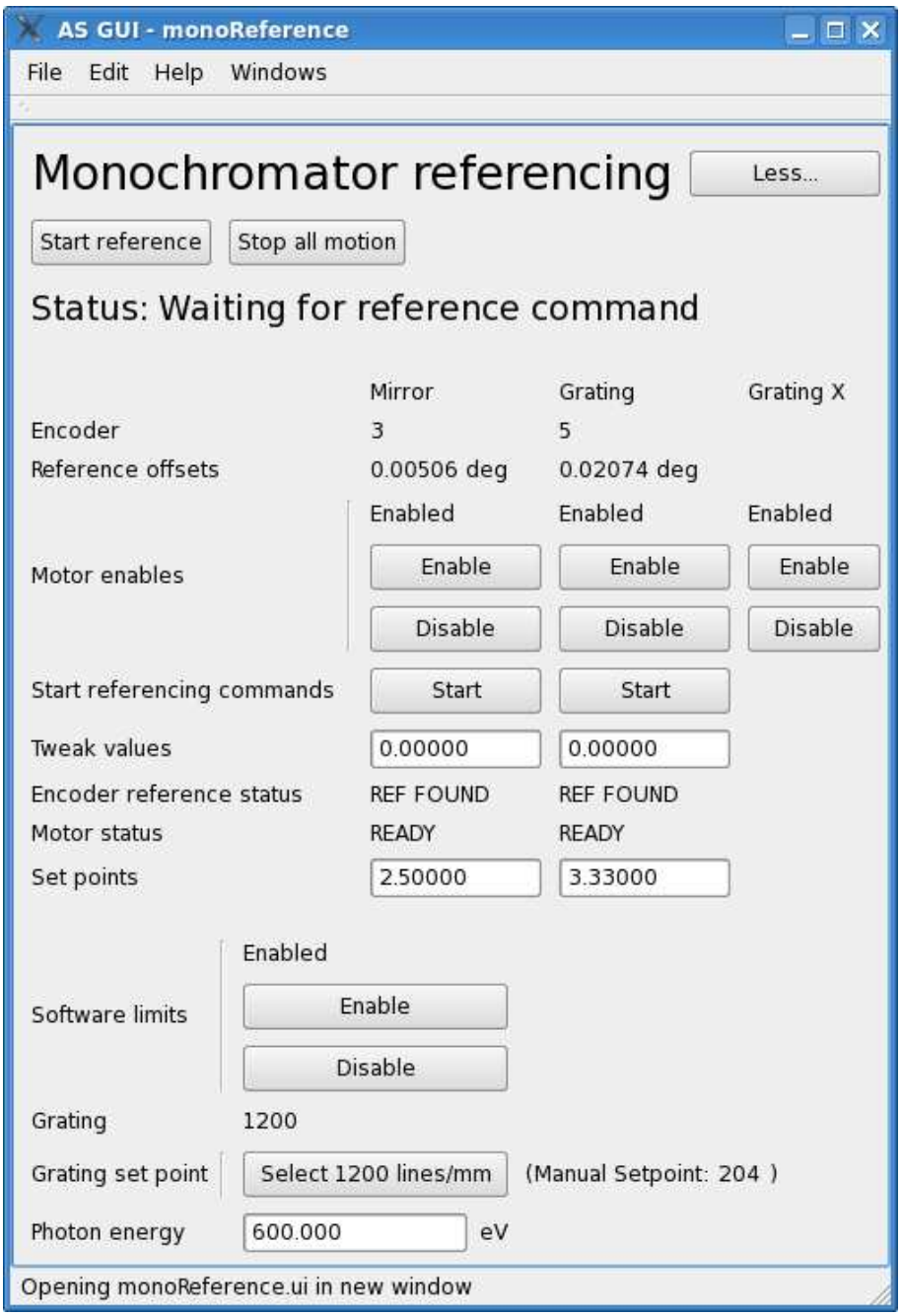


Figure 3.2: Monochromator referencing

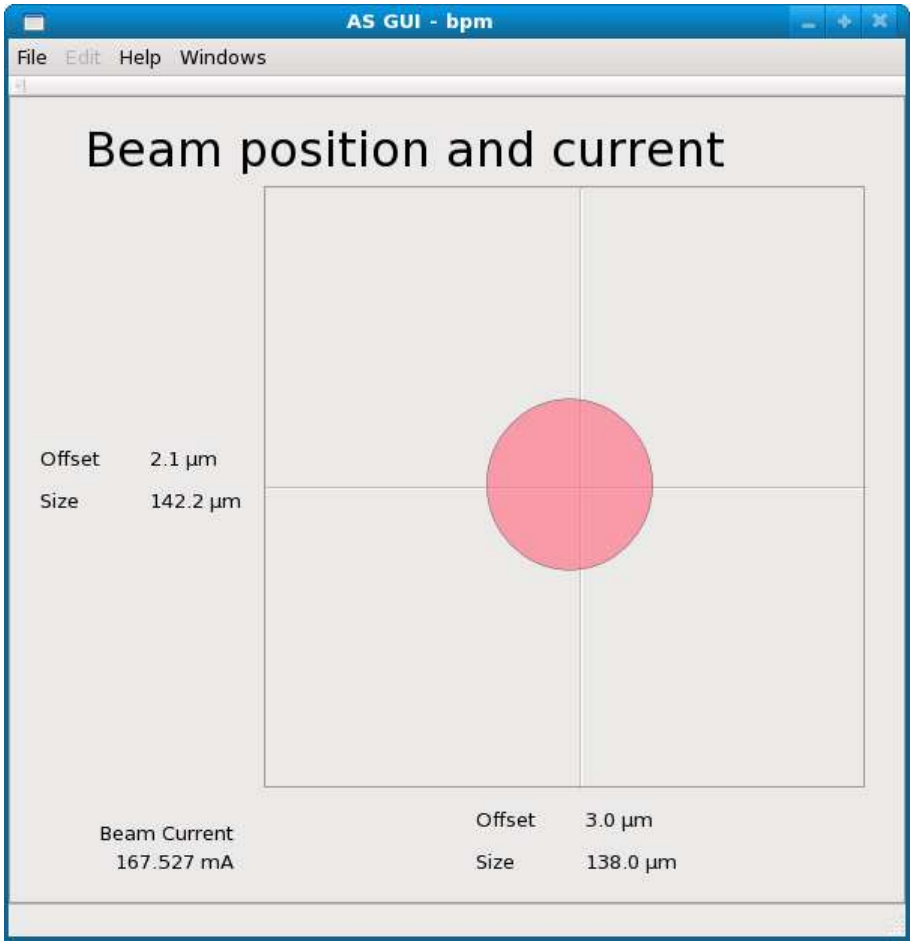


Figure 3.3: Beam position monitor

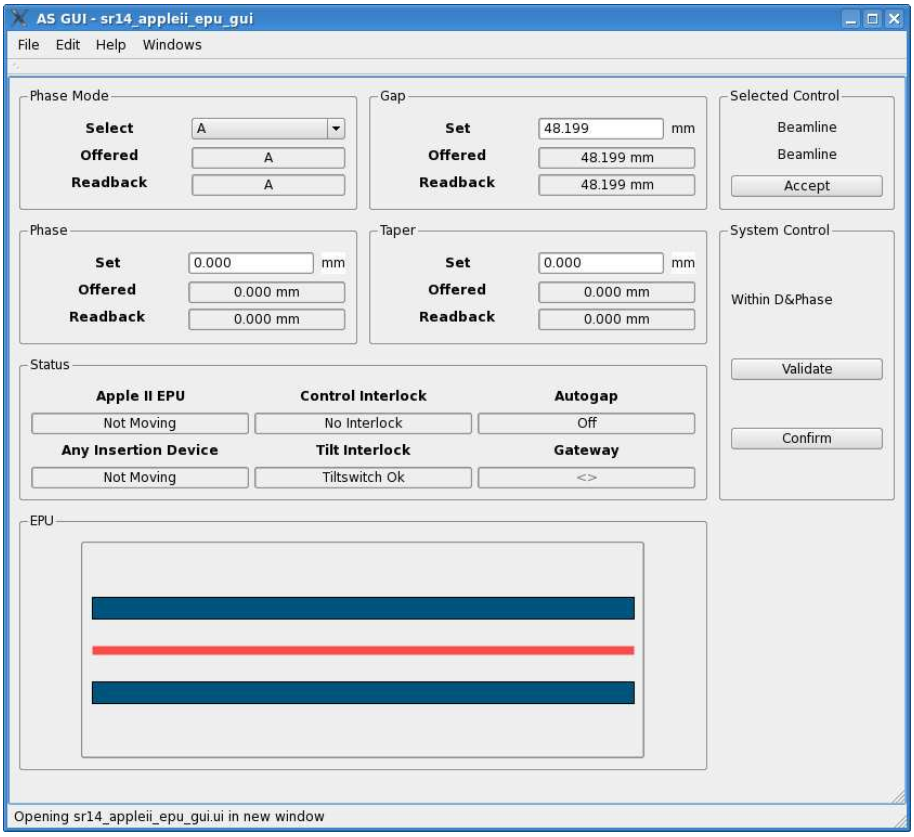


Figure 3.4: Insertion device

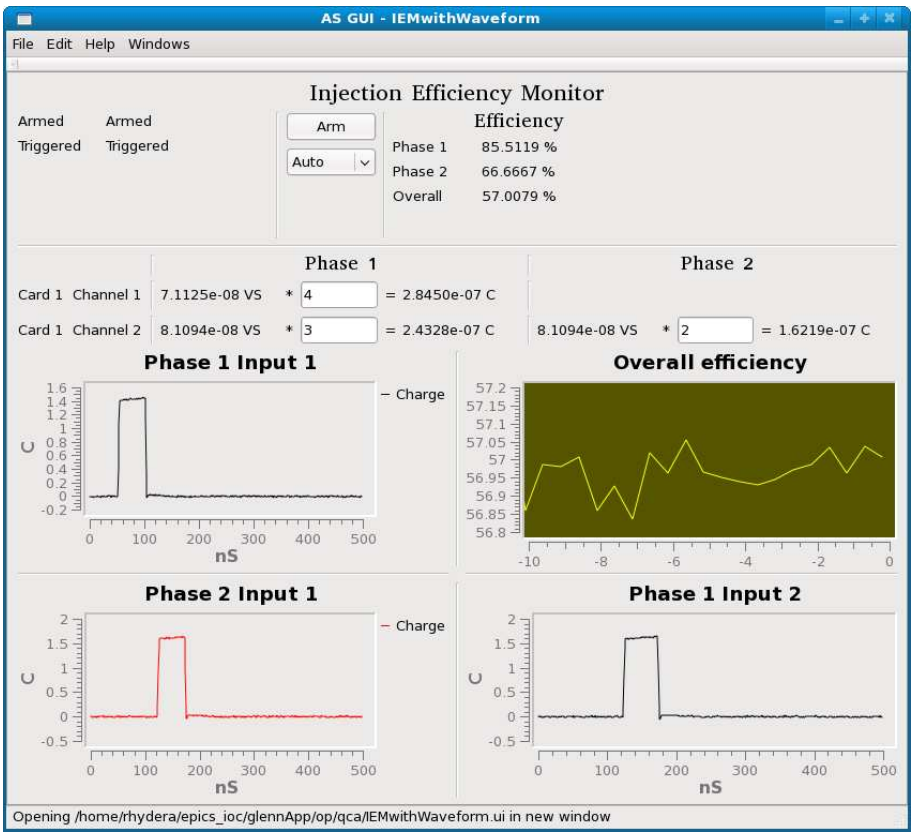


Figure 3.5: Injection efficiency monitor

Chapter 4

other applications using epicsqt widgets

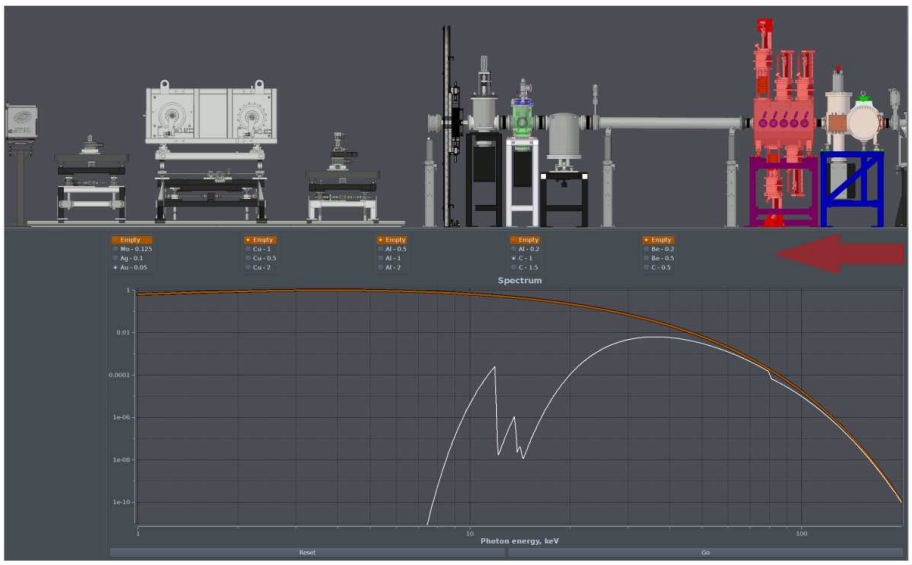


Figure 4.1: Medical Imaging beamline

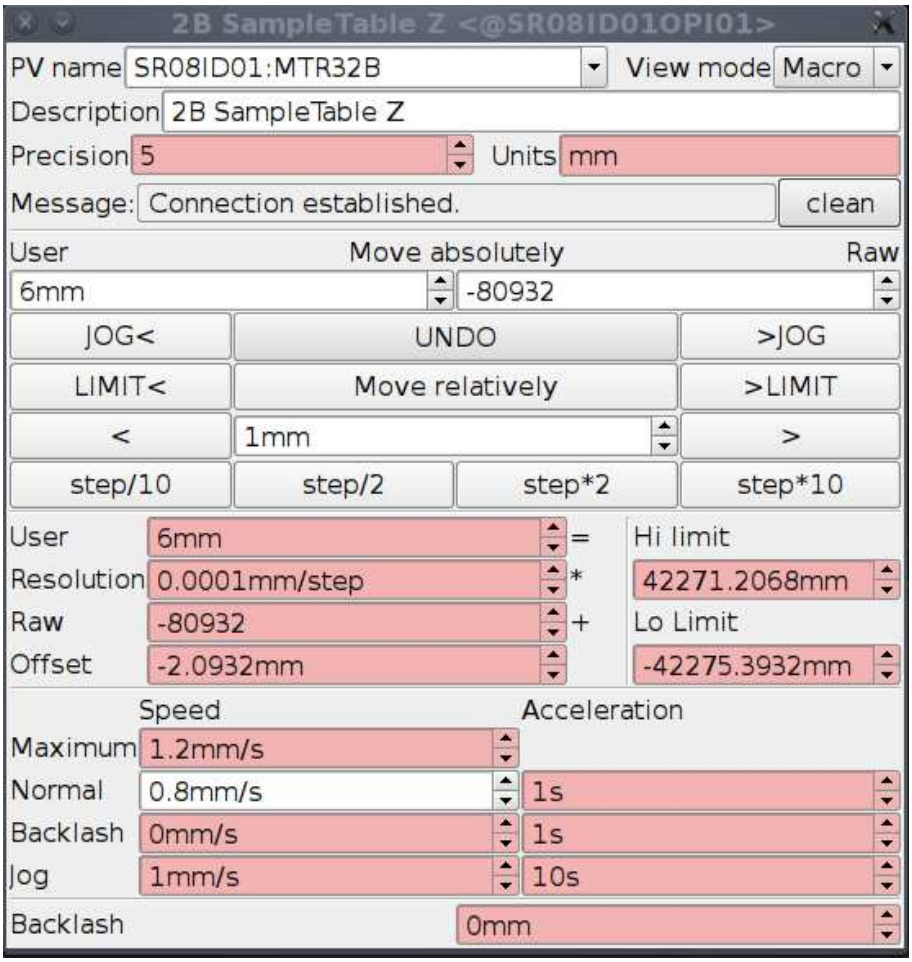


Figure 4.2: Motor controller

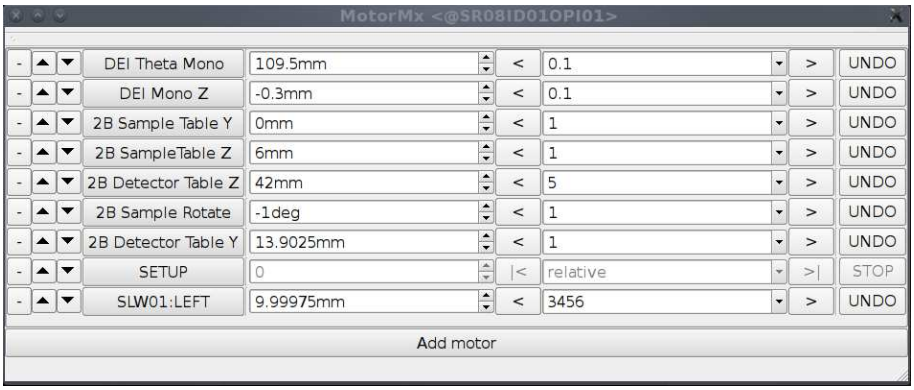


Figure 4.3: Motor controller

Chapter 5

Qt Designer

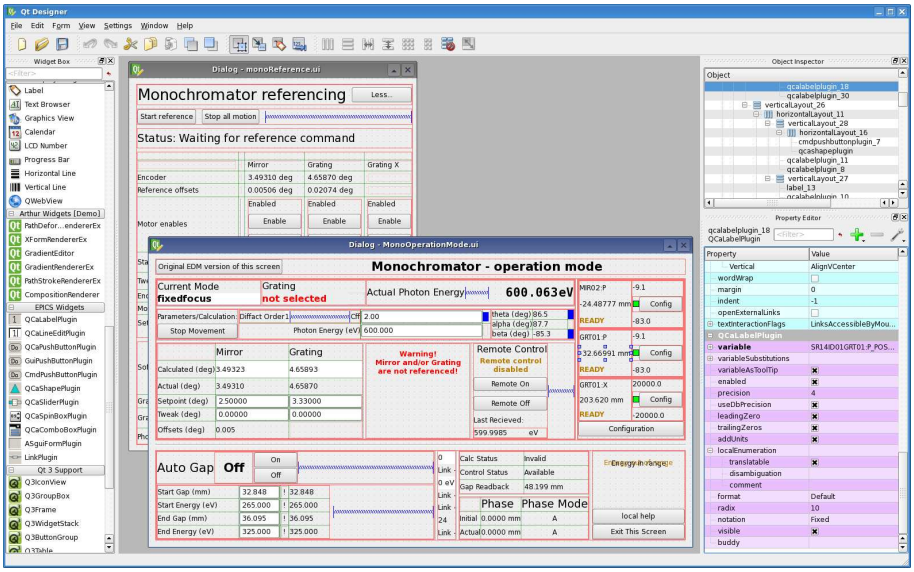


Figure 5.1: Editing multiple GUIs

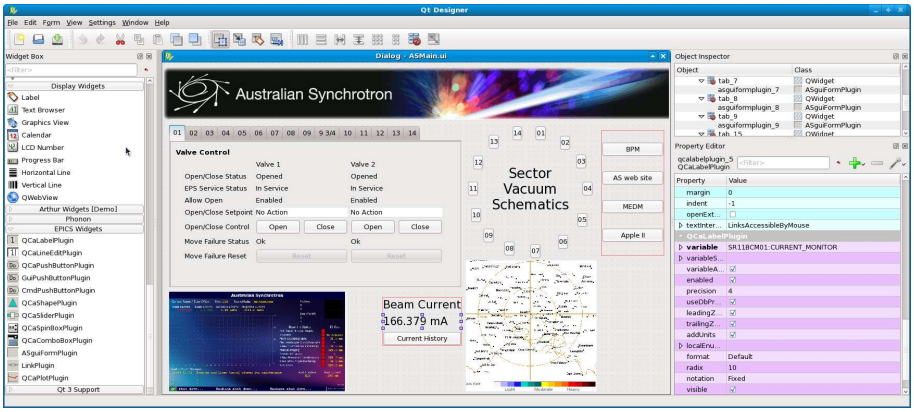


Figure 5.2: Editing a GUI

Chapter 6

Qt Creator

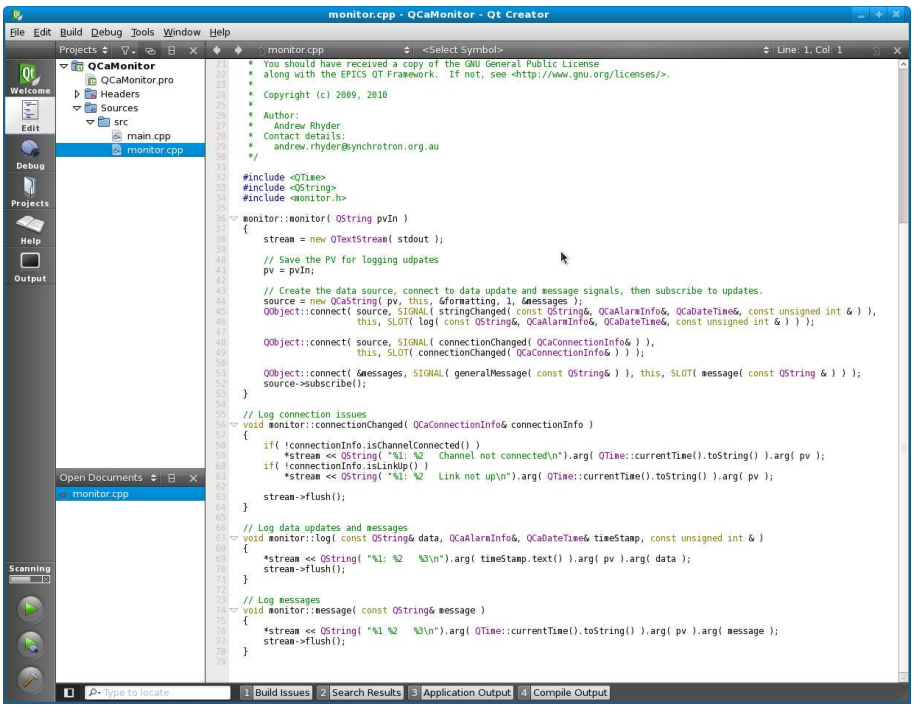


Figure 6.1: Application using epicsqt data source classes

Chapter 7

Class Index

7.1 Class Hierarchy

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

Chapter 8

Class Index

8.1 Class List

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

Chapter 9

Class Documentation

9.1 _CopyPaste Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.cpp

9.2 _Field Class Reference

Public Member Functions

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

Public Attributes

- QEWidget ∗ **qeWidget**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE

9.3 _Item Class Reference

Public Member Functions

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

Public Attributes

- QList< _Field * > **fieldList**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QEConfiguredL
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QEConfiguredL

9.4 _QDialogItem Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE

9.5 _QPushButtonGroup Class Reference

Public Slots

- void **buttonGroupClicked** ()

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QEConfiguredL
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QEConfiguredL

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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileBrowser/QEFileBr
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileBrowser/QEFileBr

9.7 _QTableWidgetLog Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELog/QELog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.cpp

9.9 applicationLauncher Class Reference

Public Types

- enum **programStartupOptions** { **PSO_NONE**, **PSO_TERMINAL**, **PSO_LOGOUTPUT**, **PSO_STDOUTPUT** }

Public Member Functions

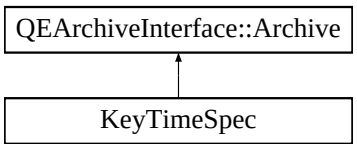
- void **launchImage** (VariableNameManager *variableNameManager, QImage image)
- void **launch** (VariableNameManager *variableNameManager, QObject *receiver)
- void **launchCommon** (VariableNameManager *variableNameManager, QTemporaryFile *tempFile=NULL, QObject *receiver=NULL)
- void **setProgram** (const QString programIn)
- QString **getProgram** () const
- void **setArguments** (const QStringList argumentsIn)
- QStringList **getArguments** () const
- void **setProgramStartupOption** (const programStartupOptions programStartupOptionIn)
- programStartupOptions **getProgramStartupOption** () const

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/applicationLauncher.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/applicationLauncher.cpp

9.10 QEArchiveInterface::Archive Struct Reference

Inheritance diagram for QEArchiveInterface::Archive::



Public Attributes

- int **key**
- QString **name**
- QString **path**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h

9.11 QEArchiveAccess::ArchiverPvInfo Struct Reference

Public Attributes

- int **key**
- QString **path**
- QCaDateTime **startTime**
- QCaDateTime **endTime**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.h

9.12 areaInfo Class Reference

Public Member Functions

- void **setX1** (long x)
- void **setY1** (long y)
- void **setX2** (long x)
- void **setY2** (long y)
- void **setR** (double r)
- void **setX** (long x)
- void **setY** (long y)
- void **setW** (long w)
- void **setH** (long h)
- void **setPoint1** (QPoint p1In)
- void **setPoint2** (QPoint p2In)
- void **clearX1** ()
- void **clearY1** ()
- void **clearX2** ()
- void **clearY2** ()
- void **clearR** ()
- void **clearX** ()
- void **clearY** ()
- void **clearW** ()
- void **clearH** ()
- bool **getStatus** ()
- QRect **getArea** ()
- QPoint **getPoint1** ()
- QPoint **getPoint2** ()
- bool **hasRotation** ()
- double **getRotation** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImage.h

9.13 QEAnalogIndicator::Band Struct Reference

Public Attributes

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogIndicator/QEAnalogIndicat

9.14 QEAnalogIndicator::BandList Class Reference

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogIndicator/QEA

9.15 carecord::ca_limit Struct Reference

Public Attributes

- double **upper**
- double **lower**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaRecord.h

9.16 caconnection::caChannel Struct Reference

Public Attributes

- bool **activated**
- int **creation**
- int **readResponse**
- int **writeResponse**
- int **state**
- short **type**
- bool **requestedElementCountSet**
- unsigned long **requestedElementCount**
- unsigned long **elementCount**
- chid **id**
- bool **writeWithCallback**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.h

9.17 caconnection::CaConnection Class Reference

Public Member Functions

- **CaConnection** (void ∗newParent)
- ca_responses **establishContext** (void(∗exceptionHandler)(struct exception_handler_args), void ∗args)
- ca_responses **establishChannel** (void(∗connectionHandler)(struct connection_handler_args), std::string channelName, priorities priority)
- ca_responses **establishSubscription** (void(∗subscriptionHandler)(struct event_handler_args), void ∗args, short initialDbrStructType, short updateDbrStructType)
- void **removeChannel** ()
- void **removeSubscription** ()
- ca_responses **readChannel** (void(∗readHandler)(struct event_handler_args), void ∗args, short dbrStructType)
- ca_responses **writeChannel** (void(∗writeHandler)(struct event_handler_args), void ∗args, short dbrStructType, int count, const void ∗dbrValue)
- void **setLinkState** (link_states newLinkState)
- link_states **getLinkState** ()
- channel_states **getChannelState** ()
- short **getChannelType** ()
- chid **getChannelId** ()
- std::string **getHostName** ()
- std::string **getFieldType** ()
- unsigned long **getElementCount** ()
- bool **getReadAccess** ()
- bool **getWriteAccess** ()
- void ∗ **getParent** ()
- bool **hasContext** ()
- bool **activatedChannel** ()
- bool **isSubscribed** ()
- void **setChannelElementCount** ()
- void **setChannelRequestedElementCount** (unsigned long requestedElementCountIn)
- void **setWriteWithCallback** (bool writeWithCallbackIn)
- bool **getWriteWithCallback** ()
- bool **getChannelActivated** ()
- unsigned long **getSubscribeElementCount** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.cpp

9.18 caconnection::caContext Struct Reference

Public Attributes

- bool **activated**
- int **creation**
- int **exception**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.h

9.19 caconnection::caLink Struct Reference

Public Attributes

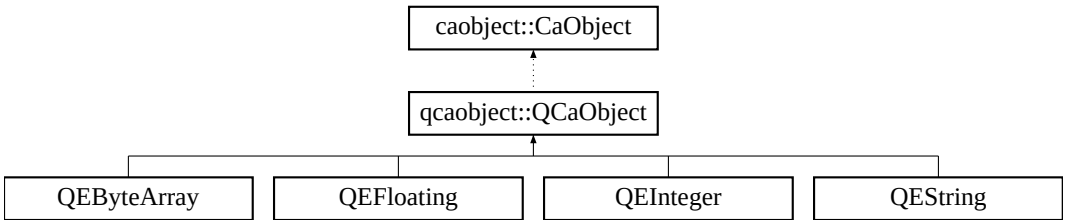
- double **searchTimeout**
- double **readTimeout**
- double **writeTimeout**
- link_states **state**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.h

9.20 caobject::CaObject Class Reference

Inheritance diagram for caobject::CaObject::



Public Member Functions

- void **setWriteWithCallback** (bool writeWithCallback)
- bool **getWriteWithCallback** ()
- virtual void **signalCallback** (callback_reasons)=0

Public Attributes

- CaRef * **myRef**
- bool **allowCallbacks**
- CaObjectPrivate * **caPrivate**

Protected Member Functions

- bool **isFirstUpdate** ()
- void * **getRecordCopyPtr** ()
- int **getEnumStateCount** ()
- std::string **getEnumState** (int position)
- int **getPrecision** ()
- double **getDisplayUpper** ()
- double **getDisplayLower** ()
- double **getAlarmUpper** ()
- double **getAlarmLower** ()
- double **getWarningUpper** ()
- double **getWarningLower** ()
- double **getControlUpper** ()
- double **getControlLower** ()
- std::string **getUnits** ()
- generic_types **getType** ()
- unsigned long **getTimeStampSeconds** ()
- unsigned long **getTimeStampNanoseconds** ()
- short **getAlarmStatus** ()

- short **getAlarmSeverity** ()
- std::string **getHostName** ()
- std::string **getFieldType** ()
- unsigned long **getElementCount** ()
- bool **getReadAccess** ()
- bool **getWriteAccess** ()
- void **inhibitCallbacks** ()

Static Protected Attributes

- static int **CA_UNIQUE_OBJECT_ID** = 0

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaObject.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaObject.cpp

9.21 CaObjectPrivate Class Reference

Public Member Functions

- **CaObjectPrivate** (CaObject ∗ownerIn)
- caconnection::ca_responses **setChannel** (std::string channelName, caconnection::priorities priority)
- caconnection::ca_responses **startSubscription** ()
- void **removeChannel** ()
- void **cancelSubscription** ()
- caconnection::ca_responses **readChannel** ()
- caconnection::ca_responses **writeChannel** (generic::Generic ∗newValue)
- caconnection::link_states **getLinkState** ()
- caconnection::channel_states **getChannelState** ()
- bool **processChannel** (struct event_handler_args args)

Static Public Member Functions

- static void **subscriptionHandler** (struct event_handler_args args)
- static void **readHandler** (struct event_handler_args args)
- static void **writeHandler** (struct event_handler_args args)
- static void **exceptionHandler** (struct exception_handler_args args)
- static void **connectionHandler** (struct connection_handler_args args)

Public Attributes

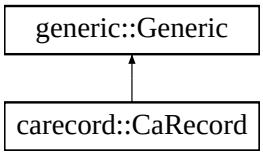
- caconnection::CaConnection ∗ **caConnection**
- carecord::CaRecord **caRecord**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaObjectPrivate.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaObject.cpp

9.22 carecord::CaRecord Class Reference

Inheritance diagram for carecord::CaRecord::



Public Member Functions

- **CaRecord** (CaRecord &param)
- CaRecord & **operator=** (CaRecord &param)
- void **setName** (std::string nameIn)
- void **setDbrType** (short dbrType)
- void **setValid** (bool newValid)
- void **updateProcessState** ()
- void **setStatus** (short newStatus)
- void **setAlarmSeverity** (short newSeverity)
- void **setPrecision** (short newPrecision)
- void **setUnits** (std::string newUnits)
- void **setTimeStamp** (unsigned long timeStampSecondsIn, unsigned long timeStampNanosecondsIn)
- void **clearEnumState** ()
- void **addEnumState** (std::string newEnumStates)
- void **setDisplayLimit** (double newUpper, double newLower)
- void **setAlarmLimit** (double newUpper, double newLower)
- void **setWarningLimit** (double newUpper, double newLower)
- void **setControlLimit** (double newUpper, double newLower)
- std::string **getName** ()
- short **getDbrType** (const dbr_translation_type type)
- short **getDbrTranslation** (const short translationMatrix[TYPE_COUNT][2], short type)
- bool **isValid** ()
- process_state **getProcessState** ()
- bool **isFirstUpdate** ()
- short **getStatus** ()
- short **getAlarmSeverity** ()
- short **getPrecision** ()
- std::string **getUnits** ()
- unsigned long **getTimeStampSeconds** ()
- unsigned long **getTimeStampNanoseconds** ()
- std::string **getEnumState** (int position)
- int **getEnumStateCount** ()

- ca_limit **getDisplayLimit** ()
- ca_limit **getAlarmLimit** ()
- ca_limit **getWarningLimit** ()
- ca_limit **getControlLimit** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaRecord.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaRecord.cpp

9.23 CaRef Class Reference

Public Member Functions

- **CaRef** (void ∗ownerIn, bool ownerIsCaObjectIn)
- void **init** (void ∗ownerIn, bool ownerIsCaObjectIn)
- void **discard** ()
- void **setPV** (std::string variableIn)
- void **setChannelId** (void ∗channelIn)
- void ∗ **getRef** (void ∗channelIn, bool ignoreZeroId=false)
- std::string **getVariable** ()

Static Public Member Functions

- static CaRef ∗ **getCaRef** (void ∗ownerIn, bool ownerIsCaObjectIn)
- static void **accessLock** ()
- static void **accessUnlock** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaRef.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaRef.cpp

9.24 caconnection::caSubscription Struct Reference

Public Attributes

- bool **activated**
- int **creation**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/CaConnection.h

9.25 changeEventFilter Class Reference

Public Member Functions

- **changeEventFilter** (styleManager ∗managerIn)

Protected Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/styleManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/styleManager.cpp

9.26 componentHostListItem Class Reference

Public Member Functions

- **componentHostListItem** (QWidget ∗widgetIn, QEActionRequests::Options creationOptionIn, bool hiddenIn, QString titleIn)
- **componentHostListItem** (componentHostListItem ∗item)

Public Attributes

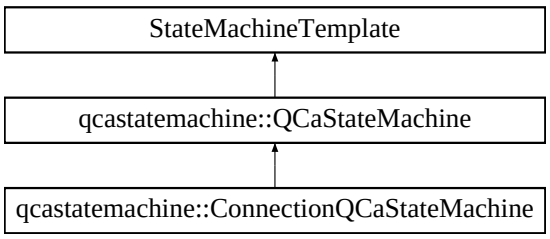
- QWidget ∗ **widget**
- QEActionRequests::Options **creationOption**
- bool **hidden**
- QString **title**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.cpp

9.27 qcastatemachine::ConnectionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ConnectionQCaStateMachine::



Public Member Functions

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

Static Public Attributes

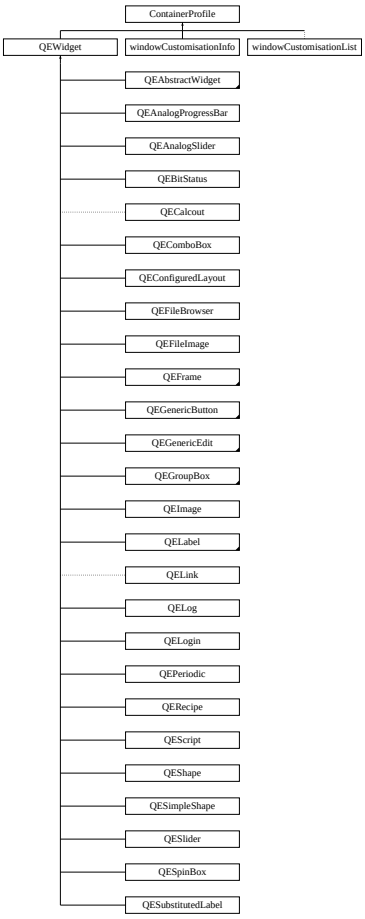
- static int **disconnectedCount** = 0
- static int **connectedCount** = 0

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.cpp

9.28 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** ()
- void **addPriorityMacroSubstitutions** (QString macroSubstitutionsIn)
- void **removePriorityMacroSubstitutions** ()

- QObject * **getGuiLaunchConsumer** ()
- QString **getPath** ()
- QStringList **getPathList** ()
- QString **getParentPath** ()
- void **setPublishedParentPath** (QString publishedParentPathIn)
- QString **getMacroSubstitutions** ()
- bool **isProfileDefined** ()
- bool **areUserLevelPasswordsSet** ()
- QStringList **getEnvPathList** ()
- QString **getUserLevelPassword** (userLevelTypes::userLevels level)
- void **setUserLevelPassword** (userLevelTypes::userLevels level, QString passwordIn)
- void **addContainedWidget** (QEWidget *containedWidget)
- QEWidget * **getNextContainedWidget** ()
- void **removeContainedWidget** (QEWidget *containedWidget)
- unsigned int **getMessageFormId** ()
- unsigned int **getPublishedMessageFormId** ()
- void **setPublishedMessageFormId** (unsigned int publishedMessageFormIdIn)
- bool **setDontActivateYet** (bool dontActivateIn)
- bool **getDontActivateYet** ()
- void **releaseProfile** ()
- void **publishOwnProfile** ()
- void **setUserLevel** (userLevelTypes::userLevels level)
- userLevelTypes::userLevels **getUserLevel** ()
- virtual void **userLevelChangedGeneral** (userLevelTypes::userLevels)
- PersistanceManager * **getPersistanceManager** ()

Static Public Member Functions

- static QChar **platformSeperator** ()
- static QString **getUserLevelName** (userLevelTypes::userLevels userLevelValue)
- static userLevelTypes::userLevels **getUserLevelValue** (QString userLevelName)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.cpp

9.29 QEArchiveInterface::Context Struct Reference

Public Attributes

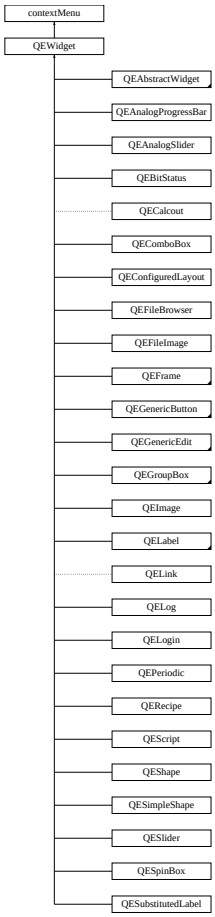
- QEArchiveInterface::Methods **method**
- QObject * **userData**
- unsigned int **requested_element**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h

9.30 contextMenu Class Reference

Inheritance diagram for contextMenu::



Public Types

- enum **contextMenuOptions** {

  CM_NOOPTION, **CM_COPY_VARIABLE**, **CM_COPY_DATA**, **CM_-PASTE**,

  CM_DRAG_VARIABLE, **CM_DRAG_DATA**, **CM_SHOW_PV_-PROPERTIES**, **CM_ADD_TO_STRIPCHART**,

  CM_ADD_TO_SCRATCH_PAD, **CM_ADD_TO_PLOTTER**, **CM_ADD_-TO_TABLE**, **CM_SHOW_AS_HISTOGRAM**,

  CM_GENERAL_PV_EDIT, **CM_SPECIFIC_WIDGETS_START_HERE** }
- typedef QSet< contextMenuOptions > **ContextMenuOptionSets**

Public Member Functions

- **contextMenu** (QEWidget *qewIn, QWidget *ownerIn)
- void **setupContextMenu** (const ContextMenuOptionSets &menuSet=contextMenu::defaultMenuSet())
- void **setEditPvUserLevel** (const userLevelTypes::userLevels level)
- bool **getEditPvUserLevel** () const
- void **clearContextMenuRequestHandling** ()
- void **setContextMenuOptions** (const ContextMenuOptionSets &menuSet)
- void **setNumberOfContextMenuItems** (const int numberOfItems)
- bool **isDraggingVariable** ()
- virtual QMenu * **buildContextMenu** ()
- virtual void **contextMenuTriggered** (int selectedItemNum)
- virtual QString **copyVariable** ()
- virtual QVariant **copyData** ()
- virtual void **paste** (QVariant)
- QAction * **showContextMenuGlobal** (const QPoint &globalPos)
- QAction * **showContextMenu** (const QPoint &pos)
- QAction * **showContextMenuGlobal** (QMenu *menu, const QPoint &global-Pos)
- QAction * **showContextMenu** (QMenu *menu, const QPoint &pos)
- void **addMenuItem** (QMenu *menu, const QString &title, const bool checkable, const bool checked, const int option)

Static Public Member Functions

- static ContextMenuOptionSets **defaultMenuSet** ()
- static bool **insertBefore** (QMenu *menu, QAction *action, const int option)
- static bool **insertAfter** (QMenu *menu, QAction *action, const int option)

Protected Member Functions

- void **setConsumer** (QObject *consumer)

Friends

- class QEContextMenuObject

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/contextMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/contextMenu.c

9.31 customisationLog Class Reference

Public Member Functions

- void **add** (const QString message)
- void **add** (const QString message, const QString param1)
- void **add** (const QString message, const QString param1, const QString param2)

- void **startLevel** ()
- void **endLevel** ()
- void **flagError** ()
- const QString **getLog** ()
- bool **getError** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h

9.32 QEPeriodic::elementInfoStruct Struct Reference

Public Attributes

- unsigned int **number**
- double **atomicWeight**
- QString **name**
- QString **symbol**
- double **meltingPoint**
- double **boilingPoint**
- double **density**
- unsigned int **group**
- double **ionizationEnergy**
- unsigned int **tableRow**
- unsigned int **tableCol**
- Category **category**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.h

9.33 FFThread Class Reference

Public Slots

- void **stopGracefully** ()

Signals

- void **updateSignal** (FFBuffer ∗buf)

Protected Member Functions

- void **run** ()

Friends

- class MpegSource

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/mpeg.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_mpeg.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/mpeg.cpp

9.34 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/flipRotateMenu
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/flipRotateMenu

9.35 fullScreenWindow Class Reference

Signals

- void **fullScreenResize** ()

Public Member Functions

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

Protected Member Functions

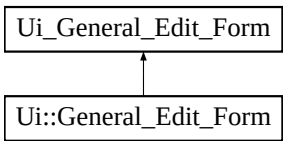
- void **resizeEvent** (QResizeEvent ∗event)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/fullScreenWindow.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_fullScreenWindow.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/fullScreenWindow.cpp

9.36 Ui::General_Edit_Form Class Reference

Inheritance diagram for Ui::General_Edit_Form::

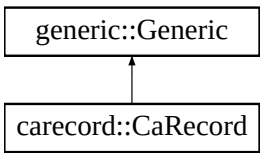


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEGeneralEdit.h

9.37 generic::Generic Class Reference

Inheritance diagram for generic::Generic::



Public Member Functions

- **Generic** (std::string newValue)
- **Generic** (short newValue)
- **Generic** (short *newValueArray, unsigned long countIn)
- **Generic** (unsigned short newValue)
- **Generic** (unsigned short *newValueArray, unsigned long countIn)
- **Generic** (unsigned char newValue)
- **Generic** (unsigned char *newValueArray, unsigned long countIn)
- **Generic** (qint32 newValue)
- **Generic** (qint32 *newValueArray, unsigned long countIn)
- **Generic** (quint32 newValue)
- **Generic** (quint32 *newValueArray, unsigned long countIn)
- **Generic** (float newValue)
- **Generic** (float *newValueArray, unsigned long countIn)
- **Generic** (double newValue)
- **Generic** (double *newValueArray, unsigned long countIn)
- **Generic** (Generic &param)
- Generic & **operator=** (Generic &param)
- void **setString** (std::string newValue)
- void **setShort** (short newValue)
- void **setShort** (short *newValueArray, unsigned long countIn)
- void **updateShort** (short newValue, unsigned long arrayIndex)
- void **setUnsignedShort** (unsigned short newValue)
- void **setUnsignedShort** (unsigned short *newValueArray, unsigned long countIn)
- void **updateUnsignedShort** (unsigned short newValue, unsigned long arrayIndex)
- void **setUnsignedChar** (unsigned char newValue)
- void **setUnsignedChar** (unsigned char *newValueArray, unsigned long countIn)
- void **updateUnsignedChar** (unsigned char newValue, unsigned long arrayIndex)
- void **setLong** (qint32 newValue)
- void **setLong** (qint32 *newValueArray, unsigned long countIn)
- void **updateLong** (qint32 newValue, unsigned long arrayIndex)

- void **setUnsignedLong** (quint32 newValue)
- void **setUnsignedLong** (quint32 *newValueArray, unsigned long countIn)
- void **updateUnsignedLong** (quint32 newValue, unsigned long arrayIndex)
- void **setFloat** (float newValue)
- void **setFloat** (float *newValueArray, unsigned long countIn)
- void **updateFloat** (float newValue, unsigned long arrayIndex)
- void **setDouble** (double newValue)
- void **setDouble** (double *newValueArray, unsigned long countIn)
- void **updateDouble** (double newValue, unsigned long arrayIndex)
- std::string **getString** ()
- void **getString** (char **valueArray, unsigned long *countOut=NULL)
- short **getShort** ()
- void **getShort** (short **valueArray, unsigned long *countOut=NULL)
- unsigned short **getUnsignedShort** ()
- void **getUnsignedShort** (unsigned short **valueArray, unsigned long *countOut=NULL)
- unsigned char **getUnsignedChar** ()
- void **getUnsignedChar** (unsigned char **valueArray, unsigned long *countOut=NULL)
- qint32 **getLong** ()
- void **getLong** (qint32 **valueArray, unsigned long *countOut=NULL)
- quint32 **getUnsignedLong** ()
- void **getUnsignedLong** (quint32 **valueArray, unsigned long *countOut=NULL)
- float **getFloat** ()
- void **getFloat** (float **valueArray, unsigned long *countOut=NULL)
- double **getDouble** ()
- void **getDouble** (double **valueArray, unsigned long *countOut=NULL)
- unsigned long **getArrayCount** ()
- generic_types **getType** ()
- void **setType** (generic_types newType)

Protected Member Functions

- void **cloneValue** (Generic *param)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/Generic.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/api/Generic.cpp

9.38 histogram Class Reference

Public Member Functions

- **histogram** (QWidget ∗parent, imageDisplayProperties ∗idp)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessContrast.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessContrast.cpp

9.39 histogramScroll Class Reference

Public Member Functions

- **histogramScroll** (QWidget ∗parent, imageDisplayProperties ∗idp)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessCont
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessCont

9.40 historicImage Class Reference

Public Member Functions

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

Public Attributes

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/recording.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImage.cpp

9.41 imageContextMenu Class Reference

Public Types

- enum **imageContextMenuOptions** {

  ICM_NONE = contextMenu::CM_SPECIFIC_WIDGETS_START_HERE, **ICM_SAVE**, **ICM_PAUSE**, **ICM_ENABLE_TIME**,

  ICM_ENABLE_CURSOR_PIXEL, **ICM_ABOUT_IMAGE**, **ICM_ENABLE_VERT1**, **ICM_ENABLE_VERT2**,

  ICM_ENABLE_VERT3, **ICM_ENABLE_VERT4**, **ICM_ENABLE_VERT5**, **ICM_ENABLE_HOZ1**,

  ICM_ENABLE_HOZ2, **ICM_ENABLE_HOZ3**, **ICM_ENABLE_HOZ4**, **ICM_ENABLE_HOZ5**,

  ICM_ENABLE_AREA1, **ICM_ENABLE_AREA2**, **ICM_ENABLE_AREA3**, **ICM_ENABLE_AREA4**,

  ICM_ENABLE_LINE, **ICM_ENABLE_TARGET**, **ICM_ENABLE_BEAM**, **ICM_DISPLAY_BUTTON_BAR**,

  ICM_DISPLAY_IMAGE_DISPLAY_PROPERTIES, **ICM_DISPLAY_RECORDER**, **ICM_ZOOM_SELECTED**, **ICM_ZOOM_FIT**,

  ICM_ZOOM_PLUS, **ICM_ZOOM_MINUS**, **ICM_ZOOM_10**, **ICM_ZOOM_25**,

  ICM_ZOOM_50, **ICM_ZOOM_75**, **ICM_ZOOM_100**, **ICM_ZOOM_150**,

  ICM_ZOOM_200, **ICM_ZOOM_300**, **ICM_ZOOM_400**, **ICM_ROTATE_NONE**,

  ICM_ROTATE_RIGHT, **ICM_ROTATE_LEFT**, **ICM_ROTATE_180**, **ICM_FLIP_HORIZONTAL**,

  ICM_FLIP_VERTICAL, **ICM_SELECT_PAN**, **ICM_SELECT_HSLICE1**, **ICM_SELECT_HSLICE2**,

  ICM_SELECT_HSLICE3, **ICM_SELECT_HSLICE4**, **ICM_SELECT_HSLICE5**, **ICM_SELECT_VSLICE1**,

  ICM_SELECT_VSLICE2, **ICM_SELECT_VSLICE3**, **ICM_SELECT_VSLICE4**, **ICM_SELECT_VSLICE5**,

  ICM_SELECT_AREA1, **ICM_SELECT_AREA2**, **ICM_SELECT_AREA3**, **ICM_SELECT_AREA4**,

  ICM_SELECT_PROFILE, **ICM_SELECT_TARGET**, **ICM_SELECT_BEAM**, **ICM_CLEAR_MARKUP**,

  ICM_SET_LEGEND, **ICM_THICKNESS_ONE_MARKUP**, **ICM_THICKNESS_SELECT_MARKUP**, **ICM_COPY_PLOT_DATA**,

  ICM_FULL_SCREEN, **ICM_DISPLAY_HSLICE1**, **ICM_DISPLAY_HSLICE2**, **ICM_DISPLAY_HSLICE3**,

  ICM_DISPLAY_HSLICE4, **ICM_DISPLAY_HSLICE5**, **ICM_DISPLAY_VSLICE1**, **ICM_DISPLAY_VSLICE2**,

ICM_DISPLAY_VSLICE3, **ICM_DISPLAY_VSLICE4**, **ICM_DISPLAY_-VSLICE5**, **ICM_DISPLAY_AREA1**,

ICM_DISPLAY_AREA2, **ICM_DISPLAY_AREA3**, **ICM_DISPLAY_-AREA4**, **ICM_DISPLAY_PROFILE**,

ICM_DISPLAY_TARGET, **ICM_DISPLAY_BEAM**, **ICM_DISPLAY_-TIMESTAMP**, **ICM_DISPLAY_ELLIPSE**,

ICM_OPTIONS }

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageContextMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageContextMenu.cpp

9.42 imageDisplayProperties Class Reference

Classes

- struct rgbPixel

Signals

- void **brightnessContrastAutoImage** ()
- void **imageDisplayPropertiesChange** ()

Public Member Functions

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

Public Attributes

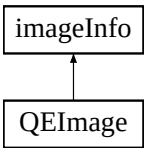
- int **zeroValue**
- int **fullValue**
- bool **defaultFullValue**
- unsigned int **range**
- unsigned int **maxP**
- unsigned int **minP**
- unsigned int **depth**
- unsigned int **bins** [HISTOGRAM_BINS]
- bool **statisticsSet**
- rgbPixel * **pixelLookup**
- QLabel * **histXLabel**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessContrast.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_brightnessContrast.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessContrast.cpp

9.43 imageInfo Class Reference

Inheritance diagram for imageInfo::



Public Member Functions

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

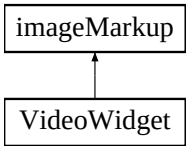
- void **infoUpdatePaused** ()
- void **infoUpdatePaused** (bool paused)
- void **setBriefInfoArea** (const bool briefIn)
- bool **getBriefInfoArea** ()
- void **freshImage** (QDateTime &time)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageInfo.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageInfo.cpp

9.44 imageMarkup Class Reference

Inheritance diagram for imageMarkup::



Public Types

- enum **markupIds** {

  MARKUP_ID_REGION1, **MARKUP_ID_REGION2**, **MARKUP_ID_-REGION3**, **MARKUP_ID_REGION4**,

  MARKUP_ID_H1_SLICE, **MARKUP_ID_H2_SLICE**, **MARKUP_ID_-H3_SLICE**, **MARKUP_ID_H4_SLICE**,

  MARKUP_ID_H5_SLICE, **MARKUP_ID_V1_SLICE**, **MARKUP_ID_-V2_SLICE**, **MARKUP_ID_V3_SLICE**,

  MARKUP_ID_V4_SLICE, **MARKUP_ID_V5_SLICE**, **MARKUP_ID_-LINE**, **MARKUP_ID_TARGET**,

  MARKUP_ID_BEAM, **MARKUP_ID_TIMESTAMP**, **MARKUP_ID_-ELLIPSE**, **MARKUP_ID_COUNT**,

  MARKUP_ID_NONE }
- enum **beamAndTargetOptions** { **CROSSHAIR1**, **CROSSHAIR2** }

Public Member Functions

- void **setShowTime** (bool visibleIn)
- bool **getShowTime** ()
- markupIds **getMode** ()
- void **setMode** (markupIds modeIn)
- void **setMarkupColor** (markupIds mode, QColor markupColorIn)
- QColor **getMarkupColor** (markupIds mode)
- bool **showMarkupMenu** (const QPoint &pos, const QPoint &globalPos)
- void **markupRegionValueChange** (int areaIndex, QRect area, bool display-Markups)
- void **markupH1ProfileChange** (int y, bool displayMarkups)
- void **markupH2ProfileChange** (int y, bool displayMarkups)
- void **markupH3ProfileChange** (int y, bool displayMarkups)
- void **markupH4ProfileChange** (int y, bool displayMarkups)
- void **markupH5ProfileChange** (int y, bool displayMarkups)
- void **markupV1ProfileChange** (int x, bool displayMarkups)
- void **markupV2ProfileChange** (int x, bool displayMarkups)

- void **markupV3ProfileChange** (int x, bool displayMarkups)
- void **markupV4ProfileChange** (int x, bool displayMarkups)
- void **markupV5ProfileChange** (int x, bool displayMarkups)
- void **markupLineProfileChange** (QPoint start, QPoint end, bool displayMarkups)
- void **markupTargetValueChange** (QPoint point, bool displayMarkups)
- void **markupBeamValueChange** (QPoint point, bool displayMarkups)
- void **markupEllipseValueChange** (QPoint point1, QPoint point2, double rotation, bool displayMarkups)
- void **markupValueChange** (int markup, bool displayMarkups, QPoint p1, QPoint p2=QPoint(), double rotation=0.0)
- QCursor **getCircleCursor** ()
- QCursor **getTargetCursor** ()
- QCursor **getVLineCursor** ()
- QCursor **getHLineCursor** ()
- QCursor **getLineCursor** ()
- QCursor **getRegionCursor** ()
- virtual void **markupSetCursor** (QCursor cursor)=0
- void **setMarkupLegend** (markupIds mode, QString legend)
- QString **getMarkupLegend** (markupIds mode)
- void **clearMarkup** (markupIds markupId)
- void **showMarkup** (markupIds markupId)
- void **displayMarkup** (markupIds markupId, bool state)
- bool **isMarkupVisible** (markupIds mode)
- double **getZoomScale** ()
- QSize **getImageSize** ()
- void **setImageSize** (const QSize &imageSizeIn)
- beamAndTargetOptions **getTargetOption** ()
- void **setTargetOption** (beamAndTargetOptions option)
- beamAndTargetOptions **getBeamOption** ()
- void **setBeamOption** (beamAndTargetOptions option)
- void **setBeamOrTargetOption** (markupIds item, beamAndTargetOptions option)

Public Attributes

- QVector< markupItem * > **items**
- QPoint **grabOffset**
- bool **markupAreasStale**
- QFont **legendFont**
- QFontMetrics * **legendFontMetrics**

Protected Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageMarkup.cpp

9.45 imageMarkupLegendSetText Class Reference

Public Member Functions

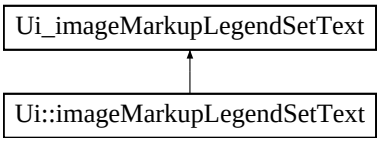
- **imageMarkupLegendSetText** (QString existingLegend, QWidget ∗parent=0)
- QString **getLegend** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageMarkupL
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageMarkupL

9.46 Ui::imageMarkupLegendSetText Class Reference

Inheritance diagram for Ui::imageMarkupLegendSetText::

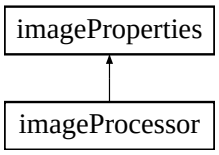


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_imageMarkupLegendSetText.h

9.47 imageProcessor Class Reference

`#include <imageProcessor.h>`Inheritance diagram for imageProcessor::



Signals

- void imageBuilt (QImage image, QString error)

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

Public Member Functions

- imageProcessor ()

  Constructor.

- ~imageProcessor ()

  Destructor.

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

  Save the image data for analysis processing and display.

- void buildImage ()

  Generate a new image.

- bool setWidth (unsigned long uValue)

  Set the image width.

- bool setHeight (unsigned long uValue)

  Set the image height.

- bool setNumDimensions (unsigned long uValue)

  Set the number of dimensions.

- bool setDimension0 (unsigned long uValue)

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

- bool setDimension1 (unsigned long uValue)

  Set the second dimension (height if two dimensions, width if three dimensions).

- bool setDimension2 (unsigned long uValue)

  Set the third dimension (unused if two dimensions, height if three dimensions).

- void setClippingOn (bool clippingOnIn)

  Set clipping flag. If true, setClippingLow() and setClippingHigh() are used to set clipping values.

- void setClippingLow (unsigned int value)

  Set pixel value below which low clip colour is displayed.

- void setClippingHigh (unsigned int value)

  Set pixel value above which high clip colour is displayed.

- int getScanOption ()

  Determine the way the input pixel data must be scanned to accommodate the required rotate and flip options.

- void getPixelTranslation ()

  Generate a lookup table to convert raw pixel values to display pixel values.

- unsigned int maxPixelValue ()

  Determine the maximum pixel value for the current format.

- unsigned int rotatedImageBuffWidth ()

  Return the image width following any rotation.

- unsigned int rotatedImageBuffHeight ()

  Return the image height following any rotation.

- imageDisplayProperties::rgbPixel getFalseColor (const unsigned char value)

  Get a false color representation for an entry fro the color lookup table.

- int getElementCount ()

  Determine the element count expected based on the available dimensions.

- bool validateDimensions ()

  Determine if the image dimensional information is valid.

- void getPixelRange (const QRect &area, unsigned int *min, unsigned int *max)

  Determine the range of pixel values an area of the image.

- bool hasImage ()

  Return true if the current image is empty.

- const unsigned char * getImageDataPtr (QPoint &pos)

Return a pointer to pixel data in the original image data.

- int getPixelValueFromData (const unsigned char *ptr)

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

- double getFloatingPixelValueFromData (const unsigned char *ptr)

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

- QImage copyImage ()

  Return a QImage based on the current image.

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

  Generate a series of pixel values from a vertical slice through the current image.

- void generateHSliceData (QVector< QPointF > &hSliceData, int y, unsigned int thickness)

  Generate a series of pixel values from a horizontal slice through the current image.

- void generateProfileData (QVector< QPointF > &profileData, QPoint point1, QPoint point2, unsigned int thickness)

  Generate a series of pseudo pixel values from an arbitrary line between two pixels.

- QRect rotateFlipToDataRectangle (const QRect &rect)

  Transform a rectangle from the image to the original data according to current rotation and flip options.

- QRect rotateFlipToDataRectangle (const QPoint &pos1, const QPoint &pos2)

  Transform a rectangle from the image to the original data according to current rotation and flip options.

- QPoint rotateFlipToDataPoint (const QPoint &pos)

  Transform a point from the image to the original data according to current rotation and flip options.

- QRect rotateFlipToImageRectangle (const QRect &rect)

  Transform a rectangle from the original data to the image according to current rotation and flip options.

- QRect rotateFlipToImageRectangle (const QPoint &pos1, const QPoint &pos2)

  Transform a rectangle from the original data to the image according to current rotation and flip options.

- QPoint rotateFlipToImagePoint (const QPoint &pos)

  Transform a point from the original data to the image according to current rotation and flip options.

- void **run** ()

Public Attributes

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

9.47.1 Detailed Description

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

9.47.2 Member Function Documentation

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

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

!! not done - copy of RGB1

!! not done - copy of RGB1

!! not done - copy of RGB1

!! not done - copy of RGB1

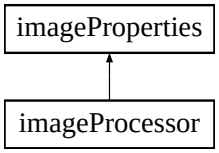
!! not done - copy of RGB1

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageProcessor.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_imageProcessor.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageProcessor.cpp

9.48 imageProperties Class Reference

`#include <imageProperties.h>`Inheritance diagram for imageProperties::



Public Types

- enum rotationOptions { ROTATION_0, ROTATION_90_RIGHT, ROTATION_-90_LEFT, ROTATION_180 }

Public Member Functions

- imageProperties ()

  Constructor.

- void **setRotation** (rotationOptions rotationIn)
- rotationOptions **getRotation** ()
- void **setFlipVert** (bool flipVertIn)
- bool **getFlipVert** ()
- void **setFlipHoz** (bool flipHozIn)
- bool **getFlipHoz** ()
- void **setImageBuffWidth** (unsigned long imageBuffWidthIn)
- void **setImageBuffHeight** (unsigned long imageBuffHeightIn)
- unsigned long **getImageBuffWidth** ()
- unsigned long **getImageBuffHeight** ()
- imageDataFormats::formatOptions **getFormat** ()
- void **setFormat** (imageDataFormats::formatOptions formatIn)
- bool **setFormat** (const QString &text)
- void **setBitDepth** (unsigned int bitDepthIn)
- unsigned int **getBitDepth** ()
- void **setElementsPerPixel** (unsigned long elementsPerPixelIn)
- void **setImageDisplayProperties** (imageDisplayProperties *imageDisplayPropsIn)
- void setWidthHeightFromDimensions ()

  // Update the image dimensions (width and height) from the area detector dimension variables.

- void invalidatePixelLookup ()

  recalculate (when next requried) pixel summary information

- QString getInfoText ()

  Generate textual information regarding the current image.

Protected Attributes

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

9.48.1 Detailed Description

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

9.48.2 Member Enumeration Documentation

9.48.2.1 enum imageProperties::rotationOptions

Image rotation options

Enumerator:

ROTATION_0 No image rotation.

ROTATION_90_RIGHT Rotate image 90 degrees clockwise.

ROTATION_90_LEFT Rotate image 90 degrees anticlockwise.

ROTATION_180 Rotate image 180 degrees.

9.48.3 Constructor & Destructor Documentation

9.48.3.1 imageProperties::imageProperties ()

Constructor. Construction. Set all image attributes to sensible defaults.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imagePropertie
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imagePropertie

9.49 imagePropertiesCore Class Reference

Public Member Functions

- **imagePropertiesCore** (QByteArray imageDataIn, unsigned long imageBuffWidthIn, unsigned long imageBuffHeightIn, int scanOptionIn, unsigned long bytesPerPixelIn, int pixelLowIn, int pixelHighIn, unsigned int bitDepthIn, imageDisplayProperties::rgbPixel *pixelLookupIn, imageDataFormats::formatOptions formatOptionIn, unsigned long imageDataSizeIn, imageDisplayProperties *imageDisplayPropsIn, unsigned int rotatedImageBuffWidthIn, unsigned int rotatedImageBuffHeightIn)
- QImage buildImageCore ()

9.49.1 Member Function Documentation

9.49.1.1 QImage imagePropertiesCore::buildImageCore ()

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

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

!! not done yet. do the same as for YUV422

!! not done yet. do the same as for YUV422

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageProperties.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageProcessor.cpp

9.50 imageUpdateIndicator Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageInfo.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/imageInfo.cpp

9.51 itemCheckInfo Class Reference

Public Member Functions

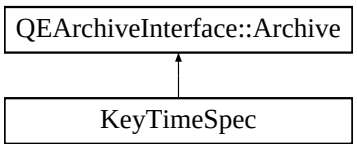
- **itemCheckInfo** (QDomElement itemElement)
- **itemCheckInfo** (const itemCheckInfo &other)
- const QString **getKey** ()
- const QString **getValue** ()
- bool **getCheckable** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cpp

9.52 KeyTimeSpec Class Reference

Inheritance diagram for KeyTimeSpec::



Public Attributes

- QCaDateTime **startTime**
- QCaDateTime **endTime**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.h

9.53 loginWidget Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.cpp

9.54 macroSubstitution Class Reference

`#include <macroSubstitution.h>`

Public Member Functions

- **macroSubstitution** (const QString &key, const QString &value)

Public Attributes

- QString **key**
- QString **value**

9.54.1 Detailed Description

Description:

A class to manage macro substitutions. Macro substitutions are often provioded as a string of keys and values.

This class parses such strings, and manages macro substitutions using a list of keys and values.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/macroSubstitution.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/macroSubstitution.cpp

9.55 macroSubstitutionList Class Reference

Public Member Functions

- **macroSubstitutionList** (const QString &string)
- void **addMacroSubstitutions** (const QString &string)
- QString **substitute** (const QString &string) const
- QString **getString** () const
- int **getCount** () const
- const QString **getKey** (const unsigned int i) const
- const QString **getValue** (const unsigned int i) const
- const QString **getValue** (const QString &key) const

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/macroSubstitution.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/macroSubstitution.cpp

9.56 MaiaFault Class Reference

Public Member Functions

- **MaiaFault** (int faultCode=0, QString faultString=QString(), QObject *parent=0)
- **MaiaFault** (const MaiaFault &other)
- QString **toString** ()

Public Attributes

- QMap< QString, QVariant > **fault**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaFault.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaFault.cpp

9.57 MaiaObject Class Reference

Public Slots

- void **parseResponse** (QString response, QNetworkReply ∗reply)

Signals

- void **aresponse** (QVariant &, QNetworkReply ∗reply)
- void **call** (const QString, const QList< QVariant >)
- void **fault** (int, const QString &, QNetworkReply ∗reply)

Public Member Functions

- **MaiaObject** (QObject ∗parent=0)
- QString **prepareCall** (QString method, QList< QVariant > args)

Static Public Member Functions

- static QDomElement **toXml** (QVariant arg)
- static QVariant **fromXml** (const QDomElement &elem)
- static QString **prepareResponse** (QVariant arg)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaObject.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaObject.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_maiaObject.cpp

9.58 MaiaXmlRpcClient Class Reference

Signals

- void **sslErrors** (QNetworkReply ∗reply, const QList< QSslError > &errors)

Public Member Functions

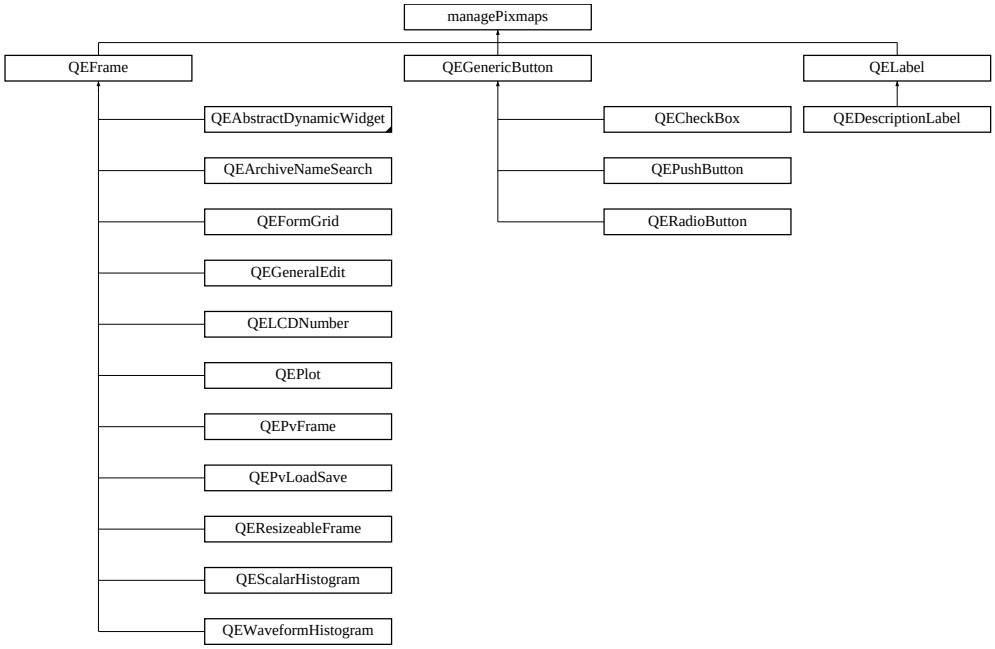
- **MaiaXmlRpcClient** (QObject ∗parent=0)
- **MaiaXmlRpcClient** (QUrl url, QObject ∗parent=0)
- **MaiaXmlRpcClient** (QUrl url, QString userAgent, QObject ∗parent=0)
- void **setUrl** (QUrl url)
- void **setUserAgent** (QString userAgent)
- QNetworkReply ∗ **call** (QString method, QList< QVariant > args, QObject ∗responseObject, const char ∗responseSlot, QObject ∗faultObject, const char ∗faultSlot)
- void **setSslConfiguration** (const QSslConfiguration &config)
- QSslConfiguration **sslConfiguration** () const

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaXmlRpcClient.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/maiaXmlRpcClient.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_maiaXmlRpcClient.cpp

9.59 managePixmaps Class Reference

Inheritance diagram for managePixmaps::



Public Member Functions

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

Protected Member Functions

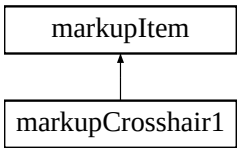
- virtual void **pixmapUpdated** (const int index)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/managePixmaps.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/managePixmaps.cpp

9.60 markupCrosshair1 Class Reference

Inheritance diagram for markupCrosshair1::



Public Member Functions

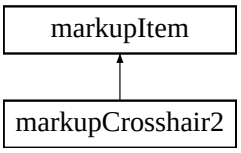
- **markupCrosshair1** (imageMarkup *ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupTarget.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupTarget.c

9.61 markupCrosshair2 Class Reference

Inheritance diagram for markupCrosshair2::



Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupBeam.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupBeam.cpp

9.62 markupDisplayMenu Class Reference

Public Member Functions

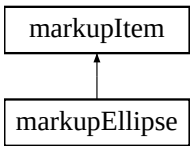
- **markupDisplayMenu** (QWidget ∗parent=0)
- void **setDisplayed** (imageContextMenu::imageContextMenuOptions option, bool state)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)
- bool **isDisplayed** (imageContextMenu::imageContextMenuOptions option)
- void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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupDisplay
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupDisplay

9.63 markupEllipse Class Reference

Inheritance diagram for markupEllipse::



Public Member Functions

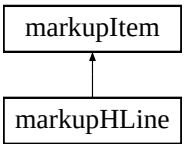
- **markupEllipse** (imageMarkup ∗ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupEllipse.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupEllipse.cpp

9.64 markupHLine Class Reference

Inheritance diagram for markupHLine::



Public Member Functions

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

9.64.1 Member Function Documentation

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

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

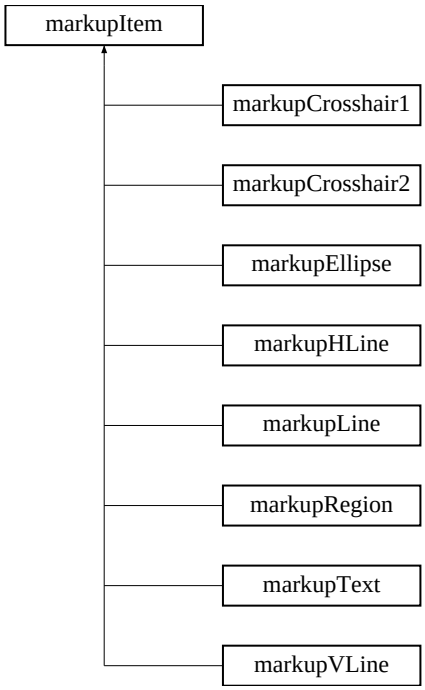
Implements markupItem.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupHLine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupHLine.c

9.65 markupItem Class Reference

Inheritance diagram for markupItem::



Public Types

- enum **markupHandles** {

  MARKUP_HANDLE_NONE, **MARKUP_HANDLE_START**, **MARKUP_-HANDLE_END**, **MARKUP_HANDLE_CENTER**,

  MARKUP_HANDLE_TL, **MARKUP_HANDLE_TR**, **MARKUP_-HANDLE_BL**, **MARKUP_HANDLE_BR**,

  MARKUP_HANDLE_T, **MARKUP_HANDLE_B**, **MARKUP_HANDLE_-L**, **MARKUP_HANDLE_R** }

Public Member Functions

- void **drawMarkupItem** (QPainter &p)
- QSize **getImageSize** ()
- virtual QPoint **origin** ()=0
- virtual void **moveTo** (const QPoint pos)=0
- virtual void **startDrawing** (const QPoint pos)=0
- virtual bool **isOver** (const QPoint point, QCursor ∗cursor)=0

- virtual QCursor **cursorForHandle** (const markupItem::markupHandles handle)=0
- virtual QPoint **getPoint1** ()=0
- virtual QPoint **getPoint2** ()=0
- virtual QCursor **defaultCursor** ()=0
- virtual void **nonInteractiveUpdate** (QPoint, QPoint, double)
- void **setThickness** (const unsigned int thicknessIn)
- unsigned int **getThickness** ()
- void **setLegend** (const QString legendIn)
- const QString **getLegend** ()
- void **setColor** (QColor colorIn)
- QColor **getColor** ()

Public Attributes

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

Protected Types

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

Protected Member Functions

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

Protected Attributes

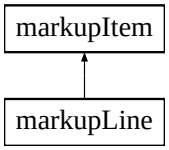
- markupHandles **activeHandle**
- imageMarkup * **owner**
- unsigned int **thickness**
- unsigned int **maxThickness**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupItem.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupItem.cpp

9.66 markupLine Class Reference

Inheritance diagram for markupLine::



Public Member Functions

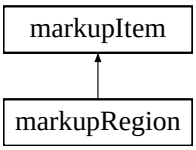
- **markupLine** (imageMarkup ∗ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupLine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupLine.cp

9.67 markupRegion Class Reference

Inheritance diagram for markupRegion::



Public Member Functions

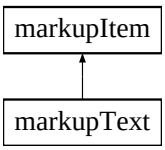
- **markupRegion** (imageMarkup ∗ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver** (const QPoint point, QCursor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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupRegion.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupRegion.cpp

9.68 markupText Class Reference

Inheritance diagram for markupText::



Public Member Functions

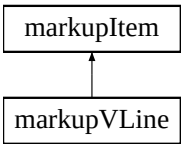
- **markupText** (imageMarkup ∗ownerIn, const bool interactiveIn, const bool reportOnMoveIn, const QString legendIn)
- void **setText** (QString textIn)
- void **startDrawing** (const QPoint pos)
- void **setArea** ()
- void **drawMarkup** (QPainter &p)
- void **moveTo** (const QPoint pos)
- bool **isOver**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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupText.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupText.cp

9.69 markupVLine Class Reference

Inheritance diagram for markupVLine::



Public Member Functions

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

9.69.1 Member Function Documentation

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

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

Implements markupItem.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupVLine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/markupVLine.cpp

9.70 menuItemToBeActivated Class Reference

Public Member Functions

- **menuItemToBeActivated** (menuItemToBeActivated *other)
- **menuItemToBeActivated** (windowCustomisationMenuItem *itemIn, QMenu *menuIn)

Public Attributes

- windowCustomisationMenuItem * **item**
- QMenu * **menu**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h

9.71 message_types Class Reference

Public Member Functions

- **message_types** (message_severities severityIn, message_kind_sets kind_setIn=MESSAGE_KIND_STANDARD)
- QString getSeverityName ()

  Function to provide string name for each message type severity.

Public Attributes

- message_severities **severity**
- message_kind_sets **kind_set**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.cpp

9.72 MpegSource Class Reference

Signals

- void **aboutToQuit** ()
- void **setDataImage** (const QByteArray &imageIn, unsigned long dataSize, unsigned long elements, unsigned long width, unsigned long height, imageDataFormats::formatOptions format, unsigned int depth)

Public Member Functions

- **MpegSource** (QObject ∗parent)
- QString **getURL** () const
- void **setURL** (const QString &urlIn)
- void **stopStream** ()
- void **startStream** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/mpeg.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_mpeg.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/mpeg.cpp

9.73 NamesResponseContext Class Reference

Public Member Functions

- **NamesResponseContext** (QEArchiveInterface *interfaceIn, QEArchiveInterface::Archive archiveIn, int i)

Public Attributes

- QEArchiveInterface * **interface**
- QEArchiveInterface::Archive **archive**
- int **instance**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp

9.74 QEGraphic::OwnPlot Class Reference

Public Member Functions

- **OwnPlot** (QEGraphic *parent=NULL)
- **OwnPlot** (const QwtText &title, QEGraphic *parent=NULL)

Protected Member Functions

- void **drawCanvas** (QPainter *painter)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphic.cpp

9.75 QEStripChartToolBar::OwnTabWidget Class Reference

Public Member Functions

- **OwnTabWidget** (QEStripChartToolBar *parent)

Public Attributes

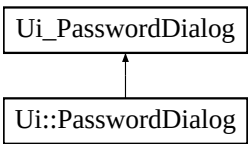
- QPushButton * **pushButtons** [NUMBER_OF_BUTTONS]
- QLabel * **yScaleStatus**
- QLabel * **timeStatus**
- QLabel * **durationStatus**
- QLabel * **numberOfOutandingRequests**
- QLabel * **timeModeStatus**
- QComboBox * **predefinedComboBox**
- QPushButton * **loadButton**
- QPushButton * **saveAsButton**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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartToolBar.c

9.76 Ui::PasswordDialog Class Reference

Inheritance diagram for Ui::PasswordDialog::



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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_PasswordDialog.h

9.77 PasswordDialog Class Reference

Public Member Functions

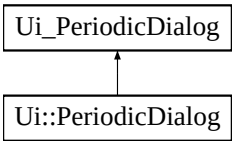
- **PasswordDialog** (QWidget ∗parent=0)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/PasswordDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/PasswordDialog.cpp

9.78 Ui::PeriodicDialog Class Reference

Inheritance diagram for Ui::PeriodicDialog::

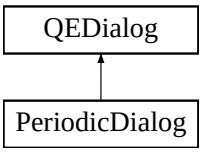


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_PeriodicDialog.h

9.79 PeriodicDialog Class Reference

Inheritance diagram for PeriodicDialog::



Public Slots

- int **exec** (QWidget *targetWidget)

Public Member Functions

- **PeriodicDialog** (QWidget *parent=0)
- void **setColourised** (const bool colouriseIn)
- bool **isColourised** () const
- void **setElement** (QString elementIn, QList< bool > &enabledList)
- Q_DECL_DEPRECATED void **setElement** (QString elementIn, QList< bool > &enabledList, QList< QString > &elementList)
- QString **getElement** () const
- int **getAtomicNumber** () const

Protected Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/PeriodicDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/PeriodicDialog.cpp

9.80 PersistanceManager Class Reference

Classes

- class ResourceLocker

Public Member Functions

- QObject * **getSaveRestoreObject** ()
- void **save** (const QString fileName, const QString rootName, const QString configName, const bool warnUser)
- void **restore** (const QString fileName, const QString rootName, const QString configName)
- void **saveWidget** (QEWidget *qewidget, const QString fileName, const QString rootName, const QString configName)
- void **restoreWidget** (QEWidget *qewidget, const QString fileName, const QString rootName, const QString configName)
- bool **isRestoring** () const
- PMElement **addNamedConfiguration** (QString name)
- PMElement **getNamedConfiguration** (QString name)
- QStringList **getConfigNames** (QString fileName, QString rootName)
- QStringList **getConfigNames** (QString fileName, QString rootName, bool &hasDefault)
- void **deleteConfigs** (const QString fileName, const QString rootName, const QStringList names, const bool warnUser)
- bool **isConfigurationPresent** (const QString fileName, const QString rootName, const QString configName)

Static Public Attributes

- static QString **defaultName**

Friends

- class PMElement

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa

9.81 playbackTimer Class Reference

Public Member Functions

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

Public Attributes

- recording ∗ **recorder**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/recording.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/recording.cpp

9.82 PMContext Class Reference

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa

9.83 PMElement Class Reference

Public Member Functions

- **PMElement** (PersistanceManager *ownerIn, QDomElement elementIn)
- PMElement **addElement** (QString name)
- void **addValue** (QString name, bool value)
- void **addValue** (QString name, int value)
- void **addValue** (QString name, double value)
- void **addValue** (QString name, QString value)
- void **addAttribute** (QString name, bool value)
- void **addAttribute** (QString name, int value)
- void **addAttribute** (QString name, double value)
- void **addAttribute** (QString name, QString value)
- PMElement **getElement** (QString name)
- PMElement **getElement** (QString name, int i)
- PMElement **getElement** (QString name, QString attrName, QString attrValue)
- PMElement **getElement** (QString name, QString attrName, int attrValue)
- PMElementList **getElementList** (QString name)
- bool **getValue** (QString name, bool &val)
- bool **getValue** (QString name, int &val)
- bool **getValue** (QString name, double &val)
- bool **getValue** (QString name, QString &val)
- bool **getAttribute** (QString name, bool &val)
- bool **getAttribute** (QString name, int &val)
- bool **getAttribute** (QString name, double &val)
- bool **getAttribute** (QString name, QString &val)
- bool **isNull** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceManager.cpp

9.84 PMElementList Class Reference

Public Member Functions

- **PMElementList** (PersistanceManager ∗ownerIn, QDomNodeList elementListIn)
- PMElement getElement (int i)
- int **count** ()

9.84.1 Member Function Documentation

9.84.1.1 PMElement PMElementList::getElement (int *i*)

!! check range of i

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa

9.85 pointInfo Class Reference

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImage.h

9.86 processManager Class Reference

Public Slots

- void **doRead** ()
- void **doReadToStandardOutput** ()
- void **doReadToStandardError** ()
- void **doFinished** (int, QProcess::ExitStatus)

Signals

- void **processCompleted** ()

Public Member Functions

- **processManager** (bool logOutput, bool useStandardIo, QTemporaryFile ∗tempFileIn)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/applicationLau
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_applicationLauncher.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/applicationLau

9.87 profilePlot Class Reference

Public Types

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

Public Member Functions

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/profilePlot.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/profilePlot.cpp

9.88 ProfilePublisher Class Reference

Public Member Functions

- **ProfilePublisher** (QEWidget *owner, const QString &prioritySubstitutions)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf

9.89 PublishedProfile Class Reference

Public Attributes

- QObject * **guiLaunchConsumer**
- QStringList **pathList**
- QString **parentPath**
- QList< QString > **macroSubstitutions**
- unsigned int **messageFormId**
- QList< WidgetRef > **containedWidgets**
- userLevelSignal **userSignal**
- QString **userLevelPassword**
- QString **scientistLevelPassword**
- QString **engineerLevelPassword**
- bool **profileDefined**
- PersistanceManager **persistanceManager**
- bool **dontActivateYet**
- bool **userLevelPasswordsSet**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.h

9.90 PushButtonSpecifications Struct Reference

Public Attributes

- int **side**
- int **gap**
- int **width**
- bool **isIcon**
- const QString **captionOrIcon**
- const QString **toolTip**
- const char * **member**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo

9.91 QEArchiveAccess::PVDataRequests Struct Reference

Public Attributes

- QObject * **userData**
- int **key**
- QString **pvName**
- QCaDateTime **startTime**
- QCaDateTime **endTime**
- int **count**
- QEArchiveInterface::How **how**
- unsigned int **element**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.h

9.92 QEArchiveAccess::PVDataResponses Struct Reference

Public Attributes

- QObject * **userData**
- bool **isSuccess**
- QCaDataPointList **pointsList**
- QString **pvName**
- QString **supplementary**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.h

9.93 QEArchiveInterface::PVName Struct Reference

Public Attributes

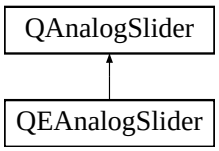
- QString **pvName**
- QCaDateTime **startTime**
- QCaDateTime **endTime**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h

9.94 QAnalogSlider Class Reference

`#include <QAnalogSlider.h>`Inheritance diagram for QAnalogSlider::



Public Slots

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

Signals

- void **valueChanged** (const double value)
- void **valueChanged** (const int value)
- void **appliedValue** (const double value)
- void **appliedValue** (const int value)

Public Member Functions

- QAnalogSlider (QWidget *parent=0)
- virtual ~QAnalogSlider ()

  Destruction.

- QSize **sizeHint** () const
- void **setColourBandList** (const QEColourBandList &bandList)
- QEColourBandList **getColourBandList** () const
- double **getValue** () const
- void setDesignPrecision (const int precision)

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

- int getDesignPrecision () const

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

- void setDesignMinimum (const double minimum)

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

- double getDesignMinimum () const

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

- void setDesignMaximum (const double maximum)

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

- double getDesignMaximum () const

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

- void setMinorInterval (const double minorInterval)

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

- double getMinorInterval () const

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

- void setMajorInterval (const double majorInterval)

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

- double getMajorInterval () const

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

- void setLeftText (const QString &leftText)

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

- QString getLeftText () const

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

- void setCentreText (const QString &centreText)

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

- QString getCentreText () const

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

- void setRightText (const QString &rightText)

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

- QString getRightText () const

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

- void **setShowSaveRevert** (const bool show)
- bool **getShowSaveRevert** () const
- void **setShowApply** (const bool show)
- bool **getShowApply** () const

Protected Slots

- virtual void **applyButtonClicked** (bool)

Protected Member Functions

- void **setIsActive** (const bool value)
- bool **getIsActive** ()
- QEAxisPainter * **getAxisPainter** ()
- virtual int **getPrecision** () const
- virtual double **getMinimum** () const
- virtual double **getMaximum** () const
- void **updateAxisAndSlider** ()

Properties

- int precision
- double minimum

  Speficies the mimimum allowed value.

- double maximum

  Speficies the maximum allowed value.

- double minorInterval
- double majorInterval
- bool tracking
- QString leftText

  Controls the left, centre and right texts.

- QString **centreText**
- QString **rightText**
- bool showSaveRevert

  Enables/disables the save-revert capability.

- bool showApply

  Enables/disables the apply value capability.

- double value

  Speficies the value.

9.94.1 Detailed Description

QAnalogSlider is a non EPICS aware slider that provides an analog equivilent of the QSlider. It is deemed analog as it can be set by/emits floating point (double) values as opposed to interger values. It is also decorated with a scale and text showing the current value; it also provides a save-restore capability.

9.94.2 Constructor & Destructor Documentation

9.94.2.1 QAnalogSlider::QAnalogSlider (QWidget * *parent* = 0) [explicit]

Create with default title.

9.94.3 Property Documentation

9.94.3.1 double QAnalogSlider::majorInterval [read, write]

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

9.94.3.2 double QAnalogSlider::minorInterval [read, write]

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

9.94.3.3 int QAnalogSlider::precision [read, write]

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

9.94.3.4 bool QAnalogSlider::tracking [read, write]

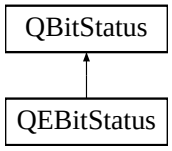
Controls when valueChanged signal is emitted. If tracking is enabled (the default), the slider emits the valueChanged () signal while the slider is being dragged. If tracking is disabled, the slider emits the valueChanged () signal only when the user releases the slider.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogSlider/QAnalogSlider.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QAnalogSlider.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogSlider/QAnalogSlider.cpp

9.95 QBitStatus Class Reference

Inheritance diagram for QBitStatus::



Public Types

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

Public Slots

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

Public Member Functions

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

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

Protected Member Functions

- void **setIsActive** (const bool value)
- bool **getIsActive** ()

Properties

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEBitStatus/QBitStatus.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEBitStatus/QBitStatus.cpp

9.96 QCaAlarmInfo Class Reference

Public Types

- typedef unsigned short **Status**
- typedef unsigned short **Severity**

Public Member Functions

- **QCaAlarmInfo** (const Status statusIn, const Severity severityIn, const QString &message="")
- bool **operator==** (const QCaAlarmInfo &other) const
- bool **operator!=** (const QCaAlarmInfo &other) const
- QString **statusName** () const
- QString **severityName** () const
- QString **messageText** () const
- bool **isInAlarm** () const
- bool **isMinor** () const
- bool **isMajor** () const
- bool **isInvalid** () const
- QString **style** () const
- QString **getStyleColorName** () const
- QString **getColorName** () const
- Severity **getSeverity** () const

Static Public Member Functions

- static Severity **getInvalidSeverity** ()
- static void **setStyleColorNames** (const QStringList &styleColorNames)
- static QStringList **getStyleColorNames** ()
- static void **setColorNames** (const QStringList &colorNames)
- static QStringList **getColorNames** ()
- static QStringList **getDefaultStyleColorNames** ()
- static QStringList **getDefaultColorNames** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaAlarmInfo.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaAlarmInfo.cpp

9.97 QCaConnectionInfo Class Reference

Public Member Functions

- **QCaConnectionInfo** (unsigned short channelStateIn, unsigned short linkStateIn, QString recordName)
- bool **isChannelConnected** ()
- bool **isLinkUp** ()
- QString **variable** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaConnectionInfo.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaConnectionInfo.cpp

9.98 QCaDataPoint Class Reference

Public Member Functions

- bool **isDisplayable** () const
- QString **toString** () const
- QString **toString** (const QCaDateTime &originDateTime) const

Public Attributes

- double **value**
- QCaDateTime **datetime**
- QCaAlarmInfo **alarm**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDataPoint.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDataPoint.cpp

9.99 QCaDataPointList Class Reference

Public Member Functions

- void **reserve** (const int size)
- void **clear** ()
- void **removeLast** ()
- void **removeFirst** ()
- void **append** (const QCaDataPointList &other)
- void **append** (const QCaDataPoint &other)
- void **replace** (const int i, const QCaDataPoint &t)
- int **count** () const
- QCaDataPoint **value** (const int j) const
- QCaDataPoint **last** () const
- void **truncate** (const int position)
- int **indexBeforeTime** (const QCaDateTime &searchTime, const int defaultIndex) const
- const QCaDataPoint * **findNearestPoint** (const QCaDateTime &searchTime) const
- void **resample** (const QCaDataPointList &source, const double interval, const QCaDateTime &endTime)
- void **compact** (const QCaDataPointList &source)
- void **toStream** (QTextStream &target, bool withIndex, bool withRelativeTime) const

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDataPoint.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDataPoint.cpp

9.100 QCaDateTime Class Reference

Public Member Functions

- **QCaDateTime** (QDateTime dt)
- QCaDateTime & **operator=** (const QCaDateTime &other)
- **QCaDateTime** (const unsigned long seconds, const unsigned long nanoseconds, const int userTag=0)
- QString **text** () const
- QString **ISOText** () const
- QCaDateTime addSeconds (const double seconds) const
- double **secondsTo** (const QDateTime &target) const
- Q_DECL_DEPRECATED double floating (const QDateTime &base) const
- unsigned long getSeconds () const

  Recover original EPICS time constructor parameters.

- unsigned long **getNanoSeconds** () const
- int **getUserTag** () const

9.100.1 Member Function Documentation

9.100.1.1 QCaDateTime QCaDateTime::addSeconds (const double *seconds*) const

Equivilent of addSecs and secsTo in base class, save that we specify the seconds as a floating point number, i,e take into account fractions of a second (i.e. mSec but not nano sec).

9.100.1.2 double QCaDateTime::floating (const QDateTime & *base*) const

Duration in seconds from base time to this time. Note: this is the opposite sense to the parent QDateTime daysTo, secsTo and msecsTo functions. Phase out

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDateTime.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaDateTime.cpp

9.101 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventFilter.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventFilter.cpp

9.102 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventUpdate.h

9.103 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventUpdate.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventUpdate.cpp

9.104 QCaInfo Class Reference

Class to pass summary information about a QCaObject.

`#include <QEWidget.h>`

Public Types

- enum **ACCESS_MODES** { **NO_ACCESS**, **READ_ONLY**, **READ_WRITE**, **UNKNOWN** }

Public Member Functions

- **QCaInfo** (QString variableIn, QString typeIn, QString valueIn, QString severityIn, QString statusIn, QString hostIn, int precisionIn, int precisionUserIn, double alarmUserMinIn, double alarmUserMaxIn, double controlLimitLowerIn, double controlLimitUpperIn, double alarmLimitLowerIn, double alarmLimitUpperIn, double warningLimitLowerIn, double warningLimitUpperIn, double driveLimitLowIn, double driveLimitHighIn, bool alarmSensitiveIn, ACCESS_MODES accessModeIn)

Public Attributes

- QString **variable**
- QString **type**
- QString **value**
- QString **severity**
- QString **status**
- QString **host**
- int **precision**
- int **precisionUser**
- double **alarmUserMin**
- double **alarmUserMax**
- double **controlLimitLower**
- double **controlLimitUpper**
- double **alarmLimitLower**
- double **alarmLimitUpper**
- double **warningLimitLower**
- double **warningLimitUpper**
- double **driveLimitLow**
- double **driveLimitHigh**
- bool **alarmSensitive**
- ACCESS_MODES **accessMode**

9.104.1 Detailed Description

Class to pass summary information about a QCaObject.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidget.h

9.105 QCaInstalledFiltersListItem Class Reference

Public Member Functions

- **QCaInstalledFiltersListItem** (QObject ∗eventObjectIn)

Public Attributes

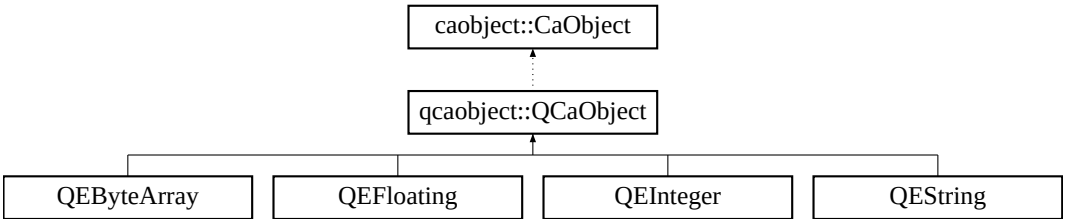
- QObject ∗ **eventObject**
- long **referenceCount**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaEventFilter.h

9.106 qcaobject::QCaObject Class Reference

Inheritance diagram for qcaobject::QCaObject::



Public Types

- enum **SignalsToSend** { **SIG_NONE** = 0x00, **SIG_VARIANT** = 0x01, **SIG_-BYTEARRAY** = 0x02 }
- enum **priorities** { **QE_PRIORITY_LOW** = 0, **QE_PRIORITY_NORMAL** = 10, **QE_PRIORITY_HIGH** = 20 }

Public Slots

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

Signals

- void **dataChanged** (const QVariant &value, QCaAlarmInfo &alarmInfo, QCa-DateTime &timeStamp, const unsigned int &variableIndex)
- void **dataChanged** (const QByteArray &value, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **connectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **connectionChanged** (QCaConnectionInfo &connectionInfo)

Public Member Functions

- **QCaObject** (const QString &recordName, QObject *eventObject, const unsigned int variableIndex, const SignalsToSend signalsToSendIn=SIG_-VARIANT, priorities priorityIn=QE_PRIORITY_NORMAL)
- **QCaObject** (const QString &recordName, QObject *eventObject, const unsigned int variableIndex, UserMessage *userMessageIn, const SignalsToSend signalsToSendIn=SIG_VARIANT, priorities priorityIn=QE_PRIORITY_-NORMAL)

- bool **subscribe** ()
- bool **singleShotRead** ()
- bool **dataTypeKnown** ()
- unsigned int **getVariableIndex** () const
- 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** ()
- void **setRequestedElementCount** (unsigned int elementCount)
- QString **getRecordName** ()
- QString **getEgu** ()
- QStringList **getEnumerations** ()
- unsigned int **getPrecision** ()
- QCaAlarmInfo **getAlarmInfo** ()
- QCaDateTime **getDateTime** ()
- double **getDisplayLimitUpper** ()
- double **getDisplayLimitLower** ()
- double **getAlarmLimitUpper** ()
- double **getAlarmLimitLower** ()
- double **getWarningLimitUpper** ()
- double **getWarningLimitLower** ()
- double **getControlLimitUpper** ()
- double **getControlLimitLower** ()
- generic::generic_types **getDataType** ()
- QString **getHostName** ()
- QString **getFieldType** ()
- unsigned long **getElementCount** ()
- bool **getReadAccess** ()
- bool **getWriteAccess** ()
- void **setArrayIndex** (const int index)
- int **getArrayIndex** () const
- void **getLastData** (bool &isDefined, QVariant &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp)
- bool **getChannelIsConnected** () const
- Q_DECL_DEPRECATED bool **getIsLinkUp** () const
- bool **getDataIsAvailable** () const
- QString **getStringValue** () const
- long **getIntegerValue** () const
- double **getFloatingValue** () const
- QVector< long > **getIntegerArray** () const
- QVector< double > **getFloatingArray** () const

Static Public Member Functions

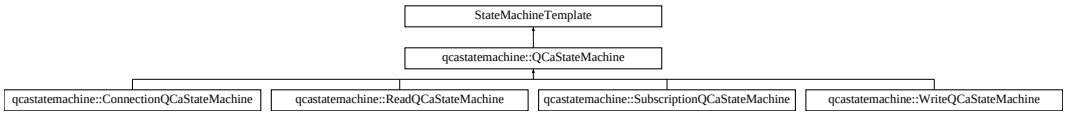
- static void **deletingEventStatic** (QCaEventUpdate *dataUpdateEvent)
- static void **processEventStatic** (QCaEventUpdate *dataUpdateEvent)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaObject.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaObject.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QCaObject.cpp

9.107 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.cpp

9.108 QCaVariableNamePropertyManager Class Reference

Signals

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

Public Member Functions

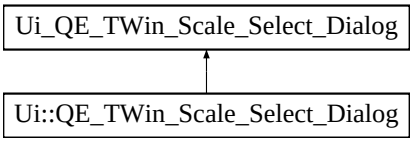
- QString **getVariableNameProperty** () const
- void **setVariableNameProperty** (QString variableNamePropertyIn)
- QString **getSubstitutionsProperty** () const
- void **setSubstitutionsProperty** (QString substitutionsPropertyIn)
- void **setVariableIndex** (unsigned int variableIndexIn)
- unsigned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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaVariableNamePropertyManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaVariableNamePropertyManager.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QCaVariableNamePropertyManager.cpp

9.109 Ui::QE_TWin_Scale_Select_Dialog Class Reference

Inheritance diagram for Ui::QE_TWin_Scale_Select_Dialog::

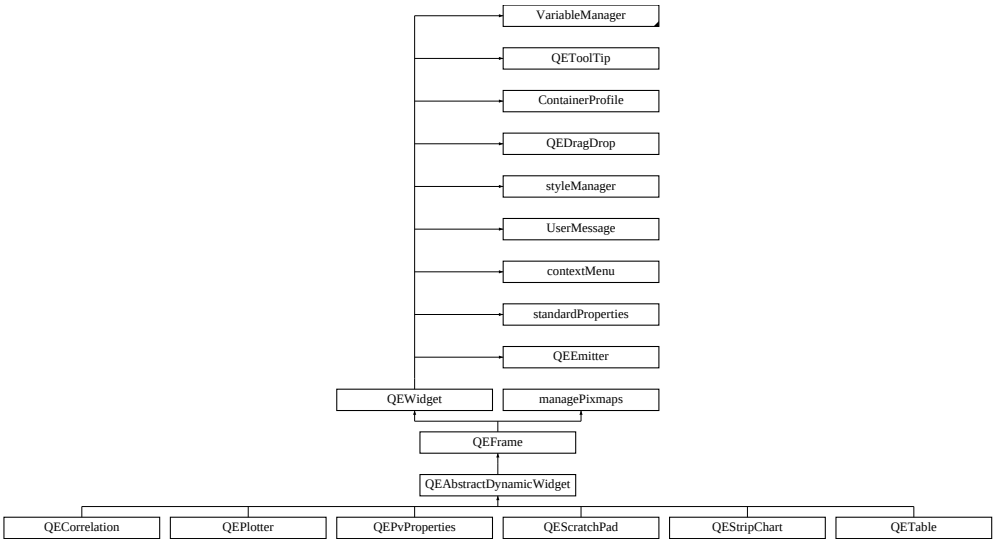


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QETwinScaleSelectDialog.h

9.110 QEAbstractDynamicWidget Class Reference

`#include <QEAbstractDynamicWidget.h>`Inheritance diagram for QEAbstractDynamicWidget::



Public Types

- enum **OwnContextMenuOptions** { **ADWCM_NONE** = CM_SPECIFIC_-WIDGETS_START_HERE, **ADWCM_LOAD_WIDGET_CONFIG**, **ADWCM_SAVE_WIDGET_CONFIG**, **ADWCM_SUB_CLASS_-WIDGETS_START_HERE** }

Public Member Functions

- **QEAbstractDynamicWidget** (QWidget ∗parent=0)
- void **setDefaultDir** (const QString &defaultDir)
- QString **getDefaultDir** () const
- void **setEnableEditPv** (const bool isEnabled)
- bool **getEnableEditPv** () const
- void **paste** (QVariant s)
- void **addPvNameList** (const QStringList &pvNameList)
- void **addPvNameSet** (const QString &pvNameSet)
- virtual int **addPvName** (const QString &pvName)=0
- virtual void **clearAllPvNames** ()=0

Protected Slots

- void **loadNamedWidetConfiguration** (const QString &filename)

- void **saveNamedWidetConfiguration** (const QString &filename)
- void **loadWidgetConfiguration** ()
- void **saveWidgetConfiguration** ()

Protected Member Functions

- QMenu * **buildContextMenu** ()
- void **contextMenuTriggered** (int selectedItemNum)
- QString **getPersistantName** () const
- virtual void **enableEditPvChanged** ()
- userLevelTypes::userLevels **minimumEditPvUserLevel** () const

Properties

- QString defaultDir
- bool enableEditPv

9.110.1 Detailed Description

Provides a common abstract base class for dynamic widgets, i.e. dynamic in the sense that the user can add/removed and modifiy PVs used by the widget at run time. It has been specifically designed to be a common base class for the QEStripChart, QEScratchPad, QEPlotter and QETable widgets. This not only minimises maintainance, but helps ensure we maintain a common look and feel user experiance.

9.110.2 Property Documentation

9.110.2.1 QString QEAbstractDynamicWidget::defaultDir [read, write]

Default directory used for loading/saving files. Default to null string which is interpreted as the current directory.

9.110.2.2 bool QEAbstractDynamicWidget::enableEditPv [read, write]

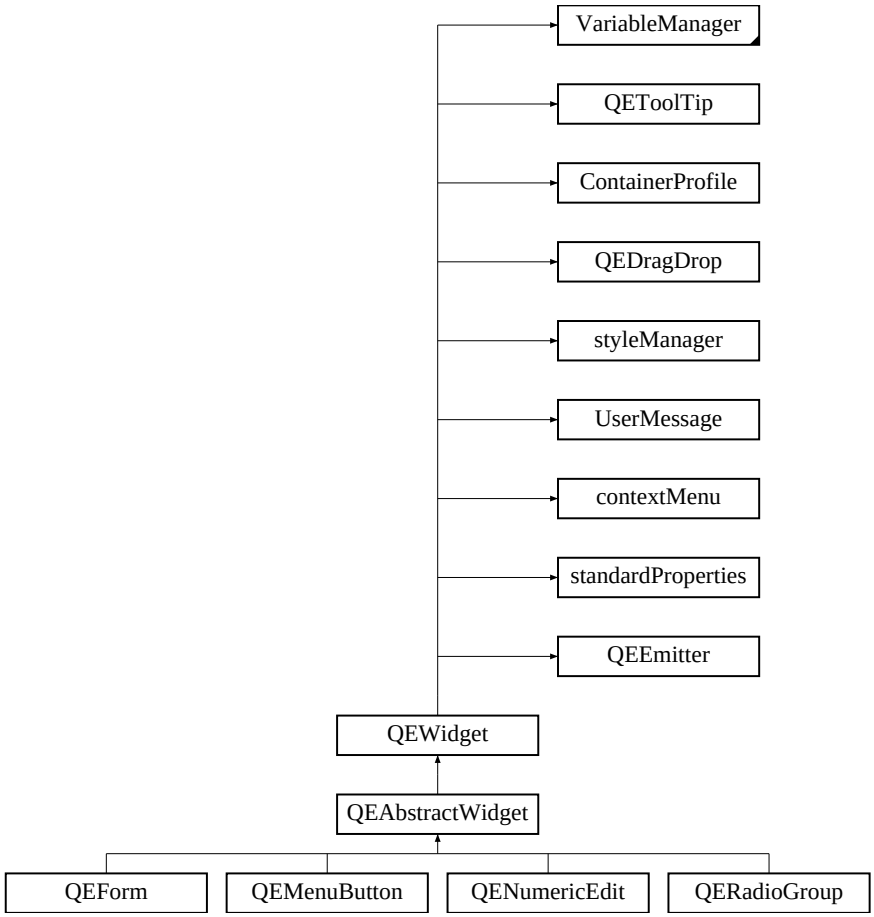
By default, the contexMenu class only adds the "Edit PV" menu entry if and only if we are using the engineer user level (provided it has also been included in the widget menu set). When enableEditPv set true, the user level required is set to user level user, i.e. always available. Note: this may apply to this widget itself, but is more likely to be applied to the contained dynamic widgets. The default value for this property is false.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAbstractWidget/QEAb
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAbstractWidget/QEAb

9.111 QEAbstractWidget Class Reference

Inheritance diagram for QEAbstractWidget::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

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

Properties

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

9.111.1 Member Enumeration Documentation

9.111.1.1 enum QEAbstractWidget::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.111.1.2 enum QEAbstractWidget::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.

9.111.2 Member Function Documentation

9.111.2.1 void QEAbstractWidget::setManagedVisible (bool *v*) [inline, slot]

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

9.111.3 Property Documentation

9.111.3.1 bool QEAbstractWidget::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.

Reimplemented in QEForm, and QEMenuButton.

9.111.3.2 QString QEAbstractWidget::defaultStyle [read, write]

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

9.111.3.3 bool QEAbstractWidget::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.111.3.4 DisplayAlarmStateOptions QEAbstractWidget::displayAlarmStateOption [read, write]

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

Reimplemented in QEForm, and QEMenuButton.

9.111.3.5 unsigned QEAbstractWidget::int [read, write]

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

Reimplemented in QEForm.

9.111.3.6 QString QEAbstractWidget::styleSheet [read, write]

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

9.111.3.7 UserLevels QEAbstractWidget::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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.111.3.8 QString QEAbstractWidget::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.

9.111.3.9 QString QEAbstractWidget::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.

9.111.3.10 QString QEAbstractWidget::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.

9.111.3.11 UserLevels QEAbstractWidget::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'.

9.111.3.12 bool QEAbstractWidget::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.

Reimplemented in QEForm, and QEMenuButton.

9.111.3.13 bool QEAbstractWidget::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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAbstractWidget/QEAb
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAbstractWidget/QEAb

9.112 QEActionRequests Class Reference

Public Types

- enum **Kinds** {

  KindNone, **KindOpenFile**, **KindOpenFiles**, **KindAction**,

  KindWidgetAction, **KindHostComponents** }
- enum **Options** {

  OptionOpen, **OptionNewTab**, **OptionNewWindow**, **OptionTopDockWindow**,

  OptionBottomDockWindow, **OptionLeftDockWindow**, **OptionRightDockWindow**, **OptionTopDockWindowTabbed**,

  OptionBottomDockWindowTabbed, **OptionLeftDockWindowTabbed**, **OptionRightDockWindowTabbed**, **OptionFloatingDockWindow** }

Public Member Functions

- **QEActionRequests** (const QString &action, const QString &pvName)
- **QEActionRequests** (const QString &filename, const QString &config, const Options optionIn, const QEFormMapper::FormHandles &formHandle=QEFormMapper::nullHandle())
- **QEActionRequests** (const QList< windowCreationListItem > windowsIn)
- **QEActionRequests** (const QString &actionIn, const QString &widgetNameIn, const QStringList &argumentsIn, const bool &initialiseIn, QAction *originator)

- **QEActionRequests** (const componentHostListItem &componentIn)
- **QEActionRequests** (const QList< componentHostListItem > &componentsIn)

- void **setKind** (const Kinds kindIn)
- Kinds **getKind** () const
- void **setArguments** (const QStringList &argumentsIn)
- QStringList **getArguments** () const
- void **setOption** (const Options optionIn)
- Options **getOption** () const
- void **setAction** (const QString actionIn)
- QString **getAction** () const
- void **setWidgetName** (const QString widgetNameIn)
- QString **getWidgetName** () const
- void **setInitialise** (const bool initialiseIn)
- bool **getInitialise** () const
- void **setOriginator** (QAction *originatorIn)
- QAction * **getOriginator** () const
- void **setFormHandle** (const QEFormMapper::FormHandles &formHandle)
- QEFormMapper::FormHandles **getFormHandle** () const
- QString **getCustomisation** () const
- QList< windowCreationListItem > **getWindows** () const
- QList< componentHostListItem > **getComponents** () const

Static Public Member Functions

- static QString **actionPvProperties** ()
- static QString **actionStripChart** ()
- static QString **actionScratchPad** ()
- static QString **actionPlotter** ()
- static QString **actionTable** ()
- static QString **actionShowInHisogram** ()
- static QString **actionGeneralPvEdit** ()
- static bool **isDockCreationOption** (const Options createOption)
- static bool **isTabbedDockCreationOption** (const Options createOption)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.cpp

9.113 QEAdaptationParameters Class Reference

`#include <QEAdaptationParameters.h>`

Public Member Functions

- **QEAdaptationParameters** (const QString &envPrefix)
- bool **getBool** (const QString &name)
- QString **getString** (const QString &name, const QString &defaultValue)
- int **getInt** (const QString &name, const int defaultValue)
- double **getFloat** (const QString &name, const double defaultValue)

9.113.1 Detailed Description

This class provides general access to adaptation parameters that may be specified in a variety of ways. Specifcally it provides a standard and consistent wrapper around the getXxxx () functions out of QEOptions, QEEnvironmentVariables and QESettings. (where Xxxx is one of Bool, String, Int and Float).

Each get getXxxx () function attempts to extract a value of each source. Where multiple values are available the highest prioity value is returned. The priorities are (from lowest to highest):

default_value - as passed into the getXxxx function. environment variable - the name is converted to upper case and is prefixed with default or specified prefix if this has been specified e.g. QE_. setting - read from adaptation section if the adaptation_-parameters_file.ini file command line option - command line parameter: --name=value. Note is case sensitive.

If a numeric value is ill-defined, then next lower priority value is used. Numeric validity is defined by QString::toInt () and QString::toDouble () Do note that QString::toInt () does not accept 0x... hexadecimal or 0... octal numbers. Users may always use getString () and parse such values themselves.

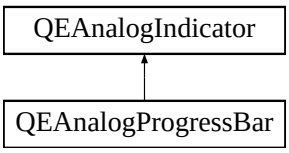
If only command line options, or setting or environment variables to be considered, then use the getXxxx () function out of the QEOptions, QEQEEnvironmentVariables or QESettings.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_-parameters/QEAdaptationParameters.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_-parameters/QEAdaptationParameters.cpp

9.114 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)
- void **setValue** (const int value)

Public Member Functions

- QEAnalogIndicator (QWidget *parent=0)

  Constructor.

- virtual ~QEAnalogIndicator ()

  Destructor.

- virtual QSize sizeHint () const

  Size hint.

- double getValue () const

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

- void setMinimum (const double value)

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

- double getMinimum () const

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

- void setMaximum (const double value)

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

- double getMaximum () const

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

- void setOrientation (const enum Orientations value)

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

- enum Orientations getOrientation () const

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

- void setMode (const enum Modes value)

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

- enum Modes getMode () const

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

- void setCentreAngle (const int value)

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

- int getCentreAngle () const

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

- void setSpanAngle (const int value)

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

- int getSpanAngle () const

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

- void setMinorInterval (const double value)

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

- double getMinorInterval () const

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

- void setMajorInterval (const double value)

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

- double getMajorInterval () const

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

- void setLogScaleInterval (const int value)

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

- int getLogScaleInterval () const

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

- void setBorderColour (const QColor value)

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

- QColor getBorderColour () const

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

- void setForegroundColour (const QColor value)

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

- QColor getForegroundColour () const

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

- void setBackgroundColour (const QColor value)

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

- QColor getBackgroundColour () const

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

- void setFontColour (const QColor value)

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

- QColor getFontColour () const

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

- void setShowText (const bool value)

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

- bool getShowText () const

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

- void setShowScale (const bool value)

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

- bool getShowScale () const

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

- void setLogScale (const bool value)

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

- bool getLogScale () const

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

Protected Member Functions

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

Properties

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

  Alternative to isEnabled. Default is true.

9.114.1 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.

9.114.2 Member Enumeration Documentation

9.114.2.1 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).

9.114.2.2 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.

9.114.3 Property Documentation

9.114.3.1 QColor QEAnalogIndicator::backgroundColour `[read, write]`

Background colour

9.114.3.2 QColor QEAnalogIndicator::borderColour `[read, write]`

Border colour

9.114.3.3 int QEAnalogIndicator::centreAngle `[read, write]`

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

9.114.3.4 QColor QEAnalogIndicator::fontColour `[read, write]`

Font colour

9.114.3.5 QColor QEAnalogIndicator::foregroundColour `[read, write]`

Foreground colour

9.114.3.6 bool QEAnalogIndicator::logScale [read, write]

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

9.114.3.7 int QEAnalogIndicator::logScaleInterval [read, write]

Log scale interval.

9.114.3.8 double QEAnalogIndicator::majorInterval [read, write]

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

9.114.3.9 double QEAnalogIndicator::maximum [read, write]

Maximum indicated value.

9.114.3.10 double QEAnalogIndicator::minimum [read, write]

Minimum indicated value.

9.114.3.11 double QEAnalogIndicator::minorInterval [read, write]

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

9.114.3.12 Modes QEAnalogIndicator::mode [read, write]

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

9.114.3.13 Orientations QEAnalogIndicator::orientation [read, write]

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

9.114.3.14 bool QEAnalogIndicator::showScale [read, write]

If set, show the scale

9.114.3.15 bool QEAnalogIndicator::showText [read, write]

If set, show textual representation of value on the indicator

9.114.3.16 int QEAnalogIndicator::spanAngle [read, write]

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

9.114.3.17 double QEAnalogIndicator::value [read, write]

Current indicated value.

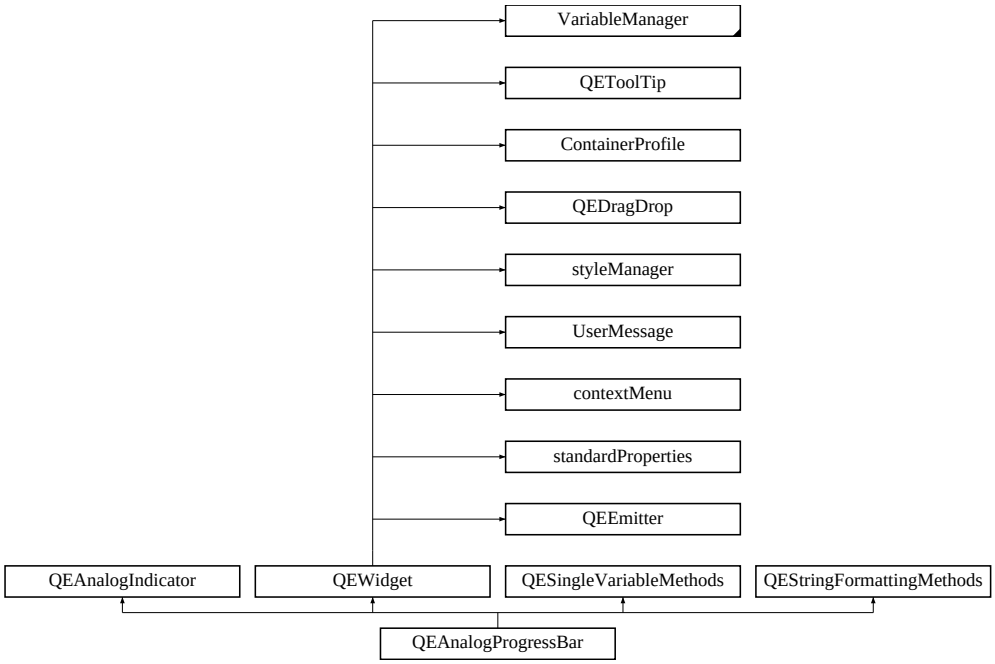
Reimplemented in QEAnalogProgressBar.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogIndicator/QEA
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogIndicator/QEA

9.115 QEAnalogProgressBar Class Reference

Inheritance diagram for QEAnalogProgressBar::



Public Types

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

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

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Underscore = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }

Public Slots

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)
- void requestResend ()

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

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

- void setFormatProperty (Formats format)

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

- Formats getFormatProperty ()

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

- void setSeparatorProperty (const Separators notation)

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

- Separators getSeparatorProperty () const

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

- void setNotationProperty (Notations notation)

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

- Notations getNotationProperty ()

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

- void setArrayActionProperty (ArrayActions arrayAction)

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

- ArrayActions getArrayActionProperty ()

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

- QEAnalogProgressBar (QWidget *parent=0)
- QEAnalogProgressBar (const QString &variableName, QWidget *parent=0)
- virtual ~QEAnalogProgressBar ()

  Destruction.

- void setUseDbDisplayLimits (bool useDbDisplayLimitsIn)

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

- bool getUseDbDisplayLimits ()

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

- void 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

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **stringFormattingChange** ()
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()
- QVariant **copyData** ()
- QString **getTextImage** ()
- BandList **getBandList** ()

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- AlarmSeverityDisplayModes alarmSeverityDisplayMode
- bool useDbDisplayLimits
- int value

- bool isActive

  Alternative to isEnabled. Default is true.

- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction

9.115.1 Member Enumeration Documentation

9.115.1.1 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.

9.115.1.2 enum QEAnalogProgressBar::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.115.1.3 enum QEAnalogProgressBar::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.115.1.4 enum QEAnalogProgressBar::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.115.1.5 enum QEAnalogProgressBar::Separators

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

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.115.1.6 enum QEAnalogProgressBar::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.115.2 Constructor & Destructor Documentation

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

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

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

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

9.115.3 Member Function Documentation

9.115.3.1 void QEAnalogProgressBar::dbConnectionChanged (const bool & *isConnected*) [signal]

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

9.115.3.2 void QEAnalogProgressBar::dbValueChanged () [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.

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

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

9.115.4 Property Documentation

9.115.4.1 bool QEAnalogProgressBar::addUnits [read, write]

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

9.115.4.2 AlarmSeverityDisplayModes QEAnalogProgressBar::alarmSeverityDisplayMode [read, write]

Visualise the EPICS alarm severity

9.115.4.3 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.

9.115.4.4 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'.

9.115.4.5 int QEAnalogProgressBar::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.115.4.6 QString QEAnalogProgressBar::defaultStyle [read, write]

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

9.115.4.7 bool QEAnalogProgressBar::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.115.4.8 DisplayAlarmStateOptions QEAnalogProgressBar::displayAlarmStateOption [read, write]

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

9.115.4.9 int QEAnalogProgressBar::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.115.4.10 bool QEAnalogProgressBar::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.115.4.11 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).

9.115.4.12 unsigned QEAnalogProgressBar::int [read, write]

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

9.115.4.13 bool QEAnalogProgressBar::leadingZero [read, write]

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

9.115.4.14 QString QEAnalogProgressBar::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.115.4.15 Notations QEAnalogProgressBar::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.115.4.16 int QEAnalogProgressBar::precision [read, write]

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

9.115.4.17 int QEAnalogProgressBar::radix [read, write]

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

9.115.4.18 Separators QEAnalogProgressBar::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.115.4.19 QString QEAnalogProgressBar::styleSheet [read, write]

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

9.115.4.20 bool QEAnalogProgressBar::trailingZeros [read, write]

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

9.115.4.21 bool QEAnalogProgressBar::useDbDisplayLimits [read, write]

Use the EPICS database display limits

9.115.4.22 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.

9.115.4.23 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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.115.4.24 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.

9.115.4.25 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.

9.115.4.26 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.

9.115.4.27 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'.

9.115.4.28 int QEAnalogProgressBar::value [read, write]

Current indicated value.

Reimplemented from QEAnalogIndicator.

9.115.4.29 QString QEAnalogProgressBar::variable [read, write]

EPICS variable name (CA PV)

9.115.4.30 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.

9.115.4.31 QString QEAnalogProgressBar::variableSubstitutions [read, write]

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

9.115.4.32 bool QEAnalogProgressBar::visible [read, write]

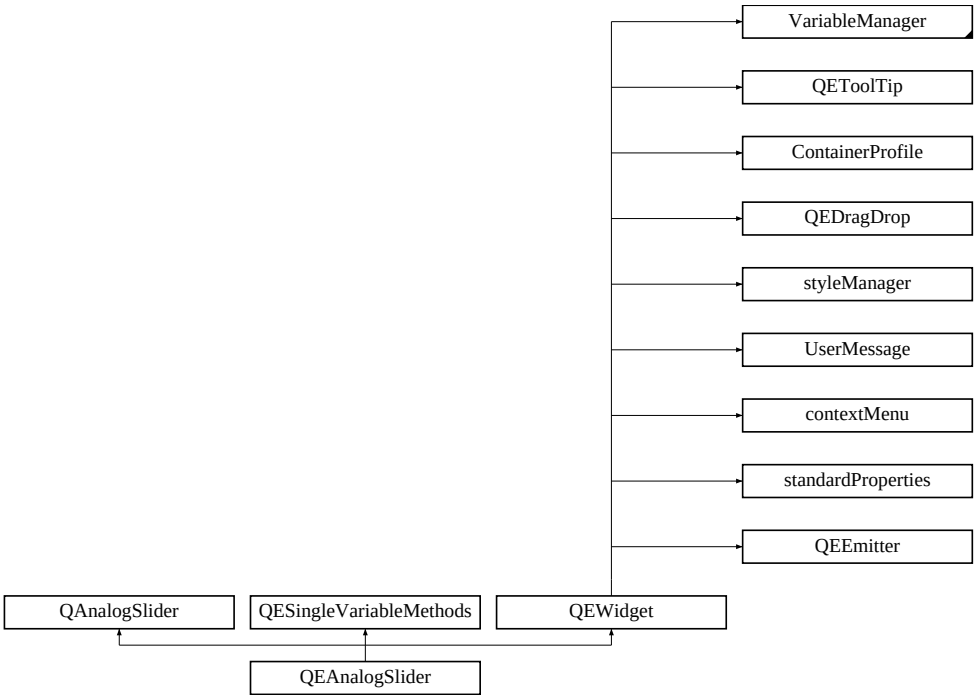
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogProgressBar/QEAnalogProg
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEAnalogProgressBar.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogProgressBar/QEAnalogProg

9.116 QEAnalogSlider Class Reference

Inheritance diagram for QEAnalogSlider::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)

- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

- QEAnalogSlider (QWidget *parent=0)
- QEAnalogSlider (const QString &variableName, const QString &readbackName="", QWidget *parent=0)
- virtual ~QEAnalogSlider ()

  Destruction.

- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- void writeNow ()
- void setVariableNameSubstitutionsProperty (const QString &substitutions)
- void **setReadbackNameProperty** (const QString &variableName)
- QString **getReadbackNameProperty** () const
- void **setReadbackElementsRequired** (const int elementsRequired)
- int **getReadbackElementsRequired** () const

- void **setReadbackArrayIndex** (const int arrayIndex)
- int **getReadbackArrayIndex** () const
- void **setSubstitutionsProperty** (const QString &substitutions)
- QString **getSubstitutionsProperty** () const
- void **setContinuousWrite** (const bool value)
- bool **getContinuousWrite** () const
- void **setAutoScale** (const bool value)
- bool **getAutoScale** () const
- void **setAlarmColours** (const bool value)
- bool **getAlarmColours** () const

Protected Member Functions

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- int **getPrecision** () const
- double **getMinimum** () const
- double **getMaximum** () const

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- QString **readbackVariable**
- int **readbackElementsRequired**
- int **readbackArrayIndex**
- bool continuousWrite
- bool autoScale
- bool axisAlarmColours
- double value

  Speficies the value.

- QString leftText

Controls the left, centre and right texts.

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

9.116.1 Member Enumeration Documentation

9.116.1.1 enum QEAnalogSlider::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.116.1.2 enum QEAnalogSlider::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.

9.116.2 Constructor & Destructor Documentation

9.116.2.1 QEAnalogSlider::QEAnalogSlider (QWidget * *parent* = 0) [explicit]

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

9.116.2.2 QEAnalogSlider::QEAnalogSlider (const QString & *variableName*, const QString & *readbackName* = "", QWidget * *parent* = 0) [explicit]

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

9.116.3 Member Function Documentation

9.116.3.1 void QEAnalogSlider::dbConnectionChanged (const bool & *isConnected*) [signal]

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

9.116.3.2 void QEAnalogSlider::dbValueChanged () [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. Note: this widget emits the numeric enumeration value as opposed to the associated text.

9.116.3.3 void QEAnalogSlider::setManagedVisible (bool *v*) [inline, slot]

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

9.116.3.4 void QEAnalogSlider::setVariableNameSubstitutionsProperty (const QString & *substitutions*)

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

Reimplemented from QESingleVariableMethods.

9.116.3.5 void QEAnalogSlider::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.116.4 Property Documentation

9.116.4.1 bool QEAnalogSlider::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.

9.116.4.2 int QEAnalogSlider::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.116.4.3 bool QEAnalogSlider::autoScale [read, write]

If true (default), display and editing of numbers using the precision, and control limits supplied with the data. If false, the precision, leadingZeros, minimum and maximum properties are used.

9.116.4.4 bool QEAnalogSlider::axisAlarmColours [read, write]

If true the backgroud axis colour refect the alarm and warning values of the (setpoint) variable If false (default) then axis uses widget colour.

9.116.4.5 bool QEAnalogSlider::continuousWrite [read, write]

If true the widget writes to the PV as the slider is moved. If false (default) a write only occurs when apply button click.

9.116.4.6 QString QEAnalogSlider::defaultStyle [read, write]

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

9.116.4.7 bool QEAnalogSlider::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.116.4.8 DisplayAlarmStateOptions QEAnalogSlider::displayAlarmStateOption [read, write]

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

9.116.4.9 int QEAnalogSlider::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.116.4.10 unsigned QEAnalogSlider::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.

9.116.4.11 QString QEAnalogSlider::styleSheet [read, write]

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

9.116.4.12 UserLevels QEAnalogSlider::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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.116.4.13 QString QEAnalogSlider::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.

9.116.4.14 QString QEAnalogSlider::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.

9.116.4.15 QString QEAnalogSlider::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.

9.116.4.16 UserLevels QEAnalogSlider::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'.

9.116.4.17 QString QEAnalogSlider::variable [read, write]

EPICS variable name (CA PV)

9.116.4.18 bool QEAnalogSlider::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.

9.116.4.19 QString QEAnalogSlider::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.

9.116.4.20 bool QEAnalogSlider::visible [read, write]

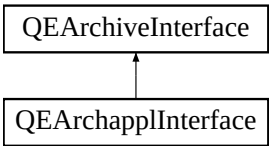
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogSlider/QEAnal
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEAnalogSlider.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEAnalogSlider/QEAnal

9.117 QEArchapplInterface Class Reference

Inheritance diagram for QEArchapplInterface::



Public Slots

- void **networkManagerResponse** (const QEArchiveInterface::Context &context, QNetworkReply ∗reply)
- void **networkManagerFault** (const QEArchiveInterface::Context &context, const QNetworkReply::NetworkError error)

Public Member Functions

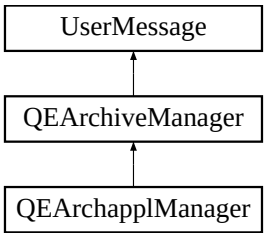
- **QEArchapplInterface** (QUrl url, QObject ∗parent=0)
- virtual void **namesRequest** (QObject ∗userData, const int key=0, QString pattern=".∗")
- virtual void **valuesRequest** (QObject ∗userData, const QCaDateTime startTime, const QCaDateTime endTime, const int count, const How how, const QStringList pvNames, const int key=0, const unsigned int requested_element=0)
- virtual void **infoRequest** (QObject ∗userData)
- virtual void **archivesRequest** (QObject ∗userData)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.cpp

9.118 QEArchapplManager Class Reference

Inheritance diagram for QEArchapplManager::



Friends

- class QEArchiveAccess

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp

9.119 QEArchapplNetworkManager Class Reference

Classes

- struct **ArchapplValuesRequest**

Signals

- void **networkManagerResponse** (const QEArchiveInterface::Context &context, QNetworkReply *reply)
- void **networkManagerFault** (const QEArchiveInterface::Context &context, const QNetworkReply::NetworkError error)

Friends

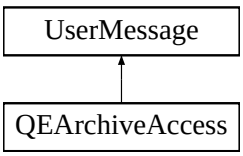
- class QEArchapplInterface

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEArchiveInterface.cpp

9.120 QEArchiveAccess Class Reference

Inheritance diagram for QEArchiveAccess::



Classes

- struct ArchiverPvInfo
- struct PVDataRequests
- struct PVDataResponses
- struct Status

Public Types

- enum **ArchiverTypes** { **CA**, **ARCHAPPL**, **Error** }
- typedef QList< Status > **StatusList**
- typedef QList< ArchiverPvInfo > **ArchiverPvInfoLists**

Signals

- void **archiveStatus** (const QEArchiveAccess::StatusList &statusList)
- void **setArchiveData** (const QObject *userData, const bool isOkay, const QCaDataPointList &pointsList, const QString &pvName, const QString &supplementary)
- void **setArchiveData** (const QObject *, const bool, const QCaDataPointList &)
- void **reInterogateArchives** ()
- void **archiveStatusRequest** ()
- void **readArchiveRequest** (const QEArchiveAccess *, const QEArchiveAccess::PVDataRequests &)

Public Member Functions

- **QEArchiveAccess** (QObject *parent=0)
- unsigned int **getMessageSourceId** ()
- void **setMessageSourceId** (unsigned int messageSourceId)
- void **resendStatus** ()
- void **reReadAvailablePVs** ()
- void **readArchive** (QObject *userData, const QString pvName, const QCaDateTime startTime, const QCaDateTime endTime, const int count, const QEArchiveInterface::How how, const unsigned int element=0)

Static Public Member Functions

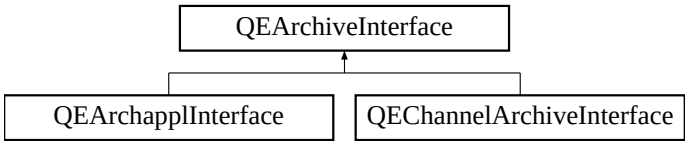
- static Q_DECL_DEPRECATED void **initialise** (const QString &archives, const QString &pattern)
- static Q_DECL_DEPRECATED void **initialise** ()
- static bool **isReady** ()
- static int **getNumberInterfaces** ()
- static QString **getPattern** ()
- static int **getNumberPVs** ()
- static QStringList **getAllPvNames** ()
- static bool **getArchivePvInformation** (const QString &pvName, QString &effectivePvName, ArchiverPvInfoLists &data)
- static ArchiverTypes **getArchiverType** ()
- static bool **registerMetaTypes** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEArchiveAccess.cpp

9.121 QEArchiveInterface Class Reference

Inheritance diagram for QEArchiveInterface::



Classes

- struct Archive
- struct Context
- struct PVName
- struct ResponseValues

Public Types

- enum **archiveAlarmSeverity** {

  archSevNone = 0, **archSevMinor** = 1, **archSevMajor** = 2, **archSevInvalid** = 3,

  archSevEstRepeat = 0x0f80, **archSevDisconnect** = 0x0f40, **archSevStopped** = 0x0f20, **archSevRepeat** = 0x0f10,

  archSevDisabled = 0x0f08 }
- enum **How** {

  Raw = 0, **SpreadSheet**, **Averaged**, **PlotBinning**,

  Linear }
- enum **Methods** {

  Information, **Archives**, **Names**, **Values**,

  Count }
- enum **States** {

  Unknown, **Updating**, **Complete**, **InComplete**,

  No_Response, **Error** }
- typedef QList< QEArchiveInterface::Archive > **ArchiveList**
- typedef QList< QEArchiveInterface::PVName > **PVNameList**
- typedef std::list< QEArchiveInterface::ResponseValues > **ResponseValueList**

Signals

- void **pvNamesResponse** (const QObject *, const bool, const QEArchiveInterface::PVNameList &)

- void **valuesResponse** (const QObject *, const bool, const QEArchiveInterface::ResponseValueList &)
- void **infoResponse** (const QObject *, const bool, const int, const QString &)
- void **archivesResponse** (const QObject *, const bool, const QEArchiveInterface::ArchiveList &)
- void **nextRequest** (const int requestIndex)

Public Member Functions

- **QEArchiveInterface** (QObject *parent=0)
- virtual void **setUrl** (QUrl url)
- QUrl **getUrl** () const
- QString **getName** () const
- int **getNumberPending** () const
- virtual void **infoRequest** (QObject *userData)=0
- virtual void **archivesRequest** (QObject *userData)=0
- virtual void **namesRequest** (QObject *userData, const int key=0, QString pattern=".*")=0
- virtual void **valuesRequest** (QObject *userData, const QCaDateTime startTime, const QCaDateTime endTime, const int count, const How how, const QStringList pvNames, const int key=0, const unsigned int requested_element=0)=0

Static Public Member Functions

- static QString **alarmSeverityName** (enum archiveAlarmSeverity severity)
- static bool **registerMetaTypes** ()

Public Attributes

- States **state**
- int **available**
- int **read**
- int **numberPVs**
- ArchiveList **archiveList**
- int **requestIndex**
- QTimer * **timer**

Static Protected Member Functions

- static QCaDateTime **convertArchiveToEpics** (const int seconds, const int nanoSecs)
- static void **convertEpicsToArchive** (const QCaDateTime &datetime, int &seconds, int &nanoSecs)

Protected Attributes

- QUrl **mUrl**
- int **pending**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEArchiveInterface.cpp

9.122 QEArchiveInterfaceAgent Class Reference

Signals

- void **xmlRpcResponse** (const QEArchiveInterface::Context &, const QVariant &)
- void **xmlRpcFault** (const QEArchiveInterface::Context &, int, const QString &)

Friends

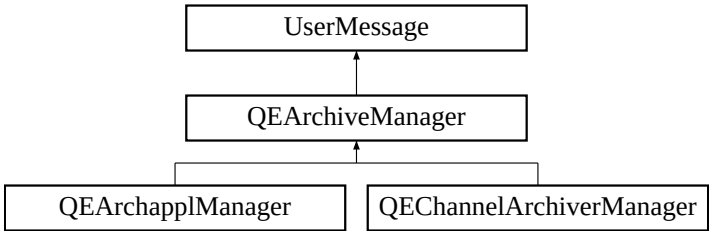
- class QEChannelArchiveInterface

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEArchiveInterface.cpp

9.123 QEArchiveManager Class Reference

Inheritance diagram for QEArchiveManager::



Signals

- void **archiveStatusResponse** (const QEArchiveAccess::StatusList &)
- void **readArchiveResponse** (const QEArchiveAccess *, const QEArchiveAccess::PVDataResponses &)

Protected Slots

- virtual void **readArchiveRequest** (const QEArchiveAccess *archiveAccess, const QEArchiveAccess::PVDataRequests &request)=0
- virtual void **pvNamesResponse** (const QObject *userData, const bool isSuccess, const QEArchiveInterface::PVNameList &pvNameList)=0

Protected Member Functions

- **QEArchiveManager** (QEArchiveManager const &)
- void **operator=** (QEArchiveManager const &)
- void **clear** ()
- void **resendStatus** ()

Protected Attributes

- QString **archives**
- QString **pattern**
- QDateTime **lastReadTime**
- QTimer * **timer**
- ArchiveInterfaceLists **archiveInterfaceList**
- PVNameToSourceSpecLookUp **pvNameToSourceLookUp**
- bool **allArchivesRead**
- int **numberArchivesRead**
- bool **environmentErrorReported**
- QEArchiveAccess::ArchiverTypes **archiverType**

Friends

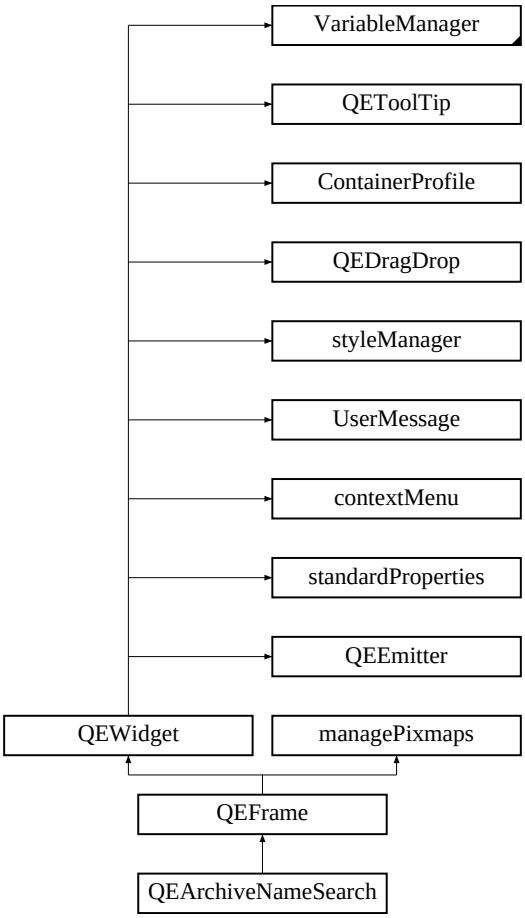
- class QEArchiveAccess

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEArchiveManager.cpp

9.124 QEArchiveNameSearch Class Reference

`#include <QEArchiveNameSearch.h>`Inheritance diagram for QEArchiveNameSearch::



Public Member Functions

- **QEArchiveNameSearch** (QWidget *parent=0)
- QStringList **getSelectedNames** () const
- void **clear** ()

Protected Member Functions

- QSize **sizeHint** () const
- void **mousePressEvent** (QMouseEvent *event)
- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()

9.124.1 Detailed Description

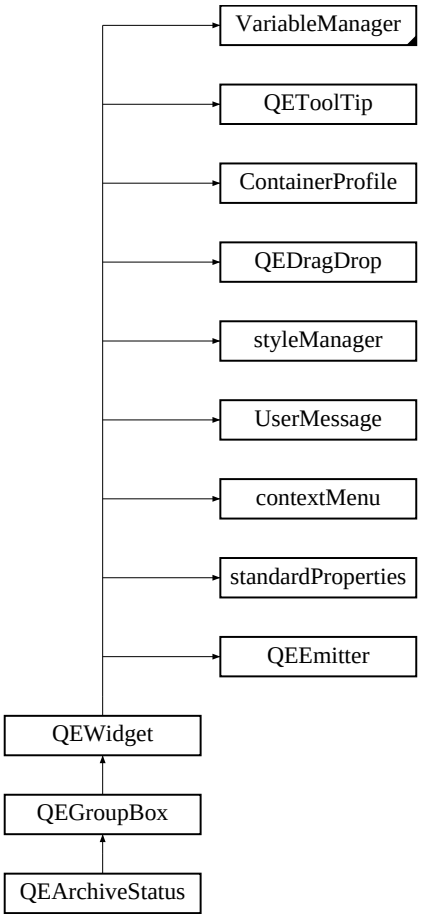
This is a non EPICS aware widget. It provides a simple user means to find archived PV names.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveNameSearch.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveNameSearch.cpp

9.125 QEArchiveStatus Class Reference

`#include <QEArchiveStatus.h>`Inheritance diagram for QEArchiveStatus::



Classes

- struct **Rows**

Public Slots

- void **reReadAvailablePVs** ()

Public Member Functions

- **QEArchiveStatus** (QWidget ∗parent=0)
- QSize **sizeHint** () const

9.125.1 Detailed Description

This is a non EPICS aware widget. It extracts and displays states information from the archive manager.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveStatus.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveStatus.cpp

9.126 QEArchiveThread Class Reference

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp

9.127 QEAxisIterator Class Reference

`#include <QEAxisIterator.h>`

Public Member Functions

- **QEAxisIterator** (const double minimum, const double maximum, const double minorInterval, const int majorMinorRatio, const bool isLogarithmic)
- void reInitialise (const double minimum, const double maximum, const double minorInterval, const int majorMinorRatio, const bool isLogarithmic)
- bool firstValue (double &value, bool &isMajor, const int maxIterations=10000)
- bool nextValue (double &value, bool &isMajor)

  Returns next value if available indicated by return value being true.

- double **getMinimum** () const
- double getMaximum () const

  Extracts iterator's minimum value.

- double getMinorInterval () const

  Extracts iterator's maximum value.

- int getMajorMinorRatio () const

  Extracts iterator's minor interval.

- bool getIsLogarithmic () const

  Extracts iterator's major/minor ratio.

9.127.1 Detailed Description

This class provides the means to iterate over a range of values such that the returned values x, are minimum <= x <= maximum together an indication of whether the interval is a minor ot major interval. It primarily intented to support the painting of axies, but could be use for other purposes.

For a linear iteration, the set of values are always an integer multiple of the given minorInterval. The major interval indication occures when the value an interger multiple of the major interval (minorInterval $*$ majorMinorRatio).

The class constructor will contstrain the minorInterval to be >= 1.0e-20 and the majorMinorRatio to be >= 1 if needs be. This is illustrates below - the minor values are indicated by a '+' character and the major intervals by an '|' character. The minimum value is -0.14, the maximum value is 1.0, the minorInterval is 0.1, and the majorMinorRatio is 4.

min max v v --+-----|-----+-----+-----+-----|-----+-----+-----+-----|-----+-----+ -0.1 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

For a logarithmic iteration, the set of values are of the form:

1.0eN, 2.0eN, 3.0eN, 4.0eN, 5.0eN, 6.0eN, 7.0eN, 8.0eN, 9.0eN, 1.0e(N+1)

Major values are of the form 1.0eM where M is a multiple of majorMinorRatio. The minorInterval parameter is not applicable when the iteration is logarithmic.

9.127.2 Member Function Documentation

9.127.2.1 bool QEAxisIterator::firstValue (double & *value*, bool & *isMajor*, const int *maxIterations* = 10000)

Resets the iterator and returns first iteration value if available (return value is true). The maxIterations parameter is a belts and braces safety check to avoid infinite looping.

9.127.2.2 void QEAxisIterator::reInitialise (const double *minimum*, const double *maximum*, const double *minorInterval*, const int *majorMinorRatio*, const bool *isLogarithmic*)

Re-initialise iterator parameters. If the iterator parameters are updated, the next call to nextValue () will always return false.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEAxisIterator.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEAxisIterator.cpp

9.128 QEAxisPainter Class Reference

Public Types

- enum Orientations { Left_To_Right, Top_To_Bottom, Right_To_Left, Bottom_-To_Top }
- enum TextPositions { BelowLeft, AboveRight }

Public Member Functions

- **QEAxisPainter** (QWidget *parent=0)
- void **setMinimum** (const double minimum)
- double **getMinimum** () const
- void **setMaximum** (const double maximum)
- double **getMaximum** () const
- void **setModulo** (const double modulo)
- double **getModulo** () const
- void **setPrecision** (const int precision)
- int **getPrecision** () const
- void **setMinorInterval** (const double minorInterval)
- double **getMinorInterval** () const
- void **setMajorMinorRatio** (const int majorMinorRatio)
- int **getMajorMinorRatio** () const
- void **setHasAxisLine** (const bool hasAxisLine)
- bool **getHasAxisLine** () const
- void **setLogScale** (const bool isLog)
- bool **getLogScale** () const
- void **setOrientation** (const Orientations orientation)
- Orientations **getOrientation** () const
- void **setTextPosition** (const TextPositions textPositions)
- TextPositions **getTextPosition** () const
- void **setPenColour** (const QColor colour)
- QColor **getPenColour** () const
- void **setIndent** (const int topLeftIndent, const int rightBottomIndent)
- int **getTopLeftIndent** () const
- int **getRightBottomIndent** () const
- void **setIndent** (const int indent)
- int **getIndent** () const
- void **setGap** (const int gap)
- int **getGap** () const
- void **setAutoFixedSize** (const bool enabled)
- bool **getAutoFixedSize** () const
- void **setMarkerColour** (const int index, const QColor &colour)
- QColor **getMarkerColour** (const int index) const
- void **setMarkerVisible** (const int index, const bool isVisible)
- bool **getMarkerVisible** (const int index) const

- void **setMarkerValue** (const int index, const double value)
- double **getMarkerValue** (const int index) const
- void **setColourBandList** (const QEColourBandList &bandList)
- QEColourBandList **getColourBandList** () const

Protected Member Functions

- void **paintEvent** (QPaintEvent ∗event)

9.128.1 Member Enumeration Documentation

9.128.1.1 enum QEAxisPainter::Orientations

The orientation of the axis

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.

9.128.1.2 enum QEAxisPainter::TextPositions

Enumerator:

BelowLeft Below when Orientation is horizontal, Left when Orientation is vertical.

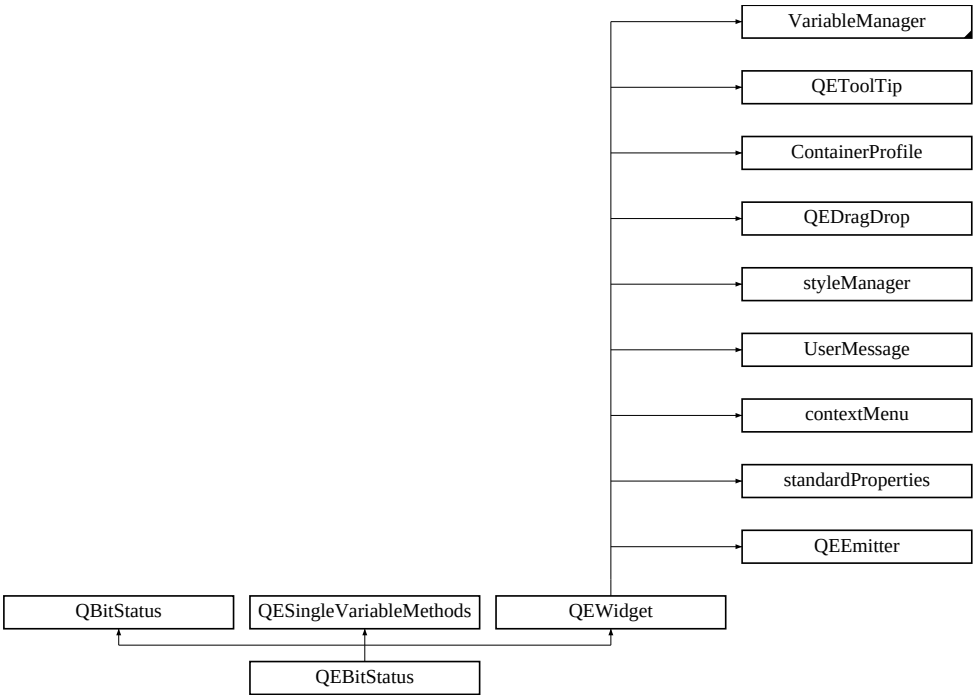
AboveRight Above when Orientation is horizontal, Right when Orientation is vertical.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEAxisPainter.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEAxisPainter.cpp

9.129 QEBitStatus Class Reference

Inheritance diagram for QEBitStatus::



Public Types

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

Public Slots

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)

- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

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

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- double **value**
- bool **isActive**
- bool **isValid**

9.129.1 Member Enumeration Documentation

9.129.1.1 enum QEBitStatus::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.129.1.2 enum QEBitStatus::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.129.2 Member Function Documentation

9.129.2.1 void QEBitStatus::dbConnectionChanged (const bool & *isConnected*) [signal]

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

9.129.2.2 void QEBitStatus::dbValueChanged () [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.

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

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

9.129.3 Property Documentation

9.129.3.1 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.

9.129.3.2 int QEBitStatus::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.129.3.3 QString QEBitStatus::defaultStyle [read, write]

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

9.129.3.4 bool QEBitStatus::displayAlarmState [read, write]

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

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

Reimplemented from standardProperties.

9.129.3.5 DisplayAlarmStateOptions QEBitStatus::displayAlarmStateOption [read, write]

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

9.129.3.6 int QEBitStatus::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.129.3.7 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.

9.129.3.8 QString QEBitStatus::styleSheet [read, write]

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

9.129.3.9 UserLevels QEBitStatus::userLevelEnabled [read, write]

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

9.129.3.10 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.

9.129.3.11 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.

9.129.3.12 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.

9.129.3.13 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'.

9.129.3.14 QString QEBitStatus::variable [read, write]

EPICS variable name (CA PV)

9.129.3.15 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.

9.129.3.16 QString QEBitStatus::variableSubstitutions [read, write]

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

9.129.3.17 bool QEBitStatus::visible [read, write]

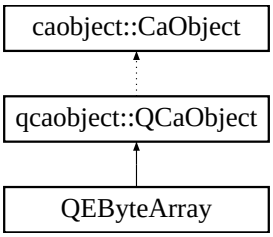
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEBitStatus/QEBitStatus.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEBitStatus.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEBitStatus/QEBitStatus.cpp

9.130 QEByteArray Class Reference

Inheritance diagram for QEByteArray::



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

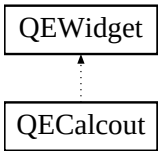
- **QEByteArray** (QString recordName, QObject *eventObject, unsigned int variableIndexIn)
- **QEByteArray** (QString recordName, QObject *eventObject, unsigned int variableIndexIn, UserMessage *userMessageIn)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEByteArray.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEByteArray.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEByteArray.cpp

9.131 QECalcout Class Reference

`#include <QECalcout.h>`Inheritance diagram for QECalcout::



Public Types

- enum **DataOptions** { **Use_CALC**, **Use_OCAL** }
- enum **OutputOptions** {
  Every_Time, **On_Change**, **When_Zero**, **When_Non_zero**,
  Transition_To_Zero, **Transition_To_Non_zero**, **Never** }

Public Slots

- void **inpa** (const double a)
- void **inpb** (const double b)
- void **inpc** (const double c)
- void **inpd** (const double d)
- void **inpe** (const double e)
- void **inpf** (const double f)
- void **inpg** (const double g)
- void **inph** (const double h)
- void **inpi** (const double i)
- void **inpj** (const double j)
- void **inpk** (const double k)
- void **inpl** (const double l)
- void **inpa** (const int a)
- void **inpb** (const int b)
- void **inpc** (const int c)
- void **inpd** (const int d)
- void **inpe** (const int e)
- void **inpf** (const int f)
- void **inpg** (const int g)
- void **inph** (const int h)
- void **inpi** (const int i)
- void **inpj** (const int j)
- void **inpk** (const int k)
- void **inpl** (const int l)

Signals

- void **out** (const bool &out)
- void **out** (const int &out)
- void **out** (const long &out)
- void **out** (const qlonglong &out)
- void **out** (const double &out)
- void **out** (const QString &out)

Public Member Functions

- **QECalcout** (QWidget ∗parent=0)
- void **setCalc** (const QString &calc)
- QString **getCalc** () const
- void **setCalcStatus** (const QString &)
- QString **getCalcStatus** () const
- void **setOcal** (const QString &ocal)
- QString **getOcal** () const
- void **setOcalStatus** (const QString &)
- QString **getOcalStatus** () const
- void **setDopt** (const DataOptions dopt)
- DataOptions **getDopt** () const
- void **setOopt** (const OutputOptions oopt)
- OutputOptions **getOopt** () const
- void **setEgu** (const QString &egu)
- QString **getEgu** () const
- void **setPrec** (const int prec)
- int **getPrec** () const
- void **setFormat** (const QChar format)
- QChar **getFormat** () const
- double **geta** () const
- double **getb** () const
- double **getc** () const
- double **getd** () const
- double **gete** () const
- double **getf** () const
- double **getg** () const
- double **geth** () const
- double **geti** () const
- double **getj** () const
- double **getk** () const
- double **getl** () const
- void **process** ()

Properties

- QString calc
- QString calcStatus
- QString ocal
- QString ocalStatus
- DataOptions **dopt**
- OutputOptions **oopt**
- QString egu
- int **prec**
- QChar **format**
- double a
- double **b**
- double **c**
- double **d**
- double **e**
- double **f**
- double **g**
- double **h**
- double **i**
- double **j**
- double **k**
- double **l**
- bool runVisible

9.131.1 Detailed Description

This widget provides a calcout-like widget. So much so that where applicable the property names have been choosen to match the calcout record. This widget can be used instead of and/or to complement QELink.

This widget has no PV variables of its own. The inputs, A to L, must be provided either by signals from other (QE) widgets or preset as 'constants' using the a to l properties.

Under the covers, this widget uses the same calculation engine as the calc/calcout record.

Traditionally, the type of GUI functionality the QECalcout widget supports has been effected by using EPICS database variables (often CALC records) to determine the state of GUI items. Where the variable is primarily a part of the control system this is appropriate. Where the variable is only present to support the GUI, then this functionality should be embedded in the GUI.

Conversely, while it is tempting to use this widget to quickly and easily do GUI-side calculations, you should ask your self whether this should really be done in an IOC. It can then be archived, alarmed, plotted, and be available to any other EPICS client.

9.131.2 Property Documentation

9.131.2.1 double QECalcout::a [read, write]

Sets the 'constant' input values.

9.131.2.2 QString QECalcout::calc [read, write]

Specifies the calculation to be performed. The syntax is identical to that used by the calc/calcout records. However it is not limited to 80 characters. The default is "0"

9.131.2.3 QString QECalcout::calcStatus [read, write]

Shows the state of the calculation expression.

9.131.2.4 QString QECalcout::egu [read, write]

Controls the string format for both the out (QString) signal and the widhets own text.

9.131.2.5 QString QECalcout::ocal [read, write]

Specifies the ouput calc to be performed. The syntax is identical to that used by the calc/calcout records. However it is not limited to 80 characters. The default is "0"

9.131.2.6 QString QECalcout::ocalStatus [read, write]

Shows the state of the output calculation expression.

9.131.2.7 bool QECalcout::runVisible [read, write]

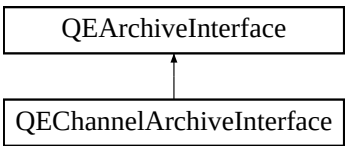
Allows the widget to be visible ar run time. Default is false. Note: always visible in designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QECalcout/QECalcout.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QECalcout.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QECalcout/QECalcout.cp

9.132 QEChannelArchiveInterface Class Reference

Inheritance diagram for QEChannelArchiveInterface::



Public Member Functions

- **QEChannelArchiveInterface** (QUrl url, QObject ∗parent=0)
- virtual void **namesRequest** (QObject ∗userData, const int key, QString pattern=".∗")
- virtual void **valuesRequest** (QObject ∗userData, const QCaDateTime startTime, const QCaDateTime endTime, const int count, const How how, const QStringList pvNames, const int key=0, const unsigned int requested_element=0)
- virtual void **infoRequest** (QObject ∗userData)
- virtual void **archivesRequest** (QObject ∗userData)
- virtual void **setUrl** (QUrl url)

Friends

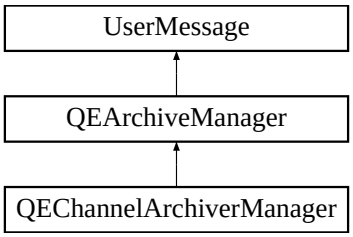
- class QEArchiveInterfaceAgent

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.cpp

9.133 QEChannelArchiverManager Class Reference

Inheritance diagram for QEChannelArchiverManager::



Friends

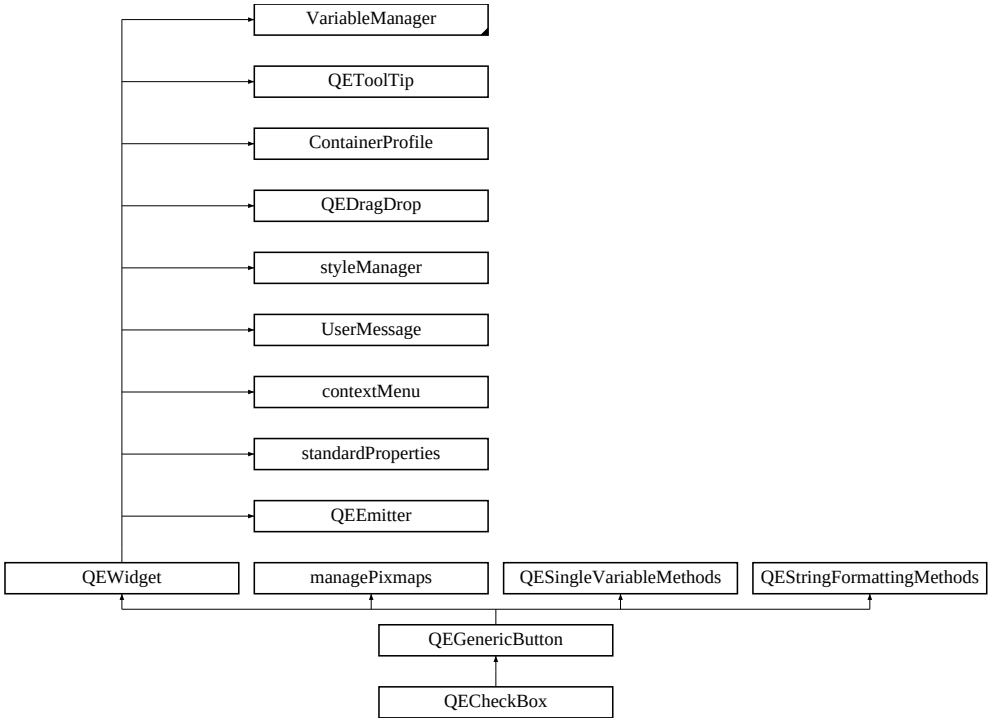
- class QEArchiveAccess

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp

9.134 QECheckBox Class Reference

Inheritance diagram for QECheckBox::



Public Types

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

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

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Underscore = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }
- enum UpdateOptions {

  Text = QEGenericButton::UPDATE_TEXT, Icon = QEGenericButton::UPDATE_ICON, TextAndIcon = QEGenericButton::UPDATE_TEXT_AND_ICON, State = QEGenericButton::UPDATE_STATE,

  TextAndState = QEGenericButton::UPDATE_TEXT_AND_STATE, IconAndState = QEGenericButton::UPDATE_ICON_AND_STATE, TextIconAndState = QEGenericButton::UPDATE_TEXT_ICON_AND_STATE }

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

- enum ProgramStartupOptionNames { None = applicationLauncher::PSO_NONE, Terminal = applicationLauncher::PSO_TERMINAL, LogOutput = applicationLauncher::PSO_LOGOUTPUT, StdOutput = applicationLauncher::PSO_STDOUTPUT }
- enum CreationOptionNames {

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

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

  DockBottomTabbed = QEActionRequests::OptionBottomDockWindowTabbed, DockLeftTabbed = QEActionRequests::OptionLeftDockWindowTabbed, DockRightTabbed = QEActionRequests::OptionRightDockWindowTabbed, DockFloating = QEActionRequests::OptionFloatingDockWindow }

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

Public Slots

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

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void requestResend ()

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

- void newGui (const QEActionRequests &request)

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

- void pressed (int value)
- void released (int value)
- void clicked (int value)
- void programCompleted ()

  Program started by button has compelted.

Public Member Functions

- QECheckBox (QWidget *parent=0)
- QECheckBox (const QString &variableName, QWidget *parent=0)
- void writeNow ()
- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

- void setFormatProperty (Formats format)

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

- Formats getFormatProperty ()

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

- void setSeparatorProperty (const Separators notation)

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

- Separators getSeparatorProperty () const

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

- void setNotationProperty (Notations notation)

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

- Notations getNotationProperty ()

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

- void setArrayActionProperty (ArrayActions arrayAction)

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

- ArrayActions getArrayActionProperty ()

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

Properties

- QString variable
- QString variableSubstitutions
- int arrayIndex
- bool subscribe
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle

- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction
- QEWidgetProperties::DisabledRecordPolicy disabledRecordPolicy
- Qt::Alignment alignment
- UpdateOptions updateOption
- QPixmap pixmap0
- QPixmap pixmap1
- QPixmap pixmap2
- QPixmap pixmap3
- QPixmap pixmap4
- QPixmap pixmap5
- QPixmap pixmap6
- QPixmap pixmap7
- QString password
- bool confirmAction
- QString confirmText
- bool writeOnPress
- bool writeOnRelease
- bool writeOnClick
- QString pressText
- QString releaseText
- QString clickText
- QString clickCheckedText
- QString labelText
- QString program
- QStringList arguments
- ProgramStartupOptionNames programStartupOption
- QString guiFile
- CreationOptionNames creationOption
- QString prioritySubstitutions
- QString customisationName

9.134.1 Member Enumeration Documentation

9.134.1.1 enum QECheckBox::ArrayActions

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

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.134.1.2 enum QECheckBox::CreationOptionNames

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

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

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

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

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

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

DockFloating Open new GUI in a floating dock window.

9.134.1.3 enum QECheckBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.134.1.4 enum QECheckBox::Formats

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

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.134.1.5 enum QECheckBox::Notations

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

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.134.1.6 enum QECheckBox::ProgramStartupOptionNames

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

Enumerator:

None Just run the program.

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

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

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

9.134.1.7 enum QECheckBox::Separators

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

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.134.1.8 enum QECheckBox::UpdateOptions

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

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

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

TextAndState Data updates will update the button text and state.

IconAndState Data updates will update the button icon and state.

TextIconAndState Data updates will update the button - the lot.

9.134.1.9 enum QECheckBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.134.2 Constructor & Destructor Documentation

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

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

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

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

9.134.3 Member Function Documentation

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

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

9.134.3.2 void QECheckBox::dbValueChanged () [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.

9.134.3.3 void QECheckBox::pressed (int *value*) [signal]

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

9.134.3.4 void QECheckBox::released (int *value*) [signal]

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

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

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

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

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

9.134.3.7 void QECheckBox::writeNow () [inline, virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.134.4 Property Documentation

9.134.4.1 bool QECheckBox::addUnits [read, write]

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

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

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

9.134.4.3 bool QECheckBox::allowDrop [read, write]

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

Reimplemented from QEDragDrop.

9.134.4.4 QStringList QECheckBox::arguments [read, write]

Arguments for program specified in the 'program' property.

9.134.4.5 ArrayActions QECheckBox::arrayAction [read, write]

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

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.

- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.134.4.6 int QECheckBox::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.134.4.7 QString QECheckBox::clickCheckedText [read, write]

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

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

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

Reimplemented from QEGenericButton.

9.134.4.8 QString QECheckBox::clickText [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.9 bool QECheckBox::confirmAction [read, write]

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

9.134.4.10 QString QECheckBox::confirmText [read, write]

Text used to confirm acion if confirmation dialog is presented

Reimplemented from QEGenericButton.

9.134.4.11 CreationOptionNames QECheckBox::creationOption [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.12 QString QECheckBox::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from QEGenericButton.

9.134.4.13 QString QECheckBox::defaultStyle [read, write]

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

9.134.4.14 QEWidgetProperties::DisabledRecordPolicy QECheckBox::disabledRecordPolicy [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.15 bool QECheckBox::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.134.4.16 DisplayAlarmStateOptions QECheckBox::displayAlarmStateOption [read, write]

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

9.134.4.17 bool QECheckBox::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.134.4.18 Formats QECheckBox::format [read, write]

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

9.134.4.19 QString QECheckBox::guiFile [read, write]

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

9.134.4.20 unsigned QECheckBox::int [read, write]

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

9.134.4.21 QString QECheckBox::labelText [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.22 bool QECheckBox::leadingZero [read, write]

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

9.134.4.23 QString QECheckBox::localEnumeration [read, write]

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

Format is:

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

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

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

may be included in a string to indicate a line break

Examples are:

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

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

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

9.134.4.24 Notations QECheckBox::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.134.4.25 QString QECheckBox::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from QEGenericButton.

9.134.4.26 QPixmap QECheckBox::pixmap0 [read, write]

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

9.134.4.27 QPixmap QECheckBox::pixmap1 [read, write]

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

9.134.4.28 QPixmap QECheckBox::pixmap2 [read, write]

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

9.134.4.29 QPixmap QECheckBox::pixmap3 [read, write]

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

9.134.4.30 QPixmap QECheckBox::pixmap4 [read, write]

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

9.134.4.31 QPixmap QECheckBox::pixmap5 [read, write]

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

9.134.4.32 QPixmap QECheckBox::pixmap6 [read, write]

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

9.134.4.33 QPixmap QECheckBox::pixmap7 [read, write]

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

9.134.4.34 int QECheckBox::precision [read, write]

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

9.134.4.35 QString QECheckBox::pressText [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.36 QString QECheckBox::prioritySubstitutions [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.37 QString QECheckBox::program [read, write]

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

9.134.4.38 ProgramStartupOptionNames QECheckBox::programStartupOption [read, write]

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

9.134.4.39 int QECheckBox::radix [read, write]

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

9.134.4.40 QString QECheckBox::releaseText [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.41 Separators QECheckBox::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.134.4.42 QString QECheckBox::styleSheet [read, write]

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

9.134.4.43 bool QECheckBox::subscribe [read, write]

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

Reimplemented from VariableManager.

9.134.4.44 bool QECheckBox::trailingZeros [read, write]

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

9.134.4.45 UpdateOptions QECheckBox::updateOption [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.46 bool QECheckBox::useDbPrecision [read, write]

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

9.134.4.47 UserLevels QECheckBox::userLevelEnabled [read, write]

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

9.134.4.48 QString QECheckBox::userLevelEngineerStyle [read, write]

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

9.134.4.49 QString QECheckBox::userLevelScientistStyle [read, write]

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

9.134.4.50 QString QECheckBox::userLevelUserStyle [read, write]

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

9.134.4.51 UserLevels QECheckBox::userLevelVisibility [read, write]

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

9.134.4.52 QString QECheckBox::variable [read, write]

EPICS variable name (CA PV)

9.134.4.53 bool QECheckBox::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

9.134.4.54 QString QECheckBox::variableSubstitutions [read, write]

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

9.134.4.55 bool QECheckBox::visible [read, write]

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

9.134.4.56 bool QECheckBox::writeOnClick [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.57 bool QECheckBox::writeOnPress [read, write]

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

Reimplemented from QEGenericButton.

9.134.4.58 bool QECheckBox::writeOnRelease [read, write]

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

Reimplemented from QEGenericButton.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QECheckBox.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QECheckBox.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QECheckBox.cpp

9.135 QEColourBand Struct Reference

Public Attributes

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEColourBandList.h

9.136 QEColourBandList Class Reference

`#include <QEColourBandList.h>`

Public Member Functions

- void **clear** ()
- void **append** (const QEColourBand &item)
- int **count** () const
- QEColourBand **value** (int j) const
- void **setAlarmColours** (const double dispLower, const double dispUpper, qcaobject::QCaObject *qca)

9.136.1 Detailed Description

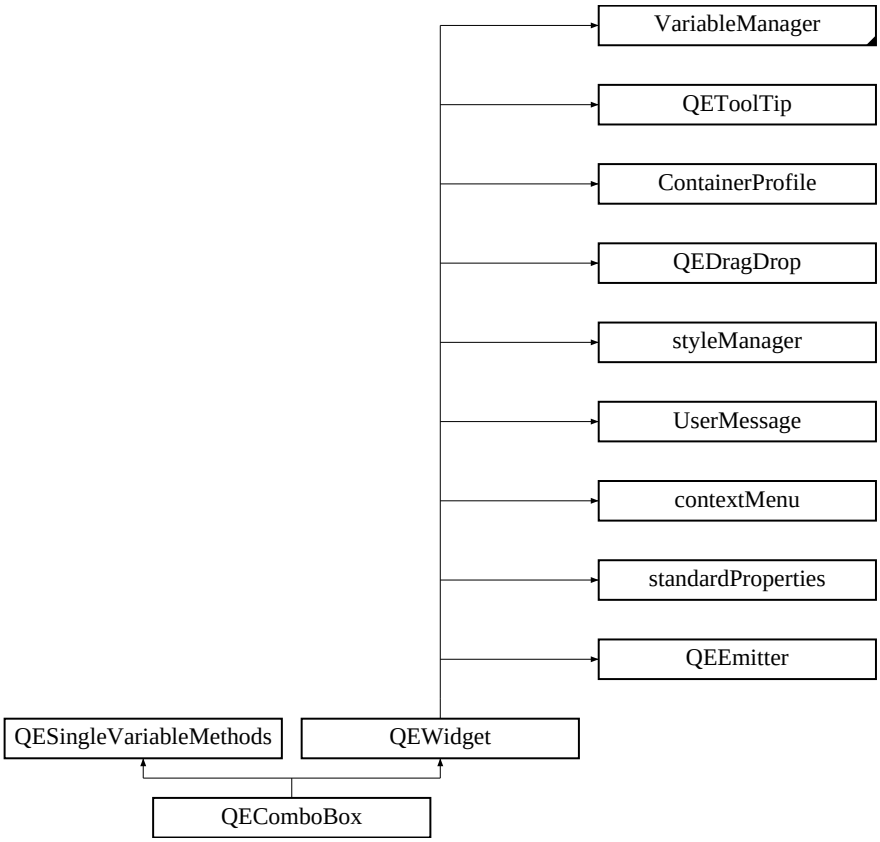
Defines a list of QEColourBands elements. Support class for QAxisPainter

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEColourBandList.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEColourBandList.cpp

9.137 QEComboBox Class Reference

Inheritance diagram for QEComboBox::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void userChange (const QString &oldValue, const QString &newValue, const QString &lastValue)

  Internal use only. Used by QEConfiguredLayout to be notified when one of its widgets has written something.

Public Member Functions

- **QEComboBox** (QWidget *parent=0)
- **QEComboBox** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** () const
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** () const
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** () const
- void **setLocalEnumerations** (const QString &localEnumerations)
- QString **getLocalEnumerations** () const
- void **setAllowFocusUpdate** (bool allowFocusUpdate)
- bool **getAllowFocusUpdate** () const
- void writeNow ()
- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

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

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Protected Attributes

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

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool subscribe
- bool allowFocusUpdate
- bool variableAsToolTip

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

9.137.1 Member Enumeration Documentation

9.137.1.1 enum QEComboBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.137.1.2 enum QEComboBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.137.2 Member Function Documentation

9.137.2.1 void QEComboBox::dbValueChanged () `[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.

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

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

9.137.2.3 void QEComboBox::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.137.3 Member Data Documentation

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

Use database enumerations - defaults to true

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

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

9.137.4 Property Documentation

9.137.4.1 bool QEComboBox::allowDrop [read, write]

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

Reimplemented from QEDragDrop.

9.137.4.2 bool QEComboBox::allowFocusUpdate [read, write]

Allow updated while widget has focus - defaults to false

9.137.4.3 int QEComboBox::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.137.4.4 QString QEComboBox::defaultStyle [read, write]

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

9.137.4.5 bool QEComboBox::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.137.4.6 DisplayAlarmStateOptions QEComboBox::displayAlarmStateOption [read, write]

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

9.137.4.7 int QEComboBox::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.137.4.8 unsigned QEComboBox::int [read, write]

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

9.137.4.9 QString QEComboBox::localEnumeration [read, write]

Enumrations values used when useDbEnumerations is false.

9.137.4.10 QString QEComboBox::styleSheet [read, write]

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

9.137.4.11 bool QEComboBox::subscribe [read, write]

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

Reimplemented from VariableManager.

9.137.4.12 UserLevels QEComboBox::userLevelEnabled [read, write]

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

9.137.4.13 QString QEComboBox::userLevelEngineerStyle [read, write]

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

9.137.4.14 QString QEComboBox::userLevelScientistStyle [read, write]

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

9.137.4.15 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.

9.137.4.16 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'.

9.137.4.17 QString QEComboBox::variable [read, write]

EPICS variable name (CA PV)

9.137.4.18 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.

9.137.4.19 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 are also used for other purposes.

9.137.4.20 bool QEComboBox::visible [read, write]

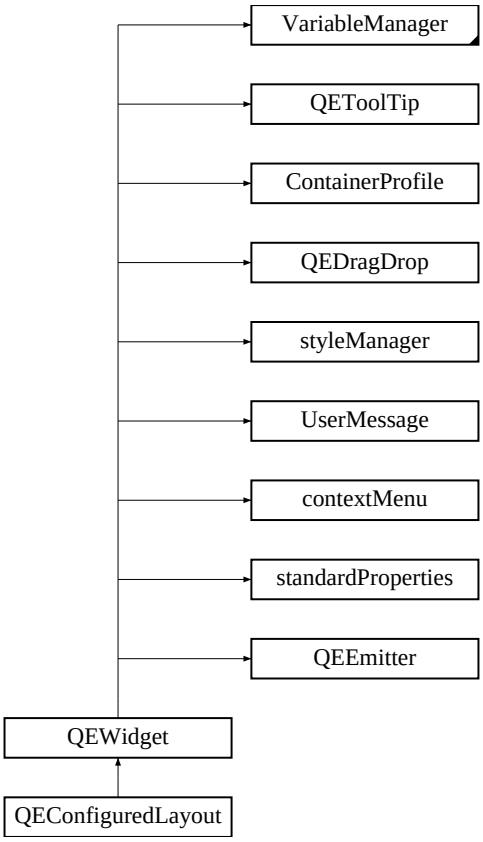
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEComboBox/QEComboBox.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEComboBox.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEComboBox/QEComboBox.cpp

9.138 QEConfiguredLayout Class Reference

Inheritance diagram for QEConfiguredLayout::



Public Types

- enum **types** {

  LABEL, **LINEEDIT**, **COMBOBOX**, **SPINBOX**,

  BUTTON }
- enum **configurationTypesProperty** { **File**, **Text** }
- enum **optionsLayoutProperty** { **Top**, **Bottom**, **Left**, **Right** }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Public Member Functions

- **QEConfiguredLayout** (QWidget *pParent=0, bool pSubscription=true)
- void **setItemDescription** (QString pValue)
- QString **getItemDescription** ()
- void **setShowItemList** (bool pValue)
- bool **getShowItemList** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- void **refreshFields** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

Public Attributes

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

Protected Attributes

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

Properties

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

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

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

9.138.1 Member Enumeration Documentation

9.138.1.1 enum QEConfiguredLayout::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.138.1.2 enum QEConfiguredLayout::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.138.2 Member Function Documentation

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

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

9.138.3 Property Documentation

9.138.3.1 bool QEConfiguredLayout::allowDrop [read, write]

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

Reimplemented from QEDragDrop.

9.138.3.2 QString QEConfiguredLayout::defaultStyle [read, write]

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

9.138.3.3 bool QEConfiguredLayout::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.138.3.4 DisplayAlarmStateOptions QEConfiguredLayout::displayAlarmStateOption [read, write]

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

9.138.3.5 unsigned QEConfiguredLayout::int [read, write]

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

9.138.3.6 QString QEConfiguredLayout::styleSheet [read, write]

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

9.138.3.7 UserLevels QEConfiguredLayout::userLevelEnabled [read, write]

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

9.138.3.8 QString QEConfiguredLayout::userLevelEngineerStyle [read, write]

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

9.138.3.9 QString QEConfiguredLayout::userLevelScientistStyle [read, write]

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

9.138.3.10 QString QEConfiguredLayout::userLevelUserStyle [read, write]

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

9.138.3.11 UserLevels QEConfiguredLayout::userLevelVisibility [read, write]

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

9.138.3.12 bool QEConfiguredLayout::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

9.138.3.13 bool QEConfiguredLayout::visible [read, write]

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEConfiguredLayout/QE

9.139 QEContextMenuObject Class Reference

Public Slots

- void **contextMenuTriggeredSlot** (QAction ∗selectedItem)
- void **showContextMenuSlot** (const QPoint &pos)

Signals

- void **requestAction** (const QEActionRequests &)

Public Member Functions

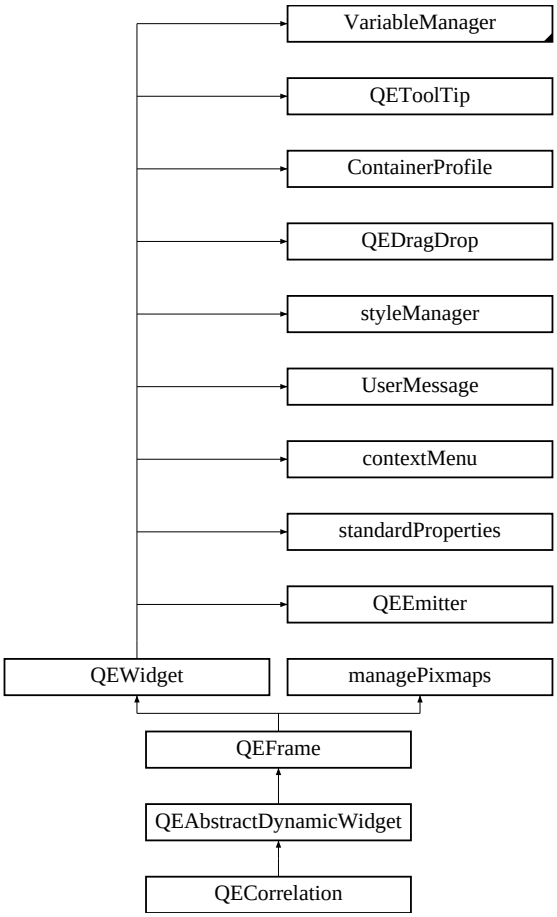
- **QEContextMenuObject** (contextMenu ∗menuIn, QObject ∗parent)
- void **sendRequestAction** (const QEActionRequests &request)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/contextMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_contextMenu.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/contextMenu.cpp

9.140 QECorrelation Class Reference

Inheritance diagram for QECorrelation::



Public Member Functions

- **QECorrelation** (QWidget ∗parent=0)
- **QECorrelation** (const QString &xVariableName, QWidget ∗parent=0)
- **QECorrelation** (const QString &xVariableName, const QString &yVariableName, QWidget ∗parent=0)
- void **setXVariableName** (const QString &pvName)
- QString **getXVariableName** () const
- void **setYVariableName** (const QString &pvName)
- QString **getYVariableName** () const
- void **setVariableSubstitutions** (const QString &variableNameSubstitutions)
- QString **getVariableSubstitutions** () const
- void **setSampleInterval** (const double interval)

- double **getSampleInterval** () const
- void **setNumberPoints** (const int number)
- int **getNumberPoints** () const
- void **setXLogarithmic** (bool visible)
- bool **getXLogarithmic** () const
- void **setYLogarithmic** (bool visible)
- bool **getYLogarithmic** () const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)
- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dragMoveEvent** (QDragMoveEvent *event)
- void **dropEvent** (QDropEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void saveConfiguration (PersistanceManager *pm)
- void restoreConfiguration (PersistanceManager *pm, restorePhases restorePhase)

Properties

- QString **variableX**
- QString **variableY**
- QString variableSubstitutions
- double **sampleInterval**
- int **numberPoints**
- bool **xLogarithmic**
- bool **yLogarithmic**

9.140.1 Member Function Documentation

9.140.1.1 void QECorrelation::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.140.1.2 void QECorrelation::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

9.140.2 Property Documentation

9.140.2.1 QString QECorrelation::variableSubstitutions [read, write]

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QECorrelation/QECorrela
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QECorrelation/QECorrela

9.141 QEDelayedText Class Reference

`#include <QEDelayedText.h>`

Public Slots

- void **setText** (const QString &)

Signals

- void **textChanged** (const QString &)

Public Member Functions

- **QEDelayedText** (const double delay, QObject ∗parent=0)
- bool **doubleConnect** (const QObject ∗sender, const char ∗signal, const QObject ∗receiver, const char ∗member) const

9.141.1 Detailed Description

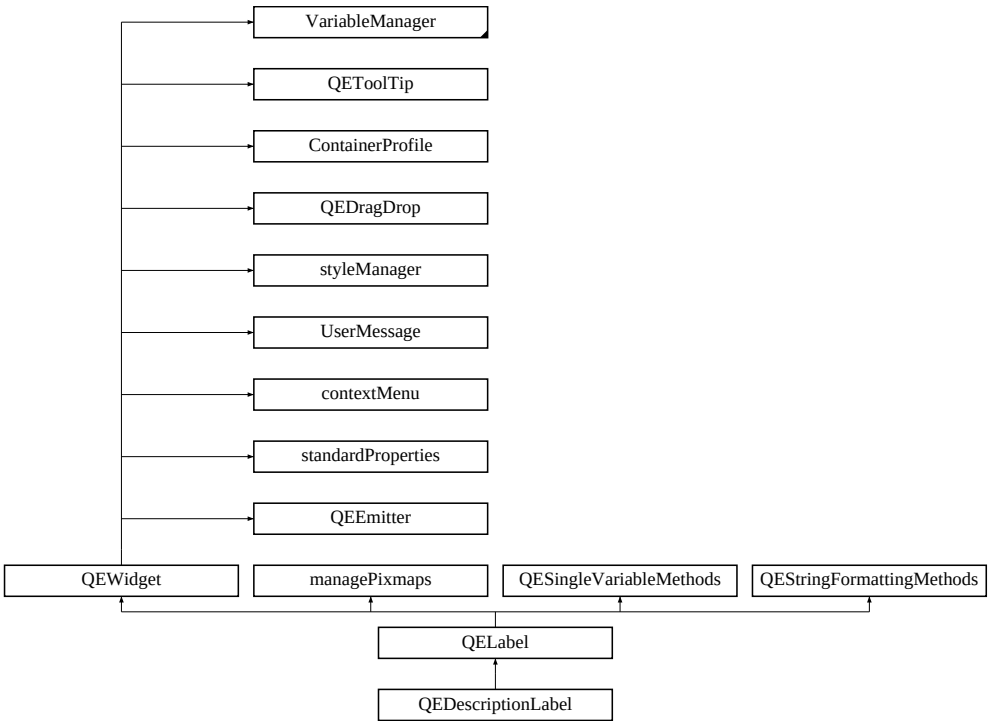
Provides a delayed text signal, noy unlike that employed by QCaVariableNamePropertyManager

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDelayedText.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDelayedText.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEDelayedText.cpp

9.142 QEDescriptionLabel Class Reference

`#include <QEDescriptionLabel.h>`Inheritance diagram for QEDescriptionLabel::



Public Member Functions

- QEDescriptionLabel (QWidget *parent=0)
- QEDescriptionLabel (const QString &variableName, QWidget *parent=0)
- ~QEDescriptionLabel ()

9.142.1 Detailed Description

This is just a QELabel, but with some different default property values. These are: a) Clear style sheet and default style. b) The indent set to -1, the QLabel default c) Set display alarm option to DISPLAY_ALARM_STATE_NEVER d) Set Font size to 8.

Note: there are no additional proerties or functionality.

9.142.2 Constructor & Destructor Documentation

9.142.2.1 QEDescriptionLabel::QEDescriptionLabel (QWidget * *parent* = 0) [explicit]

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

9.142.2.2 QEDescriptionLabel::QEDescriptionLabel (const QString & *variableName*, QWidget * *parent* = 0) [explicit]

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.

9.142.2.3 QEDescriptionLabel::~QEDescriptionLabel ()

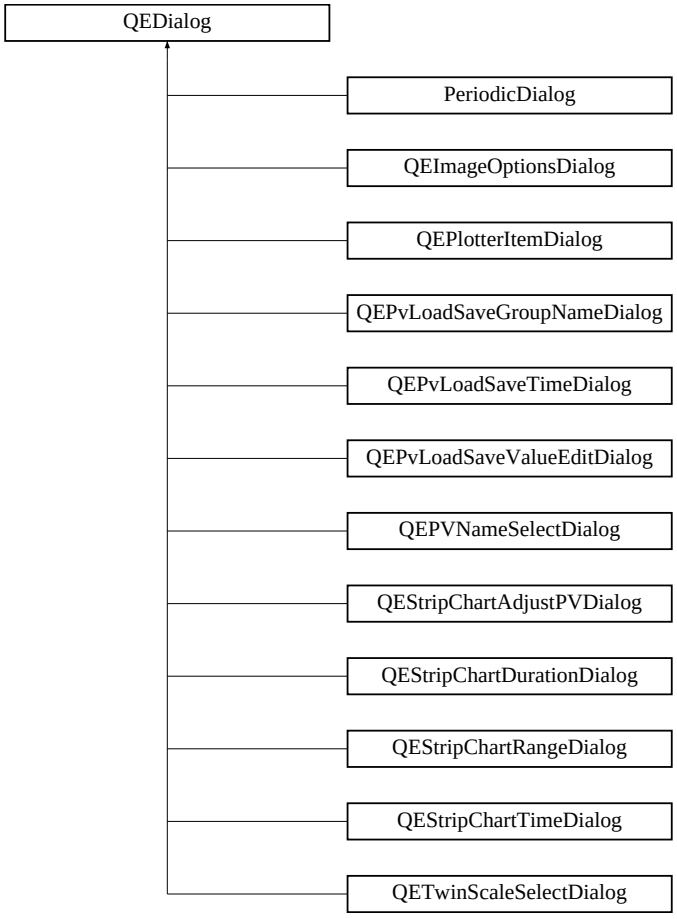
Destructor

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELabel/QEDescriptionLabel.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELabel/QEDescriptionLabel.cpp

9.143 QEDialog Class Reference

`#include <QEDialog.h>`Inheritance diagram for QEDialog::



Public Slots

- int **exec** (QWidget ∗targetWidget)

Public Member Functions

- **QEDialog** (QWidget ∗parent=0)

Protected Member Functions

- void **setSourceWidget** (QWidget ∗widget)

9.143.1 Detailed Description

This class provides a thin wrapper around the QDialog class, specifically the exec function. There are two issues with the standard QDialog when exec-ed.

a) if the dialog has no parent and there is a single application form open then the dialog is centred over the form - which is good. However if two or more forms are open, it appears in the centre of the screen - okay but not ideal; and

b) when scaling applied (even null scaling) this confuses the dialog and it appears in the top left corner of the screen which is not acceptable.

The overaloded wrapper function allows a widget to be specified, and if specified the dialog is centred over the widget.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDialog.cpp

9.144 QEDisplayRanges Class Reference

Public Member Functions

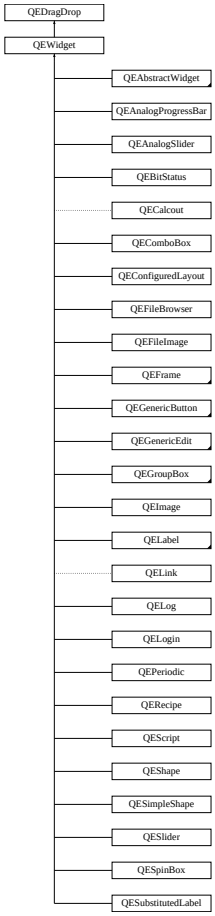
- **QEDisplayRanges** (const double min, const double max)
- bool **operator==** (const QEDisplayRanges &other) const
- bool **operator!=** (const QEDisplayRanges &other) const
- bool **isSimilar** (const QEDisplayRanges &other, const double tolerance) const
- void **clear** ()
- void **setRange** (const double min, const double max)
- void **merge** (const double d)
- void **merge** (const QEDisplayRanges &other)
- bool **getIsDefined** () const
- double **getMinimum** () const
- double **getMaximum** () const
- bool **getMinMax** (double &min, double &max) const
- void **adjustLogMinMax** (double &minOut, double &maxOut, double &majorOut) const
- void **adjustMinMax** (const int number, const bool roundToMajor, double &minOut, double &maxOut, double &majorOut) const

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDisplayRanges.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEDisplayRanges.cpp

9.145 QEDragDrop Class Reference

Inheritance diagram for QEDragDrop::



Public Member Functions

- **QEDragDrop** (QEWidget ∗qewIn, QWidget ∗ownerIn)
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** () const

Protected Member Functions

- void **qcaDragEnterEvent** (QDragEnterEvent ∗event, const bool allowSelfDrop=false)
- void **qcaDropEvent** (QDropEvent ∗event, const bool allText=false)
- void **qcaMousePressEvent** (QMouseEvent ∗event)
- virtual void **setDrop** (QVariant)

- virtual QVariant **getDrop** ()
- void **setDragDropConsumer** (QObject ∗consumer)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEDragDrop.
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEDragDrop.

9.146 QEDragDropObject Class Reference

Signals

- void **requestAction** (const QEActionRequests &)

Public Member Functions

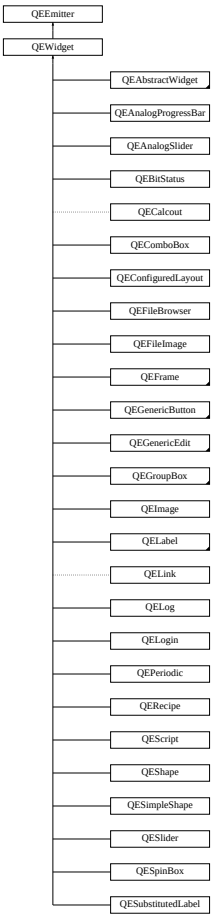
- **QEDragDropObject** (QObject *parent)
- void **sendRequestAction** (const QEActionRequests &request)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEDragDrop.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEDragDrop.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEDragDrop.cpp

9.147 QEEmitter Class Reference

Inheritance diagram for QEEmitter::



Public Member Functions

- **QEEmitter** (QEWidget ∗qew, QWidget ∗owner)

Protected Member Functions

- void **emitDbConnectionChanged** (const unsigned int variableIndex)
- void **emitDbValueChanged** (const unsigned int variableIndex)
- void **emitDbValueChanged** (const QString &formatedText, const unsigned int variableIndex)

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEEmitter.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEEmitter.cpp

9.148 QEEnvironmentShare Class Reference

Public Attributes

- QSharedMemory * **sharedMemory**
- PublishedProfile * **publishedProfile**
- bool **publishedProfileCreatedByMe**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf

9.149 QEEnvironmentVariables Class Reference

`#include <QEEnvironmentVariables.h>`

Public Member Functions

- **QEEnvironmentVariables** (const QString &prefix)
- bool **getBool** (const QString &name, const bool defaultValue) const
- QString **getString** (const QString &name, const QString &defaultValue) const
- int **getInt** (const QString &name, const int defaultValue) const
- double **getFloat** (const QString &name, const double defaultValue) const

Static Public Member Functions

- static void **setDefaultPrefix** (const QString &prefix)
- static QString **getDefaultPrefix** ()

9.149.1 Detailed Description

This class provides a means to access environment variables.

The types of data that my be extracted from an option are bool, QString, int and double; These are accessed by the associated getXxxx () functions where Xxxx is one of Bool, String, Int and Float.

Each of these function takes a default value of the appropriate type which is returned to the caller if the value is not defined or, as in the case of numerical values, is ill-defined.

Numeric validity is defined by QString::toInt () and QString::toDouble () Do note that QString::toInt () does not accept 0x... hexadecimal or 0... octal numbers.

The environment variable name is formed by concating the prefix and the suffix and conveting this to upper case if needs be. The prefix can be specified when the object is constucted otherwise the default prefix is used when no prefix provided. Example prefix could be "QE_".

This class is one of several provided to access configuration data such as from environment variables, command line options and settings. They all provide a similar API.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_parameters/QEEnvironmentVariables.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_parameters/QEEnvironmentVariables.cpp

9.150 QEExpressionEvaluation Class Reference

`#include <QEExpressionEvaluation.h>`

Public Types

- enum **InputKinds** { **Normal**, **Primed** }
- typedef double **CalculateArguments** [NumberInputKinds][NumberUserArguments]

Public Member Functions

- **QEExpressionEvaluation** (const bool allowPrimedInput)
- bool **initialise** (const QString &expression)
- QString **getCalcError** () const
- double **evaluate** (const CalculateArguments &userArgs, bool ∗okay=0) const

Static Public Member Functions

- static void **clear** (CalculateArguments &userArgs)
- static int **indexOf** (const char c)

Static Public Attributes

- static const int **NumberUserArguments** = 26
- static const int **NumberInputKinds** = 2

9.150.1 Detailed Description

Description: The QEExpressionEvaluation object allows the evaluation of expressions similar to that available in calc/calcout records; and in fact "under the covers" it uses the calcRecord's postfix functions out of the Com shared library.

The user initialises the QEExpressionEvaluation object with an expression as per the CALC field of a calc (or calcout) record, e.g. "A + LN (B/C)", and then may evaluate the expression by calling the Evaluate method with an array of Doubles, each element supplying the value for A, B, C etc. The standard CALC functions (SIN, LOG, ABS etc.) are all available.

The QEExpressionEvaluation object provides an extended input name space for the user supplied expressions, i.e. 52 names (A .. Z, A' .. Z') instead of the native 12 inputs (A .. L). However, the MAXIMUM number of inputs that may be used in any one expression is still LIMITED to 12.

That is: "B + L' + M + X + Y' " is an allowed expression, but "A + B + C + D + E + F + G + H + I + J + K + L' + M" is invalid because more than 12 inputs specified.

This input name space extension is to allow sensible use with the Strip Chart widget that has 16 available inputs and the Plotter widget that has 16 avialable inputs plus an X input.

Also, the length of the input string is not limited to the 40 characters of the CALC field, but may use the full 100 characters allowed by the underlying postfix function.

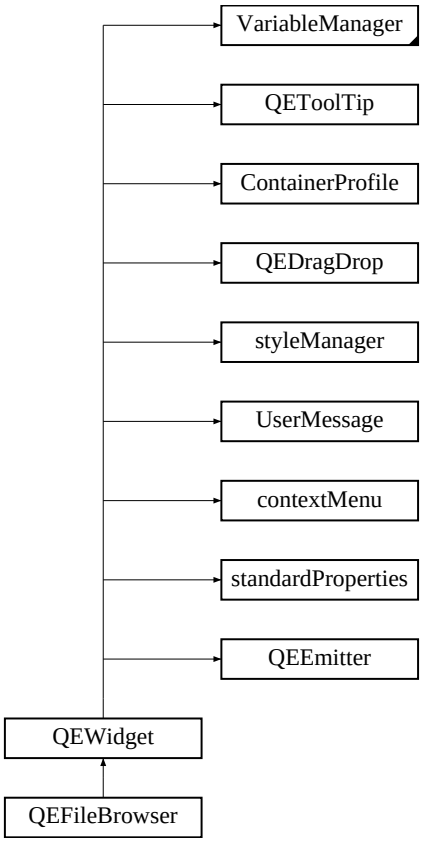
Acknowledgements: QEExpressionEvaluation is a direct crib of TCalculate out of the Delphi OPI framework. The postfix and calcPerform functions were written by Bob Dalesio (12-12-86).

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEExpressionEvaluation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEExpressionEvaluation.cpp

9.151 QEFileBrowser Class Reference

`#include <QEFileBrowser.h>`Inheritance diagram for QEFileBrowser::



Public Types

- enum **optionsLayoutProperty** { **Top**, **Bottom**, **Left**, **Right** }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Signals

- void selected (QString pFilename)

Public Member Functions

- **QEFileBrowser** (QWidget *pParent=0)
- void **setVariableName** (QString pValue)
- QString **getVariableName** ()
- void **setVariableNameSubstitutions** (QString pValue)
- QString **getVariableNameSubstitutions** ()
- void **setDirectoryPath** (QString pValue)
- QString **getDirectoryPath** ()
- void **setShowDirectoryPath** (bool pValue)
- bool **getShowDirectoryPath** ()
- void **setShowDirectoryBrowser** (bool pValue)
- bool **getShowDirectoryBrowser** ()
- void **setShowRefresh** (bool pValue)
- bool **getShowRefresh** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnSize** (bool pValue)
- bool **getShowColumnSize** ()
- void **setShowColumnFilename** (bool pValue)
- bool **getShowColumnFilename** ()
- void **setShowFileExtension** (bool pValue)
- bool **getShowFileExtension** ()
- void **setFileFilter** (QString pValue)
- QString **getFileFilter** ()
- void **setFileDialogDirectoriesOnly** (bool pValue)
- bool **getFileDialogDirectoriesOnly** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **updateTable** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

Protected Attributes

- QELineEdit * **qeLineEditDirectoryPath**
- QPushButton * **qPushButtonDirectoryBrowser**
- QPushButton * **qPushButtonRefresh**
- _QTableWidgetFileBrowser * **qTableWidgetFileBrowser**
- QString fileFilter
- bool showFileExtension

  Show/hide the extension of files.

- bool fileDialogDirectoriesOnly

  Enable/disable the browsing of directories-only when opening the dialog window.

- int **optionsLayout**

Properties

- QString variable
- QString variableSubstitutions
- QString directoryPath

  Default directory where to browse files when QEFileBrowser is launched for the first time.

- bool showDirectoryPath

  Show/hide directory path line edit where the user can specify the directory to browse files.

- bool showDirectoryBrowser

  Show/hide button to open the dialog window to browse for directories and files.

- bool showRefresh

  Show/hide button to refresh the table containing the list of files being browsed.

- bool showTable

  Show/hide table containing the list of files being browsed.

- bool showColumnTime

  Show/hide column containing the time of creation of files.

- bool showColumnSize

  Show/hide column containing the size (in bytes) of files.

- bool showColumnFilename

  Show/hide column containing the name of files.

- optionsLayoutProperty optionsLayout

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

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

9.151.1 Detailed Description

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

9.151.2 Member Enumeration Documentation

9.151.2.1 enum QEFileBrowser::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.151.2.2 enum QEFileBrowser::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.151.3 Member Function Documentation

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

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

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

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

9.151.4 Member Data Documentation

9.151.4.1 QString QEFileBrowser::fileFilter [read, write, protected]

Specify which files to browse. To specify more than one filter, please separate them with a ";". Example: ∗.py;∗.ui (this will only display files with an extension .py or .ui).

9.151.5 Property Documentation

9.151.5.1 bool QEFileBrowser::allowDrop [read, write]

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

Reimplemented from QEDragDrop.

9.151.5.2 QString QEFileBrowser::defaultStyle [read, write]

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

9.151.5.3 bool QEFileBrowser::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.151.5.4 DisplayAlarmStateOptions QEFileBrowser::displayAlarmStateOption [read, write]

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

9.151.5.5 unsigned QEFileBrowser::int [read, write]

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

9.151.5.6 QString QEFileBrowser::styleSheet [read, write]

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

9.151.5.7 UserLevels QEFileBrowser::userLevelEnabled [read, write]

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

9.151.5.8 QString QEFileBrowser::userLevelEngineerStyle [read, write]

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

9.151.5.9 QString QEFileBrowser::userLevelScientistStyle [read, write]

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

9.151.5.10 QString QEFileBrowser::userLevelUserStyle [read, write]

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

9.151.5.11 UserLevels QEFileBrowser::userLevelVisibility [read, write]

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

9.151.5.12 QString QEFileBrowser::variable [read, write]

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

9.151.5.13 bool QEFileBrowser::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

9.151.5.14 QString QEFileBrowser::variableSubstitutions [read, write]

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

9.151.5.15 bool QEFileBrowser::visible [read, write]

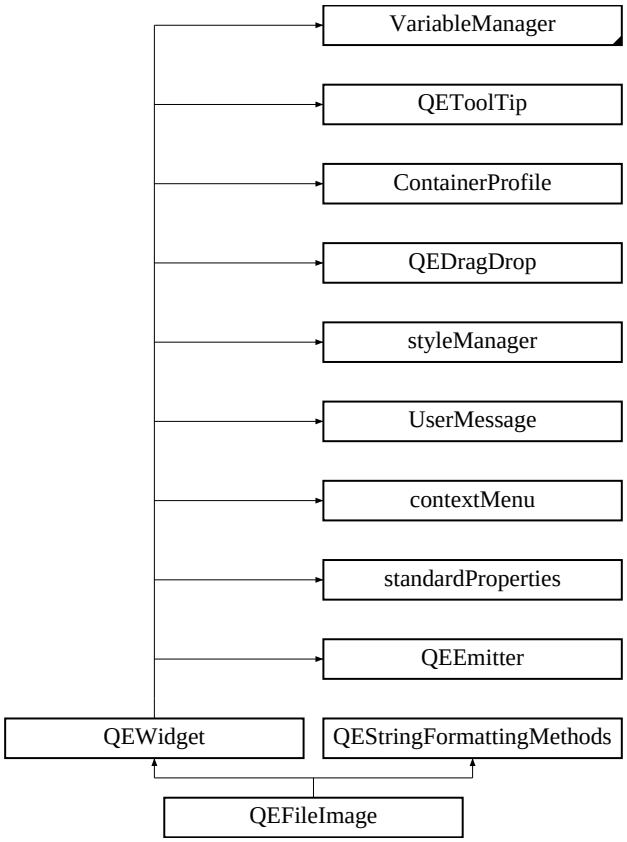
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileBrowser/QEFileBrowser.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEFileBrowser.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileBrowser/QEFileBrowser.cpp

9.152 QEFileImage Class Reference

`#include <QEFileImage.h>`Inheritance diagram for QEFileImage::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void **setImageFileName** (const QString &text)
- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- 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

- QEFileImage (QWidget *parent=0)
- QEFileImage (const QString &variableName, QWidget *parent=0)
- void setVariableNameProperty (QString variableName)

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

- QString getVariableNameProperty ()

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

- void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)

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

- QString getVariableNameSubstitutionsProperty ()

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

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

Properties

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

9.152.1 Detailed Description

This class is a EPICS aware image widget based on the Qt label widget. When a variable is defined, the label pixmap will be updated with the file specified by the variable. The label will be disabled if the variable is invalid. It is tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.152.2 Member Enumeration Documentation

9.152.2.1 enum QEFileImage::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.152.2.2 enum QEFileImage::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.

9.152.3 Constructor & Destructor Documentation

9.152.3.1 QEFileImage::QEFileImage (QWidget * *parent* = 0)

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

9.152.3.2 QEFileImage::QEFileImage (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.

9.152.4 Member Function Documentation

9.152.4.1 void QEFileImage::dbValueChanged () [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.

9.152.4.2 void QEFileImage::setManagedVisible (bool *v*) [inline, slot]

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

9.152.5 Property Documentation

9.152.5.1 bool QEFileImage::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.

9.152.5.2 QString QEFileImage::defaultStyle [read, write]

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

9.152.5.3 bool QEFileImage::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.152.5.4 DisplayAlarmStateOptions QEFileImage::displayAlarmStateOption [read, write]

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

9.152.5.5 unsigned QEFileImage::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.

9.152.5.6 QString QEFileImage::styleSheet [read, write]

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

9.152.5.7 UserLevels QEFileImage::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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.152.5.8 QString QEFileImage::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.

9.152.5.9 QString QEFileImage::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.

9.152.5.10 QString QEFileImage::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.

9.152.5.11 UserLevels QEFileImage::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'.

9.152.5.12 QString QEFileImage::variable [read, write]

EPICS variable name (CA PV)

9.152.5.13 bool QEFileImage::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.

9.152.5.14 QString QEFileImage::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.

9.152.5.15 bool QEFileImage::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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileImage/QEFileIma
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEFileImage.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFileImage/QEFileIma

9.153 QEFileMonitor Class Reference

`#include <QEFileMonitor.h>`

Signals

- void **fileChanged** (const QString &path)
- void **directoryChanged** (const QString &path)

Public Member Functions

- **QEFileMonitor** (QObject *parent=NULL)
- **QEFileMonitor** (const QString &path, QObject *parent=NULL)
- void **setPath** (const QString &path)
- QString **getPath** () const
- void **clearPath** ()

9.153.1 Detailed Description

The QEFileMonitor class provides a wrapper around a single QFileSystemWatcher object which can be used to monitor all file/directories that require watching. This class acts as an agent between the user and the singleton QFileSystemWatcher. As each instance of a QFileSystemWatcher object creates a new thread, it is far more thread efficient to use a single watcher to monitor many files/directories.

Note: each agent only sets up monitoring for a single file/directory. Multiple agents may/can monitor the same file/directory.

Refer to QFileSystemWatcher documentation on limits and monitoring features.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFileMonitor.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFileMonitor.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEFileMonitor.cpp

9.154 QEFixedPointRadix Class Reference

`#include <QEFixedPointRadix.h>`

Public Types

- enum Radicies { **Decimal** = 0, **Hexadecimal**, **Octal**, **Binary** }
  Defines allowed radix values.
- enum Separators { **None** = 0, **Comma**, **Underscore**, **Space** }
  Defines the digit 'thousands' separator character.

Public Member Functions

- **QEFixedPointRadix** (QObject *parent=0)
- **QEFixedPointRadix** (const Radicies radix, const Separators separator, QObject *parent=0)
- void **setRadix** (const Radicies radix)
- Radicies **getRadix** () const
- void **setSeparator** (const Separators separator)
- Separators **getSeparator** () const
- int **getRadixValue** () const
- bool **isRadixDigit** (const QChar qc) const
- QString **toString** (const double value, const bool sign, const int zeros, const int precision) const
- double **toValue** (const QString &image, bool &okay) const

9.154.1 Detailed Description

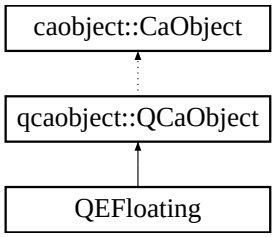
The QString class provides general value to/from radix string representation, but only for integer types. This class provdies similar functionality for the double type to/from a fixed point radix representation. The class restricts the allowed radix values to 2, 8, 10 and 16. The class allows the for the insertion of 'thousands' separators, typically comma, but space and under score are also allowed.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFixedPointRadix.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFixedPointRadix.cpp

9.155 QEFloating Class Reference

Inheritance diagram for QEFloating::



Public Slots

- void **writeFloating** (const double &data)
- void **writeFloatingElement** (const double &data)
- void **writeFloating** (const QVector< 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloating.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloating.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEFloating.cpp

9.156 QEFloatingArray Class Reference

`#include <QEFloatingArray.h>`

Public Member Functions

- **QEFloatingArray** (int size)
- **QEFloatingArray** (int size, const double &t)
- **QEFloatingArray** (const QVector< double > &other)
- QEFloatingArray & **operator=** (const QVector< double > &other)
- double **minimumValue** (const double &defaultValue=0.0)
- double **maximumValue** (const double &defaultValue=0.0)
- QEFloatingArray **calcDyByDx** (const QVector< double > &x)
- QEFloatingArray **medianFilter** (const int window)

9.156.1 Detailed Description

This class provides short hand for QVector<double> together with some basic double vector operations.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloatingArray.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloatingArray.cpp

9.157 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, int arrayIndex=0)
- QVector< double > formatFloatingArray (const QVariant &value)
- QVariant **formatValue** (const double &floatingValue, generic::generic_types valueType)
- QVariant **formatValue** (const QVector< double > &floatingValue, generic::generic_types valueType)
- void **setPrecision** (unsigned int precision)
- void **setFormat** (formats format)
- unsigned int **getPrecision** ()
- int **getFormat** ()

9.157.1 Member Function Documentation

9.157.1.1 double QEFloatingFormatting::formatFloating (const QVariant & *value*, int *arrayIndex* = 0)

Given a data value of any type, format it as an double 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 a double.

9.157.1.2 QVector< double > QEFloatingFormatting::formatFloatingArray (const QVariant & *value*)

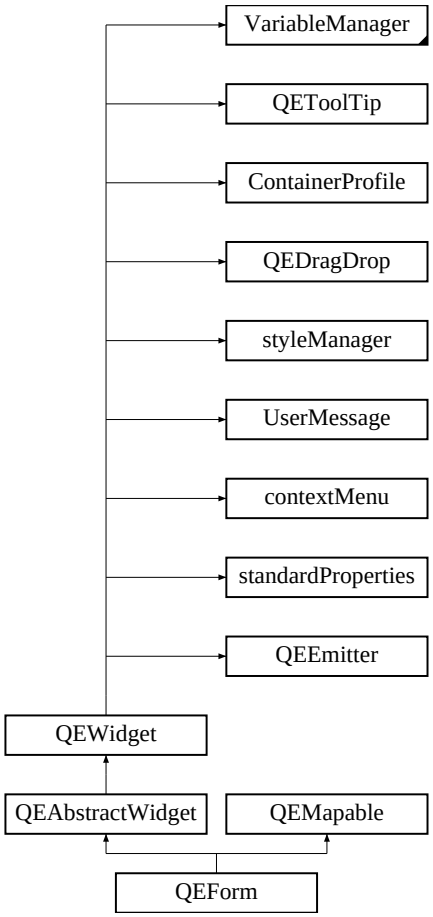
Given a data value of any type, format it as an array of doubles according to the formatting instructions held by the class. This is used to convert the QVariant value received from a QCaObject, which is still based on the data variable type, to a double array. Typically used where the input QVariant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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloatingFormatting.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEFloatingFormatting.cpp

9.158 QEForm Class Reference

Inheritance diagram for QEForm::



Public Types

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

Public Slots

- bool readUiFile ()

  Find a widget within the ui loaded by the QEForm. Returns NULL if no UI is loaded yet or if the named widget can't be found.

- void requestAction (const QEActionRequests &request)

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

Signals

- void **formLoaded** (bool fileLoaded)

Public Member Functions

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

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

- QString getFullFileName ()

  Get the title to be used as the window or form title.

- QString getUiFileName ()

  Get the standard, absolute UI file name.

- void setFileMonitoringIsEnabled (bool fileMonitoringIsEnabled)

  Get the fully substituted file name (Not the uiFile property).

- bool getFileMonitoringIsEnabled ()

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

- void setHandleGuiLaunchRequests (bool handleGuiLaunchRequests)

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

- bool getHandleGuiLaunchRequests ()

  Set flag indicating form should handle gui form launch requests.

- void setResizeContents (bool resizeContentsIn)

  Get flag indicating form should handle gui form launch requests.

- bool getResizeContents ()

  Set flag indicating form should resize contents to match form size (otherwise resize form to match contents).

- QString getContainedFrameworkVersion ()

  Get flag indicating form should resize contents to match form size (otherwise resize form to match contents).

- QString getUniqueIdentifier ()

  Get the version of the first QE widget (if any) of QE widgets by QUILoader.

- void setUniqueIdentifier (QString name)

  Get a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.

- void clearUiFileNames ()

  Set a unique identifier string for this form. This identifier should be persistant across application runs as it is based on the QEForm's position in the widget hierarchy. The same widget will generate the same identifier when opened within the same GUI.

- int **getDisconnectedCount** ()
- int getConnectedCount ()

  Return the count of disconnected variables.

- QWidget * getChild (QString name)

  Return the count of connected variables.

- void **setUiFileNameProperty** (QString uiFileName)
- QString **getUiFileNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)

Protected Member Functions

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

Protected Attributes

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

Properties

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

- MessageFilterOptions messageSourceFilter
- bool variableAsToolTip
- bool allowDrop
- DisplayAlarmStateOptions displayAlarmStateOption

9.158.1 Member Data Documentation

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

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

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

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

9.158.2 Property Documentation

9.158.2.1 bool QEForm::allowDrop [read, write]

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

Reimplemented from QEAbstractWidget.

9.158.2.2 DisplayAlarmStateOptions QEForm::displayAlarmStateOption [read, write]

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

Reimplemented from QEAbstractWidget.

9.158.2.3 unsigned QEForm::int [read, write]

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

Reimplemented from QEAbstractWidget.

9.158.2.4 MessageFilterOptions QEForm::messageFormFilter [read, write]

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

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

9.158.2.5 MessageFilterOptions QEForm::messageSourceFilter [read, write]

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

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

9.158.2.6 QString QEForm::uiFile [read, write]

File name of the .ui file being presented within the QEForm widget.

9.158.2.7 bool QEForm::variableAsToolTip [read, write]

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

Reimplemented from QEAbstractWidget.

9.158.2.8 QString QEForm::variableSubstitutions [read, write]

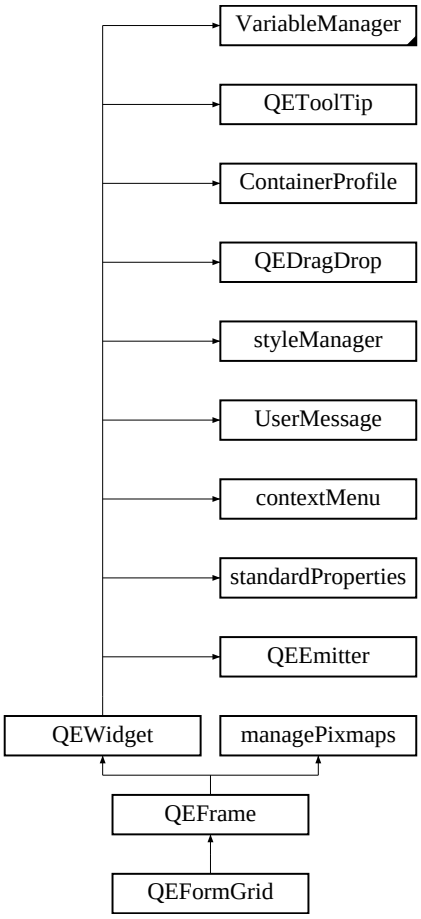
Macro substitutions to be applied to this widget, and all QE widgets that are opened when the .ui file is presented. Note, despite the name, the macro substitutions are general macro substitutions, and do not just apply to a variable name (in fact a QEForm widget does not even have a variable name property.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEForm/QEForm.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEForm.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEForm/QEForm.cpp

9.159 QEFormGrid Class Reference

`#include <QEFormGrid.h>`Inheritance diagram for QEFormGrid::



Classes

- class **MacroData**

Public Types

- enum GridOrders { **RowMajor** = 0, **ColMajor** = 1 }

Public Member Functions

- QEFormGrid (QWidget *parent=0)
- **QEFormGrid** (const QString &uiFile, const int number, const int cols, QWidget *parent=0)

- virtual ~QEFormGrid ()

  Destruction.

- void **setUiFile** (QString uiFileName)
- QString **getUiFile** ()
- void **setGridVariableSubstitutions** (QString variableSubstitutions)
- QString **getGridVariableSubstitutions** ()
- void **setNumber** (int n)
- int **getNumber** ()
- void **setColumns** (int n)
- int **getColumns** ()
- void **setGridOrder** (GridOrders go)
- GridOrders **getGridOrder** ()
- void **setMargin** (int n)
- int **getMargin** ()
- void **setSpacing** (int n)
- int **getSpacing** ()
- int **getRows** ()

Protected Member Functions

- QSize **sizeHint** () const

Properties

- QString uiFile
- QString **variableSubstitutions**
- int number
- int columns
- GridOrders gridOrder
- int margin
- int spacing
- QString slotMacroPrefix
- int slotNumberOffset
- int slotNumberWidth
- QStringList slotStrings
- QString rowMacroPrefix
- int rowNumberOffset
- int rowNumberWidth
- QStringList rowStrings
- QString colMacroPrefix
- int colNumberOffset
- int colNumberWidth
- QStringList colStrings
- bool variableAsToolTip
- bool allowDrop
- DisplayAlarmStateOptions displayAlarmStateOption

Friends

- class **MacroData**

9.159.1 Detailed Description

This class load a grid QEForms.

9.159.2 Member Enumeration Documentation

9.159.2.1 enum QEFormGrid::GridOrders

GridOrders specfies how grid elements are layed out.

Row major (example 10, items, 3 cols): 0 1 2 3 4 5 6 7 8 9 - -

Col major (example 10, items, 3 cols): 0 4 8 1 5 9 2 6 - 3 7 -

9.159.3 Constructor & Destructor Documentation

9.159.3.1 QEFormGrid::QEFormGrid (QWidget * *parent* = 0) [explicit]

Create a grid widget with default parameters.

9.159.4 Property Documentation

9.159.4.1 bool QEFormGrid::allowDrop [read, write]

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

Reimplemented from QEFrame.

9.159.4.2 QString QEFormGrid::colMacroPrefix [read, write]

Specified formal slot macro name prefix Default value: COL

9.159.4.3 int QEFormGrid::colNumberOffset [read, write]

Specifies the COL macro number offset. Default value: 1.

9.159.4.4 int QEFormGrid::colNumberWidth [read, write]

Specifies the COL macro width format. Default value: 2.

9.159.4.5 QStringList QEFormGrid::colStrings [read, write]

Specifies the COLNAME macro values applied to each QEFrom grid element in a spefic column. Indexing of the colStrings is NOT impacted by the colNumberOffset. Default value: ""

9.159.4.6 int QEFormGrid::columns [read, write]

Specifies the number of columns. This is restricted to the range 1 to 200. Default value: 1.

9.159.4.7 DisplayAlarmStateOptions QEFormGrid::displayAlarmStateOption [read, write]

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

Reimplemented from QEFrame.

9.159.4.8 GridOrders QEFormGrid::gridOrder [read, write]

Specifies the gridOrder. Default value: RowMajor

9.159.4.9 int QEFormGrid::margin [read, write]

Margin of the internal QGridLayout object. Default value: 2.

9.159.4.10 int QEFormGrid::number [read, write]

The total number of QEForms. This is restricted to the range 1 to 2000. Default value: 4.

Reimplemented from QEToolTip.

9.159.4.11 QString QEFormGrid::rowMacroPrefix [read, write]

Specified formal slot macro name prefix Default value: ROW

9.159.4.12 int QEFormGrid::rowNumberOffset [read, write]

Specifies the ROW macro number offset. Default value: 1.

9.159.4.13 int QEFormGrid::rowNumberWidth [read, write]

Specifies the COL macro width format. Default value: 2.

9.159.4.14 QStringList QEFormGrid::rowStrings [read, write]

Specifies the ROWNAME macro values applied to each QEFrom grid element in a spefic row. Indexing of the rowStrings is NOT impacted by the rowNumberOffset. Default value: ""

9.159.4.15 QString QEFormGrid::slotMacroPrefix [read, write]

Specified formal slot macro name prefix Default value: SLOT

9.159.4.16 int QEFormGrid::slotNumberOffset [read, write]

Specifies the SLOT macro number offset. Default value: 1.

9.159.4.17 int QEFormGrid::slotNumberWidth [read, write]

Specifies the SLOT macro width format. Default value: 2.

9.159.4.18 QStringList QEFormGrid::slotStrings [read, write]

Specifies the SLOTNAME macro values applied to each QEFrom grid element. Slot number allocation is impacted by the gridOrder property. Indexing of the slotStrings is NOT impacted by the slotNumberOffset. Default value: ""

9.159.4.19 int QEFormGrid::spacing [read, write]

Spaceing of the internal QGridLayout object. Default value: 2.

9.159.4.20 QString QEFormGrid::uiFile [read, write]

The uiFile loaded into each QEForm element. Default value: ""

9.159.4.21 bool QEFormGrid::variableAsToolTip [read, write]

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

Reimplemented from QEFrame.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFormGrid/QEFormGr
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFormGrid/QEFormGr

9.160 QEFormMapper Class Reference

`#include <QEFormMapper.h>`

Public Types

- typedef quint64 **FormHandles**

Static Public Member Functions

- static FormHandles **nullHandle** ()
- static FormHandles **allocateHandle** ()
- static QWidget * **findWidget** (const FormHandles &handle)
- static QWidget * **findWidget** (const FormHandles &handle, const QString &className)

Friends

- class QEMapable

9.160.1 Detailed Description

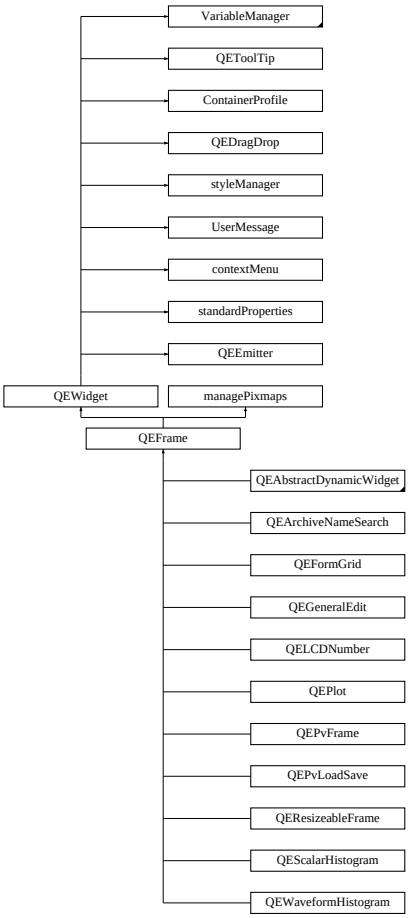
The QEFormMapper mapper class is a singleton class that allows a user allocated form handle to be associated with an QEForm, specifically the form handle is included in the action request (along with the ui filename, macros etc.) emitted and then received up by the QEGui (or other) display manager. When the QEForm object is created the user alllocated handle is associated with the QEForm object. This allows the originator of the request to find the QEForm that was created as a result of the request.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFormMapper.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFormMapper.cpp

9.161 QEFrame Class Reference

Inheritance diagram for QEFrame::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

- void **setSelectPixmap** (const int index)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

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

Protected Member Functions

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

Properties

- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet

- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- QPixmap pixmap
- bool scaledContents
- QPixmap pixmap0
- QPixmap pixmap1
- QPixmap pixmap2
- QPixmap pixmap3
- QPixmap pixmap4
- QPixmap pixmap5
- QPixmap pixmap6
- QPixmap pixmap7

9.161.1 Member Enumeration Documentation

9.161.1.1 enum QEFrame::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.161.1.2 enum QEFrame::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.161.2 Member Function Documentation

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

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

9.161.3 Property Documentation

9.161.3.1 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.

Reimplemented in QEFormGrid, and QEPlot.

9.161.3.2 QString QEFrame::defaultStyle [read, write]

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

9.161.3.3 bool QEFrame::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.161.3.4 DisplayAlarmStateOptions QEFrame::displayAlarmStateOption [read, write]

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

Reimplemented in QEFormGrid.

9.161.3.5 unsigned QEFrame::int [read, write]

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

9.161.3.6 QPixmap QEFrame::pixmap [read, write]

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

9.161.3.7 QPixmap QEFrame::pixmap0 [read, write]

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

9.161.3.8 QPixmap QEFrame::pixmap1 [read, write]

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

9.161.3.9 QPixmap QEFrame::pixmap2 [read, write]

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

9.161.3.10 QPixmap QEFrame::pixmap3 [read, write]

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

9.161.3.11 QPixmap QEFrame::pixmap4 [read, write]

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

9.161.3.12 QPixmap QEFrame::pixmap5 [read, write]

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

9.161.3.13 QPixmap QEFrame::pixmap6 [read, write]

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

9.161.3.14 QPixmap QEFrame::pixmap7 [read, write]

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

9.161.3.15 bool QEFrame::scaledContents [read, write]

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

9.161.3.16 QString QEFrame::styleSheet [read, write]

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

9.161.3.17 UserLevels QEFrame::userLevelEnabled [read, write]

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

9.161.3.18 QString QEFrame::userLevelEngineerStyle [read, write]

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

9.161.3.19 QString QEFrame::userLevelScientistStyle [read, write]

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

will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.161.3.20 QString QEFrame::userLevelUserStyle [read, write]

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

9.161.3.21 UserLevels QEFrame::userLevelVisibility [read, write]

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

9.161.3.22 bool QEFrame::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

Reimplemented in QEFormGrid.

9.161.3.23 bool QEFrame::visible [read, write]

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFrame/QEFrame.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFrame/QEFrame.cpp

9.162 QEFrameworkVersion Class Reference

Static Public Member Functions

- static unsigned int **getMajor** ()
- static unsigned int **getMinor** ()
- static unsigned int **getRelease** ()
- static const QString **getStage** ()
- static const QString **getString** ()
- static const QString **getDateTime** ()
- static const QString **getAttributes** ()
- static const QString **getQtVersionStr** ()
- static const QString **getQwtVersionStr** ()
- static const QString **getEpicsVersionStr** ()

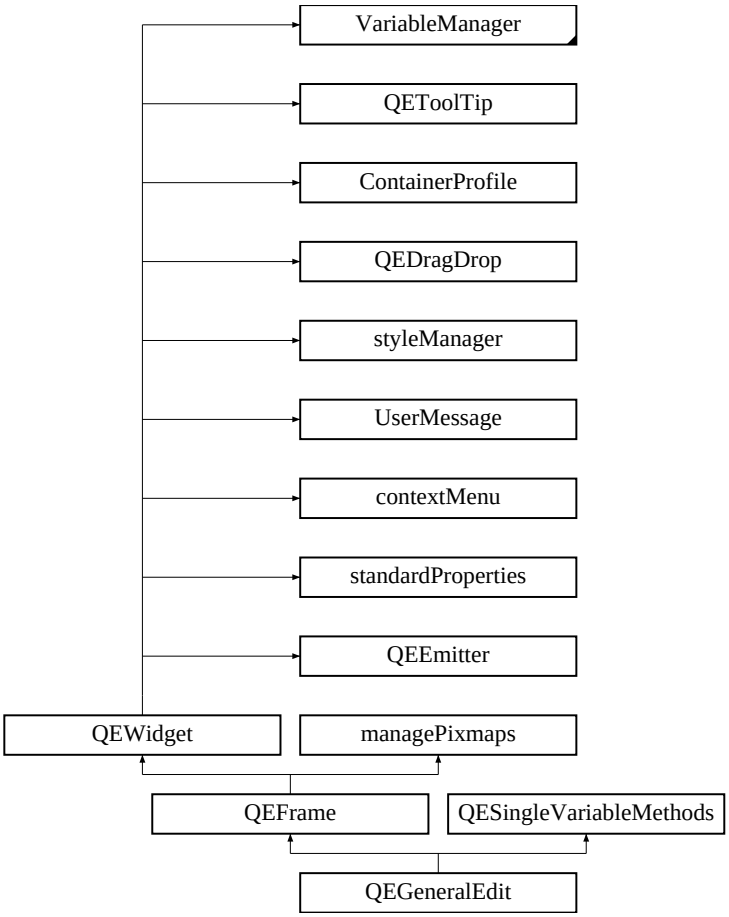
The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFrameworkVersion.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFrameworkVersion.cpp

9.163 QEGeneralEdit Class Reference

The QEGeneralEdit class This class provides a general PV edit widget, presenting one off a QELineEdit, a QENumericEdit or a QERadioGroup for string, numerical and enumeration data kinds respectively.

`#include <QEGeneralEdit.h>`Inheritance diagram for QEGeneralEdit::



Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- QEGeneralEdit (QWidget ∗parent=0)
- QEGeneralEdit (const QString &variableName, QWidget ∗parent=0)
- virtual ∼QEGeneralEdit ()

  Destruction.

- void setArrayIndex (const int arrayIndex)
- void **setButtonStyle** (const QRadioGroup::ButtonStyles style)
- QRadioGroup::ButtonStyles **getButtonStyle** () const
- void **setButtonOrder** (const QRadioGroup::ButtonOrders order)
- QRadioGroup::ButtonOrders **getButtonOrder** () const
- void **setUseApplyButton** (const bool useApplyButton)
- bool **getUseApplyButton** () const

Protected Member Functions

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

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject ∗ createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- QRadioGroup::ButtonStyles buttonStyle
- QRadioGroup::ButtonOrders buttonOrder
- bool useApplyButton

9.163.1 Detailed Description

The QEGeneralEdit class This class provides a general PV edit widget, presenting one off a QELineEdit, a QENumericEdit or a QERadioGroup for string, numerical and enumeration data kinds respectively. This widget is intented for use within a qegui predefined form, displayed in response to context menu request to edit an arbitary PV.

9.163.2 Constructor & Destructor Documentation

9.163.2.1 QEGeneralEdit::QEGeneralEdit (QWidget * *parent* = 0) [explicit]

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

9.163.2.2 QEGeneralEdit::QEGeneralEdit (const QString & *variableName*, QWidget * *parent* = 0) [explicit]

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.

9.163.3 Member Function Documentation

9.163.3.1 void QEGeneralEdit::dbConnectionChanged (const bool & *isConnected*) [signal]

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

9.163.3.2 void QEGeneralEdit::dbValueChanged () [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.

9.163.3.3 void QEGeneralEdit::setArrayIndex (const int *arrayIndex*)

Property access function for arrayIndex property. Array element to access if variable is an array variable. Defaults to 0, i.e. first element. arrayIndex value is restricted to be >= 0

If the assocated qcaobject::QCaObject exists then calls its setArrayIndex function and then requests that the object resend last data. The function adjusts the elementsRequired property value if necessary.

Reimplemented from QESingleVariableMethods.

9.163.4 Property Documentation

9.163.4.1 int QEGeneralEdit::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.163.4.2 QRadioGroup::ButtonOrders QEGeneralEdit::buttonOrder [read, write]

Button order for enumerations edits.

9.163.4.3 QRadioGroup::ButtonStyles QEGeneralEdit::buttonStyle [read, write]

Button style for enumerations edits.

9.163.4.4 int QEGeneralEdit::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.163.4.5 bool QEGeneralEdit::useApplyButton [read, write]

For numeric and string edits, the cnage can be applied on enter/lose focus, or applied via explicity clicking an apply button. Setting this proerty false uses the former, true the latter. The default property value is false.

9.163.4.6 QString QEGeneralEdit::variable [read, write]

EPICS variable name (CA PV)

9.163.4.7 QString QEGeneralEdit::variableSubstitutions [read, write]

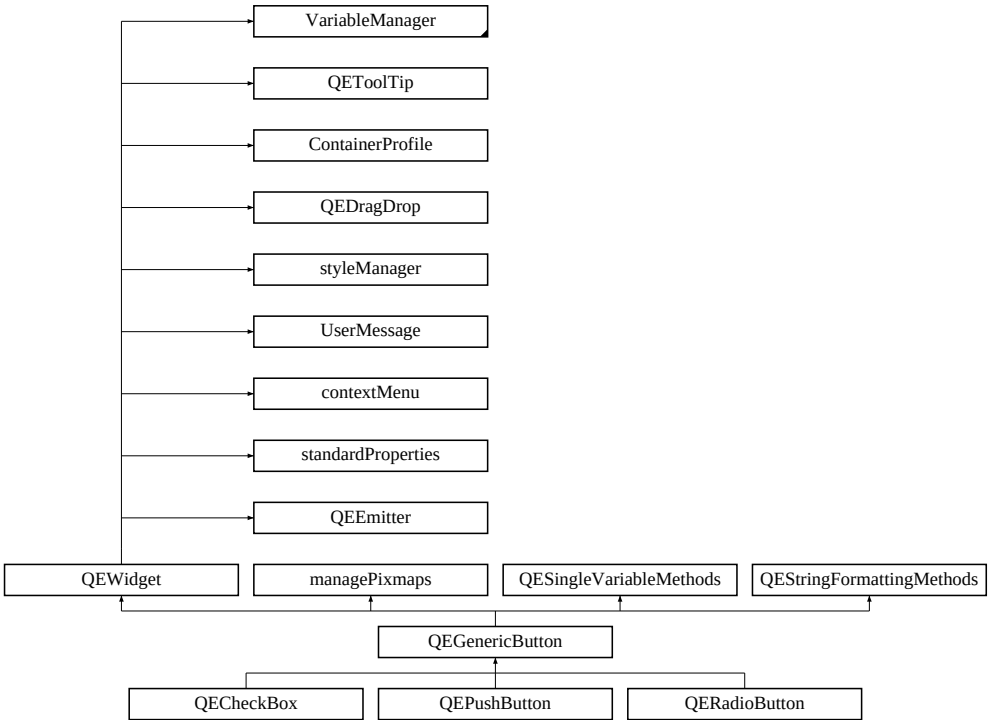
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEGeneralEdit/QEGeneralEdit.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEGeneralEdit.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEGeneralEdit/QEGeneralEdit.cpp

9.164 QEGenericButton Class Reference

Inheritance diagram for QEGenericButton::



Public Types

- enum **VariableAllocation** {

  VAR_PRIMARY = 0, **VAR_READBACK**, **VAR_DISA**, **VAR_DISV**,

  NUMBER_OF_VARIABLES }
- enum **updateOptions** {

  UPDATE_NONE = 0x00, **UPDATE_TEXT** = 0x01, **UPDATE_ICON** = 0x02, **UPDATE_TEXT_AND_ICON** = 0x03,

  UPDATE_STATE = 0x04, **UPDATE_TEXT_AND_STATE** = 0x05, **UPDATE_ICON_AND_STATE** = 0x06, **UPDATE_TEXT_ICON_AND_-STATE** = 0x07 }

Public Member Functions

- **QEGenericButton** (QAbstractButton ∗owner)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setUpdateOption** (updateOptions updateOptionIn)

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

- void **writeClickedNow** (const bool checked=false)

Protected Member Functions

- void **useGenericNewVariableName** (const QString &variableName, const QString &variableNameSubstitutions, const unsigned int variableIndex)
- void **connectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **setGenericButtonText** (const QString &text, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **setGenericDISAvalue** (const long &value, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **setGenericDISVvalue** (const long &value, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void **userPressed** ()
- void **userReleased** ()
- void **userClicked** (bool checked)
- void **processWriteNow** (const bool checked)
- virtual updateOptions **getDefaultUpdateOption** ()=0
- void **setup** ()
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **calcStyleOption** ()

Protected Attributes

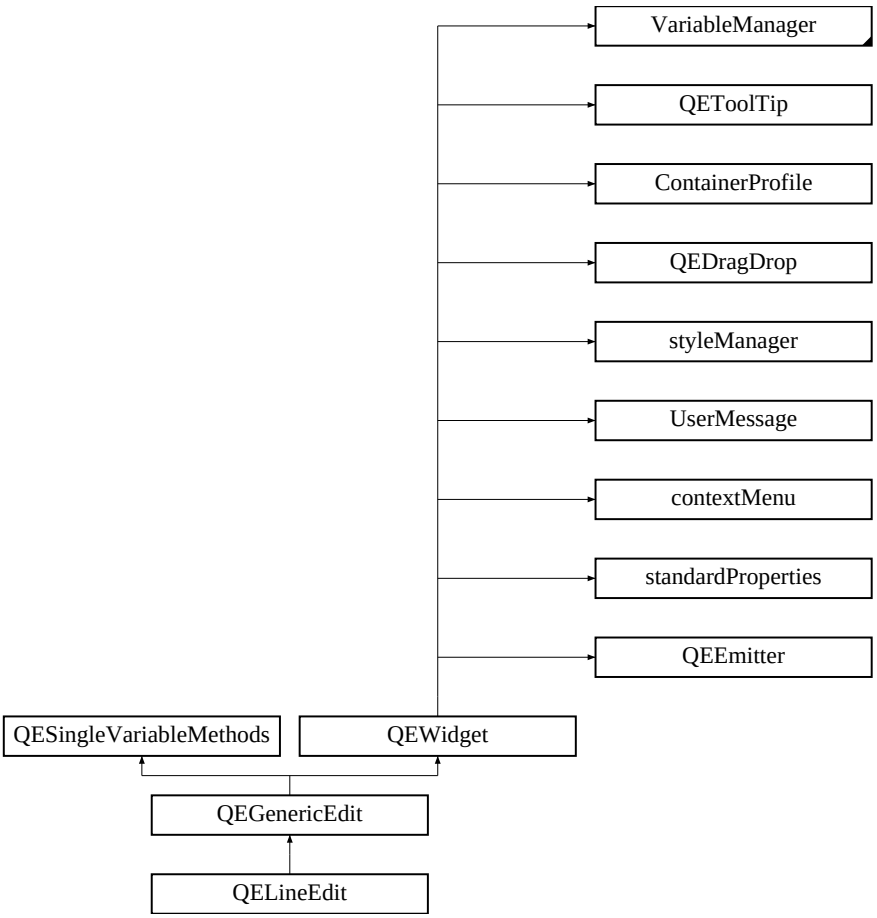
- applicationLauncher **programLauncher**
- QESingleVariableMethods * **altReadback**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QEGenericBut
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QEGenericBut

9.165 QEGenericEdit Class Reference

Inheritance diagram for QEGenericEdit::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

Signals

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

  Internal use only. Used by QEConfiguredLayout to be notified when one of its widgets has written something.

- void requestResend ()

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

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

- QEGenericEdit (QWidget *parent=0)
- QEGenericEdit (const QString &variableName, QWidget *parent=0)
- void setWriteOnLoseFocus (bool writeOnLoseFocus)
- bool getWriteOnLoseFocus ()
- void setWriteOnEnter (bool writeOnEnter)

- bool getWriteOnEnter ()
- void setWriteOnFinish (bool writeOnFinish)
- bool getWriteOnFinish ()
- void setConfirmWrite (bool confirmWrite)
- bool getConfirmWrite ()
- void setAllowFocusUpdate (bool allowFocusUpdate)
- bool getAllowFocusUpdate () const

  Returns 'true' if this widget configured to allow updates while it has focus.

- void setSubscribe (bool subscribe)
- bool getSubscribe ()
- void **writeValue** (qcaobject::QCaObject *qca, QVariant newValue)
- void writeNow ()

Protected Member Functions

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

Protected Attributes

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

Properties

- QString **text**
- QString variable
- QString variableSubstitutions
- int arrayIndex
- bool subscribe
- bool writeOnLoseFocus
- bool writeOnEnter
- bool writeOnFinish
- bool confirmWrite
- bool allowFocusUpdate

  Allow updated while widget has focus - defaults to false.

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

9.165.1 Member Enumeration Documentation

9.165.1.1 enum QEGenericEdit::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.165.1.2 enum QEGenericEdit::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.165.2 Constructor & Destructor Documentation

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

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

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

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

9.165.3 Member Function Documentation

9.165.3.1 bool QEGenericEdit::getConfirmWrite ()

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

9.165.3.2 bool QEGenericEdit::getSubscribe ()

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

9.165.3.3 bool QEGenericEdit::getWriteOnEnter ()

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

9.165.3.4 bool QEGenericEdit::getWriteOnFinish ()

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

9.165.3.5 bool QEGenericEdit::getWriteOnLoseFocus ()

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

9.165.3.6 void QEGenericEdit::setAllowFocusUpdate (bool *allowFocusUpdate*)

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

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

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

9.165.3.8 void QEGenericEdit::setManagedVisible (bool *v*) [inline, slot]

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

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

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

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

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

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

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

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

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

9.165.3.13 void QEGenericEdit::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.165.4 Property Documentation

9.165.4.1 bool QEGenericEdit::allowDrop [read, write]

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

Reimplemented from QEDragDrop.

9.165.4.2 int QEGenericEdit::arrayIndex [read, write]

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

Reimplemented from QESingleVariableMethods.

9.165.4.3 bool QEGenericEdit::confirmWrite [read, write]

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

9.165.4.4 QString QEGenericEdit::defaultStyle [read, write]

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

9.165.4.5 bool QEGenericEdit::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.165.4.6 DisplayAlarmStateOptions QEGenericEdit::displayAlarmStateOption [read, write]

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

9.165.4.7 unsigned QEGenericEdit::int [read, write]

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

9.165.4.8 QString QEGenericEdit::styleSheet [read, write]

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

9.165.4.9 bool QEGenericEdit::subscribe [read, write]

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

Reimplemented from VariableManager.

9.165.4.10 UserLevels QEGenericEdit::userLevelEnabled [read, write]

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

9.165.4.11 QString QEGenericEdit::userLevelEngineerStyle [read, write]

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

9.165.4.12 QString QEGenericEdit::userLevelScientistStyle [read, write]

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

9.165.4.13 QString QEGenericEdit::userLevelUserStyle [read, write]

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

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

9.165.4.14 UserLevels QEGenericEdit::userLevelVisibility [read, write]

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

9.165.4.15 QString QEGenericEdit::variable [read, write]

EPICS variable name (CA PV)

9.165.4.16 bool QEGenericEdit::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

9.165.4.17 QString QEGenericEdit::variableSubstitutions [read, write]

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

9.165.4.18 bool QEGenericEdit::visible [read, write]

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

9.165.4.19 bool QEGenericEdit::writeOnEnter [read, write]

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

9.165.4.20 bool QEGenericEdit::writeOnFinish [read, write]

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

9.165.4.21 bool QEGenericEdit::writeOnLoseFocus [read, write]

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELineEdit/QEGenericE
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEGenericEdit.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELineEdit/QEGenericE

9.166 QEGlobalStyle Class Reference

`#include <QEGlobalStyle.h>`

Public Member Functions

- QEGlobalStyle ()
- ~QEGlobalStyle ()
- void apply ()

9.166.1 Detailed Description

This class sets the QApplication instancse style sheet, and hence is a global style that is applicable to all widgets. As this functionality is invoked from the QEWidget constructor, this style applies equally within designer, qegui or any other display mangers using at least one QEWidget.

9.166.2 Constructor & Destructor Documentation

9.166.2.1 QEGlobalStyle::QEGlobalStyle ()

Constructor

9.166.2.2 QEGlobalStyle::~QEGlobalStyle ()

Destructor

9.166.3 Member Function Documentation

9.166.3.1 void QEGlobalStyle::apply ()

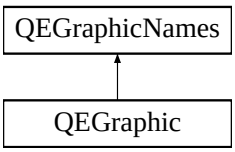
Checks for the existance of the QE_GLOBAL_STYLE_SHEET environment variable and if defined appends the specified style to the application style sheet. If the (trimmed) value of the environment variable starts with "file://" then the rest of the environment variable will be interpreted as file name path.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEGlobalStyle.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEGlobalStyle.cpp

9.167 QEGraphic Class Reference

`#include <QEGraphic.h>`Inheritance diagram for QEGraphic::



Classes

- class **Axis**
- class OwnPlot
- struct **TextItems**

Signals

- void **mouseMove** (const QPointF &posn)
- void **wheelRotate** (const QPointF &posn, const int delta)
- void **areaDefinition** (const QPointF &from, const QPointF &to)
- void **lineDefinition** (const QPointF &from, const QPointF &to)
- void **crosshairsMove** (const QPointF &posn)
- void **markupMove** (const QEGraphicNames::Markups markup, const QPointF &position)

Public Member Functions

- **QEGraphic** (QWidget ∗parent=0)
- **QEGraphic** (const QString &title, QWidget ∗parent=0)
- void **releaseCurves** ()
- void **attchOwnCurve** (QwtPlotCurve ∗curve)
- void **setBackgroundColour** (const QColor colour)
- void **setGridPen** (const QPen &pen)
- void **setAvailableMarkups** (const MarkupFlags graphicMarkupsSet)
- MarkupFlags **getAvailableMarkups** () const
- void **setMarkupVisible** (const Markups markup, const bool isVisible)
- bool **getMarkupVisible** (const Markups markup) const
- void **setMarkupEnabled** (const Markups markup, const bool isEnabled)
- bool **getMarkupEnabled** (const Markups markup) const
- void **setMarkupSelected** (const Markups markup, const bool selected)
- bool **getMarkupIsSelected** (const Markups markup) const
- void **setMarkupPosition** (const Markups markup, const QPointF &position)
- QPointF **getMarkupPosition** (const Markups markup) const
- void **setMarkupData** (const Markups markup, const QVariant &text)

- QVariant **getMarkupData** (const Markups markup) const
- Q_DECL_DEPRECATED void setCrosshairsVisible (const bool isVisible)

  Depricated.

- Q_DECL_DEPRECATED void setCrosshairsVisible (const bool isVisible, const QPointF &position)

  Depricated.

- Q_DECL_DEPRECATED bool getCrosshairsVisible () const

  Depricated.

- void plotCurveData (const DoubleVector &xData, const DoubleVector &yData, const QwtPlot::Axis yAxis=QwtPlot::yLeft)
- void drawText (const QPointF &posn, const QString &text, const TextPositions option, bool isCentred=true)
- void **drawText** (const QPoint &posn, const QString &text, const TextPositions option, bool isCentred=true)
- void **setXRange** (const double min, const double max, const AxisMajorIntervalModes mode, const double value, const bool immediate)
- void getXRange (double &min, double &max) const
- void **setYRange** (const double min, const double max, const AxisMajorIntervalModes mode, const double value, const bool immediate, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)
- void getYRange (double &min, double &max, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void **replot** ()
- Q_DECL_DEPRECATED bool **rightButtonPressed** () const
- QPointF **getRealMousePosition** () const
- bool **globalPosIsOverCanvas** (const QPoint &golbalPos) const
- bool **getSlopeIsDefined** (QPointF &slope) const
- void **setAxisEnableX** (const bool enable)
- bool **getAxisEnableX** () const
- void setAxisEnableY (const bool enable, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)
- bool getAxisEnableY (const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void **setXScale** (const double scale)
- double **getXScale** () const
- void **setXOffset** (const double offset)
- double **getXOffset** () const
- void **setXLogarithmic** (const bool isLog)
- bool **getXLogarithmic** () const
- void setYScale (const double scale, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)
- double getYScale (const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void setYOffset (const double offset, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)

- double getYOffset (const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void setYLogarithmic (const bool isLog, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)
- bool getYLogarithmic (const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void setYColor (const QColor color, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft)
- void **setCurvePen** (const QPen &pen)
- QPen **getCurvePen** () const
- void **setCurveBrush** (const QBrush &brush)
- QBrush **getCurveBrush** () const
- void **setTextFont** (const QFont &font)
- QFont **getTextFont** () const
- void **setTextPointSize** (const int pointSize)
- int **getTextPointSize** () const
- void **setCurveRenderHint** (const QwtPlotItem::RenderHint hint, const bool on=true)
- QwtPlotItem::RenderHint **getCurveRenderHint** () const
- bool **getCurveRenderHintOn** () const
- void **setCurveStyle** (const QwtPlotCurve::CurveStyle style)
- QwtPlotCurve::CurveStyle **getCurveStyle** ()
- QPoint **pixelDistance** (const QPointF &from, const QPointF &to) const
- QPointF **realOffset** (const QPoint &offset, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- QPointF **realOffset** (const QPointF &offset, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- QPointF **pointToReal** (const QPoint &pos, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- QPointF **pointToReal** (const QPointF &pos, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- QPoint **realToPoint** (const QPointF &pos, const QwtPlot::Axis selectedYAxis=QwtPlot::yLeft) const
- void **enableAxis** (int axisId, bool tf=true)
- void **setAxisScale** (int axisId, double min, double max, double step=0)
- void **installCanvasEventFilter** (QObject *filter)
- bool **isCanvasObject** (QObject *obj) const
- QRect **getEmbeddedCanvasGeometry** () const
- QwtPlot * **getEmbeddedQwtPlot** () const

Protected Member Functions

- void **canvasMousePress** (QMouseEvent *mouseEvent)
- void **canvasMouseRelease** (QMouseEvent *mouseEvent)
- void **canvasMouseMove** (QMouseEvent *mouseEvent, const bool isButtonAction)
- bool **eventFilter** (QObject *obj, QEvent *event)

Friends

- class **OwnPlot**
- class QEGraphicMarkup
- class QEGraphicAreaMarkup
- class QEGraphicLineMarkup
- class QEGraphicCrosshairsMarkup
- class QEGraphicHorizontalMarkup
- class QEGraphicVerticalMarkup

9.167.1 Detailed Description

Provides a basic wrapper around QwtPlot, which:

a) Allocates and attaches curves and grids, and releases these on delete, and releases curves on request;

b) Interprets mouse events with real world co-ordinates;

c) Provides a log scale mode (X and/or Y);

d) Provides a consistant means to round down/up min/max values and the selection of a major interval value, e.g.: 2.1 .. 7.83 (user min/max) => 2.0 .. 8.0, 1.0 (display min/max, major);

e) Standardised mouse and wheel zoomimg;

f) Smart axis re-scaling;

g) Provides markups; and

h) Provides wrapper functions to hide QWT version API changes.

9.167.2 Member Function Documentation

9.167.2.1 void QEGraphic::drawText (const QPointF & *posn*, const QString & *text*, const TextPositions *option*, bool *isCentred* = `true`)

Draw text at position specifying centre of bottom left corner. Position may be real world coordinates or pixel coordates.

Parameters:

posn The postion to draw text

text The text to draw

option Specifies if position is in cavas pixels or real world co-ordinates

pointSize Defines text point size. Honors QEScaling parameters.

isCentred When true text is centered about point, otherwise point defines bottom left corner of the generated text

9.167.2.2 bool QEGraphic::getAxisEnableY (const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`) const

Is left/right Y axis enabled/disabled

Parameters:

selectedYAxis Left/right Y axis. If not provided left axis is used by default

Returns:

Boolean signaling if axis is enabled/disabled

9.167.2.3 void QEGraphic::getXRange (double & *min*, double & *max*) const

Returns the range of X axis

Parameters:

min Minimum X value

max Maximum X value

9.167.2.4 bool QEGraphic::getYLogarithmic (const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`) const

Is the scale set to be logarithmic or not on left/right axis

Parameters:

selectedYAxis Left/right Y axis. If not provided left axis is used by default

Returns:

Boolean flag defining if the scale set is logarithmic

9.167.2.5 double QEGraphic::getYOffset (const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`) const

Returns the offset set on left/right Y axis.

Parameters:

selectedYAxis Left/right Y axis. If not provided left axis is used by default.

Returns:

Offset set

9.167.2.6 void QEGraphic::getYRange (double & *min*, double & *max*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`) const

Returns the rance of either left or right Y axis

Parameters:

min Reference to variable to be used to store minimum value

max Reference to variable to be used to store maximumSize() value

selectedYAxis Range for which Y axis is required. If not provided left axis is used by default

9.167.2.7 double QEGraphic::getYScale (const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`) const

Returns the scale set on left/right Y axis.

Parameters:

selectedYAxis Left/right Y axis. If not provided left axis is used by default.

Returns:

Scale set

9.167.2.8 void QEGraphic::plotCurveData (const DoubleVector & *xData*, const DoubleVector & *yData*, const QwtPlot::Axis *yAxis* = `QwtPlot::yLeft`)

Allocates a curve, sets current curve attibutes and attaches to plot.

Parameters:

xData Vector of X axis values

yData Vector of Y axis values

yAxis Which Y axis should the yData be plotted against. If not provided left axis is used by default

9.167.2.9 void QEGraphic::setAxisEnableY (const bool *enable*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`)

Enables or disables left or right Y axis

Parameters:

enable Enable/disable Y axis

selectedYAxis Left/right Y axis. If not provided left axis is used by default

9.167.2.10 void QEGraphic::setYColor (const QColor *color*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`)

Sets the color of left/right Y axis

Parameters:

color Color to be set

selectedYAxis Left/right Y axis. If not provided left axis is used by default

9.167.2.11 void QEGraphic::setYLogarithmic (const bool *isLog*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`)

Sets the scale on Y axis to be logarithmic or not

Parameters:

isLog Boolean flag defining if the scale is to be logarithmic

selectedYAxis Left/right Y axis. If not provided left axis is used by default

9.167.2.12 void QEGraphic::setYOffset (const double *offset*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`)

Sets offset on either left or right Y axis

Parameters:

offset Offset to be set

selectedYAxis Left/right Y axis. If not provided left axis is used by default

9.167.2.13 void QEGraphic::setYScale (const double *scale*, const QwtPlot::Axis *selectedYAxis* = `QwtPlot::yLeft`)

Sets scale on either left or right Y axis

Parameters:

scale Scale to be set

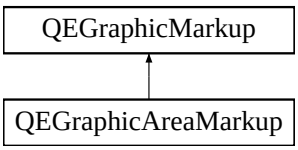
selectedYAxis Left/right Y axis. If not provided left axis is used by default

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphic.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphic.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEGraphic.cpp

9.168 QEGraphicAreaMarkup Class Reference

Inheritance diagram for QEGraphicAreaMarkup::



Public Member Functions

- **QEGraphicAreaMarkup** (QEGraphic ∗owner)
- void **mousePress** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseRelease** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseMove** (const QPointF &realMousePosition)

Protected Member Functions

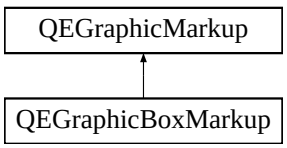
- void **plotMarkup** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.169 QEGraphicBoxMarkup Class Reference

Inheritance diagram for QEGraphicBoxMarkup::



Public Member Functions

- **QEGraphicBoxMarkup** (QEGraphic *owner)
- bool **isOver** (const QPointF &point, int &distance) const
- void **setSelected** (const bool selected)

Protected Member Functions

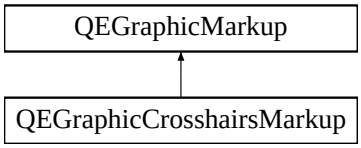
- void **plotMarkup** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.170 QEGraphicCrosshairsMarkup Class Reference

Inheritance diagram for QEGraphicCrosshairsMarkup::



Public Member Functions

- **QEGraphicCrosshairsMarkup** (QEGraphic ∗owner)
- void **setVisible** (const bool visible)
- bool **isOver** (const QPointF &point, int &distance) const
- void **mousePress** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseRelease** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseMove** (const QPointF &realMousePosition)

Protected Member Functions

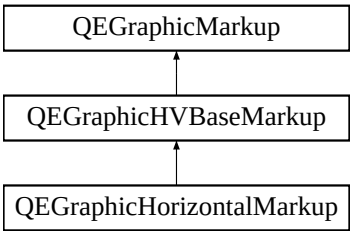
- void **plotMarkup** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.171 QEGraphicHorizontalMarkup Class Reference

Inheritance diagram for QEGraphicHorizontalMarkup::



Public Member Functions

- **QEGraphicHorizontalMarkup** (const QEGraphicNames::Markups markup, QEGraphic ∗owner)
- bool **isOver** (const QPointF &point, int &distance) const
- void **relocate** ()

Protected Member Functions

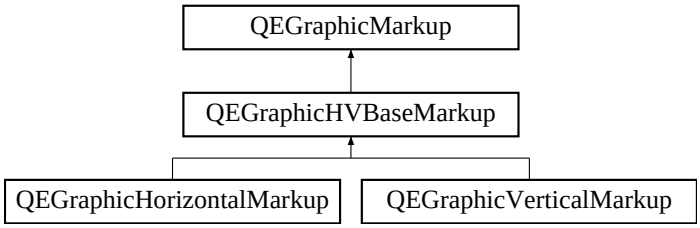
- void **getLine** (double &xmin, double &xmax, double &ymin, double &ymax)
- void **getShape** (QPoint shape[ ])

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.172 QEGraphicHVBaseMarkup Class Reference

Inheritance diagram for QEGraphicHVBaseMarkup::



Public Member Functions

- **QEGraphicHVBaseMarkup** (QEGraphicNames::Markups markup, QEGraphic ∗owner)
- void **setInUse** (const bool inUse)
- void **mousePress** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseRelease** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseMove** (const QPointF &realMousePosition)

Protected Member Functions

- void **setColours** (const unsigned int baseRGB)
- void **plotMarkup** ()
- virtual void **getLine** (double &xmin, double &xmax, double &ymin, double &ymax)=0
- virtual void **getShape** (QPoint shape[ ])=0

Protected Attributes

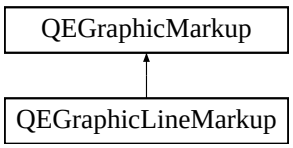
- QColor **brushDisabled**
- QColor **brushEnabled**
- QColor **brushSelected**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.173 QEGraphicLineMarkup Class Reference

Inheritance diagram for QEGraphicLineMarkup::



Public Member Functions

- **QEGraphicLineMarkup** (QEGraphic ∗owner)
- QPointF **getSlope** () const
- void **mousePress** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseRelease** (const QPointF &realMousePosition, const Qt::MouseButton button)
- void **mouseMove** (const QPointF &realMousePosition)

Protected Member Functions

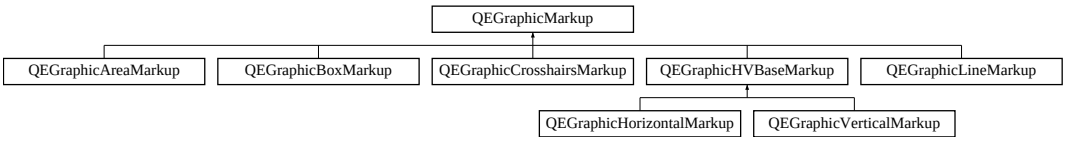
- void **plotMarkup** ()

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.174 QEGraphicMarkup Class Reference

Inheritance diagram for QEGraphicMarkup::



Public Member Functions

- **QEGraphicMarkup** (const QEGraphicNames::Markups markup, QEGraphic ∗owner)
- QEGraphicNames::Markups **getMarkup** () const
- void **setCurrentPosition** (const QPointF &currentPosition)
- QPointF **getCurrentPosition** () const
- void **setData** (const QVariant &data)
- QVariant **getData** () const
- virtual void **setInUse** (const bool inUse)
- virtual bool **isInUse** () const
- virtual void **setVisible** (const bool visible)
- virtual bool **isVisible** () const
- virtual void **setEnabled** (const bool enabled)
- virtual bool **isEnabled** () const
- virtual void **setSelected** (const bool selected)
- virtual bool **isSelected** () const
- virtual void **mousePress** (const QPointF &realMousePosition, const Qt::MouseButton button)
- virtual void **mouseRelease** (const QPointF &realMousePosition, const Qt::MouseButton button)
- virtual void **mouseMove** (const QPointF &realMousePosition)
- virtual QCursor **getCursor** () const
- virtual bool **isOver** (const QPointF &point, int &distance) const
- virtual void **relocate** ()
- void **plot** ()

Protected Member Functions

- QEGraphic ∗ **getOwner** () const
- bool **isOverHere** (const QPointF &here, const QPointF &point, int &distance) const
- void **plotCurve** (const QEGraphicNames::DoubleVector &xData, const QEGraphicNames::DoubleVector &yData)
- QFontMetrics **getFontMetrics** ()
- void **emitCurrentPostion** ()
- virtual void **plotMarkup** ()=0

Protected Attributes

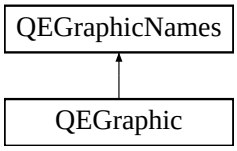
- QPointF **positon**
- QVariant **data**
- QPen **pen**
- QBrush **brush**
- QwtPlotCurve::CurveStyle **curveStyle**
- QCursor **cursor**
- Qt::MouseButton **activationButton**
- bool **inUse**
- bool **visible**
- bool **enabled**
- bool **selected**
- const QEGraphicNames::Markups **markup**
- QEGraphic * **owner**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.175 QEGraphicNames Class Reference

Inheritance diagram for QEGraphicNames::



Public Types

- enum **TextPositions** { **RealWorldPosition**, **PixelPosition** }
- enum **AxisMajorIntervalModes** { **SelectByValue**, **SelectBySize**, **UserInterval** }
- enum **Markups** {

  None = 0x0000, **Area** = 0x0001, **Line** = 0x0002, **CrossHair** = 0x0004,

  HorizontalLine_1 = 0x0010, **HorizontalLine_2** = 0x0020, **HorizontalLine_3** = 0x0040, **HorizontalLine_4** = 0x0080,

  VerticalLine_1 = 0x0100, **VerticalLine_2** = 0x0200, **VerticalLine_3** = 0x0400, **VerticalLine_4** = 0x0800,

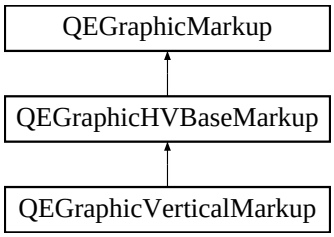
  Box = 0x1000 }
- typedef QVector< double > **DoubleVector**

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicNames.h

9.176 QEGraphicVerticalMarkup Class Reference

Inheritance diagram for QEGraphicVerticalMarkup::



Public Member Functions

- **QEGraphicVerticalMarkup** (const QEGraphicNames::Markups markup, QEGraphic ∗owner)
- bool **isOver** (const QPointF &point, int &distance) const
- void **relocate** ()

Protected Member Functions

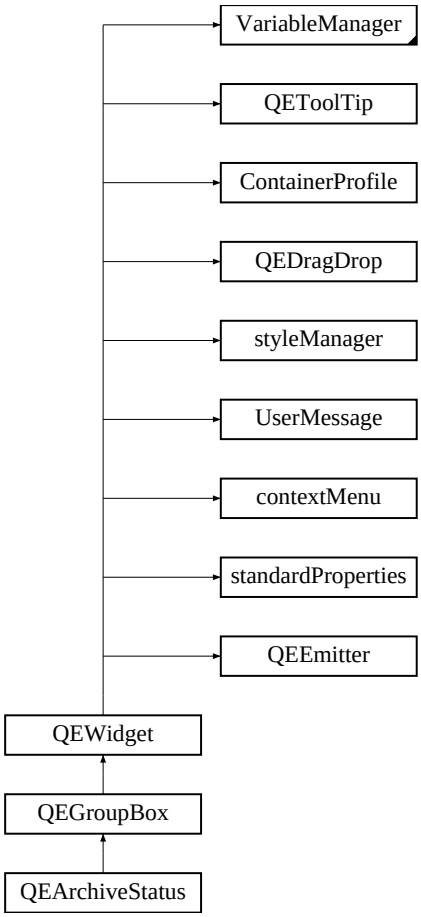
- void **getLine** (double &xmin, double &xmax, double &ymin, double &ymax)
- void **getShape** (QPoint shape[ ])

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEGraphicMarkup.cpp

9.177 QEGroupBox Class Reference

Inheritance diagram for QEGroupBox::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

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

- void setUserLevelVisibilityProperty (UserLevels level)

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

- UserLevels getUserLevelEnabledProperty ()

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

- void setUserLevelEnabledProperty (UserLevels level)

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

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

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

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

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

- **QEGroupBox** (QWidget *parent=0)
- **QEGroupBox** (const QString &title, QWidget *parent=0)
- QSize **sizeHint** () const

Protected Member Functions

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

Properties

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

- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- QString substitutedTitle
- QString textSubstitutions

9.177.1 Member Enumeration Documentation

9.177.1.1 enum QEGroupBox::DisplayAlarmStateOptions

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

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.177.1.2 enum QEGroupBox::UserLevels

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

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.177.2 Member Function Documentation

9.177.2.1 void QEGroupBox::setManagedVisible (bool *v*) `[inline, slot]`

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

9.177.3 Property Documentation

9.177.3.1 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.

9.177.3.2 QString QEGroupBox::defaultStyle [read, write]

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

9.177.3.3 bool QEGroupBox::displayAlarmState [read, write]

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

Reimplemented from standardProperties.

9.177.3.4 DisplayAlarmStateOptions QEGroupBox::displayAlarmStateOption [read, write]

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

9.177.3.5 unsigned QEGroupBox::int [read, write]

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

9.177.3.6 QString QEGroupBox::styleSheet [read, write]

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

9.177.3.7 QString QEGroupBox::substitutedTitle [read, write]

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

9.177.3.8 QString QEGroupBox::textSubstitutions [read, write]

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

9.177.3.9 UserLevels QEGroupBox::userLevelEnabled [read, write]

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

9.177.3.10 QString QEGroupBox::userLevelEngineerStyle [read, write]

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

9.177.3.11 QString QEGroupBox::userLevelScientistStyle [read, write]

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

9.177.3.12 QString QEGroupBox::userLevelUserStyle [read, write]

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

9.177.3.13 UserLevels QEGroupBox::userLevelVisibility [read, write]

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

mode is set application wide through the QELogin widget, or 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'.

9.177.3.14 bool QEGroupBox::variableAsToolTip [read, write]

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

Reimplemented from QEToolTip.

9.177.3.15 bool QEGroupBox::visible [read, write]

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

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEGroupBox/QEGroupB
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEGroupBox/QEGroupB

9.178 QEHistogram Class Reference

`#include <QEHistogram.h>`

Public Types

- typedef QVector< double > **DataArray**

Public Slots

- void **setColour** (const int index, const QColor &value)
- void **setValue** (const int index, const double value)
- void **setValues** (const DataArray &values)

Signals

- void **mouseIndexChanged** (const int index)
- void **mouseIndexPressed** (const int index, const Qt::MouseButton button)

Public Member Functions

- **QEHistogram** (QWidget *parent=0)
- virtual QSize **sizeHint** () const
- int **count** () const
- int **indexOfPosition** (const int x, const int y) const
- int **indexOfPosition** (const QPoint &p) const
- QRect **positionOfIndex** (const int index) const
- void **clearValue** (const int index)
- void **clearColour** (const int index)
- void **clear** ()
- double **value** (const int index) const
- DataArray **values** () const

Protected Member Functions

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

Properties

- bool **autoBarGapWidths**
- int **barWidth**
- int **gap**
- int **margin**
- bool **autoScale**

- double **minimum**
- double **maximum**
- double **baseLine**
- bool **drawAxies**
- bool **showScale**
- int **precision**
- bool showGrid

  This is the value grid selection.

- bool **logScale**
- QColor backgroundColour

  Where possible I spell colour properly.

- QColor **secondBgColour**
- int **secondBgSize**
- bool **showSecondBg**
- QColor **barColour**
- bool **drawBorder**
- Qt::Orientation orientation
- int **testSize**

9.178.1 Detailed Description

The QEHistogram class is a non-EPICS aware histogram widget. The value of, i.e. the length of each bar, and colour may be set indepedently.

9.178.2 Property Documentation

9.178.2.1 Qt::Orientation QEHistogram::orientation `[read, write]`

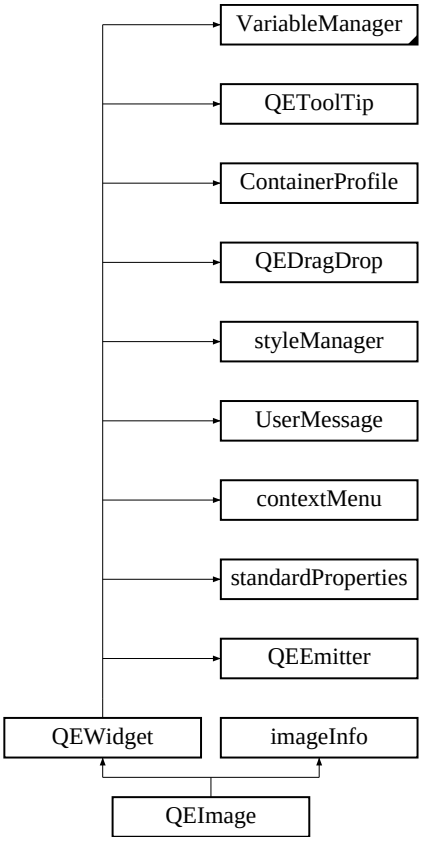
orientation horizontal (default) or vertical. Horizontal means each element displayed horzontally from left to right with the bar represting the value increasing vertically from bottom to top.

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

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEHistogr
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEHistogram.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEHistogr

9.179 QEImage Class Reference

`#include <QEImage.h>`Inheritance diagram for QEImage::



Public Types

- enum selectOptions {
  SO_NONE, SO_PANNING, SO_VSLICE1, SO_VSLICE2,
  SO_VSLICE3, SO_VSLICE4, SO_VSLICE5, SO_HSLICE1,
  SO_HSLICE2, SO_HSLICE3, SO_HSLICE4, SO_HSLICE5,
  SO_AREA1, **SO_AREA2**, **SO_AREA3**, SO_AREA4,
  SO_PROFILE, SO_TARGET, SO_BEAM }
- enum **imageUses** { **IMAGE_USE_DISPLAY**, **IMAGE_USE_SAVE**, **IMAGE_USE_DISPLAY_AND_SAVE** }
- enum resizeOptions { RESIZE_OPTION_ZOOM, RESIZE_OPTION_FIT }
- enum ellipseVariableDefinitions { BOUNDING_RECTANGLE, **CENTRE_-AND_SIZE** }

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

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

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

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

Public Slots

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

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

- void setSelectVSliceMode ()

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

- void setSelectHSliceMode ()

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

- void setSelectArea1Mode ()

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

- void setSelectArea2Mode ()

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

- void setSelectArea3Mode ()

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

- void setSelectArea4Mode ()

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

- void setSelectProfileMode ()

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

- void setSelectTargetMode ()

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

- void setSelectBeamMode ()

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

- void setSelectVSlice1Mode ()

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

- void setSelectVSlice2Mode ()

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

- void setSelectVSlice3Mode ()

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

- void setSelectVSlice4Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectVSlice5Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectHSlice1Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectHSlice2Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectHSlice3Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectHSlice4Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void setSelectHSlice5Mode ()

  Framework use only. Slot to allow external setting of selection menu options.

- void pauseClicked ()

  Framework use only. Slot to allow external setting of selection menu options.

- void saveClicked ()

  Framework use only. Slot to allow external setting of selection menu options.

- void targetClicked ()

  Framework use only. Slot to allow external setting of selection menu options.

- void imageDisplayPropsDestroyed (QObject ∗)

  Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void vSliceDisplayDestroyed (QObject ∗)

  Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void hSliceDisplayDestroyed (QObject ∗)

  Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void profileDisplayDestroyed (QObject ∗)

  Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void recorderDestroyed (QObject ∗)

  Framework use only. Slot to catch deletion of components (such as profile plots) that have been passed to the application for presentation.

- void showProfile ()

  Show the arbitrary line (profile) markup - refer to enableProfileSelection property and displayMarkups property for details.

- void showProfile (bool show)

  Show or hide the arbitrary line (profile) markup. Note that when hiding if its PV changes it will reshow unless DisplayMarkups has been set to off - refer to enableProfileSelection property and displayMarkups property for details.

- void hideProfile ()

  Hide the arbitrary line (profile) markup but note that if its PV changes it will reshow unless DisplayMarkups has been set to off - refer to enableProfileSelection property and displayMarkups property for details.

- void showArea1 ()

Show the area1 markup - refer to enableArea1Selection property and displayMarkups property for details.

- void showArea1 (bool show)

  show or hide the area1 markup. Note that when hiding if its PV changes it will reshow unless DisplayMarkups has been set to off - refer to enableArea1Selection property and displayMarkups property for details

- void hideArea1 ()

  Hide the area1 markup but note that if its PV changes it will reshow unless DisplayMarkups has been set to off - refer to enableArea1Selection property and displayMarkups property for details.

- void setDisplayMarkupsOn ()

  Set markup display to on to show all markups that change either due to user or PV activity, even if their setDisplay????Selection is off - refer to displayMarkups property for details.

- void setDisplayMarkupsOn (bool on)

  Set markup display to on to or off show all markups that change either due to user or PV activity, even if their setDisplay????Selection is off - refer to displayMarkups property for details.

- void setDisplayMarkupsOff ()

  Set markup display to off to stop PV controlled pvs from showing even if they change, unless their setDisplay????Selection is on - refer to displayMarkups property for details.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged (const QString &out)
- void requestResend ()

  Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

- void **componentHostRequest** (const QEActionRequests &request)

Public Member Functions

- QEImage (QWidget *parent=0)
- QEImage (const QString &variableName, QWidget *parent=0)
- ~QEImage ()

  Destructor.

- selectOptions **getSelectionOption** ()

- void setBitDepth (unsigned int bitDepthIn)

  Access function for bitDepth property - refer to bitDepth property for details.

- unsigned int getBitDepth ()

  Access function for bitDepth property - refer to bitDepth property for details.

- void setFormatOption (imageDataFormats::formatOptions formatOption)

  Access function for formatOption property - refer to formatOption property for details.

- imageDataFormats::formatOptions getFormatOption ()

  Access function for formatOption property - refer to formatOption property for details.

- void setResizeOption (resizeOptions resizeOptionIn)

  Access function for resizeOption property - refer to resizeOption property for details.

- resizeOptions getResizeOption ()

  Access function for resizeOption property - refer to resizeOption property for details.

- void setZoom (int zoomIn)

  Access function for zoom property - refer to zoom property for details.

- int getZoom ()

  Access function for zoom property - refer to zoom property for details.

- void setXStretch (double XStretchIn)

  Access function for XStretch property - refer to XStretch property for details.

- double getXStretch ()

  Access function for XStretch property - refer to XStretch property for details.

- void setYStretch (double YStretchIn)

  Access function for YStretch property - refer to YStretch property for details.

- double getYStretch ()

  Access function for YStretch property - refer to YStretch property for details.

- void setRotation (imageProperties::rotationOptions rotationIn)

  Access function for rotation property - refer to rotation property for details.

- imageProperties::rotationOptions getRotation ()

  Access function for rotation property - refer to rotation property for details.

- void setHorizontalFlip (bool flipHozIn)

  Access function for horizontalFlip property - refer to horizontalFlip property for details.

- bool getHorizontalFlip ()

  Access function for horizontalFlip property - refer to horizontalFlip property for details.

- void setVerticalFlip (bool flipVertIn)

  Access function for verticalFlip property - refer to verticalFlip property for details.

- bool getVerticalFlip ()

  Access function for verticalFlip property - refer to verticalFlip property for details.

- void setInitialHozScrollPos (int initialHosScrollPosIn)

  Access function for initialHosScrollPos property - refer to initialHosScrollPos property for details.

- int getInitialHozScrollPos ()

  Access function for initialHosScrollPos property - refer to initialHosScrollPos property for details.

- void setInitialVertScrollPos (int initialVertScrollPosIn)

  Access function for initialVertScrollPos property - refer to initialVertScrollPos property for details.

- int getInitialVertScrollPos ()

  Access function for initialVertScrollPos property - refer to initialVertScrollPos property for details.

- void setDisplayButtonBar (bool displayButtonBarIn)

  Access function for displayButtonBar property - refer to displayButtonBar property for details.

- bool getDisplayButtonBar ()

  Access function for displayButtonBar property - refer to displayButtonBar property for details.

- void setShowTime (bool pValue)

  Access function for showTime property - refer to showTime property for details.

- bool getShowTime ()

  Access function for showTime property - refer to showTime property for details.

- void setUseFalseColour (bool pValue)

  Access function for useFalseColour property - refer to useFalseColour property for details.

- bool getUseFalseColour ()

  Access function for useFalseColour property - refer to useFalseColour property for details.

- void setVertSlice1MarkupColor (QColor pValue)

  Access function for vertSliceColor property - refer to vertSliceColor property for details.

- QColor getVertSlice1MarkupColor ()

  Access function for vertSliceColor property - refer to vertSliceColor property for details.

- void setVertSlice2MarkupColor (QColor pValue)

  Access function for vertSlice2Color property - refer to vertSlice2Color property for details.

- QColor getVertSlice2MarkupColor ()

  Access function for vertSlice2Color property - refer to vertSlice2Color property for details.

- void setVertSlice3MarkupColor (QColor pValue)

  Access function for vertSlice3Color property - refer to vertSlice3Color property for details.

- QColor getVertSlice3MarkupColor ()

  Access function for vertSlice3Color property - refer to vertSlice3Color property for details.

- void setVertSlice4MarkupColor (QColor pValue)

  Access function for vertSlice4Color property - refer to vertSlice4Color property for details.

- QColor getVertSlice4MarkupColor ()

  Access function for vertSlice4Color property - refer to vertSlice4Color property for details.

- void setVertSlice5MarkupColor (QColor pValue)

  Access function for vertSlice5Color property - refer to vertSlice5Color property for details.

- QColor getVertSlice5MarkupColor ()

  Access function for vertSlice5Color property - refer to vertSlice5Color property for details.

- void setHozSlice1MarkupColor (QColor pValue)

  Access function for hozSliceColor property - refer to hozSliceColor property for details.

- QColor getHozSlice1MarkupColor ()

  Access function for hozSliceColor property - refer to hozSliceColor property for details.

- void setHozSlice2MarkupColor (QColor pValue)

  Access function for hozSlice2Color property - refer to hozSlice2Color property for details.

- QColor getHozSlice2MarkupColor ()

  Access function for hozSlice2Color property - refer to hozSlice2Color property for details.

- void setHozSlice3MarkupColor (QColor pValue)

  Access function for hozSlice3Color property - refer to hozSlice3Color property for details.

- QColor getHozSlice3MarkupColor ()

  Access function for hozSlice3Color property - refer to hozSlice3Color property for details.

- void setHozSlice4MarkupColor (QColor pValue)

  Access function for hozSlice4Color property - refer to hozSlice4Color property for details.

- QColor getHozSlice4MarkupColor ()

  Access function for hozSlice4Color property - refer to hozSlice4Color property for details.

- void setHozSlice5MarkupColor (QColor pValue)

  Access function for hozSlice5Color property - refer to hozSlice5Color property for details.

- QColor getHozSlice5MarkupColor ()

  Access function for hozSlice5Color property - refer to hozSlice5Color property for details.

- void setProfileMarkupColor (QColor pValue)

  Access function for profileColor property - refer to profileColor property for details.

- QColor getProfileMarkupColor ()

  Access function for profileColor property - refer to profileColor property for details.

- void setAreaMarkupColor (QColor pValue)

  Access function for areaColor property - refer to areaColor property for details.

- QColor getAreaMarkupColor ()

  Access function for areaColor property - refer to areaColor property for details.

- void setTargetMarkupColor (QColor pValue)

  Access function for targetColor property - refer to targetColor property for details.

- QColor getTargetMarkupColor ()

  Access function for targetColor property - refer to targetColor property for details.

- void setBeamMarkupColor (QColor pValue)

  Access function for beamColor property - refer to beamColor property for details.

- QColor getBeamMarkupColor ()

  Access function for beamColor property - refer to beamColor property for details.

- void setTimeMarkupColor (QColor pValue)

  Access function for timeColor property - refer to timeColor property for details.

- QColor getTimeMarkupColor ()

  Access function for timeColor property - refer to timeColor property for details.

- void setEllipseMarkupColor (QColor markupColor)

  Access function for ellipseColor property - refer to ellipseColor property for details.

- QColor getEllipseMarkupColor ()

  Access function for ellipseColor property - refer to ellipseColor property for details.

- void setDisplayCursorPixelInfo (bool displayCursorPixelInfo)

  Access function for displayCursorPixelInfo property - refer to displayCursorPixelInfo property for details.

- bool getDisplayCursorPixelInfo ()

  Access function for displayCursorPixelInfo property - refer to displayCursorPixelInfo property for details.

- void setContrastReversal (bool contrastReversalIn)

  Access function for contrastReversal property - refer to contrastReversal property for details.

- bool getContrastReversal ()

  Access function for contrastReversal property - refer to contrastReversal property for details.

- void setLog (bool log)

  Access function for logBrightness property - refer to logBrightness property for details.

- bool getLog ()

  Access function for logBrightness property - refer to logBrightness property for details.

- void setEnableVertSlice1Selection (bool enableVSliceSelection)

Access function for enableVertSlice1Selection property - refer to enableVertSlice1Selection property for details.

- bool getEnableVertSlice1Selection ()

  Access function for enableVertSlice1Selection property - refer to enableVertSlice1Selection property for details.

- void setEnableVertSlice2Selection (bool enableVSliceSelection)

  Access function for enableVertSlice2Selection property - refer to enableVertSlice2Selection property for details.

- bool getEnableVertSlice2Selection ()

  Access function for enableVertSlice2Selection property - refer to enableVertSlice2Selection property for details.

- void setEnableVertSlice3Selection (bool enableVSliceSelection)

  Access function for enableVertSlice3Selection property - refer to enableVertSlice3Selection property for details.

- bool getEnableVertSlice3Selection ()

  Access function for enableVertSlice3Selection property - refer to enableVertSlice3Selection property for details.

- void setEnableVertSlice4Selection (bool enableVSliceSelection)

  Access function for enableVertSlice4Selection property - refer to enableVertSlice4Selection property for details.

- bool getEnableVertSlice4Selection ()

  Access function for enableVertSlice4Selection property - refer to enableVertSlice4Selection property for details.

- void setEnableVertSlice5Selection (bool enableVSliceSelection)

  Access function for enableVertSlice5Selection property - refer to enableVertSlice5Selection property for details.

- bool getEnableVertSlice5Selection ()

  Access function for enableVertSlice5Selection property - refer to enableVertSlice5Selection property for details.

- void setEnableHozSlice1Selection (bool enableHSliceSelection)

  Access function for enableHozSlice1Selection property - refer to enableHozSlice1Selection property for details.

- bool getEnableHozSlice1Selection ()

  Access function for enableHozSlice1Selection property - refer to enableHozSlice1Selection property for details.

- void setEnableHozSlice2Selection (bool enableHSliceSelection)

Access function for enableHozSlice2Selection property - refer to enableHozSlice2Selection property for details.

- bool getEnableHozSlice2Selection ()

  Access function for enableHozSlice2Selection property - refer to enableHozSlice2Selection property for details.

- void setEnableHozSlice3Selection (bool enableHSliceSelection)

  Access function for enableHozSlice3Selection property - refer to enableHozSlice3Selection property for details.

- bool getEnableHozSlice3Selection ()

  Access function for enableHozSlice3Selection property - refer to enableHozSlice3Selection property for details.

- void setEnableHozSlice4Selection (bool enableHSliceSelection)

  Access function for enableHozSlice4Selection property - refer to enableHozSlice4Selection property for details.

- bool getEnableHozSlice4Selection ()

  Access function for enableHozSlice4Selection property - refer to enableHozSlice4Selection property for details.

- void setEnableHozSlice5Selection (bool enableHSliceSelection)

  Access function for enableHozSlice5Selection property - refer to enableHozSlice5Selection property for details.

- bool getEnableHozSlice5Selection ()

  Access function for enableHozSlice5Selection property - refer to enableHozSlice5Selection property for details.

- void setEnableArea1Selection (bool enableAreaSelectionIn)

  Access function for enableArea1Selection property - refer to enableArea1Selection property for details.

- bool getEnableArea1Selection ()

  Access function for enableArea1Selection property - refer to enableArea1Selection property for details.

- void setEnableArea2Selection (bool enableAreaSelectionIn)

  Access function for enableArea2Selection property - refer to enableArea2Selection property for details.

- bool getEnableArea2Selection ()

  Access function for enableArea2Selection property - refer to enableArea2Selection property for details.

- void setEnableArea3Selection (bool enableAreaSelectionIn)

Access function for enableArea3Selection property - refer to enableArea3Selection property for details.

- bool getEnableArea3Selection ()

  Access function for enableArea3Selection property - refer to enableArea3Selection property for details.

- void setEnableArea4Selection (bool enableAreaSelectionIn)

  Access function for enableArea4Selection property - refer to enableArea4Selection property for details.

- bool getEnableArea4Selection ()

  Access function for enableArea4Selection property - refer to enableArea4Selection property for details.

- void setEnableProfileSelection (bool enableProfileSelectionIn)

  Access function for enableProfileSelection property - refer to enableProfileSelection property for details.

- bool getEnableProfileSelection ()

  Access function for enableProfileSelection property - refer to enableProfileSelection property for details.

- void setEnableTargetSelection (bool enableTargetSelectionIn)

  Access function for enableTargetSelection property - refer to enableTargetSelection property for details.

- bool getEnableTargetSelection ()

  Access function for enableTargetSelection property - refer to enableTargetSelection property for details.

- void setEnableBeamSelection (bool enableBeamSelectionIn)

  Access function for enableBeamSelection property - refer to enableBeamSelection property for details.

- bool getEnableBeamSelection ()

  Access function for enableBeamSelection property - refer to enableBeamSelection property for details.

- void setEnableImageDisplayProperties (bool enableImageDisplayPropertiesIn)

  Access function for enableImageDisplayProperties property - refer to enableImageDisplayProperties property for details.

- bool getEnableImageDisplayProperties ()

  Access function for enableImageDisplayProperties property - refer to enableImageDisplayProperties property for details.

- void setEnableRecording (bool enableRecordingIn)

Access function for enableRecording property - refer to enableRecording property for details.

- bool getEnableRecording ()

  Access function for enableRecording property - refer to enableRecording property for details.

- void setAutoBrightnessContrast (bool autoBrightnessContrastIn)

  Access function for autoBrightnessContrast property - refer to autoBrightnessContrast property for details.

- bool getAutoBrightnessContrast ()

  Access function for autoBrightnessContrast property - refer to autoBrightnessContrast property for details.

- void setExternalControls (bool externalControlsIn)

  Access function for externalControls property - refer to externalControls property for details.

- bool getExternalControls ()

  Access function for externalControls property - refer to externalControls property for details.

- void setFullContextMenu (bool fullContextMenuIn)

  Access function for fullContextMenu property - refer to fullContextMenu property for details.

- bool getFullContextMenu ()

  Access function for fullContextMenu property - refer to fullContextMenu property for details.

- void setEnableProfilePresentation (bool enableProfilePresentationIn)

  Access function for enableProfilePresentation property - refer to enableProfilePresentation property for details.

- bool getEnableProfilePresentation ()

  Access function for enableProfilePresentation property - refer to enableProfilePresentation property for details.

- void setEnableHozSlicePresentation (bool enableHozSlicePresentationIn)

  Access function for enableHozSlicePresentation property - refer to enableHozSlicePresentation property for details.

- bool getEnableHozSlicePresentation ()

  Access function for enableHozSlicePresentation property - refer to enableHozSlicePresentation property for details.

- void setEnableVertSlicePresentation (bool enableVertSlicePresentationIn)

Access function for enableVertSlicePresentation property - refer to enableVertSliceP-resentation property for details.

- bool getEnableVertSlicePresentation ()

  Access function for enableVertSlicePresentation property - refer to enableVertSliceP-resentation property for details.

- void setDisplayVertSlice1Selection (bool displayVSliceSelection)

  Access function for displayVertSlice1Selection property - refer to displayVert-Slice1Selection property for details.

- bool getDisplayVertSlice1Selection ()

  Access function for displayVertSlice1Selection property - refer to displayVert-Slice1Selection property for details.

- void setDisplayVertSlice2Selection (bool displayVSliceSelection)

  Access function for displayVertSlice2Selection property - refer to displayVert-Slice2Selection property for details.

- bool getDisplayVertSlice2Selection ()

  Access function for displayVertSlice2Selection property - refer to displayVert-Slice2Selection property for details.

- void setDisplayVertSlice3Selection (bool displayVSliceSelection)

  Access function for displayVertSlice3Selection property - refer to displayVert-Slice3Selection property for details.

- bool getDisplayVertSlice3Selection ()

  Access function for displayVertSlice3Selection property - refer to displayVert-Slice3Selection property for details.

- void setDisplayVertSlice4Selection (bool displayVSliceSelection)

  Access function for displayVertSlice4Selection property - refer to displayVert-Slice4Selection property for details.

- bool getDisplayVertSlice4Selection ()

  Access function for displayVertSlice4Selection property - refer to displayVert-Slice4Selection property for details.

- void setDisplayVertSlice5Selection (bool displayVSliceSelection)

  Access function for displayVertSlice5Selection property - refer to displayVert-Slice5Selection property for details.

- bool getDisplayVertSlice5Selection ()

  Access function for displayVertSlice5Selection property - refer to displayVert-Slice5Selection property for details.

- void setDisplayHozSlice1Selection (bool displayHSliceSelection)

Access function for displayHozSlice1Selection property - refer to displayHozSlice1Selection property for details.

- bool getDisplayHozSlice1Selection ()

  Access function for displayHozSlice1Selection property - refer to displayHozSlice1Selection property for details.

- void setDisplayHozSlice2Selection (bool displayHSliceSelection)

  Access function for displayHozSlice2Selection property - refer to displayHozSlice2Selection property for details.

- bool getDisplayHozSlice2Selection ()

  Access function for displayHozSlice2Selection property - refer to displayHozSlice2Selection property for details.

- void setDisplayHozSlice3Selection (bool displayHSliceSelection)

  Access function for displayHozSlice3Selection property - refer to displayHozSlice3Selection property for details.

- bool getDisplayHozSlice3Selection ()

  Access function for displayHozSlice3Selection property - refer to displayHozSlice3Selection property for details.

- void setDisplayHozSlice4Selection (bool displayHSliceSelection)

  Access function for displayHozSlice4Selection property - refer to displayHozSlice4Selection property for details.

- bool getDisplayHozSlice4Selection ()

  Access function for displayHozSlice4Selection property - refer to displayHozSlice4Selection property for details.

- void setDisplayHozSlice5Selection (bool displayHSliceSelection)

  Access function for displayHozSlice5Selection property - refer to displayHozSlice5Selection property for details.

- bool getDisplayHozSlice5Selection ()

  Access function for displayHozSlice5Selection property - refer to displayHozSlice5Selection property for details.

- void setDisplayArea1Selection (bool displayAreaSelection)

  Access function for displayArea1Selection property - refer to displayArea1Selection property for details.

- bool getDisplayArea1Selection ()

  Access function for displayArea1Selection property - refer to displayArea1Selection property for details.

- void setDisplayArea2Selection (bool displayAreaSelection)

Access function for displayArea2Selection property - refer to displayArea2Selection property for details.

- bool getDisplayArea2Selection ()

  Access function for displayArea2Selection property - refer to displayArea2Selection property for details.

- void setDisplayArea3Selection (bool displayAreaSelection)

  Access function for displayArea3Selection property - refer to displayArea3Selection property for details.

- bool getDisplayArea3Selection ()

  Access function for displayArea3Selection property - refer to displayArea3Selection property for details.

- void setDisplayArea4Selection (bool displayAreaSelection)

  Access function for displayArea4Selection property - refer to displayArea4Selection property for details.

- bool getDisplayArea4Selection ()

  Access function for displayArea4Selection property - refer to displayArea4Selection property for details.

- void setDisplayProfileSelection (bool displayProfileSelection)

  Access function for displayProfileSelection property - refer to displayProfileSelection property for details.

- bool getDisplayProfileSelection ()

  Access function for displayProfileSelection property - refer to displayProfileSelection property for details.

- void setDisplayTargetSelection (bool displayTargetSelection)

  Access function for displayTargetSelection property - refer to displayTargetSelection property for details.

- bool getDisplayTargetSelection ()

  Access function for displayTargetSelection property - refer to displayTargetSelection property for details.

- void setDisplayBeamSelection (bool displayBeamSelection)

  Access function for displayBeamSelection property - refer to displayBeamSelection property for details.

- bool getDisplayBeamSelection ()

  Access function for displayBeamSelection property - refer to displayBeamSelection property for details.

- void setDisplayEllipse (bool displayEllipse)

Access function for displayEllipse property - refer to displayEllipse property for details.

- bool getDisplayEllipse ()

  Access function for displayEllipse property - refer to displayEllipse property for details.

- ellipseVariableDefinitions getEllipseVariableDefinition ()

  Access function for ellipseVariableDefinition property - refer to ellipseVariableDefinition property for details.

- void setEllipseVariableDefinition (ellipseVariableDefinitions def)

  Access function for ellipseVariableDefinition property - refer to ellipseVariableDefinition property for details.

- void setDisplayMarkups (bool displayMarkupsIn)

  Access function for displayMarkups property - refer to displayMarkups property for details.

- bool getDisplayMarkups ()

  Access function for displayMarkups property - refer to displayMarkups property for details.

- void setName (QString nameIn)

  Access function for name property - refer to name property for details.

- QString getName ()

  Access function for name property - refer to name property for details.

- void setProgram1 (QString program)

  Access function for program1 property - refer to program1 property for details.

- QString getProgram1 ()

  Access function for program1 property - refer to program1 property for details.

- void setProgram2 (QString program)

  Access function for program2 property - refer to program2 property for details.

- QString getProgram2 ()

  Access function for program2 property - refer to program2 property for details.

- void setArguments1 (QStringList arguments)

  Access function for arguments1 property - refer to arguments1 property for details.

- QStringList getArguments1 ()

  Access function for arguments1 property - refer to arguments1 property for details.

- void setArguments2 (QStringList arguments)

  Access function for arguments2 property - refer to arguments2 property for details.

- QStringList getArguments2 ()

  Access function for arguments2 property - refer to arguments2 property for details.

- void setProgramStartupOption1 (applicationLauncher::programStartupOptions programStartupOption)

  Access function for programStartupOption1 property - refer to programStartupOption1 property for details.

- applicationLauncher::programStartupOptions getProgramStartupOption1 ()

  Access function for programStartupOption1 property - refer to programStartupOption1 property for details.

- void setProgramStartupOption2 (applicationLauncher::programStartupOptions programStartupOption)

  Access function for programStartupOption2 property - refer to programStartupOption2 property for details.

- applicationLauncher::programStartupOptions getProgramStartupOption2 ()

  Access function for programStartupOption2 property - refer to programStartupOption2 property for details.

- QString getHozSlice1Legend ()

  Access function for hozSlice1Legend property - refer to hozSlice1Legend property for details.

- void setHozSlice1Legend (QString legend)

  Access function for hozSlice1Legend property - refer to hozSlice1Legend property for details.

- QString getHozSlice2Legend ()

  Access function for hozSlice2Legend property - refer to hozSlice2Legend property for details.

- void setHozSlice2Legend (QString legend)

  Access function for hozSlice2Legend property - refer to hozSlice2Legend property for details.

- QString getHozSlice3Legend ()

  Access function for hozSlice3Legend property - refer to hozSlice3Legend property for details.

- void setHozSlice3Legend (QString legend)

  Access function for hozSlice3Legend property - refer to hozSlice3Legend property for details.

- QString getHozSlice4Legend ()

  Access function for hozSlice4Legend property - refer to hozSlice4Legend property for details.

- void setHozSlice4Legend (QString legend)

  Access function for hozSlice4Legend property - refer to hozSlice4Legend property for details.

- QString getHozSlice5Legend ()

  Access function for hozSlice5Legend property - refer to hozSlice5Legend property for details.

- void setHozSlice5Legend (QString legend)

  Access function for hozSlice5Legend property - refer to hozSlice5Legend property for details.

- QString getVertSlice1Legend ()

  Access function for vertSlice1Legend property - refer to vertSlice1Legend property for details.

- void setVertSlice1Legend (QString legend)

  Access function for vertSlice1Legend property - refer to vertSlice1Legend property for details.

- QString getVertSlice2Legend ()

  Access function for vertSlice2Legend property - refer to vertSlice2Legend property for details.

- void setVertSlice2Legend (QString legend)

  Access function for vertSlice2Legend property - refer to vertSlice2Legend property for details.

- QString getVertSlice3Legend ()

  Access function for vertSlice3Legend property - refer to vertSlice3Legend property for details.

- void setVertSlice3Legend (QString legend)

  Access function for vertSlice3Legend property - refer to vertSlice3Legend property for details.

- QString getVertSlice4Legend ()

  Access function for vertSlice4Legend property - refer to vertSlice4Legend property for details.

- void setVertSlice4Legend (QString legend)

  Access function for vertSlice4Legend property - refer to vertSlice4Legend property for details.

- QString getVertSlice5Legend ()

  Access function for vertSlice5Legend property - refer to vertSlice5Legend property for details.

- void setVertSlice5Legend (QString legend)

  Access function for vertSlice5Legend property - refer to vertSlice5Legend property for details.

- QString getprofileLegend ()

  Access function for profileLegend property - refer to profileLegend property for details.

- void setProfileLegend (QString legend)

  Access function for profileLegend property - refer to profileLegend property for details.

- QString getAreaSelection1Legend ()

  Access function for areaSelection1Legend property - refer to areaSelection1Legend property for details.

- void setAreaSelection1Legend (QString legend)

  Access function for areaSelection1Legend property - refer to areaSelection1Legend property for details.

- QString getAreaSelection2Legend ()

  Access function for areaSelection2Legend property - refer to areaSelection2Legend property for details.

- void setAreaSelection2Legend (QString legend)

  Access function for areaSelection2Legend property - refer to areaSelection2Legend property for details.

- QString getAreaSelection3Legend ()

  Access function for areaSelection3Legend property - refer to areaSelection3Legend property for details.

- void setAreaSelection3Legend (QString legend)

  Access function for areaSelection3Legend property - refer to areaSelection3Legend property for details.

- QString getAreaSelection4Legend ()

  Access function for areaSelection4Legend property - refer to areaSelection4Legend property for details.

- void setAreaSelection4Legend (QString legend)

  Access function for areaSelection4Legend property - refer to areaSelection4Legend property for details.

- QString getTargetLegend ()

  Access function for targetLegend property - refer to targetLegend property for details.

- void setTargetLegend (QString legend)

  Access function for targetLegend property - refer to targetLegend property for details.

- QString getBeamLegend ()

  Access function for beamLegend property - refer to beamLegend property for details.

- void setBeamLegend (QString legend)

  Access function for beamLegend property - refer to beamLegend property for details.

- QString getEllipseLegend ()

  Access function for ellipseLegend property - refer to ellipseLegend property for details.

- void setEllipseLegend (QString legend)

  Access function for ellipseLegend property - refer to ellipseLegend property for details.

- bool getFullScreen ()

  Access function for fullScreen property - refer to fullScreen property for details.

- void setFullScreen (bool fullScreenIn)

  Access function for fullScreen property - refer to fullScreen property for details.

- void setSubstitutedUrl (QString urlIn)

  Access function for URL property - refer to URL property for deta.

- QString getSubstitutedUrl ()

  Access function for URL property - refer to URL property for deta.

- void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- QString getVariableNameSubstitutionsProperty ()

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setFormatOptionProperty (FormatOptions formatOption)

  Access function for formatOption property - refer to formatOption property for details.

- FormatOptions getFormatOptionProperty ()

  Access function for formatOption property - refer to formatOption property for details.

- void setBitDepthProperty (unsigned int bitDepth)

  Access function for bitDepth property - refer to bitDepth property for details.

- unsigned int getBitDepthProperty ()

  Access function for bitDepth property - refer to bitDepth property for details.

- EllipseVariableDefinitions getEllipseVariableDefinitionProperty ()

  Access function for EllipseVariableDefinition property - refer to EllipseVariableDefinition property for details.

- void setEllipseVariableDefinitionProperty (EllipseVariableDefinitions variableUsage)

  Access function for EllipseVariableDefinitions property - refer to EllipseVariableDefinitions property for details.

- TargetOptions getTargetOptionProperty ()

  Access function for targetOption property - refer to targetOption property for details.

- void setTargetOptionProperty (TargetOptions option)

  Access function for targetOption property - refer to targetOption property for details.

- TargetOptions getBeamOptionProperty ()

  Access function for beamOption property - refer to beamOption property for details.

- void setBeamOptionProperty (TargetOptions option)

  Access function for beamOption property - refer to beamOption property for details.

- void setResizeOptionProperty (ResizeOptions resizeOption)

  Access function for resizeOption property - refer to resizeOption property for details.

- ResizeOptions getResizeOptionProperty ()

  Access function for resizeOption property - refer to resizeOption property for details.

- void setRotationProperty (RotationOptions rotation)

  Access function for rotation property - refer to rotation property for details.

- RotationOptions getRotationProperty ()

  Access function for rotation property - refer to rotation property for details.

- void setProgramStartupOptionProperty1 (ProgramStartupOptionNames programStartupOption)

  Access function for ProgramStartupOptionNames1 property - refer to ProgramStartupOptionNames1 property for details.

- ProgramStartupOptionNames getProgramStartupOptionProperty1 ()

  Access function for ProgramStartupOptionNames1 property - refer to ProgramStartupOptionNames1 property for details.

- void setProgramStartupOptionProperty2 (ProgramStartupOptionNames programStartupOption)

  Access function for ProgramStartupOptionNames2 property - refer to ProgramStartupOptionNames2 property for details.

- ProgramStartupOptionNames getProgramStartupOptionProperty2 ()

  Access function for ProgramStartupOptionNames2 property - refer to ProgramStartupOptionNames2 property for details.

Protected Types

- enum **variableIndexes** {

  IMAGE_VARIABLE, **FORMAT_VARIABLE**, **BIT_DEPTH_VARIABLE**, **DATA_TYPE_VARIABLE**,

  WIDTH_VARIABLE, **HEIGHT_VARIABLE**, **NUM_DIMENSIONS_VARIABLE**, **DIMENSION_0_VARIABLE**,

  DIMENSION_1_VARIABLE, **DIMENSION_2_VARIABLE**, **ROI1_X_VARIABLE**, **ROI1_Y_VARIABLE**,

ROI1_W_VARIABLE, **ROI1_H_VARIABLE**, **ROI2_X_VARIABLE**, **ROI2_Y_VARIABLE**,

ROI2_W_VARIABLE, **ROI2_H_VARIABLE**, **ROI3_X_VARIABLE**, **ROI3_Y_VARIABLE**,

ROI3_W_VARIABLE, **ROI3_H_VARIABLE**, **ROI4_X_VARIABLE**, **ROI4_Y_VARIABLE**,

ROI4_W_VARIABLE, **ROI4_H_VARIABLE**, **TARGET_X_VARIABLE**, **TARGET_Y_VARIABLE**,

BEAM_X_VARIABLE, **BEAM_Y_VARIABLE**, **TARGET_TRIGGER_-VARIABLE**, **CLIPPING_ONOFF_VARIABLE**,

CLIPPING_LOW_VARIABLE, **CLIPPING_HIGH_VARIABLE**, **PROFILE_H1_VARIABLE**, **PROFILE_H1_THICKNESS_VARIABLE**,

PROFILE_H2_VARIABLE, **PROFILE_H2_THICKNESS_VARIABLE**, **PROFILE_H3_VARIABLE**, **PROFILE_H3_THICKNESS_VARIABLE**,

PROFILE_H4_VARIABLE, **PROFILE_H4_THICKNESS_VARIABLE**, **PROFILE_H5_VARIABLE**, **PROFILE_H5_THICKNESS_VARIABLE**,

PROFILE_V1_VARIABLE, **PROFILE_V1_THICKNESS_VARIABLE**, **PROFILE_V2_VARIABLE**, **PROFILE_V2_THICKNESS_VARIABLE**,

PROFILE_V3_VARIABLE, **PROFILE_V3_THICKNESS_VARIABLE**, **PROFILE_V4_VARIABLE**, **PROFILE_V4_THICKNESS_VARIABLE**,

PROFILE_V5_VARIABLE, **PROFILE_V5_THICKNESS_VARIABLE**, **LINE_PROFILE_X1_VARIABLE**, **LINE_PROFILE_Y1_VARIABLE**,

LINE_PROFILE_X2_VARIABLE, **LINE_PROFILE_Y2_VARIABLE**, **LINE_PROFILE_THICKNESS_VARIABLE**, **PROFILE_H_ARRAY**,

PROFILE_V_ARRAY, **PROFILE_LINE_ARRAY**, **ELLIPSE_X_-VARIABLE**, **ELLIPSE_Y_VARIABLE**,

ELLIPSE_W_VARIABLE, **ELLIPSE_H_VARIABLE**, **ELLIPSE_R_-VARIABLE**, **QEIMAGE_NUM_VARIABLES** }

Protected Member Functions

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **redisplayAllMarkups** ()
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()

- QVariant **copyData** ()
- void **paste** (QVariant v)
- void **resizeEvent** (QResizeEvent ∗)

Protected Attributes

- QEStringFormatting **stringFormatting**
- QEIntegerFormatting **integerFormatting**
- QEFloatingFormatting **floatingFormatting**
- resizeOptions **resizeOption**
- int zoom

  Zoom percentage. Used when resizeOption is Zoom.

- double XStretch

  Stretch X factor. Used when generating canvas in which fully processed image is presented.

- double YStretch

  Stretch Y factor. Used when generating canvas in which fully processed image is presented.

- int **initialHozScrollPos**
- int initialVertScrollPos
- bool displayButtonBar

Properties

- QString imageVariable
- QString formatVariable
- QString bitDepthVariable
- QString dataTypeVariable
- QString widthVariable
- QString heightVariable
- QString dimensionsVariable
- QString dimension1Variable
- QString dimension2Variable
- QString dimension3Variable
- QString regionOfInterest1XVariable
- QString regionOfInterest1YVariable
- QString regionOfInterest1WVariable
- QString regionOfInterest1HVariable
- QString regionOfInterest2XVariable
- QString regionOfInterest2YVariable
- QString regionOfInterest2WVariable
- QString regionOfInterest2HVariable
- QString regionOfInterest3XVariable

- QString regionOfInterest3YVariable
- QString regionOfInterest3WVariable
- QString regionOfInterest3HVariable
- QString regionOfInterest4XVariable
- QString regionOfInterest4YVariable
- QString regionOfInterest4WVariable
- QString regionOfInterest4HVariable
- QString targetXVariable
- QString targetYVariable
- QString beamXVariable
- QString beamYVariable
- QString targetTriggerVariable
- QString clippingOnOffVariable
- QString clippingLowVariable
- QString clippingHighVariable
- QString **profileHozVariable**
- QString profileHoz1Variable
- QString **profileHozThicknessVariable**
- QString profileHoz1ThicknessVariable
- QString profileHoz2Variable
- QString profileHoz2ThicknessVariable
- QString profileHoz3Variable
- QString profileHoz3ThicknessVariable
- QString profileHoz4Variable
- QString profileHoz4ThicknessVariable
- QString profileHoz5Variable
- QString profileHoz5ThicknessVariable
- QString **profileVertVariable**
- QString profileVert1Variable
- QString **profileVertThicknessVariable**
- QString profileVert1ThicknessVariable
- QString profileVert2Variable
- QString profileVert2ThicknessVariable
- QString profileVert3Variable
- QString profileVert3ThicknessVariable
- QString profileVert4Variable
- QString profileVert4ThicknessVariable
- QString profileVert5Variable
- QString profileVert5ThicknessVariable
- QString lineProfileX1Variable
- QString lineProfileY1Variable
- QString lineProfileX2Variable
- QString lineProfileY2Variable
- QString lineProfileThicknessVariable
- QString profileHozArrayVariable
- QString profileVertArrayVariable

- QString lineProfileArrayVariable
- QString ellipseXVariable
- QString ellipseYVariable
- QString ellipseWVariable
- QString ellipseHVariable
- QString ellipseRotationVariable
- QString variableSubstitutions
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- FormatOptions formatOption
- bool **enableVertSliceSelection**
- bool enableVertSlice1Selection
- bool enableVertSlice2Selection
- bool enableVertSlice3Selection
- bool enableVertSlice4Selection
- bool enableVertSlice5Selection
- bool **enableHozSliceSelection**
- bool enableHozSlice1Selection
- bool enableHozSlice2Selection
- bool enableHozSlice3Selection
- bool enableHozSlice4Selection
- bool enableHozSlice5Selection
- bool enableProfileSelection
- bool enableArea1Selection
- bool enableArea2Selection
- bool enableArea3Selection
- bool enableArea4Selection
- bool enableTargetSelection
- bool enableBeamSelection
- QString **hozSliceLegend**
- QString hozSlice1Legend

  Name of horizontal slice 1 markup.

- QString hozSlice2Legend

  Name of horizontal slice 2 markup.

- QString hozSlice3Legend

  Name of horizontal slice 3 markup.

- QString hozSlice4Legend

  Name of horizontal slice 4 markup.

- QString hozSlice5Legend

  Name of horizontal slice 5 markup.

- QString **vertSliceLegend**
- QString vertSlice1Legend

  Name of vertical slice 1 markup.

- QString vertSlice2Legend

  Name of vertical slice 2 markup.

- QString vertSlice3Legend

  Name of vertical slice 3 markup.

- QString vertSlice4Legend

  Name of vertical slice 4 markup.

- QString vertSlice5Legend

  Name of vertical slice 5 markup.

- QString profileLegend

  Name of arbitrary profile markup.

- QString areaSelection1Legend

  Name of area selection 1 markup.

- QString areaSelection2Legend

  Name of area selection 2 markup.

- QString areaSelection3Legend

  Name of area selection 3 markup.

- QString areaSelection4Legend

  Name of area selection 4 markup.

- QString targetLegend

  Name of target markup.

- QString beamLegend

  Name of beam markup.

- QString ellipseLegend

  Name of ellipse markup.

- bool **displayVertSliceSelection**
- bool displayVertSlice1Selection
- bool displayVertSlice2Selection
- bool displayVertSlice3Selection
- bool displayVertSlice4Selection
- bool displayVertSlice5Selection
- bool **displayHozSliceSelection**
- bool displayHozSlice1Selection
- bool displayHozSlice2Selection
- bool displayHozSlice3Selection
- bool displayHozSlice4Selection
- bool displayHozSlice5Selection
- bool displayProfileSelection
- bool displayArea1Selection
- bool displayArea2Selection
- bool displayArea3Selection
- bool displayArea4Selection
- bool displayTargetSelection
- bool displayBeamSelection
- bool displayEllipse
- EllipseVariableDefinitions ellipseVariableDefinition

  Definition of how ellipse variables are to be used.

- TargetOptions targetOption

  Definition of target markup options.

- TargetOptions beamOption

  Definition of beam markup options.

- bool displayCursorPixelInfo
- bool contrastReversal
- bool logBrightness
- bool showTime
- bool useFalseColour
- QColor **vertSliceColor**
- QColor vertSlice1Color
- QColor vertSlice2Color
- QColor vertSlice3Color
- QColor vertSlice4Color
- QColor vertSlice5Color
- QColor **hozSliceColor**
- QColor hozSlice1Color

- QColor hozSlice2Color
- QColor hozSlice3Color
- QColor hozSlice4Color
- QColor hozSlice5Color
- QColor profileColor
- QColor areaColor
- QColor beamColor
- QColor targetColor
- QColor timeColor
- QColor ellipseColor
- ResizeOptions resizeOption
- RotationOptions rotation
- bool verticalFlip
- bool horizontalFlip
- int initialHosScrollPos
- bool enableImageDisplayProperties

  If true, the local Image Display Properties controls are displayed.

- bool enableRecording

  If true, the recording controls are displayed.

- bool autoBrightnessContrast
- bool externalControls
- bool briefInfoArea
- QString program1
- QStringList arguments1
- ProgramStartupOptionNames programStartupOption1
- QString program2
- QStringList arguments2
- ProgramStartupOptionNames programStartupOption2
- QString URL

9.179.1 Detailed Description

This class is a EPICS aware image widget. When image related variables are defined the image will be displayed. Many PVs may be defined to allow user interaction, such as selecting regions of interest. It is tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.179.2 Member Enumeration Documentation

9.179.2.1 enum QEImage::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.179.2.2 enum QEImage::EllipseVariableDefinitions

User friendly enumerations for ellipseVariableDefinition property - refer to ellipseVariableDefinition property for details.

Enumerator:

BoundingRectangle Refer to BOUNDING_RECTANGLE for details.

CenterAndSize Refer to CENTRE_AND_SIZE for details.

9.179.2.3 enum QEImage::ellipseVariableDefinitions

Options for the use of ellipse markup variables.

Enumerator:

BOUNDING_RECTANGLE Variables define bounding rectagle of ellipse.

9.179.2.4 enum QEImage::FormatOptions

User friendly enumerations for formatOption property - refer to formatOption property and formatOptions enumeration for details.

Enumerator:

Mono Grey scale.

Bayer Colour (Bayer Red Green).

BayerGB Colour (Bayer Green Blue).

BayerBG Colour (Bayer Blue Green).

BayerGR Colour (Bayer Green Red).

BayerRG Colour (Bayer Red Green).

rgb1 Colour (24 bit RGB).

rgb2 Colour (??? bit RGB).

rgb3 Colour (??? bit RGB).

yuv444 Colour (???).

yuv422 Colour (???).

9.179.2.5 enum QEImage::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

Enumerator:

None Just run the program.

Terminal Run the program in a termainal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir').

LogOutput Run the program, and log the output in the QE message system.

StdOutput Run the program, and send doutput to standard output and standard error.

9.179.2.6 enum QEImage::ResizeOptions

User friendly enumerations for resizeOption property

Enumerator:

Zoom Zoom to selected percentage.

Fit Zoom to fit the current window size.

9.179.2.7 enum QEImage::resizeOptions

Image resize options

Enumerator:

RESIZE_OPTION_ZOOM Zoom to selected percentage.

RESIZE_OPTION_FIT Zoom to fit the current window size.

9.179.2.8 enum QEImage::RotationOptions

User friendly enumerations for rotation property

Enumerator:

NoRotation No image rotation.

Rotate90Right Rotate image 90 degrees clockwise.

Rotate90Left Rotate image 90 degrees anticlockwise.

Rotate180 Rotate image 180 degrees.

9.179.2.9 enum QEImage::selectOptions

Internal use only. Selection options. What will happen when the user interacts with the image area

Enumerator:

SO_NONE Do nothing.

SO_PANNING User is panning.

SO_VSLICE1 Select the vertical slice 1 point.

SO_VSLICE2 Select the vertical slice 2 point.

SO_VSLICE3 Select the vertical slice 3 point.

SO_VSLICE4 Select the vertical slice 4 point.

SO_VSLICE5 Select the vertical slice 5 point.

SO_HSLICE1 Select the horizontal slice 1 point.

SO_HSLICE2 Select the horizontal slice 2 point.

SO_HSLICE3 Select the horizontal slice 3 point.

SO_HSLICE4 Select the horizontal slice 4 point.

SO_HSLICE5 Select the horizontal slice 5 point.

SO_AREA4 User is selecting an area (for region of interest).

SO_PROFILE Select an arbitrary line across the image (to determine a profile).

SO_TARGET Mark the target point.

SO_BEAM Mark the current beam location.

9.179.2.10 enum QEImage::TargetOptions

User friendly enumerations for targetOptions property - refer to targetOptions property for details.

Enumerator:

DottedFullCrosshair Refer to CROSSHAIR1 for details.

SolidSmallCrosshair Refer to CROSSHAIR2 for details.

9.179.2.11 enum QEImage::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.179.3 Constructor & Destructor Documentation

9.179.3.1 QEImage::QEImage (QWidget * *parent* = 0)

Create without a variable. Use setVariableName'n'Property() - where 'n' is a number from 0 to 40 - and setSubstitutionsProperty() to define variables and, optionally, macro substitutions later. Note, each variable property is named by function (such as imageVariable and widthVariable) but given a numeric get and set property access function such as setVariableName22Property(). Refer to the property definitions to determine what 'set' and 'get' function is used for each varible, or use Qt library functions to set or get the variable names by name.

9.179.3.2 QEImage::QEImage (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. The variable is set up as the first variable. This is consistant with other widgets, but will not result in an updating image as the width and height variables are required as a minimum.

9.179.4 Member Function Documentation

9.179.4.1 void QEImage::dbValueChanged (const QString & *out*) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.179.4.2 void QEImage::setImageFile (QString *name*) [slot]

!! memcpy will be more efficient.

9.179.4.3 void QEImage::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.179.5 Member Data Documentation

9.179.5.1 bool QEImage::displayButtonBar [read, write, protected]

If true, a button bar will be displayed above the image. If not displayed, all buttons in the button bar are still available in the right click menu.

9.179.5.2 int QEImage::initialVertScrollPos [read, write, protected]

Sets the initial position of the vertical scroll bar, if present. Used to set up an initial view when zoomed in.

9.179.6 Property Documentation

9.179.6.1 bool QEImage::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEDragDrop.

9.179.6.2 QColor QEImage::areaColor [read, write]

Used to select the color of the area selection markups.

9.179.6.3 QStringList QEImage::arguments1 [read, write]

Arguments for program specified in the 'program1' property.

9.179.6.4 QStringList QEImage::arguments2 [read, write]

Arguments for program specified in the 'program2' property.

9.179.6.5 bool QEImage::autoBrightnessContrast [read, write]

If true, auto set local brightness and contrast when any area is selected. The brightness and contrast is set to use the full range of pixels in the selected area.

9.179.6.6 QColor QEImage::beamColor [read, write]

Used to select the color of the beam marker.

9.179.6.7 QString QEImage::beamXVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the selected beam X position.

9.179.6.8 QString QEImage::beamYVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the selected beam Y position.

9.179.6.9 QString QEImage::bitDepthVariable [read, write]

EPICS variable name (CA PV). This variable is used to read the bit depth of the image.

9.179.6.10 bool QEImage::briefInfoArea [read, write]

If true, the information area willl be brief (one row)

9.179.6.11 QString QEImage::clippingHighVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping high level.

9.179.6.12 QString QEImage::clippingLowVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping low level.

9.179.6.13 QString QEImage::clippingOnOffVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector clipping on/off command.

9.179.6.14 bool QEImage::contrastReversal [read, write]

If true, the image will undergo contrast reversal.

9.179.6.15 QString QEImage::dataTypeVariable [read, write]

EPICS variable name (CA PV). This variable is used to infer the bit depth of the image.

9.179.6.16 QString QEImage::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.179.6.17 QString QEImage::dimension1Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the first area detector dimension of the image. If there are 2 dimensions, this will be the image width. If there are 3 dimensions, this will be the number of elements per pixel.

9.179.6.18 QString QEImage::dimension2Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the second area detector dimension of the image. If there are 2 dimensions, this will be the image height. If there are 3 dimensions, this will be the image width.

9.179.6.19 QString QEImage::dimension3Variable [read, write]

EPICS variable name (CA PV). This variable is used to read the third area detector dimension of the image. If there are 3 dimensions, this will be the image height.

9.179.6.20 QString QEImage::dimensionsVariable [read, write]

EPICS variable name (CA PV). This variable is used to read the number of area detector dimensions of the image. If used, this will be 2 (one element per pixel arranged by width and height) or 3 (multiple elements per pixel arranged by pixel, width and height)

9.179.6.21 bool QEImage::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.179.6.22 DisplayAlarmStateOptions QEImage::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.179.6.23 bool QEImage::displayArea1Selection [read, write]

If true, selected area 1 will be displayed on the image. Note, this property is ignored unless the enableArea1Selection property is true.

9.179.6.24 bool QEImage::displayArea2Selection [read, write]

If true, selected area 2 will be displayed on the image. Note, this property is ignored unless the enableArea2Selection property is true.

9.179.6.25 bool QEImage::displayArea3Selection [read, write]

If true, selected area 3 will be displayed on the image. Note, this property is ignored unless the enableArea3Selection property is true.

9.179.6.26 bool QEImage::displayArea4Selection [read, write]

If true, selected area 4 will be displayed on the image. Note, this property is ignored unless the enableArea4Selection property is true.

9.179.6.27 bool QEImage::displayBeamSelection [read, write]

If true, beam selection will be displayed on the image. Note, this property is ignored unless the enableBeamSelection property is true.

9.179.6.28 bool QEImage::displayCursorPixelInfo [read, write]

If true, an area will be presented under the image with textual information about the pixel under the cursor, and for other selections such as selected areas.

9.179.6.29 bool QEImage::displayEllipse [read, write]

If true, the ellipse markup will be displayed on the image.

9.179.6.30 bool QEImage::displayHozSlice1Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the enableHozSlice1Selection property is true.

9.179.6.31 bool QEImage::displayHozSlice2Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the enableHozSlice2Selection property is true.

9.179.6.32 bool QEImage::displayHozSlice3Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the enableHozSlice3Selection property is true.

9.179.6.33 bool QEImage::displayHozSlice4Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the enableHozSlice4Selection property is true.

9.179.6.34 bool QEImage::displayHozSlice5Selection [read, write]

If true, the selected horizontal slice will be displayed on the image. Note, this property is ignored unless the enableHozSlice5Selection property is true.

9.179.6.35 bool QEImage::displayProfileSelection [read, write]

If true, the selected arbirtary line will be displayed on the image. Note, this property is ignored unless the enableProfileSelection property is true.

9.179.6.36 bool QEImage::displayTargetSelection [read, write]

If true, target selection will be displayed on the image. Note, this property is ignored unless the enableTargetSelection property is true.

9.179.6.37 bool QEImage::displayVertSlice1Selection [read, write]

If true, the selected vertical slice 1 will be displayed on the image. Note, this property is ignored unless the enableVertSlice1Selection property is true.

9.179.6.38 bool QEImage::displayVertSlice2Selection [read, write]

If true, the selected vertical slice 2 will be displayed on the image. Note, this property is ignored unless the enableVertSlice2Selection property is true.

9.179.6.39 bool QEImage::displayVertSlice3Selection [read, write]

If true, the selected vertical slice 3 will be displayed on the image. Note, this property is ignored unless the enableVertSlice3Selection property is true.

9.179.6.40 bool QEImage::displayVertSlice4Selection [read, write]

If true, the selected vertical slice 4 will be displayed on the image. Note, this property is ignored unless the enableVertSlice4Selection property is true.

9.179.6.41 bool QEImage::displayVertSlice5Selection [read, write]

If true, the selected vertical slice 5 will be displayed on the image. Note, this property is ignored unless the enableVertSlice5Selection property is true.

9.179.6.42 QColor QEImage::ellipseColor [read, write]

Used to select the color of the ellipse marker.

9.179.6.43 QString QEImage::ellipseHVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse height

9.179.6.44 QString QEImage::ellipseRotationVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse height

9.179.6.45 QString QEImage::ellipseWVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse width.

9.179.6.46 QString QEImage::ellipseXVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse X (center or top left corner of bounding rectangle depending on property ellipseDefinition).

9.179.6.47 QString QEImage::ellipseYVariable [read, write]

EPICS variable name (CA PV). This variable is used to read an ellipse Y (center or top left corner of bounding rectangle depending on property ellipseDefinition).

9.179.6.48 bool QEImage::enableArea1Selection [read, write]

If true, the user will be able to select area 1. These are used for selection of Region of Interests, and for zooming to area 1

9.179.6.49 bool QEImage::enableArea2Selection [read, write]

If true, the user will be able to select area 2. These are used for selection of Region of Interests, and for zooming to area 2

9.179.6.50 bool QEImage::enableArea3Selection [read, write]

If true, the user will be able to select area 3. These are used for selection of Region of Interests, and for zooming to area 3

9.179.6.51 bool QEImage::enableArea4Selection [read, write]

If true, the user will be able to select area 4. These are used for selection of Region of Interests, and for zooming to area 4

9.179.6.52 bool QEImage::enableBeamSelection [read, write]

If true, the user will be able to select points on the image to mark a beam position. This can be used for automatic beam positioning.

9.179.6.53 bool QEImage::enableHozSlice1Selection [read, write]

If true, the option to select a horizontal slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the profileHoz1Variable property. The profile will only be presented to the user if enableHozSlicePresentation property is true.

9.179.6.54 bool QEImage::enableHozSlice2Selection [read, write]

If true, the option to select a second horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileHoz2Variable property.

9.179.6.55 bool QEImage::enableHozSlice3Selection [read, write]

If true, the option to select a third horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileHoz3Variable property.

9.179.6.56 bool QEImage::enableHozSlice4Selection [read, write]

If true, the option to select a fourth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileHoz4Variable property.

9.179.6.57 bool QEImage::enableHozSlice5Selection [read, write]

If true, the option to select a fifth horizontal slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileHoz5Variable property.

9.179.6.58 bool QEImage::enableProfileSelection [read, write]

If true, the option to select an arbitrary line through any part of the image will be available to the user. This will be used to generate a pixel profile.

9.179.6.59 bool QEImage::enableTargetSelection [read, write]

If true, the user will be able to select points on the image to mark a target position. This can be used for automatic beam positioning.

9.179.6.60 bool QEImage::enableVertSlice1Selection [read, write]

If true, the option to select a vertical slice through the image will be available to the user. This will be used to generate a horizontal pixel profile, and write the position of the slice to the optional variable specified by the profileVert1Variable property. The profile will only be presented to the user if enableVertSlicePresentation property is true.

9.179.6.61 bool QEImage::enableVertSlice2Selection [read, write]

If true, the option to select a second vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileVert2Variable property.

9.179.6.62 bool QEImage::enableVertSlice3Selection [read, write]

If true, the option to select a third vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileVert3Variable property.

9.179.6.63 bool QEImage::enableVertSlice4Selection [read, write]

If true, the option to select a fourth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileVert4Variable property.

9.179.6.64 bool QEImage::enableVertSlice5Selection [read, write]

If true, the option to select a fifth vertical slice through the image will be available to the user. This will be used to write the position of the slice to the optional variable specified by the profileVert5Variable property.

9.179.6.65 bool QEImage::externalControls [read, write]

If true, image controls and views such as brightness controls and profile plots are hosted by the application as dock windows, toolbars, etc. Refer to the ContainerProfile class and the windowCustomisation class to see how this class asks an application to act as a host.

9.179.6.66 FormatOptions QEImage::formatOption [read, write]

Video format. EPICS data type size will typically be adequate for the number of bits required (one byte for 8 bits, 2 bytes for 12 and 16 bits), but can be larger (4 bytes for 24 bits.)

9.179.6.67 QString QEImage::formatVariable [read, write]

EPICS variable name (CA PV). This variable is used to read the format of the image.

9.179.6.68 QString QEImage::heightVariable [read, write]

EPICS variable name (CA PV). This variable is used to read the height of the image.

9.179.6.69 bool QEImage::horizontalFlip [read, write]

If true, flip image horizontally.

9.179.6.70 QColor QEImage::hozSlice1Color [read, write]

Used to select the color of the horizontal slice 1 markup.

9.179.6.71 QColor QEImage::hozSlice2Color [read, write]

Used to select the color of the horizontal slice 2 markup.

9.179.6.72 QColor QEImage::hozSlice3Color [read, write]

Used to select the color of the horizontal slice 3 markup.

9.179.6.73 QColor QEImage::hozSlice4Color [read, write]

Used to select the color of the horizontal slice 4 markup.

9.179.6.74 QColor QEImage::hozSlice5Color [read, write]

Used to select the color of the horizontal slice 5 markup.

9.179.6.75 QString QEImage::imageVariable [read, write]

EPICS variable name (CA PV). This variable is used as the source the image waveform.

9.179.6.76 int QEImage::initialHosScrollPos [read, write]

Sets the initial position of the horizontal scroll bar, if present. Used to set up an initial view when zoomed in.

9.179.6.77 unsigned QEImage::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

Bit depth. Note, EPICS data type size will typically be adequate for the number of bits required (one byte for up to 8 bits, 2 bytes for up to 16 bits, etc), but can be larger (for example, 4 bytes for 24 bits) and may be larger than nessesary (4 bytes for 8 bits).

9.179.6.78 QString QEImage::lineProfileArrayVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile array.

9.179.6.79 QString QEImage::lineProfileThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

9.179.6.80 QString QEImage::lineProfileX1Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start X.

9.179.6.81 QString QEImage::lineProfileX2Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end X.

9.179.6.82 QString QEImage::lineProfileY1Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile start Y.

9.179.6.83 QString QEImage::lineProfileY2Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector arbitrary line profile end Y.

9.179.6.84 bool QEImage::logBrightness [read, write]

If true, the image will be displayed using a logarithmic brightness scale.

9.179.6.85 QColor QEImage::profileColor [read, write]

Used to select the color of the arbitrarty profile line markup.

9.179.6.86 QString QEImage::profileHoz1ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile thickness.

9.179.6.87 QString QEImage::profileHoz1Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector first horizontal profile.

9.179.6.88 QString QEImage::profileHoz2ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile thickness.

9.179.6.89 QString QEImage::profileHoz2Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector second horizontal profile.

9.179.6.90 QString QEImage::profileHoz3ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile thickness.

9.179.6.91 QString QEImage::profileHoz3Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector third horizontal profile.

9.179.6.92 QString QEImage::profileHoz4ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile thickness.

9.179.6.93 QString QEImage::profileHoz4Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fourth horizontal profile.

9.179.6.94 QString QEImage::profileHoz5ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile thickness.

9.179.6.95 QString QEImage::profileHoz5Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fifth horizontal profile.

9.179.6.96 QString QEImage::profileHozArrayVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector horizontal profile array.

9.179.6.97 QString QEImage::profileVert1ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

9.179.6.98 QString QEImage::profileVert1Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector first vertical profile.

9.179.6.99 QString QEImage::profileVert2ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

9.179.6.100 QString QEImage::profileVert2Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector second vertical profile.

9.179.6.101 QString QEImage::profileVert3ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

9.179.6.102 QString QEImage::profileVert3Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector third vertical profile.

9.179.6.103 QString QEImage::profileVert4ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

9.179.6.104 QString QEImage::profileVert4Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fourth vertical profile.

9.179.6.105 QString QEImage::profileVert5ThicknessVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

9.179.6.106 QString QEImage::profileVert5Variable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector fifth vertical profile.

9.179.6.107 QString QEImage::profileVertArrayVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the areadetector vertical profile array.

9.179.6.108 QString QEImage::program1 [read, write]

Program to run when a request is made to pass on the current image to the first external application. No attempt to run a program is made if this property is empty. Example: paint.exe

9.179.6.109 QString QEImage::program2 [read, write]

Program to run when a request is made to pass on the current image to the second external application. No attempt to run a program is made if this property is empty. Example: paint.exe

9.179.6.110 ProgramStartupOptionNames QEImage::programStartupOption1 [read, write]

Startup options for the program specified in the 'program1' property. Just run the command, run the command within a terminal, or display the output in QE message system.

9.179.6.111 ProgramStartupOptionNames QEImage::programStartupOption2 [read, write]

Startup options for the program specified in the 'program2' property. Just run the command, run the command within a terminal, or display the output in QE message system.

9.179.6.112 QString QEImage::regionOfInterest1HVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest height.

9.179.6.113 QString QEImage::regionOfInterest1WVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest width.

9.179.6.114 QString QEImage::regionOfInterest1XVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest X position.

9.179.6.115 QString QEImage::regionOfInterest1YVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the first region of interest Y position.

9.179.6.116 QString QEImage::regionOfInterest2HVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest height.

9.179.6.117 QString QEImage::regionOfInterest2WVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest width.

9.179.6.118 QString QEImage::regionOfInterest2XVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest X position.

9.179.6.119 QString QEImage::regionOfInterest2YVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the second region of interest Y position.

9.179.6.120 QString QEImage::regionOfInterest3HVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest height.

9.179.6.121 QString QEImage::regionOfInterest3WVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest width.

9.179.6.122 QString QEImage::regionOfInterest3XVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest X position.

9.179.6.123 QString QEImage::regionOfInterest3YVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the third region of interest Y position.

9.179.6.124 QString QEImage::regionOfInterest4HVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest height.

9.179.6.125 QString QEImage::regionOfInterest4WVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest width.

9.179.6.126 QString QEImage::regionOfInterest4XVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest X position.

9.179.6.127 QString QEImage::regionOfInterest4YVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the fourth region of interest Y position.

9.179.6.128 ResizeOptions QEImage::resizeOption [read, write]

Resize option. Zoom to zoom to the percentage given by the zoom property, or fit to the window size.

9.179.6.129 RotationOptions QEImage::rotation [read, write]

Image rotation option.

9.179.6.130 bool QEImage::showTime [read, write]

If true, the image timestamp will be written in the top left of the image.

9.179.6.131 QString QEImage::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.179.6.132 QColor QEImage::targetColor [read, write]

Used to select the color of the target marker.

9.179.6.133 QString QEImage::targetTriggerVariable [read, write]

EPICS variable name (CA PV). This variable is used to write a 'trigger' to initiate movement of the target into the beam as defined by the target and beam X and Y positions.

9.179.6.134 QString QEImage::targetXVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the selected target X position.

9.179.6.135 QString QEImage::targetYVariable [read, write]

EPICS variable name (CA PV). This variable is used to write the selected target Y position.

9.179.6.136 QColor QEImage::timeColor [read, write]

Used to select the color of the timestamp.

9.179.6.137 QString QEImage::URL [read, write]

MPEG stream URL. If this is specified, this will be used as the source of the image in preference to variables (variables defining the image data, width, and height will be ignored)

9.179.6.138 bool QEImage::useFalseColour [read, write]

If true, the apply false colour to the image.

9.179.6.139 UserLevels QEImage::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.179.6.140 QString QEImage::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.179.6.141 QString QEImage::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.179.6.142 QString QEImage::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example,

'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.179.6.143 UserLevels QEImage::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.179.6.144 bool QEImage::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from QEToolTip.

9.179.6.145 QString QEImage::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'CAM=1, NAME = "Image 1"' These substitutions are applied to all the variable names.

9.179.6.146 bool QEImage::verticalFlip [read, write]

If true, flip image vertically.

9.179.6.147 QColor QEImage::vertSlice1Color [read, write]

Used to select the color of the vertical slice 1 markup.

9.179.6.148 QColor QEImage::vertSlice2Color [read, write]

Used to select the color of the vertical slice 2 markup.

9.179.6.149 QColor QEImage::vertSlice3Color [read, write]

Used to select the color of the vertical slice 3 markup.

9.179.6.150 QColor QEImage::vertSlice4Color [read, write]

Used to select the color of the vertical slice 4 markup.

9.179.6.151 QColor QEImage::vertSlice5Color [read, write]

Used to select the color of the vertical slice 5 markup.

9.179.6.152 bool QEImage::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

9.179.6.153 QString QEImage::widthVariable [read, write]

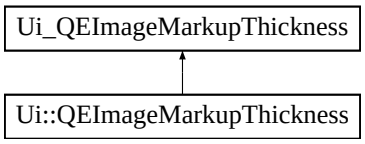
EPICS variable name (CA PV). This variable is used to read the width of the image.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImage.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEImage.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImage.cpp

9.180 Ui::QEImageMarkupThickness Class Reference

Inheritance diagram for Ui::QEImageMarkupThickness::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEImageMarkupThickness.h

9.181 QEImageMarkupThickness Class Reference

Public Member Functions

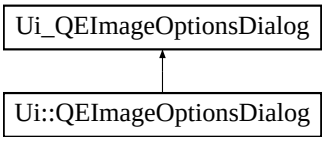
- **QEImageMarkupThickness** (QWidget ∗parent=0)
- void **setThickness** (unsigned int thicknessIn)
- unsigned int **getThickness** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImageMarkupThickness.
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImageMarkupThickness.

9.182 Ui::QEImageOptionsDialog Class Reference

Inheritance diagram for Ui::QEImageOptionsDialog::

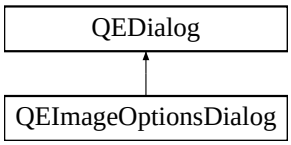


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEImageOptionsDialog.h

9.183 QEImageOptionsDialog Class Reference

Inheritance diagram for QEImageOptionsDialog::



Signals

- void **optionChange** (imageContextMenu::imageContextMenuOptions option, bool checked)

Public Member Functions

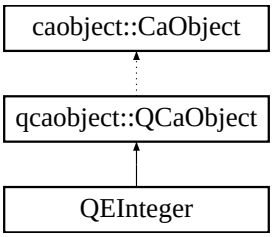
- **QEImageOptionsDialog** (QWidget ∗parent=0)
- void **initialise** ()
- void **optionSet** (imageContextMenu::imageContextMenuOptions option, bool checked)
- bool **optionGet** (imageContextMenu::imageContextMenuOptions option)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImageOptionsDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEImageOptionsDialog.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/QEImageOptionsDialog.cpp

9.184 QEInteger Class Reference

Inheritance diagram for QEInteger::



Public Slots

- void **writeInteger** (const long &data)
- void **writeIntegerElement** (const long &data)
- void **writeInteger** (const QVector< 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEInteger.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEInteger.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEInteger.cpp

9.185 QEIntegerArray Class Reference

`#include <QEIntegerArray.h>`

Public Member Functions

- **QEIntegerArray** (int size)
- **QEIntegerArray** (int size, const long &t)
- **QEIntegerArray** (const QVector< long > &other)
- QEIntegerArray & **operator=** (const QVector< long > &other)
- long **minimumValue** (const long &defaultValue=0)
- long **maximumValue** (const long &defaultValue=0)

9.185.1 Detailed Description

This class provides short hand for QVector<long> together with some basic long vector operations.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEIntegerArray.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEIntegerArray.cpp

9.186 QEIntegerFormatting Class Reference

`#include <QEIntegerFormatting.h>`

Public Member Functions

- QEIntegerFormatting ()

  Constructor.

- long formatInteger (const QVariant &value, int arrayIndex=0)
- QVector< long > formatIntegerArray (const QVariant &value)
- QVariant formatValue (const long &integerValue, generic::generic_types valueType)
- QVariant **formatValue** (const QVector< long > &integerValue, generic::generic_types valueType)
- 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.

9.186.1 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.

9.186.2 Member Function Documentation

9.186.2.1 long QEIntegerFormatting::formatInteger (const QVariant & *value*, int *arrayIndex* = 0)

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.

9.186.2.2 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.

9.186.2.3 QVariant QEIntegerFormatting::formatValue (const long & *integerValue*, generic::generic_types *valueType*)

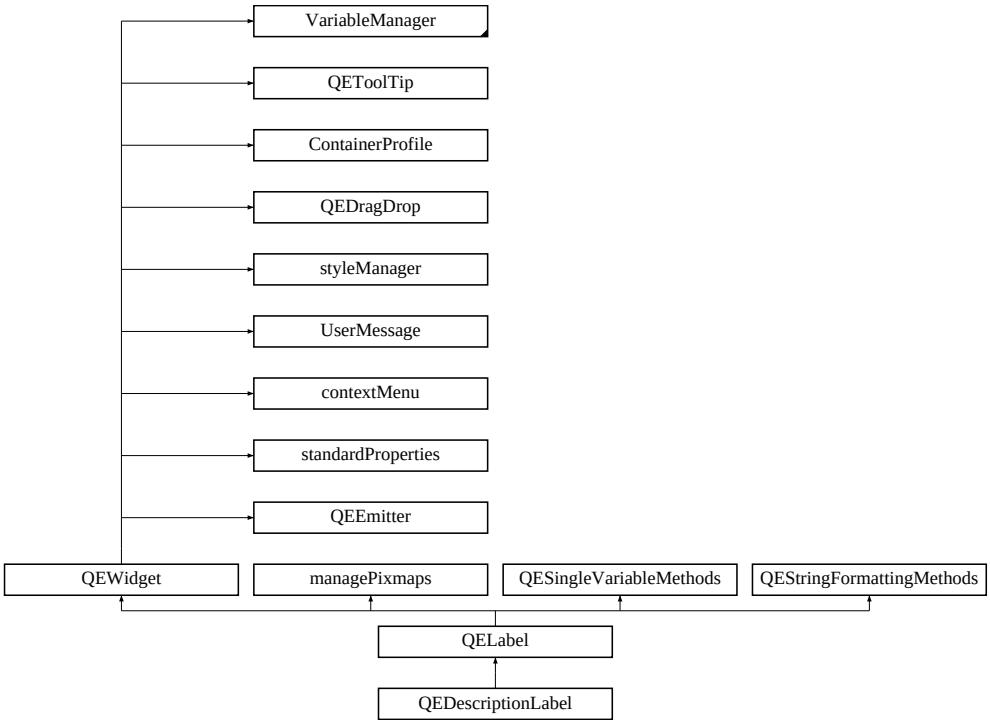
Given an integer value, format it as a data value of the specified type, according to the formatting instructions held by the class. This is used when writing integer data to a QCaObject.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEIntegerFormatting.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEIntegerFormatting.cpp

9.187 QELabel Class Reference

`#include <QELabel.h>`Inheritance diagram for QELabel::



Public Types

- enum updateOptions { UPDATE_TEXT, UPDATE_PIXMAP }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }
- enum Formats {

  Default = QEStringFormatting::FORMAT_DEFAULT, Floating = QEStringFormatting::FORMAT_FLOATING, Integer = QEStringFormatting::FORMAT_INTEGER, UnsignedInteger = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_-NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Under-

score = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }
- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }
- enum UpdateOptions { Text = QELabel::UPDATE_TEXT, Picture = QELabel::UPDATE_PIXMAP }

  User friendly enumerations for updateOption property - refer to QELabel::updateOptions for details.

Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void requestResend ()

  Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

Public Member Functions

- QELabel (QWidget *parent=0)
- QELabel (const QString &variableName, QWidget *parent=0)
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setFormatProperty (Formats format)

  Access function for format property - refer to format property for details.

- Formats getFormatProperty ()

  Access function for format property - refer to format property for details.

- void setSeparatorProperty (const Separators notation)

  Access function for separator property - refer to separator property for details.

- Separators getSeparatorProperty () const

  Access function for separator property - refer to separator property for details.

- void setNotationProperty (Notations notation)

  Access function for notation property - refer to notation property for details.

- Notations getNotationProperty ()

  Access function for notation property - refer to notation property for details.

- void setArrayActionProperty (ArrayActions arrayAction)

  Access function for arrayAction property - refer to arrayAction property for details.

- ArrayActions getArrayActionProperty ()

  Access function for arrayAction property - refer to arrayAction property for details.

- void setUpdateOptionProperty (UpdateOptions updateOption)

  Access function for updateOption property - refer to updateOption property for details.

- UpdateOptions getUpdateOptionProperty ()

  Access function for updateOption property - refer to updateOption property for details.

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction
- QString **text**
- UpdateOptions updateOption
- QPixmap pixmap0
- QPixmap pixmap1
- QPixmap pixmap2
- QPixmap pixmap3
- QPixmap pixmap4
- QPixmap pixmap5
- QPixmap pixmap6
- QPixmap pixmap7

9.187.1 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 tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.187.2 Member Enumeration Documentation

9.187.2.1 enum QELabel::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.187.2.2 enum QELabel::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.187.2.3 enum QELabel::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.187.2.4 enum QELabel::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.187.2.5 enum QELabel::Separators

User friendly enumerations for seprator property - refer to QEStringFormatting::formats for details.

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.187.2.6 enum QELabel::UpdateOptions

User friendly enumerations for updateOption property - refer to QELabel::updateOptions for details.

Enumerator:

Text Data updates will update the label text.

Picture Data updates will update the label icon.

9.187.2.7 enum QELabel::updateOptions

Options for updating the label. The formatted text is used to update the label text, or select a background pixmap.

Enumerator:

UPDATE_TEXT Update the label text.

UPDATE_PIXMAP Update the label background pixmap.

9.187.2.8 enum QELabel::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.187.3 Constructor & Destructor Documentation

9.187.3.1 QELabel::QELabel (QWidget * *parent* = 0)

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.187.3.2 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.

9.187.4 Member Function Documentation

9.187.4.1 void QELabel::dbValueChanged () [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.

9.187.4.2 void QELabel::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.187.5 Property Documentation

9.187.5.1 bool QELabel::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.187.5.2 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.

9.187.5.3 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'.

9.187.5.4 int QELabel::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.187.5.5 QString QELabel::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.187.5.6 bool QELabel::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.187.5.7 DisplayAlarmStateOptions QELabel::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.187.5.8 int QELabel::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.187.5.9 bool QELabel::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.187.5.10 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).

9.187.5.11 unsigned QELabel::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

9.187.5.12 bool QELabel::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.187.5.13 QString QELabel::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.187.5.14 Notations QELabel::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.187.5.15 QPixmap QELabel::pixmap0 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 0.

9.187.5.16 QPixmap QELabel::pixmap1 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 1.

9.187.5.17 QPixmap QELabel::pixmap2 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 2.

9.187.5.18 QPixmap QELabel::pixmap3 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 3.

9.187.5.19 QPixmap QELabel::pixmap4 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 4.

9.187.5.20 QPixmap QELabel::pixmap5 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 5.

9.187.5.21 QPixmap QELabel::pixmap6 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 6.

9.187.5.22 QPixmap QELabel::pixmap7 [read, write]

Pixmap displayed when updateOption property is 'Picture' and data is interpreted as 7.

9.187.5.23 int QELabel::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.187.5.24 int QELabel::radix [read, write]

Base used for when formatting integers. Default is 10 (duh!)

9.187.5.25 Separators QELabel::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.187.5.26 QString QELabel::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.187.5.27 bool QELabel::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.187.5.28 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.

9.187.5.29 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.

9.187.5.30 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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.187.5.31 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.

9.187.5.32 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.

9.187.5.33 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.

9.187.5.34 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'.

9.187.5.35 QString QELabel::variable [read, write]

EPICS variable name (CA PV)

9.187.5.36 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.

9.187.5.37 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 are also used for other purposes.

9.187.5.38 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.

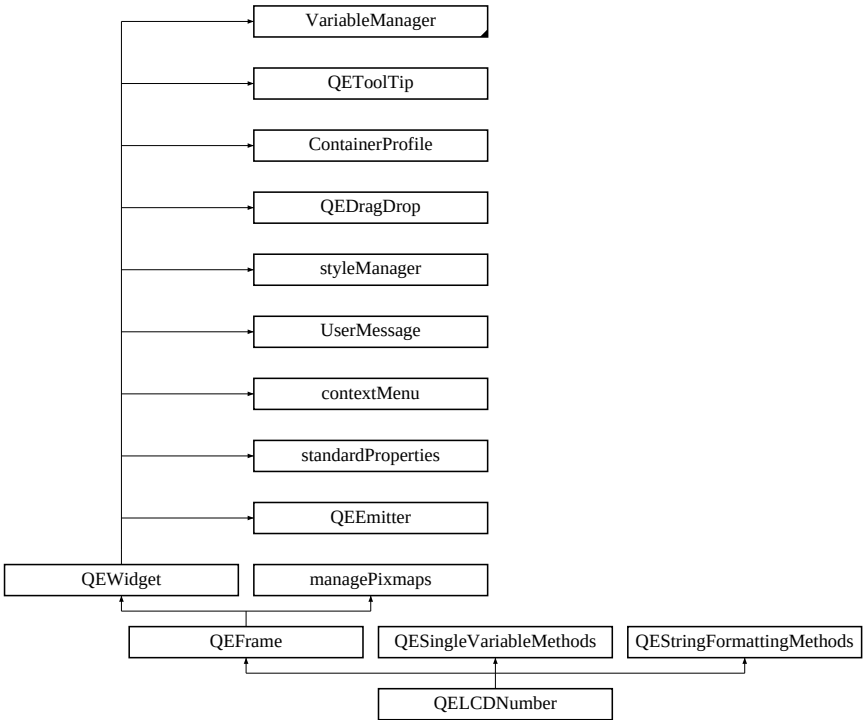
The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELabel/QELabel.h

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QELabel.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELabel/QELabel.cpp

9.188 QELCDNumber Class Reference

`#include <QELCDNumber.h>`Inheritance diagram for QELCDNumber::



Public Types

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)
- void requestResend ()

Public Member Functions

- QELCDNumber (QWidget *parent=0)
- QELCDNumber (const QString &variableName, QWidget *parent=0)
- virtual ~QELCDNumber ()

  Destruction.

- QSize sizeHint () const

  Size hint for designer.

- void setNotationProperty (Notations notation)

  Access function for notation property - refer to notation property for details.

- Notations getNotationProperty () const

  Access function for notation property - refer to notation property for details.

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **stringFormattingChange** ()
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant v)

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool **smallDecimalPoint**
- QLCDNumber::SegmentStyle **segmentStyle**
- int precision
- bool useDbPrecision
- Notations notation

9.188.1 Detailed Description

This class provides an EPICS aware extention to the QLCDNumber widget.

9.188.2 Member Enumeration Documentation

9.188.2.1 enum QELCDNumber::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.188.3 Constructor & Destructor Documentation

9.188.3.1 QELCDNumber::QELCDNumber (QWidget * *parent* = 0) [explicit]

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.188.3.2 QELCDNumber::QELCDNumber (const QString & *variableName*, QWidget * *parent* = 0) [explicit]

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.

9.188.4 Member Function Documentation

9.188.4.1 void QELCDNumber::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.188.4.2 void QELCDNumber::dbValueChanged () [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.

9.188.4.3 void QELCDNumber::requestResend () [signal]

Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

9.188.5 Property Documentation

9.188.5.1 int QELCDNumber::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.188.5.2 int QELCDNumber::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.188.5.3 Notations QELCDNumber::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.188.5.4 int QELCDNumber::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.188.5.5 bool QELCDNumber::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

9.188.5.6 QString QELCDNumber::variable [read, write]

EPICS variable name (CA PV)

9.188.5.7 QString QELCDNumber::variableSubstitutions [read, write]

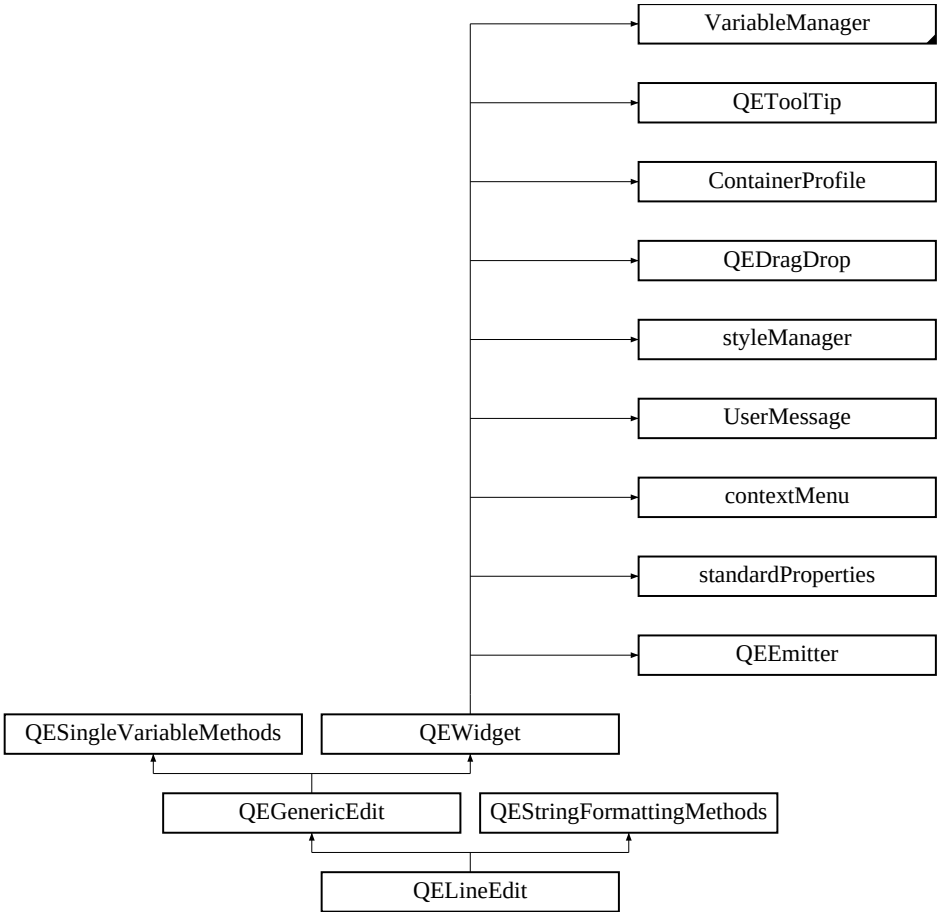
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELCDNumber/QELCD
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QELCDNumber.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELCDNumber/QELCD

9.189 QELineEdit Class Reference

Inheritance diagram for QELineEdit::



Public Types

- enum Formats {

  Default = QEStringFormatting::FORMAT_DEFAULT, Floating = QEStringFormatting::FORMAT_FLOATING, Integer = QEStringFormatting::FORMAT_INTEGER, UnsignedInteger = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Underscore = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void userChange (const QString &oldValue, const QString &newValue, const QString &lastValue)

  Internal use only. Used by QEConfiguredLayout to be notified when one of its widgets has written something.

- void requestResend ()

  Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

Public Member Functions

- void setFormatProperty (Formats format)

  Access function for format property - refer to format property for details.

- Formats getFormatProperty ()

  Access function for format property - refer to format property for details.

- void setSeparatorProperty (const Separators notation)

  Access function for separator property - refer to separator property for details.

- Separators getSeparatorProperty () const

  Access function for separator property - refer to separator property for details.

- void setNotationProperty (Notations notation)

  Access function for notation property - refer to notation property for details.

- Notations getNotationProperty ()

Access function for notation property - refer to notation property for details.

- void setArrayActionProperty (ArrayActions arrayAction)

  Access function for arrayAction property - refer to arrayAction property for details.

- ArrayActions getArrayActionProperty ()

  Access function for arrayAction property - refer to arrayAction property for details.

- QELineEdit (QWidget *parent=0)
- QELineEdit (const QString &variableName, QWidget *parent=0)

Properties

- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction

9.189.1 Member Enumeration Documentation

9.189.1.1 enum QELineEdit::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.189.1.2 enum QELineEdit::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.189.1.3 enum QELineEdit::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.189.1.4 enum QELineEdit::Separators

User friendly enumerations for seprator property - refer to QEStringFormatting::formats for details.

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.189.2 Constructor & Destructor Documentation

9.189.2.1 QELineEdit::QELineEdit (QWidget * *parent* = 0)

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.189.2.2 QELineEdit::QELineEdit (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.189.3 Member Function Documentation

9.189.3.1 void QELineEdit::dbValueChanged () [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.

9.189.4 Property Documentation

9.189.4.1 bool QELineEdit::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.189.4.2 ArrayActions QELineEdit::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.189.4.3 bool QELineEdit::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.189.4.4 Formats QELineEdit::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.189.4.5 bool QELineEdit::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.189.4.6 QString QELineEdit::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.189.4.7 Notations QELineEdit::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.189.4.8 int QELineEdit::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.189.4.9 int QELineEdit::radix [read, write]

Base used for when formatting integers. Default is 10 (duh!)

9.189.4.10 Separators QELineEdit::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.189.4.11 bool QELineEdit::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.189.4.12 bool QELineEdit::useDbPrecision [read, write]

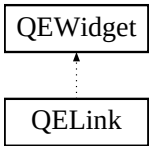
If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELineEdit/QELineEdit.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QELineEdit.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELineEdit/QELineEdit.cpp

9.190 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 int &in)
- void **in** (const long &in)
- void **in** (const qlonglong &in)
- void **in** (const double &in)
- void **in** (const QString &in)
- void **autoFillBackground** (const bool &enable)

Signals

- void **out** (const bool &out)
- void **out** (const int &out)
- void **out** (const long &out)
- void **out** (const qlonglong &out)
- void **out** (const double &out)
- void **out** (const QString &out)

Public Member Functions

- **QELink** (QWidget ∗parent=0)
- void **setCondition** (conditions conditionIn)
- conditions **getCondition** ()
- void **setComparisonValue** (QString comparisonValue)
- QString **getComparisonValue** ()
- void **setSignalTrue** (bool signalTrue)
- bool **getSignalTrue** ()
- void **setSignalFalse** (bool signalFalse)
- bool **getSignalFalse** ()
- void **setOutTrueValue** (QString outTrueValue)
- QString **getOutTrueValue** ()
- void **setOutFalseValue** (QString outFalseValue)
- QString **getOutFalseValue** ()
- void **setConditionProperty** (ConditionNames condition)
- ConditionNames **getConditionProperty** ()

Protected Attributes

- conditions **condition**
- QVariant **comparisonValue**
- bool **signalTrue**
- bool **signalFalse**
- QVariant **outTrueValue**
- QVariant **outFalseValue**

Properties

- ConditionNames **condition**
- QString **comparisonValue**
- QString **outTrueValue**
- QString **outFalseValue**
- bool **runVisible**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELink/QELink.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QELink.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELink/QELink.cpp

9.191 QELocalEnumeration Class Reference

`#include <QELocalEnumeration.h>`

Classes

- class **localEnumerationItem**

Public Member Functions

- QELocalEnumeration ()
- QELocalEnumeration (const QString &localEnumeration)
- void setLocalEnumeration (const QString &localEnumeration)
- QString getLocalEnumeration () const
- bool isDefined () const
- QString valueToText (const QVariant &value, bool &match) const
- QVariant textToValue (const QString &text, bool &ok) const
- int textToInt (const QString &text, bool &ok) const
- double textToDouble (const QString &text, bool &ok) const

9.191.1 Detailed Description

This class allows a user defined two-way value to enumeration map. The map is define using a single string, typically a widget property string. This may then be used to replace the enumeration values provided by EPICS and/or provide an enueration set of more that 16 values. See setLocalEnumeration() for the use of 'localEnumeration'.

This functionality that this class provided was formerly embedded within QEString-Formatting.

9.191.2 Constructor & Destructor Documentation

9.191.2.1 QELocalEnumeration::QELocalEnumeration ()

Constructors

9.191.2.2 QELocalEnumeration::QELocalEnumeration (const QString & *localEnumeration*)

Constructor with localEnumeration

9.191.3 Member Function Documentation

9.191.3.1 QString QELocalEnumeration::getLocalEnumeration () const

Get the local enumeration strings. See setLocalEnumeration() for the use of 'localEnumeration'.

9.191.3.2 bool QELocalEnumeration::isDefined () const

Evaluates: getLocalEnumeration.count() > 0

9.191.3.3 void QELocalEnumeration::setLocalEnumeration (const QString & *localEnumeration*)

Parse the local enumeration string.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

Will completely re-initialises the object.

9.191.3.4 double QELocalEnumeration::textToDouble (const QString & *text*, bool & *ok*) const

Generate a double value given a string, using formatting defined within this class. If the value can be formatted the formatted value is returned and 'ok' is true. If the value

can't be formatted then 0.0 is returned and 'ok' is false.

9.191.3.5 int QELocalEnumeration::textToInt (const QString & *text*, bool & *ok*) const

Generate an integer value given a string, using formatting defined within this class. If the value can be formatted the formatted value is returned and 'ok' is true. If the value can't be formatted then 0 is returned and 'ok' is false.

9.191.3.6 QVariant QELocalEnumeration::textToValue (const QString & *text*, bool & *ok*) const

Generate a value given a string, using formatting defined within this class. If the value can be formatted the formatted value is returned and 'ok' is true. If the value can't be formatted an error string is returned and 'ok' is false

9.191.3.7 QString QELocalEnumeration::valueToText (const QVariant & *value*, bool & *match*) const

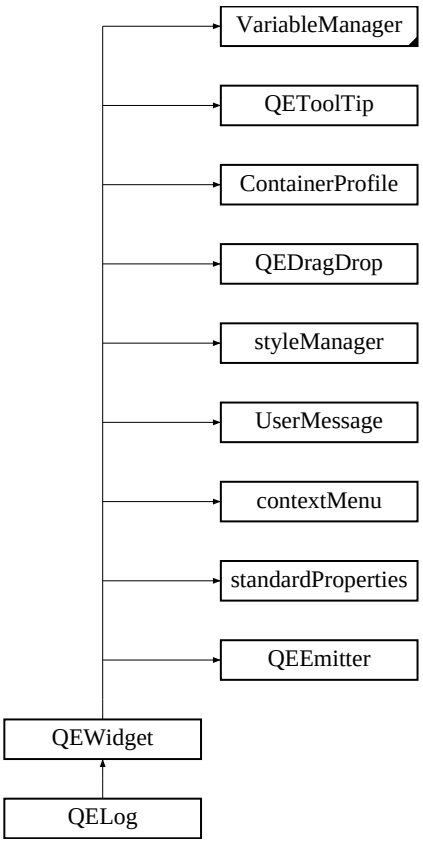
Format a variant value using local enumeration list. If the value is numeric, then the value is compared to the numeric interpretation of the enumeration values, if the value is textual, then the value is compared to the textual enumeration values.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QELocalEnumeration.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QELocalEnumeration.cpp

9.192 QELog Class Reference

Inheritance diagram for QELog::



Public Types

- enum **optionsLayoutProperty** { **Top**, **Bottom**, **Left**, **Right** }
- enum **MessageFilterOptions** { **Any** = UserMessage::MESSAGE_FILTER_ANY, **Match** = UserMessage::MESSAGE_FILTER_MATCH, **None** = UserMessage::MESSAGE_FILTER_NONE }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Public Member Functions

- **QELog** (QWidget *parent=0)
- void **setShowColumnTime** (bool pValue)
- bool **getShowColumnTime** ()
- void **setShowColumnType** (bool pValue)
- bool **getShowColumnType** ()
- void **setShowColumnMessage** (bool pValue)
- bool **getShowColumnMessage** ()
- void **setShowMessageFilter** (bool pValue)
- bool **getShowMessageFilter** ()
- void **setShowClear** (bool pValue)
- bool **getShowClear** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setScrollToBottom** (bool pValue)
- bool **getScrollToBottom** ()
- void **setInfoColor** (QColor pValue)
- QColor **getInfoColor** ()
- void **setWarningColor** (QColor pValue)
- QColor **getWarningColor** ()
- void **setErrorColor** (QColor pValue)
- QColor **getErrorColor** ()
- void **clearLog** ()
- void **addLog** (int pType, QString pMessage)
- void **refreshLog** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- MessageFilterOptions **getMessageFormFilter** ()
- void **setMessageFormFilter** (MessageFilterOptions messageFormFilter)
- MessageFilterOptions **getMessageSourceFilter** ()
- void **setMessageSourceFilter** (MessageFilterOptions messageSourceFilter)
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Protected Attributes

- _QTableWidgetLog * **qTableWidgetLog**
- QCheckBox * **qCheckBoxInfoMessage**
- QCheckBox * **qCheckBoxWarningMessage**
- QCheckBox * **qCheckBoxErrorMessage**
- QPushButton * **qPushButtonClear**
- QPushButton * **qPushButtonSave**
- QColor **qColorInfo**
- QColor **qColorWarning**
- QColor **qColorError**
- bool **scrollToBottom**
- int **optionsLayout**

Properties

- bool **showColumnTime**
- bool **showColumnType**
- bool **showColumnMessage**
- bool **showMessageFilter**
- bool **showClear**
- bool **showSave**
- optionsLayoutProperty **optionsLayout**
- QColor **infoColor**
- QColor **warningColor**
- QColor **errorColor**
- MessageFilterOptions **messageFormFilter**
- MessageFilterOptions **messageSourceFilter**
- bool variableAsToolTip

- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption

9.192.1 Member Enumeration Documentation

9.192.1.1 enum QELog::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.192.1.2 enum QELog::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.192.2 Member Function Documentation

9.192.2.1 void QELog::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.192.3 Property Documentation

9.192.3.1 bool QELog::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEDragDrop.

9.192.3.2 QString QELog::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.192.3.3 bool QELog::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.192.3.4 DisplayAlarmStateOptions QELog::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.192.3.5 unsigned QELog::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

9.192.3.6 QString QELog::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.192.3.7 UserLevels QELog::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.192.3.8 QString QELog::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.192.3.9 QString QELog::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.192.3.10 QString QELog::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.192.3.11 UserLevels QELog::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or 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'.

9.192.3.12 bool QELog::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from QEToolTip.

9.192.3.13 bool QELog::visible [read, write]

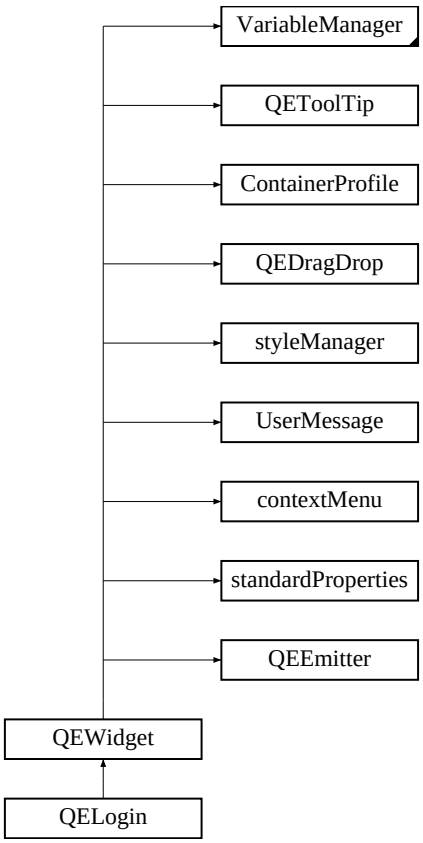
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELog/QELog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELog/QELog.cpp

9.193 QELogin Class Reference

Inheritance diagram for QELogin::



Signals

- void **login** ()

Public Member Functions

- **QELogin** (QWidget *pParent=0)
- bool **login** (userLevelTypes::userLevels level, QString password)
- QString **getPriorityUserPassword** ()
- QString **getPriorityScientistPassword** ()
- QString **getPriorityEngineerPassword** ()
- void **setUserPassword** (QString pValue)
- QString **getUserPassword** ()
- void **setScientistPassword** (QString pValue)
- QString **getScientistPassword** ()

- void **setEngineerPassword** (QString pValue)
- QString **getEngineerPassword** ()
- void **setCompactStyle** (bool compactStyle)
- bool **getCompactStyle** ()
- void **setStatusOnly** (bool statusOnlyIn)
- bool **getStatusOnly** ()
- QString **getUserTypeName** (userLevelTypes::userLevels type)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QELogin.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.cpp

9.194 QELoginDialog Class Reference

Public Member Functions

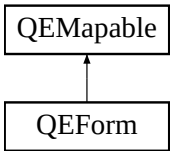
- **QELoginDialog** (QELogin ∗ownerIn)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QELogin/QELogin.cpp

9.195 QEMapable Class Reference

Inheritance diagram for QEMapable::



Public Member Functions

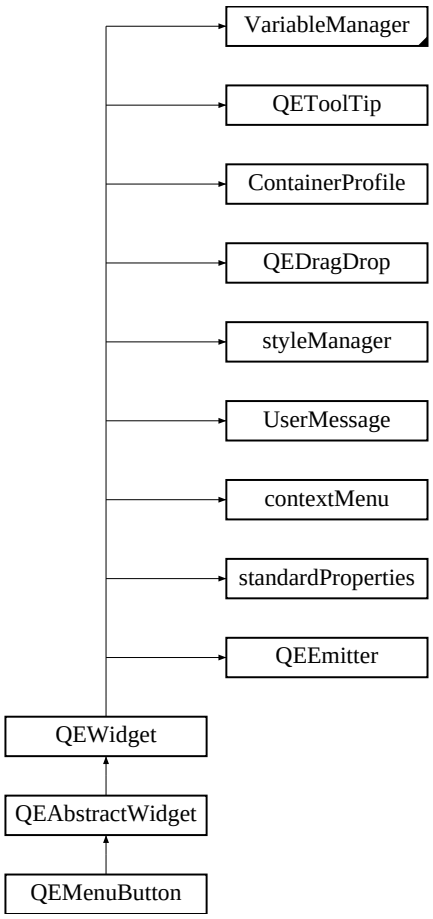
- **QEMapable** (QWidget ∗ownerIn)
- bool **setFormHandle** (const QEFormMapper::FormHandles &formHandle)
- QEFormMapper::FormHandles **getFormHandle** () const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFormMapper.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEFormMapper.cpp

9.196 QEMenuButton Class Reference

`#include <QEMenuButton.h>`Inheritance diagram for QEMenuButton::



Signals

- void newGui (const QEActionRequests &request)

  Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

Public Member Functions

- **QEMenuButton** (QWidget *parent=0)
- void **setSubstitutionsProperty** (const QString &substitutions)
- QString **getSubstitutionsProperty** () const
- void **setLabelTextProperty** (const QString &labelText)

- QString **getLabelTextProperty** () const
- void **setMenuString** (const QString &s)
- QString **getMenuString** () const

Protected Member Functions

- QSize **sizeHint** () const
- bool **eventFilter** (QObject *watched, QEvent *event)

Protected Attributes

- QMenu * **buttonMainMenu**
- QString **theMenuString**

Properties

- bool allowDrop
- bool variableAsToolTip
- DisplayAlarmStateOptions displayAlarmStateOption
- QString defaultSubstitutions
- QString labelText

  Label Text.

- QString menuEntries

  Specifies the menu entry values, encoded and an XML string.

9.196.1 Detailed Description

This widget provides a menu attached to a button. Each menu/submenu can tigger same items as a QEPushButton, i.e: a) write a value to a PV (click only); and/or b) run a nominated program (with arguments); and/or c) open a specified ui file.

9.196.2 Property Documentation

9.196.2.1 bool QEMenuButton::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 QEAbstractWidget.

9.196.2.2 QString QEMenuButton::defaultSubstitutions [read, write]

Default 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 progran names and arguments etc.

9.196.2.3 DisplayAlarmStateOptions QEMenuButton::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from QEAbstractWidget.

9.196.2.4 bool QEMenuButton::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 QEAbstractWidget.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEMenuButton.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu

9.197 QEMenuButtonData Class Reference

Public Member Functions

- QVariant **toVariant** () const
- bool **setValue** (const QVariant &data)

Static Public Member Functions

- static QString **psoToString** (const applicationLauncher::programStartupOptions value)
- static applicationLauncher::programStartupOptions **stringToPso** (const QString &image)
- static QString **optionToString** (const QEActionRequests::Options value)
- static QEActionRequests::Options **stringToOption** (const QString &image)
- static QString **formatToString** (const QEStringFormatting::formats value)
- static QEStringFormatting::formats **stringToFormat** (const QString &image)
- static QString **join** (const QStringList &x)
- static QStringList **split** (const QString &x)

Public Attributes

- bool **separator**
- QString **programName**
- QStringList **programArguments**
- applicationLauncher::programStartupOptions **programStartupOption**
- QString **uiFilename**
- QString **prioritySubstitutions**
- QEActionRequests::Options **creationOption**
- QString **customisationName**
- QString **variable**
- QString **variableValue**
- QEStringFormatting::formats **format**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenuButtonData.
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenuButtonData.

9.198 QEMenuButtonItem Class Reference

`#include <QEMenuButtonItem.h>`

Public Member Functions

- **QEMenuButtonItem** (const QString &name, const bool isSubMenuContainer, QEMenuButtonItem *parent=NULL)
- QString **getName** () const
- bool **getIsSubMenuContainer** () const
- QDomElement **createDomElement** (QDomDocument &doc) const
- bool **extractFromDomElement** (const QDomElement &element)
- QAction * **constructAction** (QMenu *parent)
- QMenu * **constructMenu** (QMenu *parent)
- int **columnCount** () const
- QEMenuButtonItem * **getChild** (const int position) const
- QEMenuButtonItem * **getParent** () const
- int **childCount** () const
- int **childPosition** () const
- QVariant **getData** (const int column) const
- bool **insertChild** (const int position, QEMenuButtonItem *child)
- bool **removeChildren** (const int position, const int count)

Public Attributes

- QString **name**
- bool **isSubMenuContainer**
- QEMenuButtonData **data**

9.198.1 Detailed Description

This class is based on the TreeItem example specified in: http://qt-project.org/doc/qt-4.8/itemviews-editabletreemodel.html

Copyright (C) 2013 Digia Plc and/or its subsidiary(-ies). Contact: http://www.qt-project.org/legal

Differences: there is no itemData variant array - these values calculated as an when needed. Also some function name changes such as parent => getParent and some changes just to follow QE prefered style.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu

9.199 QEMenuButtonModel Class Reference

Public Member Functions

- **QEMenuButtonModel** (QObject *parent=0)
- bool **parseXml** (const QString &xml)
- QString **serialiseXml** () const
- bool **constructMenu** (QMenu *rootMenu)
- QEMenuButtonItem * **indexToItem** (const QModelIndex &index) const
- QModelIndex **getIndex** (const QEMenuButtonItem *item) const
- bool **addItemToModel** (QEMenuButtonItem *item, QEMenuButtonItem *parentItem, const int position=-1)
- bool **removeItemFromModel** (QEMenuButtonItem *item)
- void **modelUpdated** ()
- void **itemUpdated** (const QEMenuButtonItem *item)

Static Public Member Functions

- static QString **emptyXml** ()

Protected Member Functions

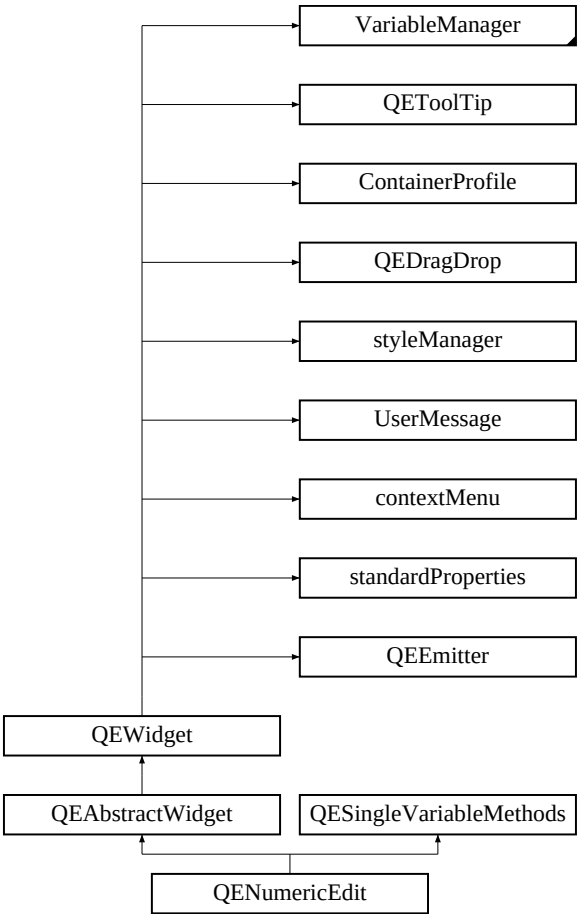
- QVariant **data** (const QModelIndex &index, int role) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const
- QModelIndex **parent** (const QModelIndex &child) const
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const
- Qt::ItemFlags **flags** (const QModelIndex &index) const
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::EditRole)
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role=Qt::EditRole)
- bool **insertRows** (int position, int rows, const QModelIndex &parent=QModelIndex())
- bool **removeRows** (int position, int rows, const QModelIndex &parent=QModelIndex())
- Qt::DropActions **supportedDropActions** () const
- QStringList **mimeTypes** () const
- QMimeData * **mimeData** (const QModelIndexList &indexes) const
- bool **dropMimeData** (const QMimeData *data, Qt::DropAction action, int row, int column, const QModelIndex &parent)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEMenuButton/QEMenu

9.200 QENumericEdit Class Reference

Inheritance diagram for QENumericEdit::



Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)

- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- QENumericEdit (QWidget *parent=0)
- QENumericEdit (const QString &variableName, QWidget *parent=0)
- virtual ~QENumericEdit ()

  Destruction.

- void writeNow ()
- double **getValue** () const
- void **setValue** (const double value, const bool isUserUpdate=false)
- double **getNumericValue** () const
- void **setNumericValue** (const double value, const bool isUserUpdate=false)
- void **setAutoScale** (const bool)
- bool **getAutoScale** () const
- void **setAddUnits** (const bool)
- bool **getAddUnits** () const
- void **setWriteOnLoseFocus** (const bool)
- bool **getWriteOnLoseFocus** () const
- void **setWriteOnEnter** (const bool)
- bool **getWriteOnEnter** () const
- void **setWriteOnFinish** (const bool)
- bool **getWriteOnFinish** () const
- void **setWriteOnChange** (const bool)
- bool **getWriteOnChange** () const
- void **setConfirmWrite** (const bool)
- bool **getConfirmWrite** () const
- void **setAllowFocusUpdate** (const bool)
- bool **getAllowFocusUpdate** () const
- void **setLeadingZeros** (const int value)
- int **getLeadingZeros** () const
- void **setPrecision** (const int value)
- int **getPrecision** () const
- void **setMinimum** (const double value)
- double **getMinimum** () const
- void **setMaximum** (const double value)
- double **getMaximum** () const
- QString **getCleanText** () const

Protected Member Functions

- bool **eventFilter** (QObject ∗watched, QEvent ∗event)
- void **focusInEvent** (QFocusEvent ∗event)
- QMenu ∗ getDefaultContextMenu ()

  Flag indicating this widget is running inside Qt's 'designer'.

- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject ∗ createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool frame
- Qt::Alignment alignment
- QString cleanText
- bool autoScale
- QNumericEdit::Notations notation
- QEFixedPointRadix::Radicies radix
- QEFixedPointRadix::Separators separator
- int leadingZeros
- int precision
- double minimum
- double maximum
- bool addUnits
- bool writeOnLoseFocus
- bool writeOnEnter
- bool writeOnFinish
- bool writeOnChange
- bool confirmWrite
- bool allowFocusUpdate

9.200.1 Constructor & Destructor Documentation

9.200.1.1 QENumericEdit::QENumericEdit (QWidget * *parent* = 0) [explicit]

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.200.1.2 QENumericEdit::QENumericEdit (const QString & *variableName*, QWidget * *parent* = 0) [explicit]

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.

9.200.2 Member Function Documentation

9.200.2.1 void QENumericEdit::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.200.2.2 void QENumericEdit::dbValueChanged () [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.

9.200.2.3 void QENumericEdit::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.200.3 Property Documentation

9.200.3.1 bool QENumericEdit::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.200.3.2 Qt::Alignment QENumericEdit::alignment [read, write]

This property holds the alignment of the numeric edit. Both horizontal and vertical alignment is allowed here, Qt::AlignJustify will map to Qt::AlignLeft. By default, this

property contains a combination of Qt::AlignRight and Qt::AlignVCenter.

9.200.3.3 bool QENumericEdit::allowFocusUpdate [read, write]

Allow updated while widget has focus - defaults to false.

9.200.3.4 int QENumericEdit::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.200.3.5 bool QENumericEdit::autoScale [read, write]

If true (default), display and editing of numbers using the PV's precision and control limits supplied with the data. If false, the precision, leadingZeros, minimum and maximum propertie values are used.

9.200.3.6 QString QENumericEdit::cleanText [read]

This property holds the displayed text. Not a property available to designer.

9.200.3.7 bool QENumericEdit::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). Note: writeOnChange and confirmWrite are mutually exclusive.

9.200.3.8 int QENumericEdit::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.200.3.9 bool QENumericEdit::frame [read, write]

This property holds whether the numeric edit draws itself with a frame. If enabled (the default) the numeric edit draws itself inside a frame, otherwise the line edit draws itself without any frame.

9.200.3.10 int QENumericEdit::leadingZeros [read, write]

Speficies the number of leading zeros. The default is 3. This is only used if autoScale is false. When autoScale is true the PV's control range is used to determine the number of required leading zeros. Stictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with ints.

9.200.3.11 double QENumericEdit::maximum [read, write]

Speficies the maximum allowed value. This is only used if autoScale is false.

9.200.3.12 double QENumericEdit::minimum [read, write]

Speficies the mimimum allowed value. This is only used if autoScale is false.

9.200.3.13 QNumericEdit::Notations QENumericEdit::notation [read, write]

Notation used for formatting/editing. Default is fixed.

9.200.3.14 int QENumericEdit::precision [read, write]

Precision used for the display and editing of numbers. The default is 2. This is only used if autoScale is false. When autoScale is true the PV's precision is used. Stictly speaking, this should be an unsigned int, but designer properties editor much 'nicer' with ints.

9.200.3.15 QEFixedPointRadix::Radicies QENumericEdit::radix [read, write]

Specify radix, default is Decimal.

9.200.3.16 QEFixedPointRadix::Separators QENumericEdit::separator [read, write]

Specify digit 'thousands' separator character, default is none.

9.200.3.17 QString QENumericEdit::variable [read, write]

EPICS variable name (CA PV)

9.200.3.18 QString QENumericEdit::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.

9.200.3.19 bool QENumericEdit::writeOnChange [read, write]

If true the widget writes to the PV as value are changes. If false (default) a write only occurs when as per writeOnLoseFocus, writeOnEnter and/or writeOnFinish values. Note: writeOnChange and confirmWrite are mutually exclusive.

9.200.3.20 bool QENumericEdit::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').

9.200.3.21 bool QENumericEdit::writeOnFinish [read, write]

Sets if this widget writes any changes when the user finished editing (the underlying QLineEdit 'editingFinished' signal is emitted). No writing occurs if no changes were made. Default is 'true' (writes any changes when the QLineEdit 'editingFinished' signal is emitted).

9.200.3.22 bool QENumericEdit::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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QENumericEdit/QENumericEdit.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QENumericEdit.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QENumericEdit/QENumericEdit.cpp

9.201 QEOneToOne< D, C > Class Template Reference

`#include <QEOneToOne.h>`

Public Member Functions

- void clear ()
- bool insertF (const D &dkey, const C &ckey)
- bool **insertI** (const C &ckey, const D &dkey)
- bool containsF (const D &key) const
- bool **containsI** (const C &key) const
- C valueF (const D &key) const
- D **valueI** (const C &key) const
- C valueF (const D &key, const C &defaultValue) const
- D **valueI** (const C &key, const D &defaultValue) const
- bool removeF (const D &dkey)
- bool **removeI** (const C &ckey)

9.201.1 Detailed Description

template<class D, class C> class QEOneToOne< D, C >

This template provides a one-to-one mapping from one type, the Domain type, to a another, possiblely the same, type, the CoDomain type. Despite the terminology used here (i.e. domain/codomain), the association between the two types is symetrical, therefore QEOneToOne <T1, T2> provides the exact same functionality as QEOneToOne <T2, T1>

Most functions are available in both forward (domain to co-domain) and in the inverse (co-domain to domain) modes of operation. These are named as <func>F and <func>I. We cannot use on overloading as user may require one to one mapping between the same type, e.g. an int to int association.

The undelying mappings are provided by a QHash. This may change.

9.201.2 Member Function Documentation

9.201.2.1 template<class D, class C> void QEOneToOne< D, C >::clear () [inline]

Clear all mapping relationships.

9.201.2.2 template<class D, class C> bool QEOneToOne< D, C >::containsF (const D & *key*) const [inline]

Indicates if assoication contains specified key.

9.201.2.3 template<class D, class C> bool QEOneToOne< D, C >::insertF (const D & *dkey*, const C & *ckey*) [inline]

Insert a key-key mapping into the association. Returns true if successfull. Returns false on fail. The insertF function fails if the association currently contains the domain key dkey and/or the codomain key ckey.

9.201.2.4 template<class D, class C> bool QEOneToOne< D, C >::removeF (const D & *dkey*) [inline]

Removes the assoication containing specified key.

9.201.2.5 template<class D, class C> C QEOneToOne< D, C >::valueF (const D & *key*, const C & *defaultValue*) const [inline]

Extracts the value associated with key. If the association does not contain the key, then returns specified default value.

9.201.2.6 template<class D, class C> C QEOneToOne< D, C >::valueF (const D & *key*) const [inline]

Extracts the b/a value associated with a/b key. If the association does not contain a/b key, then returns A ()/ B() as default.

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEOneToOne.h

9.202 QEOptions Class Reference

`#include <QEOptions.h>`

Public Member Functions

- **QEOptions** (const QStringList &args)
- int **getParameterCount** ()
- QString **getParameter** (const int i)
- bool **isSpecified** (const QString &option, const QChar letter)
- bool **isSpecified** (const QString &option)
- bool **isSpecified** (const QChar letter)
- bool **getBool** (const QString &option, const QChar letter)
- bool **getBool** (const QString &option)
- bool **getBool** (const QChar letter)
- QString **getString** (const QString &option, const QChar letter, const QString &defaultValue)
- QString **getString** (const QString &option, const QString &defaultValue)
- QString **getString** (const QChar letter, const QString &defaultValue)
- int **getInt** (const QString &option, const QChar letter, const int defaultValue)
- int **getInt** (const QString &option, const int defaultValue)
- int **getInt** (const QChar letter, const int defaultValue)
- double **getFloat** (const QString &option, const QChar letter, const double defaultValue)
- double **getFloat** (const QString &option, const double defaultValue)
- double **getFloat** (const QChar letter, const double defaultValue)

9.202.1 Detailed Description

This class provides a basic means to parse command line options or any other arbitary list of strings. Options are named and the name format may be either the short format (single letter) and/or the long format.

The types of data that my be extracted from an option are bool, QString, int and double; These are accessed by the associated getXxxx () functions where Xxxx is one of Bool, String, Int and Float.

Each of these function takes a default value of the appropriate type which is returned to the caller if the value is not defined or, as in the case of numerical values, is ill-defined.

Numeric validity is defined by QString::toInt () and QString::toDouble () Do note that QString::toInt () does not accept 0x... hexadecimal or 0... octal numbers.

Examples: program -v -- boolean or flag (short format) program --verbose -- boolean or flag (long format) program -n=31 -- integer (short format) program -n 31 -- integer (alternate short format) program --number=31 -- integer (long format) program --number 31 -- integer (alternate long format) program --output="fred.txt" -- string (long format) program --output fred.txt -- string (alternate long format) program --output= -- specifies a null string

Note: Currently it provides no means to "compress" multiple flags, e.g. to interpret: program -cap as program -c -a -p

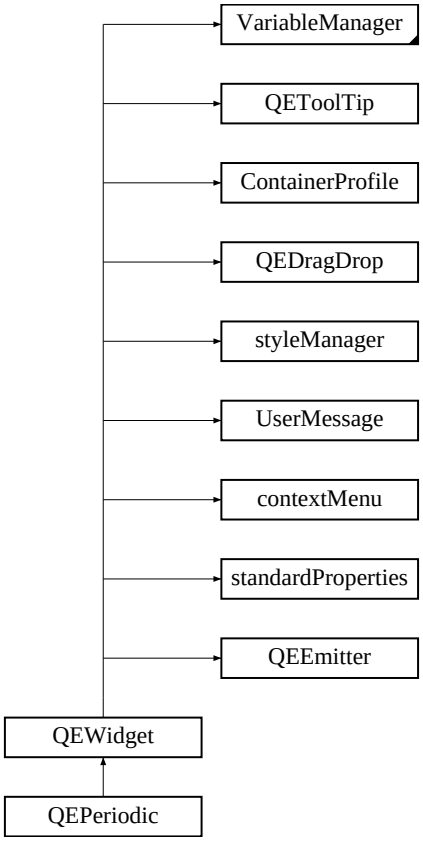
This class is one of several provided to access configuration data such as from environment variables, command line options and settings. They all provide a similar API.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_parameters/QEOptions.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_parameters/QEOptions.cpp

9.203 QEPeriodic Class Reference

Inheritance diagram for QEPeriodic::



Classes

- struct elementInfoStruct
- struct userInfoStructArray

Public Types

- enum **Category** {

  Hydrogen = 0, **AlkaliMetal**, **AlkalineEarthMetal**, **TransitionMetal**,

  PostTransitionMetal, **Metalloid**, **ReactiveNonMetal**, **NobleGas**,

  Lanthanide, **Actinide**, **UnknownProperties**, **NUMBER_OF_CATEGORIES**
  }
- enum **variableTypes** {

VARIABLE_TYPE_NUMBER, **VARIABLE_TYPE_ATOMIC_WEIGHT**, **VARIABLE_TYPE_MELTING_POINT**, **VARIABLE_TYPE_BOILING_-POINT**,

VARIABLE_TYPE_DENSITY, **VARIABLE_TYPE_GROUP**, **VARIABLE_TYPE_IONIZATION_ENERGY**, **VARIABLE_TYPE_-USER_VALUE_1**,

VARIABLE_TYPE_USER_VALUE_2 }

- enum **presentationOptions** { **PRESENTATION_BUTTON_AND_LABEL**, **PRESENTATION_BUTTON_ONLY**, **PRESENTATION_LABEL_ONLY** }
- enum **userInfoSourceOptions** { **USER_INFO_SOURCE_TEXT**, **USER_-INFO_SOURCE_FILE** }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }
- enum **PresentationOptions** { **buttonAndLabel** = QEPeriodic::PRESENTATION_BUTTON_AND_LABEL, **buttonOnly** = QEPeriodic::PRESENTATION_BUTTON_ONLY, **labelOnly** = QEPeriodic::PRESENTATION_LABEL_ONLY }
- enum **VariableTypes** {

  Number = QEPeriodic::VARIABLE_TYPE_NUMBER, **atomicWeight** = QEPeriodic::VARIABLE_TYPE_ATOMIC_WEIGHT, **meltingPoint** = QEPeriodic::VARIABLE_TYPE_MELTING_POINT, **boilingPoint** = QEPeriodic::VARIABLE_TYPE_BOILING_POINT,

  density = QEPeriodic::VARIABLE_TYPE_DENSITY, **group** = QEPeriodic::VARIABLE_TYPE_GROUP, **ionizationEnergy** = QEPeriodic::VARIABLE_TYPE_IONIZATION_ENERGY, **userValue1** = QEPeriodic::VARIABLE_TYPE_USER_VALUE_1,

  userValue2 = QEPeriodic::VARIABLE_TYPE_USER_VALUE_2 }
- enum **UserInfoSourceOptions** { **userInfoSourceText** = QEPeriodic::USER_-INFO_SOURCE_TEXT, **userInfoSourceFile** = QEPeriodic::USER_INFO_-SOURCE_FILE }

Public Slots

- void **setElement** (const QString symbol)
- void **setAtomicNumber** (const int atomicNumber)

Signals

- void userElementChanged (const QString &symbol)

  Sent when the element is changed by the user selecting an element.

- void **userAtomicNumberChanged** (const int atomicNumber)
- void dbValueChanged (const double &out)
- void dbElementChanged (const QString &out)
- void **dbAtomicNumberChanged** (const int atomicNumber)
- void requestResend ()

Public Member Functions

- **QEPeriodic** (QWidget *parent=0)
- **QEPeriodic** (const QString &variableName, QWidget *parent=0)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** () const
- void **setPresentationOption** (presentationOptions presentationOptionIn)
- presentationOptions **getPresentationOption** () const
- void **setVariableType1** (variableTypes variableType1In)
- variableTypes **getVariableType1** () const
- void **setVariableType2** (variableTypes variableType2In)
- variableTypes **getVariableType2** () const
- void **setVariableTolerance1** (double variableTolerance1In)
- double **getVariableTolerance1** () const
- void **setVariableTolerance2** (double variableTolerance2In)
- double **getVariableTolerance2** () const
- void **setUserInfo** (QString userInfo)
- QString **getUserInfo** () const
- void **setUserInfoText** (QString userInfo)
- QString **getUserInfoText** () const
- void **setUserInfoFile** (QString userInfoFileIn)
- QString **getUserInfoFile** () const
- void **setColourised** (const bool colouriseIn)
- bool **isColourised** () const
- void **setUserInfoSourceOption** (userInfoSourceOptions userInfoSourceOptionIn)
- userInfoSourceOptions **getUserInfoSourceOption** () const
- void **updateUserInfoSource** ()
- bool **getElementValues** (QString symbol, double *value1, double *value2) const
- QString **getSelectedSymbol** () const
- int **getSelectedAtomicNumber** () const
- void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- QString getVariableNameSubstitutionsProperty ()

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void **setPresentationOptionProperty** (PresentationOptions presentationOption)
- PresentationOptions **getPresentationOptionProperty** ()
- void **setVariableType1Property** (VariableTypes variableType)
- void **setVariableType2Property** (VariableTypes variableType)
- VariableTypes **getVariableType1Property** ()
- VariableTypes **getVariableType2Property** ()
- void **setUserInfoSourceOptionProperty** (UserInfoSourceOptions userInfoSourceOption)
- UserInfoSourceOptions **getUserInfoSourceOptionProperty** ()

Static Public Member Functions

- static QColor **categoryColour** (const Category cat)

Public Attributes

- userInfoStruct **userInfo** [NUM_ELEMENTS]

Static Public Attributes

- static const elementInfoStruct **elementInfo** [NUM_ELEMENTS]

Protected Types

- enum **variableIndexes** {

  WRITE_VARIABLE_1, **WRITE_VARIABLE_2**, **READ_VARIABLE_1**, **READ_VARIABLE_2**,

  QEPERIODIC_NUM_VARIABLES }

Protected Member Functions

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool allowDrop
- variableTypes **variableType1**
- variableTypes **variableType2**
- double **variableTolerance1**
- double **variableTolerance2**

Properties

- QString writeButtonVariable1
- QString writeButtonVariable2
- QString readbackLabelVariable1
- QString readbackLabelVariable2
- QString variableSubstitutions
- bool subscribe
- bool variableAsToolTip
- bool visible

- unsigned int
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- PresentationOptions **presentationOption**
- VariableTypes **variableType1**
- VariableTypes **variableType2**
- QString **userInfo**
- UserInfoSourceOptions **userInfoSourceOption**
- bool colourised

9.203.1 Member Enumeration Documentation

9.203.1.1 enum QEPeriodic::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.203.1.2 enum QEPeriodic::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.203.2 Member Function Documentation

9.203.2.1 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.

9.203.2.2 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.

9.203.2.3 void QEPeriodic::requestResend () [signal]

Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

9.203.3 Member Data Documentation

9.203.3.1 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.

9.203.4 Property Documentation

9.203.4.1 bool QEPeriodic::colourised [read, write]

This property sets element colourised enable/disable on the PeridicDialog.

9.203.4.2 bool QEPeriodic::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.203.4.3 DisplayAlarmStateOptions QEPeriodic::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.203.4.4 unsigned QEPeriodic::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

9.203.4.5 QString QEPeriodic::readbackLabelVariable1 [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the first of two positioners to determine which (if any) element is currently selected.

9.203.4.6 QString QEPeriodic::readbackLabelVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to read the value to the second of two positioners to determine which (if any) element is currently selected.

9.203.4.7 bool QEPeriodic::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from VariableManager.

9.203.4.8 UserLevels QEPeriodic::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.203.4.9 QString QEPeriodic::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.203.4.10 QString QEPeriodic::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.203.4.11 QString QEPeriodic::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.203.4.12 UserLevels QEPeriodic::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.203.4.13 bool QEPeriodic::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from QEToolTip.

9.203.4.14 QString QEPeriodic::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.203.4.15 bool QEPeriodic::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

9.203.4.16 QString QEPeriodic::writeButtonVariable1 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the first of two positioners that will position the select element.

9.203.4.17 QString QEPeriodic::writeButtonVariable2 [read, write]

EPICS variable name (CA PV). This variable is used to write a value to the second of two positioners that will position the select element.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPeriodic.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.cpp

9.204 QEPeriodicComponentData Class Reference

Public Attributes

- unsigned int **variableIndex1**
- double **lastData1**
- bool **haveLastData1**
- unsigned int **variableIndex2**
- double **lastData2**
- bool **haveLastData2**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.h

9.205 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 **valueInited** (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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/qepicspv.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/qepicspv.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_qepicspv.cpp

9.206 QEPlatform Class Reference

`#include <QEPlatform.h>`

Static Public Member Functions

- static bool isNaN (const double x)

  This function test if the specified double floating point number is 'Not a Number'.

- static bool isInf (const double x)

  This function test if the specified double floating point number is +/-Infinity.

9.206.1 Detailed Description

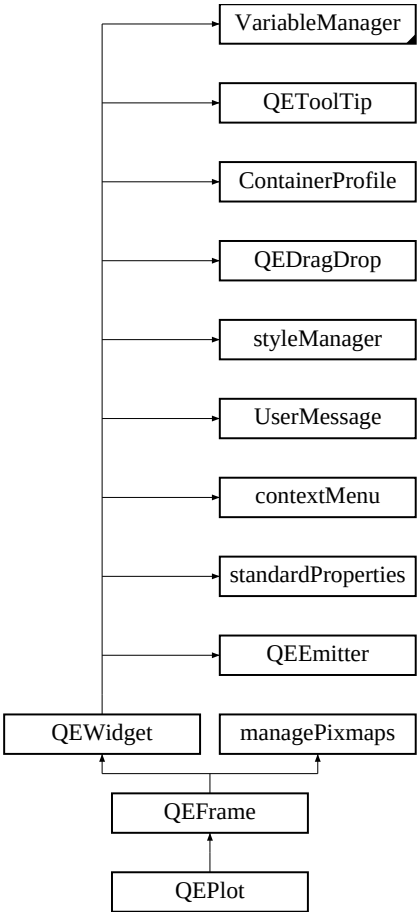
Geneal purpose platform specific functions used to hide compiler, environment and version specific features. This locates all the indef and the like tests into one centralised and consistent location. We use a class of static methods as opposed to a set of regular functions. This provide a name space, but also allows inheritance if needs be.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPlatform.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPlatform.cpp

9.207 QEPlot Class Reference

Inheritance diagram for QEPlot::



Classes

- class Trace

Public Types

- enum **TraceStyles** { **Lines** = 1, **Sticks**, **Steps**, **Dots** }

Signals

- void **mouseMove** (const QPointF &posn)
- void dbValueChanged (const double &out)
- void dbValueChanged (const QVector< double > &out)

Public Member Functions

- **QEPlot** (QWidget ∗parent=0)
- **QEPlot** (const QString &variableName, QWidget ∗parent=0)
- QSize **sizeHint** () const
- void **setYMin** (double yMin)
- double **getYMin** () const
- void **setYMax** (double yMax)
- double **getYMax** () const
- void **setAutoScale** (bool autoScale)
- bool **getAutoScale** () const
- void **setAxisEnableX** (bool axisEnableXIn)
- bool **getAxisEnableX** () const
- void **setAxisEnableY** (bool axisEnableYIn)
- bool **getAxisEnableY** () const
- void **setTitle** (const QString &title)
- QString **getTitle** () const
- void **setBackgroundColor** (const QColor backgroundColor)
- QColor **getBackgroundColor** () const
- void **setVariableNameIndexProperty** (const QString &variableName, const unsigned int variableIndex)
- void **setVariableName1Property** (const QString &pvName)
- void **setVariableName2Property** (const QString &pvName)
- void **setVariableName3Property** (const QString &pvName)
- void **setVariableName4Property** (const QString &pvName)
- QString **getVariableNameIndexProperty** (const unsigned int variableIndex) const
- QString **getVariableName1Property** () const
- QString **getVariableName2Property** () const
- QString **getVariableName3Property** () const
- QString **getVariableName4Property** () const
- void **setTraceStyle** (const TraceStyles traceStyle, const unsigned int variableIndex)
- void **setTraceStyle1** (const TraceStyles traceStyle)
- void **setTraceStyle2** (const TraceStyles traceStyle)
- void **setTraceStyle3** (const TraceStyles traceStyle)
- void **setTraceStyle4** (const TraceStyles traceStyle)
- TraceStyles **getTraceStyle** (const unsigned int variableIndex) const
- TraceStyles **getTraceStyle1** () const
- TraceStyles **getTraceStyle2** () const
- TraceStyles **getTraceStyle3** () const
- TraceStyles **getTraceStyle4** () const
- void **setTraceColor** (const QColor traceColor, const unsigned int variableIndex)
- void **setTraceColor1** (const QColor traceColor)
- void **setTraceColor2** (const QColor traceColor)
- void **setTraceColor3** (const QColor traceColor)

- void **setTraceColor4** (const QColor traceColor)
- QColor **getTraceColor** (const unsigned int variableIndex) const
- QColor **getTraceColor1** () const
- QColor **getTraceColor2** () const
- QColor **getTraceColor3** () const
- QColor **getTraceColor4** () const
- void **setTraceLegend** (const QString &traceLegend, const unsigned int variableIndex)
- void **setTraceLegend1** (const QString &traceLegend)
- void **setTraceLegend2** (const QString &traceLegend)
- void **setTraceLegend3** (const QString &traceLegend)
- void **setTraceLegend4** (const QString &traceLegend)
- QString **getTraceLegend** (const unsigned int variableIndex) const
- QString **getTraceLegend1** () const
- QString **getTraceLegend2** () const
- QString **getTraceLegend3** () const
- QString **getTraceLegend4** () const
- void **setXUnit** (const QString &xUnit)
- QString **getXUnit** () const
- void **setYUnit** (const QString &yUnit)
- QString **getYUnit** () const
- void **setGridEnableMajorX** (bool gridEnableMajorXIn)
- bool **getGridEnableMajorX** () const
- void **setGridEnableMajorY** (bool gridEnableMajorYIn)
- bool **getGridEnableMajorY** () const
- void **setGridEnableMinorX** (bool gridEnableMinorXIn)
- bool **getGridEnableMinorX** () const
- void **setGridEnableMinorY** (bool gridEnableMinorYIn)
- bool **getGridEnableMinorY** () const
- void **setGridMajorColor** (QColor gridMajorColorIn)
- QColor **getGridMajorColor** () const
- void **setGridMinorColor** (QColor gridMinorColorIn)
- QColor **getGridMinorColor** () const
- void **setXStart** (double xStart)
- double **getXStart** () const
- void **setXIncrement** (double xIncrement)
- double **getXIncrement** () const
- void **setTimeSpan** (int timeSpan)
- int **getTimeSpan** () const
- void **setTickRate** (int tickRate)
- int **getTickRate** () const
- void **setMargin** (const int margin)
- int **getMargin** () const
- void setVariableNameSubstitutionsProperty (const QString &variableNameSubstitutions)

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- QString getVariableNameSubstitutionsProperty () const

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

Protected Member Functions

- void **canvasMouseMove** (QMouseEvent *mouseEvent)
- bool **eventFilter** (QObject *obj, QEvent *event)
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool allowDrop

Properties

- QString variable1
- QString variable2
- QString variable3
- QString variable4
- QString variableSubstitutions
- 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**
- QString **xUnit**
- QString **yUnit**
- int **margin**

9.207.1 Member Function Documentation

9.207.1.1 void QEPlot::dbValueChanged (const QVector< double > & *out*) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.207.1.2 void QEPlot::dbValueChanged (const double & *out*) [signal]

Sent when the widget is updated following a data change Can be used to pass on EPICS data (as presented in this widget) to other widgets. For example a QList widget could log updates from this widget.

9.207.2 Member Data Documentation

9.207.2.1 bool QEPlot::allowDrop [protected]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEFrame.

9.207.3 Property Documentation

9.207.3.1 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.

9.207.3.2 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.

9.207.3.3 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.

9.207.3.4 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.

9.207.3.5 QString QEPlot::variableSubstitutions [read, write]

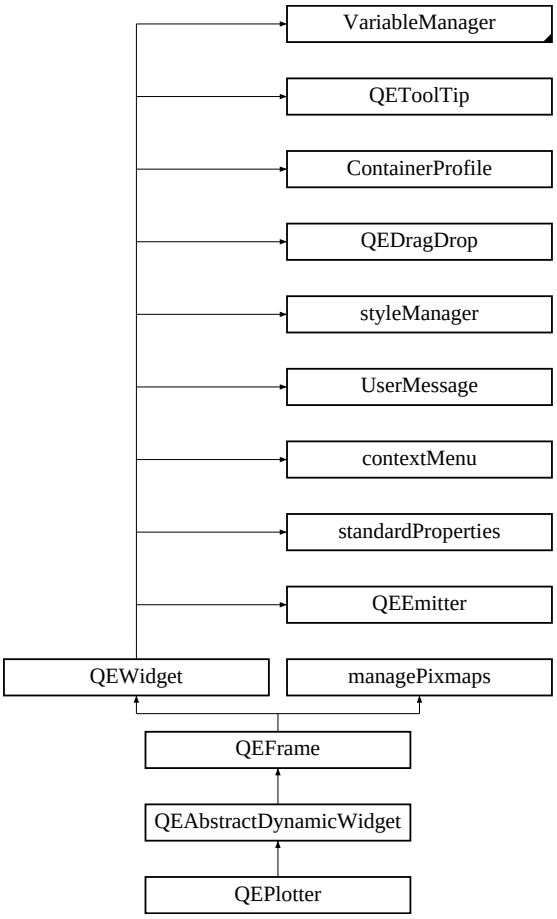
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlot/QEPlot.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPlot.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlot/QEPlot.cpp

9.208 QEPlotter Class Reference

Inheritance diagram for QEPlotter::



Classes

- class **DataSets**

Public Types

- enum **Constants** { **NUMBER_OF_PLOTS** = 16, **NUMBER_OF_SLOTS** = 17 }

Public Slots

- void **setXYDataPV** (const int slot, const QString &pvName)
- void **setXYAlias** (const int slot, const QString &alias)

- void **setXRange** (const double xMinimum, const double xMaximum)
- void **setYRange** (const double yMinimum, const double yMaximum)
- void **setDataPvNameSet** (const QStringList &pvNameSet)
- void **setAliasNameSet** (const QStringList &aliasNameSet)
- void **setPlotterEntry** (const int slot, const QString &pvName, const QString &alias)
- void **setNewVariableName** (QString variableName, QString variableNameSubstitutions, unsigned int variableIndex)
- void **dataConnectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **dataArrayChanged** (const QVector< double > &values, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **sizeConnectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **sizeValueChanged** (const long &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)

Signals

- void **crosshairIndexChanged** (int value)
- void **coordinateSelected** (QPointF xyvalue)
- void **xCoordinateSelected** (double xvalue)
- void **yCoordinateSelected** (double yvalue)
- void **pvDataNameSetChanged** (const QStringList &nameSet)
- void **alaisNameSetChanged** (const QStringList &nameSet)
- void **requestAction** (const QEActionRequests &)

Public Member Functions

- **QEPlotter** (QWidget *parent=0)
- QSize **sizeHint** () const
- QString **getXYDataPV** (const int slot) const
- void **setXYSizePV** (const int slot, const QString &pvName)
- QString **getXYSizePV** (const int slot) const
- QString **getXYAlias** (const int slot) const
- void **setXYColour** (const int slot, const QColor &colour)
- QColor **getXYColour** (const int slot) const
- void **setXYLineVisible** (const int slot, const bool isVisible)
- bool **getXYLineVisible** (const int slot) const
- void **setXYLineBold** (const int slot, const bool isBold)
- bool **getXYLineBold** (const int slot) const
- void **setXYLineDashed** (const int slot, const bool isDashed)
- bool **getXYLineDashed** (const int slot) const
- void **setXYLineHasDots** (const int slot, const bool hasDots)
- bool **getXYLineHasDots** (const int slot) const
- void **setEnableConextMenu** (bool enable)

- bool **getEnableConextMenu** () const
- void **setMenuEmitText** (const QString &text)
- QString **getMenuEmitText** () const
- void **setToolBarVisible** (bool visible)
- bool **getToolBarVisible** () const
- void **setPvItemsVisible** (bool visible)
- bool **getPvItemsVisible** () const
- void **setStatusVisible** (bool visible)
- bool **getStatusVisible** () const
- void **setAxisEnableX** (bool axisEnableX)
- bool **getAxisEnableX** () const
- void **setAxisEnableY** (bool axisEnableY)
- bool **getAxisEnableY** () const
- void **setXLogarithmic** (bool visible)
- bool **getXLogarithmic** () const
- void **setYLogarithmic** (bool visible)
- bool **getYLogarithmic** () const
- void **setXMinimum** (const double xMinimum)
- double **getXMinimum** () const
- void **setXMaximum** (const double xMaximum)
- double **getXMaximum** () const
- void **setYMinimum** (const double yMinimum)
- double **getYMinimum** () const
- void **setYMaximum** (const double yMaximum)
- double **getYMaximum** () const
- void **setIsPaused** (bool paused)
- bool **getIsPaused** () const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()
- int **getCrosshairIndex** () const
- QStringList **getDataPvNameSet** () const
- QStringList **getAliasNameSet** () const
- QEGraphic * **getGraphic** () const

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- bool **eventFilter** (QObject *obj, QEvent *event)

- void **mousePressEvent** (QMouseEvent ∗event)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void saveConfiguration (PersistanceManager ∗pm)
- void restoreConfiguration (PersistanceManager ∗pm, restorePhases restorePhase)
- int **findSlot** (QObject ∗obj)
- QString **getXYExpandedDataPV** (const int slot) const
- QString **getXYExpandedSizePV** (const int slot) const

Properties

- QString variableSubstitutions
- bool **enableContextMenu**
- bool **toolBarIsVisible**
- bool **pvItemsIsVisible**
- bool **statusIsVisible**
- bool **xLogarithmic**
- bool **yLogarithmic**
- bool **axisEnableX**
- bool **axisEnableY**
- double **xMinimum**
- double **xMaximum**
- double **yMinimum**
- double **yMaximum**
- QString **contextMenuEmitText**
- QString **DataVariableX**
- QString **DataVariableA**
- QString **DataVariableB**
- QString **DataVariableC**
- QString **DataVariableD**
- QString **DataVariableE**
- QString **DataVariableF**
- QString **DataVariableG**
- QString **DataVariableH**
- QString **DataVariableI**
- QString **DataVariableJ**
- QString **DataVariableK**
- QString **DataVariableL**
- QString **DataVariableM**
- QString **DataVariableN**
- QString **DataVariableO**
- QString **DataVariableP**
- QString **SizeVariableX**

- QString **SizeVariableA**
- QString **SizeVariableB**
- QString **SizeVariableC**
- QString **SizeVariableD**
- QString **SizeVariableE**
- QString **SizeVariableF**
- QString **SizeVariableG**
- QString **SizeVariableH**
- QString **SizeVariableI**
- QString **SizeVariableJ**
- QString **SizeVariableK**
- QString **SizeVariableL**
- QString **SizeVariableM**
- QString **SizeVariableN**
- QString **SizeVariableO**
- QString **SizeVariableP**
- QString **AliasNameX**
- QString **AliasNameA**
- QString **AliasNameB**
- QString **AliasNameC**
- QString **AliasNameD**
- QString **AliasNameE**
- QString **AliasNameF**
- QString **AliasNameG**
- QString **AliasNameH**
- QString **AliasNameI**
- QString **AliasNameJ**
- QString **AliasNameK**
- QString **AliasNameL**
- QString **AliasNameM**
- QString **AliasNameN**
- QString **AliasNameO**
- QString **AliasNameP**
- QColor **ColourA**
- QColor **ColourB**
- QColor **ColourC**
- QColor **ColourD**
- QColor **ColourE**
- QColor **ColourF**
- QColor **ColourG**
- QColor **ColourH**
- QColor **ColourI**
- QColor **ColourJ**
- QColor **ColourK**
- QColor **ColourL**
- QColor **ColourM**
- QColor **ColourN**
- QColor **ColourO**
- QStringList aliasNames

Friends

- class **DataSets**

9.208.1 Member Function Documentation

9.208.1.1 void QEPlotter::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.208.1.2 void QEPlotter::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

9.208.2 Property Documentation

9.208.2.1 QStringList QEPlotter::aliasNames [read, write]

Allows specification of alias as a 'single property', which is also a slot.

9.208.2.2 QString QEPlotter::variableSubstitutions [read, write]

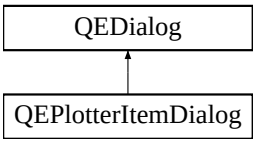
Default macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotter.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPlotter.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotter.cpp

9.209 QEPlotterItemDialog Class Reference

Inheritance diagram for QEPlotterItemDialog::



Public Member Functions

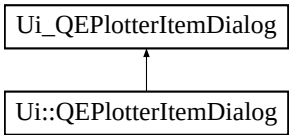
- **QEPlotterItemDialog** (QWidget ∗parent=0)
- void **setFieldInformation** (const QString dataIn, const QString aliasIn, const QString sizeIn)
- void **getFieldInformation** (QString &dataOut, QString &aliasOut, QString &sizeOut)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterItemDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterItemDialog.cpp

9.210 Ui::QEPlotterItemDialog Class Reference

Inheritance diagram for Ui::QEPlotterItemDialog::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPlotterItemDialog.h

9.211 QEPlotterMenu Class Reference

`#include <QEPlotterMenu.h>`

Signals

- void **selected** (const QEPlotterNames::MenuActions action, const int slot)

Public Member Functions

- **QEPlotterMenu** (QWidget *parent=0)
- **QEPlotterMenu** (const int slot, QWidget *parent=0)
- void **setActionChecked** (const QEPlotterNames::MenuActions action, const bool checked)
- void **setActionEnabled** (const QEPlotterNames::MenuActions action, const bool enabled)
- void **setActionVisible** (const QEPlotterNames::MenuActions action, const bool visible)
- void **setActionText** (const QEPlotterNames::MenuActions action, const QString &caption)
- void **setCheckedStates** (const bool isDisplayed, const bool isBold, const bool isDashed, const bool showDots)

9.211.1 Detailed Description

QEPlotter PV item specific context menu.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPlotterMenu.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterMenu.cpp

9.212 QEPlotterNames Class Reference

Public Types

- enum **MenuActions** {

  PLOTTER_FIRST = contextMenu::CM_SPECIFIC_WIDGETS_START_HERE, **PLOTTER_SHOW_HIDE_CROSSHAIRS**, **PLOTTER_SHOW_HIDE_TOOLBAR**, **PLOTTER_SHOW_HIDE_PV_ITEMS**,

  PLOTTER_SHOW_HIDE_STATUS, **PLOTTER_EMIT_COORDINATES**, **PLOTTER_PREV**, **PLOTTER_NEXT**,

  PLOTTER_NORMAL_VIDEO, **PLOTTER_REVERSE_VIDEO**, **PLOTTER_LINEAR_Y_SCALE**, **PLOTTER_LOG_Y_SCALE**,

  PLOTTER_MANUAL_Y_RANGE, **PLOTTER_CURRENT_Y_RANGE**, **PLOTTER_DYNAMIC_Y_RANGE**, **PLOTTER_NORAMLISED_Y_RANGE**,

  PLOTTER_FRACTIONAL_Y_RANGE, **PLOTTER_LINEAR_X_SCALE**, **PLOTTER_LOG_X_SCALE**, **PLOTTER_MANUAL_X_RANGE**,

  PLOTTER_CURRENT_X_RANGE, **PLOTTER_DYNAMIC_X_RANGE**, **PLOTTER_MANUAL_XY_RANGE**, **PLOTTER_PLAY**,

  PLOTTER_PAUSE, **PLOTTER_LOAD_CONFIG**, **PLOTTER_SAVE_CONFIG**, **PLOTTER_COPY_VARIABLE**,

  PLOTTER_COPY_DATA, **PLOTTER_PASTE**, **PLOTTER_DRAG_VARIABLE**, **PLOTTER_DRAG_DATA**,

  PLOTTER_LINE_BOLD, **PLOTTER_LINE_DASHED**, **PLOTTER_LINE_DOTS**, **PLOTTER_LINE_VISIBLE**,

  PLOTTER_LINE_NO_MEDIAN_FILTER, **PLOTTER_LINE_MEDIAN_3_FILTER**, **PLOTTER_LINE_MEDIAN_5_FILTER**, **PLOTTER_LINE_COLOUR**,

  PLOTTER_DATA_SELECT, **PLOTTER_DATA_DIALOG**, **PLOTTER_DATA_CLEAR**, **PLOTTER_SCALE_TO_MIN_MAX**,

  PLOTTER_SCALE_TO_ZERO_MAX, **PLOTTER_COPY_DATA_VARIABLE**, **PLOTTER_COPY_DATA_DATA**, **PLOTTER_PASTE_DATA_PV**,

  PLOTTER_SHOW_DATA_PV_PROPERTIES, **PLOTTER_ADD_DATA_PV_TO_STRIPCHART**, **PLOTTER_ADD_DATA_PV_TO_SCRATCH_PAD**, **PLOTTER_GENERAL_DATA_PV_EDIT**,

  PLOTTER_COPY_SIZE_VARIABLE, **PLOTTER_COPY_SIZE_DATA**, **PLOTTER_PASTE_SIZE_PV**, **PLOTTER_SHOW_SIZE_PV_PROPERTIES**,

  PLOTTER_ADD_SIZE_PV_TO_STRIPCHART, **PLOTTER_ADD_SIZE_PV_TO_SCRATCH_PAD**, **PLOTTER_GENERAL_SIZE_PV_EDIT**, **PLOTTER_LAST** }
- enum **ScaleModes** { **smFixed**, **smNormalised**, **smFractional**, **smDynamic** }

Public Member Functions

- **QEPlotterNames** (QObject ∗parent=0)

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterNames.h

9.213 QEPlotterPushButtonSpecifications Struct Reference

Public Attributes

- int **gap**
- int **width**
- bool **isIcon**
- const QString **captionOrIcon**
- QEPlotterNames::MenuActions **action**
- const QString **toolTip**

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterTool

9.214 QEPlotterState Class Reference

Public Member Functions

- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Public Attributes

- bool **isXLogarithmic**
- bool **isYLogarithmic**
- bool **isReverse**
- bool **isPaused**
- double **xMinimum**
- double **xMaximum**
- double **yMinimum**
- double **yMaximum**
- QEPlotterNames::ScaleModes **xScaleMode**
- QEPlotterNames::ScaleModes **yScaleMode**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterState.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterState.cpp

9.215 QEPlotterStateList Class Reference

Public Member Functions

- void **clear** ()
- void **push** (const QEPlotterState &state)
- bool **prev** (QEPlotterState &state)
- bool **next** (QEPlotterState &state)
- bool **prevAvailable** ()
- bool **nextAvailable** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterState
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterState

9.216 QEPlotterToolBar Class Reference

`#include <QEPlotterToolBar.h>`

Signals

- void **selected** (const QEPlotterNames::MenuActions action, const int slot)

Public Member Functions

- **QEPlotterToolBar** (QWidget *parent=0)
- void **setEnabled** (const QEPlotterNames::MenuActions action, const bool enabled)

Static Public Attributes

- static const int **designHeight** = 32

Protected Member Functions

- void **resizeEvent** (QResizeEvent *event)

9.216.1 Detailed Description

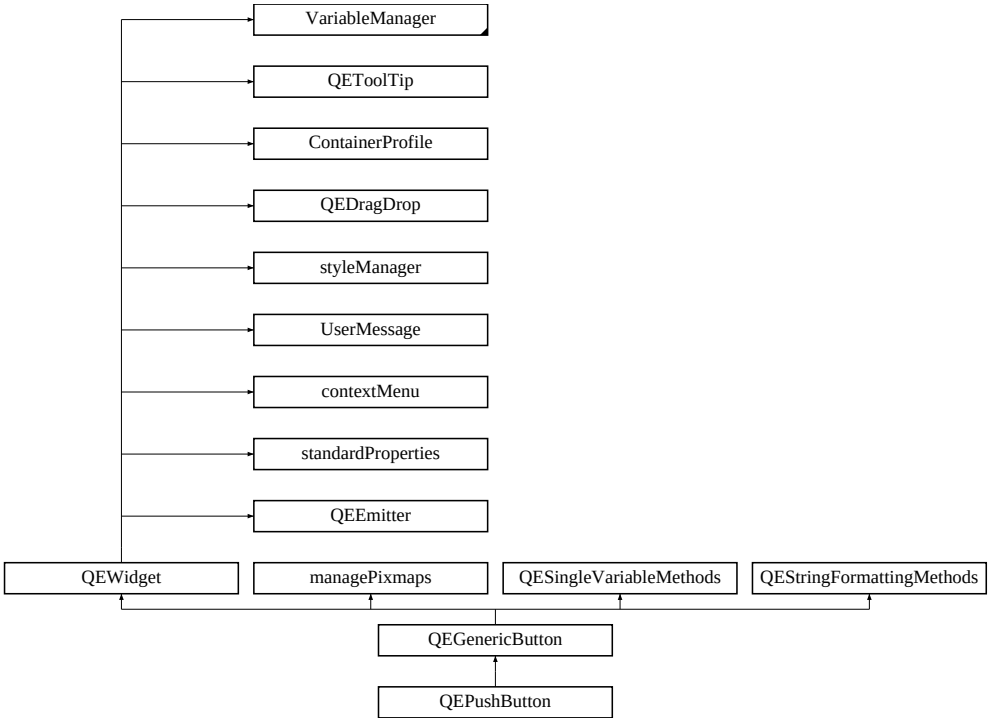
This class holds all the QEPlotter tool bar widget.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterToolBar.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPlotterToolBar.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlotter/QEPlotterToolBar.cpp

9.217 QEPushButton Class Reference

Inheritance diagram for QEPushButton::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }
- enum Formats {

  Default = QEStringFormatting::FORMAT_DEFAULT, Floating = QEStringFormatting::FORMAT_FLOATING, Integer = QEStringFormatting::FORMAT_INTEGER, UnsignedInteger = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_-NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Underscore = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }
- enum UpdateOptions {

  Text = QEGenericButton::UPDATE_TEXT, Icon = QEGenericButton::UPDATE_ICON, TextAndIcon = QEGenericButton::UPDATE_TEXT_AND_ICON, State = QEGenericButton::UPDATE_STATE,

  TextAndState = QEGenericButton::UPDATE_TEXT_AND_STATE, IconAndState = QEGenericButton::UPDATE_ICON_AND_STATE, TextIconAndState = QEGenericButton::UPDATE_TEXT_ICON_AND_STATE }

  User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

- enum ProgramStartupOptionNames { None = applicationLauncher::PSO_NONE, Terminal = applicationLauncher::PSO_TERMINAL, LogOutput = applicationLauncher::PSO_LOGOUTPUT, StdOutput = applicationLauncher::PSO_STDOUTPUT }
- enum CreationOptionNames {

  Open = QEActionRequests::OptionOpen, NewTab = QEActionRequests::OptionNewTab, NewWindow = QEActionRequests::OptionNewWindow, DockTop = QEActionRequests::OptionTopDockWindow,

  DockBottom = QEActionRequests::OptionBottomDockWindow, DockLeft = QEActionRequests::OptionLeftDockWindow, DockRight = QEActionRequests::OptionRightDockWindow, DockTopTabbed = QEActionRequests::OptionTopDockWindowTabbed,

  DockBottomTabbed = QEActionRequests::OptionBottomDockWindowTabbed, DockLeftTabbed = QEActionRequests::OptionLeftDockWindowTabbed, DockRightTabbed = QEActionRequests::OptionRightDockWindowTabbed, DockFloating = QEActionRequests::OptionFloatingDockWindow }

  Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Public Slots

- void requestAction (const QEActionRequests &request)
- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void requestResend ()

  Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

- void newGui (const QEActionRequests &request)

  Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

- void pressed (int value)
- void released (int value)
- void clicked (int value)
- void programCompleted ()

  Program started by button has compelted.

Public Member Functions

- QEPushButton (QWidget *parent=0)
- QEPushButton (const QString &variableName, QWidget *parent=0)
- ~QEPushButton ()

  Destructor.

- void writeNow ()
- void setVariableNameSubstitutionsProperty (const QString &substitutions)
- void **setAltReadbackProperty** (const QString &variableName)
- QString **getAltReadbackProperty** () const
- void **setAltReadbackArrayIndex** (const int arrayIndex)
- int **getAltReadbackArrayIndex** () const
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setFormatProperty (Formats format)

  Access function for format property - refer to format property for details.

- Formats getFormatProperty ()

  Access function for format property - refer to format property for details.

- void setSeparatorProperty (const Separators notation)

  Access function for separator property - refer to separator property for details.

- Separators getSeparatorProperty () const

  Access function for separator property - refer to separator property for details.

- void setNotationProperty (Notations notation)

  Access function for notation property - refer to notation property for details.

- Notations getNotationProperty ()

  Access function for notation property - refer to notation property for details.

- void setArrayActionProperty (ArrayActions arrayAction)

  Access function for arrayAction property - refer to arrayAction property for details.

- ArrayActions getArrayActionProperty ()

  Access function for arrayAction property - refer to arrayAction property for details.

Properties

- QString variable
- QString variableSubstitutions

- int arrayIndex
- QString altReadbackVariable
- int **altReadbackArrayIndex**
- bool subscribe
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction
- QEWidgetProperties::DisabledRecordPolicy disabledRecordPolicy
- Qt::Alignment alignment
- UpdateOptions updateOption
- QPixmap pixmap0
- QPixmap pixmap1
- QPixmap pixmap2
- QPixmap pixmap3
- QPixmap pixmap4
- QPixmap pixmap5
- QPixmap pixmap6
- QPixmap pixmap7
- QString password
- bool confirmAction
- QString confirmText
- bool writeOnPress
- bool writeOnRelease
- bool writeOnClick

- QString pressText
- QString releaseText
- QString clickText
- QString clickCheckedText
- QString labelText
- QString program
- QStringList arguments
- ProgramStartupOptionNames programStartupOption
- QString guiFile
- CreationOptionNames creationOption
- QString prioritySubstitutions
- QString customisationName

9.217.1 Member Enumeration Documentation

9.217.1.1 enum QEPushButton::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.217.1.2 enum QEPushButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

DockTopTabbed Open new GUI in a top dock window (tabbed with any existing dock in that area).

DockBottomTabbed Open new GUI in a bottom dock window (tabbed with any existing dock in that area).

DockLeftTabbed Open new GUI in a left dock window (tabbed with any existing dock in that area).

DockRightTabbed Open new GUI in a right dock window (tabbed with any existing dock in that area).

DockFloating Open new GUI in a floating dock window.

9.217.1.3 enum QEPushButton::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.217.1.4 enum QEPushButton::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.217.1.5 enum QEPushButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.217.1.6 enum QEPushButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

Enumerator:

None Just run the program.

Terminal Run the program in a termainal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir').

LogOutput Run the program, and log the output in the QE message system.

StdOutput Run the program, and send doutput to standard output and standard error.

9.217.1.7 enum QEPushButton::Separators

User friendly enumerations for seprator property - refer to QEStringFormatting::formats for details.

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.217.1.8 enum QEPushButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

State Data updates will update the button state (checked or unchecked).

TextAndState Data updates will update the button text and state.

IconAndState Data updates will update the button icon and state.

TextIconAndState Data updates will update the button - the lot.

9.217.1.9 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.

9.217.2 Constructor & Destructor Documentation

9.217.2.1 QEPushButton::QEPushButton (QWidget * *parent* = 0)

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.217.2.2 QEPushButton::QEPushButton (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.217.3 Member Function Documentation

9.217.3.1 void QEPushButton::clicked (int *value*) [signal]

Button has been Clicked. The value emitted is the integer interpretation of the clickText property (or the clickCheckedText property if the button was checked)

9.217.3.2 void QEPushButton::dbValueChanged () [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.

9.217.3.3 void QEPushButton::pressed (int *value*) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the pressText property

9.217.3.4 void QEPushButton::released (int *value*) [signal]

Button has been Released The value emitted is the integer interpretation of the release-Text property

9.217.3.5 void QEPushButton::requestAction (const QEActionRequests & *request*) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.217.3.6 void QEPushButton::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.217.3.7 void QEPushButton::setVariableNameSubstitutionsProperty (const QString & *substitutions*)

Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

Reimplemented from QESingleVariableMethods.

9.217.3.8 void QEPushButton::writeNow () [inline, virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOn-Change, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.217.4 Property Documentation

9.217.4.1 bool QEPushButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.217.4.2 Qt::Alignment QEPushButton::alignment [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

9.217.4.3 bool QEPushButton::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEDragDrop.

9.217.4.4 QString QEPushButton::altReadbackVariable [read, write]

EPICS variable name (CA PV). This variable is used to provide a readback value when different to the variable written to by a button press.

9.217.4.5 QStringList QEPushButton::arguments [read, write]

Arguments for program specified in the 'program' property.

9.217.4.6 ArrayActions QEPushButton::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.
- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.217.4.7 int QEPushButton::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.217.4.8 QString QEPushButton::clickCheckedText [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates.

When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to diaplay a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to diaplay a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from QEGenericButton.

9.217.4.9 QString QEPushButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from QEGenericButton.

9.217.4.10 bool QEPushButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

9.217.4.11 QString QEPushButton::confirmText [read, write]

Text used to confirm acion if confirmation dialog is presented

Reimplemented from QEGenericButton.

9.217.4.12 CreationOptionNames QEPushButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. the creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from QEGenericButton.

9.217.4.13 QString QEPushButton::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be

applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from QEGenericButton.

9.217.4.14 QString QEPushButton::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.217.4.15 QEWidgetProperties::DisabledRecordPolicy QEPushButton::disabledRecordPolicy [read, write]

Set the widget's disabled record policy, i.e. the action to be taken when the under lying record is disabled, i.e. when the assiociated record's DISA and DISV field values are equal. Note: this is only applicable IOC process variables. When the policy is ignore, then no special action is taken. This is the default policy. When the policy is grayout, the widget is style is set as if disconnected when the record is disabled.

Reimplemented from QEGenericButton.

9.217.4.16 bool QEPushButton::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.217.4.17 DisplayAlarmStateOptions QEPushButton::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.217.4.18 bool QEPushButton::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.217.4.19 Formats QEPushButton::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.217.4.20 QString QEPushButton::guiFile [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the QEPushButton is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See QEWidget::openQEFile() in QEWidget.cpp for details.

9.217.4.21 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.

9.217.4.22 QString QEPushButton::labelText [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from QEGenericButton.

9.217.4.23 bool QEPushButton::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.217.4.24 QString QEPushButton::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.217.4.25 Notations QEPushButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.217.4.26 QString QEPushButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from QEGenericButton.

9.217.4.27 QPixmap QEPushButton::pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

9.217.4.28 QPixmap QEPushButton::pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

9.217.4.29 QPixmap QEPushButton::pixmap2 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

9.217.4.30 QPixmap QEPushButton::pixmap3 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

9.217.4.31 QPixmap QEPushButton::pixmap4 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

9.217.4.32 QPixmap QEPushButton::pixmap5 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

9.217.4.33 QPixmap QEPushButton::pixmap6 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

9.217.4.34 QPixmap QEPushButton::pixmap7 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

9.217.4.35 int QEPushButton::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.217.4.36 QString QEPushButton::pressText [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from QEGenericButton.

9.217.4.37 QString QEPushButton::prioritySubstitutions [read, write]

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substittions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from QEGenericButton.

9.217.4.38 QString QEPushButton::program [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

9.217.4.39 ProgramStartupOptionNames QEPushButton::programStartupOption [read, write]

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

9.217.4.40 int QEPushButton::radix [read, write]

Base used for when formatting integers. Default is 10 (duh!)

9.217.4.41 QString QEPushButton::releaseText [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from QEGenericButton.

9.217.4.42 Separators QEPushButton::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.217.4.43 QString QEPushButton::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.217.4.44 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 VariableManager.

9.217.4.45 bool QEPushButton::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.217.4.46 UpdateOptions QEPushButton::updateOption [read, write]

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from QEGenericButton.

9.217.4.47 bool QEPushButton::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

9.217.4.48 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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.217.4.49 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.

9.217.4.50 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.

9.217.4.51 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.

9.217.4.52 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'.

9.217.4.53 QString QEPushButton::variable [read, write]

EPICS variable name (CA PV)

9.217.4.54 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.

9.217.4.55 QString QEPushButton::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

9.217.4.56 bool QEPushButton::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

9.217.4.57 bool QEPushButton::writeOnClick [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from QEGenericButton.

9.217.4.58 bool QEPushButton::writeOnPress [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from QEGenericButton.

9.217.4.59 bool QEPushButton::writeOnRelease [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false

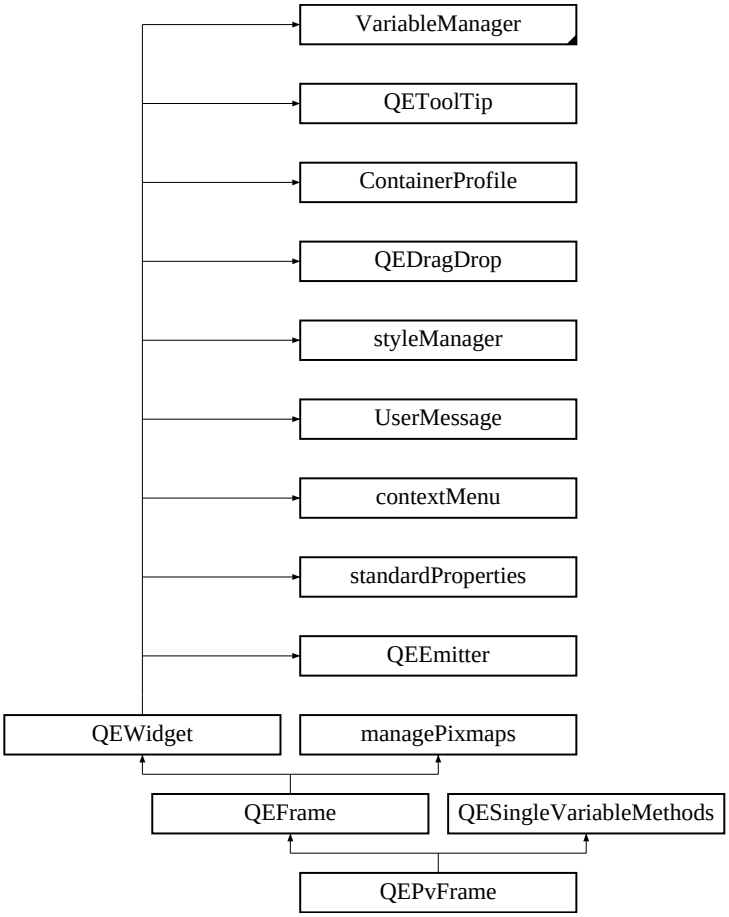
Reimplemented from QEGenericButton.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QEPushButton.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPushButton.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QEPushButton.cpp

9.218 QEPvFrame Class Reference

`#include <QEPvFrame.h>`Inheritance diagram for QEPvFrame::



Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- **QEPvFrame** (QWidget ∗parent=0)

- **QEPvFrame** (const QString &variableName, QWidget *parent=0)

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- QString **copyVariable** ()
- QVariant **copyData** ()

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex

9.218.1 Detailed Description

The QEPvFrame class provides an extension to the QEFrame class in that it allows the value of a nominated PV to select one of 8 pixmaps to use as frame background.

9.218.2 Member Function Documentation

9.218.2.1 void QEPvFrame::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.218.2.2 void QEPvFrame::dbValueChanged () [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.

9.218.3 Property Documentation

9.218.3.1 int QEPvFrame::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.218.3.2 int QEPvFrame::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.218.3.3 QString QEPvFrame::variable [read, write]

EPICS variable name (CA PV)

9.218.3.4 QString QEPvFrame::variableSubstitutions [read, write]

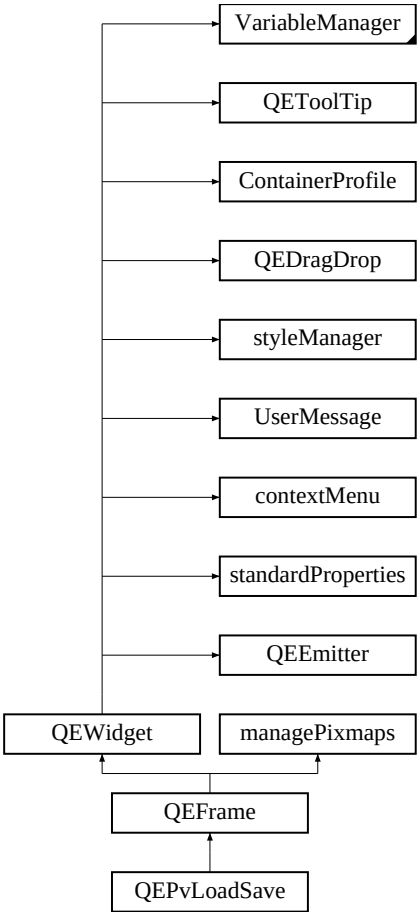
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFrame/QEPvFrame.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPvFrame.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEFrame/QEPvFrame.cp

9.219 QEPvLoadSave Class Reference

Inheritance diagram for QEPvLoadSave::



Classes

- class **Halves**

Signals

- void **requestAction** (const QEActionRequests &)

Public Member Functions

- QEPvLoadSave (QWidget *parent=0)
- virtual ~QEPvLoadSave ()

Destruction.

- virtual QSize sizeHint () const

  Size hint.

- void **setConfigurationFileLeft** (QString configurationFile)
- QString **getConfigurationFileLeft** ()
- void **setConfigurationFileRight** (QString configurationFile)
- QString **getConfigurationFileRight** ()
- void **setSubstitutions** (QString configurationFileSubstitutions)
- QString **getSubstitutions** ()
- void **setDefaultDir** (const QString &defaultDir)
- QString **getDefaultDir** () const
- void **setConfirmAction** (bool confirmRequiredIn)
- bool **getConfirmAction** () const

Static Public Attributes

- static const int **NumberOfButtons** = 15

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **resizeEvent** (QResizeEvent *)
- bool **eventFilter** (QObject *obj, QEvent *event)

Properties

- QString configurationFileLeft
- QString **configurationFileRight**
- QString defaultSubstitutions
- QString defaultDir
- bool confirmAction

Friends

- class QEPvLoadSaveCompare
- class QEPvLoadSaveModel

9.219.1 Constructor & Destructor Documentation

9.219.1.1 QEPvLoadSave::QEPvLoadSave (QWidget * *parent* = 0)

Create without a nominated config file.

9.219.2 Property Documentation

9.219.2.1 QString QEPvLoadSave::configurationFileLeft [read, write]

configurationFile

9.219.2.2 bool QEPvLoadSave::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the PV write actions should be carried out. Defaults to true.

9.219.2.3 QString QEPvLoadSave::defaultDir [read, write]

Default directory used for loading/saving files. Default to null string which is interpreted as the current directory.

9.219.2.4 QString QEPvLoadSave::defaultSubstitutions [read, write]

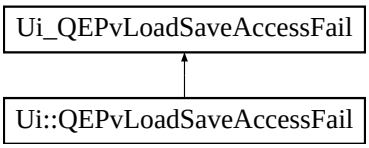
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSave.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPvLoadSave.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSave.cpp

9.220 Ui::QEPvLoadSaveAccessFail Class Reference

Inheritance diagram for Ui::QEPvLoadSaveAccessFail::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveAccessFail.h

9.221 QEPvLoadSaveAccessFail Class Reference

`#include <QEPvLoadSaveAccessFail.h>`

Public Member Functions

- **QEPvLoadSaveAccessFail** (QEPvLoadSave ∗owner, QWidget ∗parent=0)
- void **clear** ()
- void **addPVName** (const QString &pvName)
- void **addPVNames** (const QStringList &pvNameList)

9.221.1 Detailed Description

This widget displays a list of PVs that failed to extracted, applied or read from the archive,

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveAcces
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveAcces

9.222 QEPvLoadSaveCommon Class Reference

Public Types

- enum **ActionKinds** {
  NullAction = 0, **Apply**, **Extract**, **ReadArchive**,
  Update }
- enum **ColumnKinds** {
  NodeName = 0, **LoadSave**, **Live**, **Delta**,
  NUMBER_OF_COLUMNS }
- typedef QMap< QString, double > **PvNameValueMaps**

Static Public Member Functions

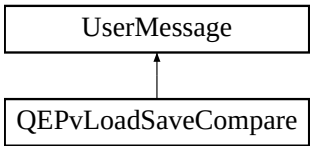
- static PvNameValueMaps **merge** (const PvNameValueMaps &a, const PvNameValueMaps &b)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo

9.223 QEPvLoadSaveCompare Class Reference

`#include <QEPvLoadSaveCompare.h>`Inheritance diagram for QEPvLoadSaveCompare::



Classes

- struct **ValuePairs**

Public Member Functions

- **QEPvLoadSaveCompare** (QEPvLoadSave ∗owner, const int side, QWidget ∗parent=0)
- void **processModelData** ()

9.223.1 Detailed Description

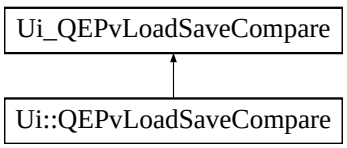
This widget displays graphically the difference between two PV data sets from the left/right hand side of a QEPvLoadSave.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveComp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveComp

9.224 Ui::QEPvLoadSaveCompare Class Reference

Inheritance diagram for Ui::QEPvLoadSaveCompare::

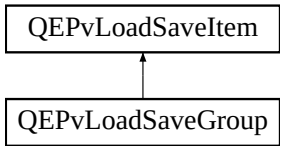


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveCompare.h

9.225 QEPvLoadSaveGroup Class Reference

Inheritance diagram for QEPvLoadSaveGroup::



Public Member Functions

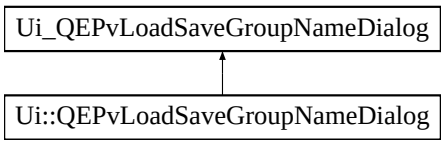
- **QEPvLoadSaveGroup** (const QString &groupName, QEPvLoadSaveItem *parent=0)
- QVariant **getData** (int column) const
- bool **getIsGroup** () const
- QEPvLoadSaveItem * **clone** (QEPvLoadSaveItem *parent)
- void **actionConnect** (QObject *actionCompleteObject, const char *actionCompleteSlot, const char *actionInCompleteSlot)
- void **extractPVData** ()
- void **applyPVData** ()
- void **readArchiveData** (const QCaDateTime &dateTime)
- void **abortAction** ()
- int **leafCount** () const
- QEPvLoadSaveCommon::PvNameValueMaps **getPvNameValueMap** () const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveItem.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveItem.c

9.226 Ui::QEPvLoadSaveGroupNameDialog Class Reference

Inheritance diagram for Ui::QEPvLoadSaveGroupNameDialog::

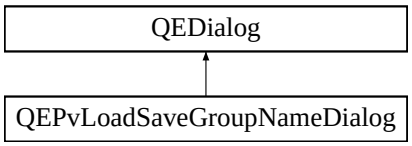


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveGroupNameDialog.h

9.227 QEPvLoadSaveGroupNameDialog Class Reference

Inheritance diagram for QEPvLoadSaveGroupNameDialog::



Public Member Functions

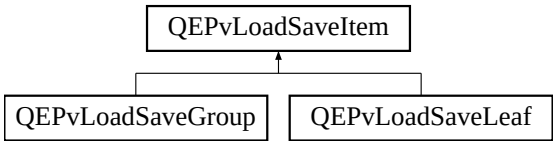
- **QEPvLoadSaveGroupNameDialog** (QWidget *parent=0)
- void **setGroupName** (QString pvNameIn)
- QString **getGroupName** ()
- bool **isClear** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveGroup
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveGroup

9.228 QEPvLoadSaveItem Class Reference

`#include <QEPvLoadSaveItem.h>`Inheritance diagram for QEPvLoadSaveItem::



Public Member Functions

- int **columnCount** () const
- QEPvLoadSaveItem * **getChild** (int position) const
- QEPvLoadSaveItem * **getParent** () const
- int **childCount** () const
- int **childPosition** () const
- virtual QVariant **getData** (int column) const
- bool **insertChild** (int position, QEPvLoadSaveItem *getChild)
- bool **removeChildren** (int position, int count)
- void **appendChild** (QEPvLoadSaveItem *getChild)
- QEPvLoadSaveItem * **getNamedChild** (const QString &searchName)
- virtual QEPvLoadSaveItem * **clone** (QEPvLoadSaveItem *parent)
- virtual void **actionConnect** (QObject *actionCompleteObject, const char *actionCompleteSlot, const char *actionInCompleteSlot)
- QStringList **getNodePath** ()
- virtual void **setNodeName** (const QString &nodeName)
- QString **getNodeName** () const
- void **setNodeValue** (const QVariant &value)
- QVariant **getNodeValue** () const
- int **getElementCount** () const
- virtual bool **getIsPV** () const
- virtual bool **getIsGroup** () const
- virtual void **extractPVData** ()
- virtual void **applyPVData** ()
- virtual void **readArchiveData** (const QCaDateTime &dateTime)
- virtual void **abortAction** ()
- virtual int **leafCount** () const
- virtual QEPvLoadSaveCommon::PvNameValueMaps **getPvNameValueMap** () const

Static Public Member Functions

- static void **initReadArchiveData** ()

Protected Member Functions

- **QEPvLoadSaveItem** (const QString &nodeName, const QVariant &value, QEPvLoadSaveItem *parent=0)

Protected Attributes

- QList< QEPvLoadSaveItem * > **childItems**
- QEPvLoadSaveItem * **parentItem**
- QVariant **value**
- QVariant **liveValue**
- QString **nodeName**

Static Protected Attributes

- static int **readArchiveCount** = 0

9.228.1 Detailed Description

This class is based on the TreeItem example specified in: http://qt-project.org/doc/qt-4.8/itemviews-editabletreemodel.html

A major difference is that it is derived from QObject because each leaf item is associated with a PV and needs slots to receive value data. A consequence of this is that this class must be exposed in a header file even though it is essentially a QEPvLoadSave private class. Also there is no itemData variant array - these values calculated as an when needed. Lastly some function name changes such as parent => getParent (as parent already inherited from QObject) and some changes just to follow my prefered style.

Note: although QObjects provide a parent/child linking, this class manages it's own parent child relationships as per the TreeItem example.

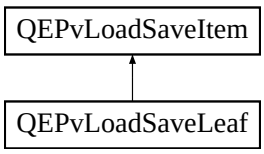
QEPvLoadSaveItem are created in one of two flavours: a/ node - used for groups b/ leaf - used for PVs. both of which inherited from the QEPvLoadSaveItem base class

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveItem.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveItem.c

9.229 QEPvLoadSaveLeaf Class Reference

Inheritance diagram for QEPvLoadSaveLeaf::



Signals

- void **reportActionComplete** (const QEPvLoadSaveItem *item, const QEPvLoadSaveCommon::ActionKinds action, const bool actionSuccessful)
- void **reportActionInComplete** (const QEPvLoadSaveItem *item, const QEPvLoadSaveCommon::ActionKinds action)

Public Member Functions

- **QEPvLoadSaveLeaf** (const QString &setPointPvName, const QString &readBackPvName, const QString &archiverPvName, const QVariant &value, QEPvLoadSaveItem *parent=0)
- void **setSetPointPvName** (const QString &pvName)
- QString **getSetPointPvName** () const
- void **setReadBackPvName** (const QString &pvName)
- QString **getReadBackPvName** () const
- void **setArchiverPvName** (const QString &pvName)
- QString **getArchiverPvName** () const
- QVariant **getData** (int column) const
- void **setNodeName** (const QString &nodeName)
- bool **getIsPV** () const
- QEPvLoadSaveItem * **clone** (QEPvLoadSaveItem *parent)
- QEPvLoadSaveCommon::PvNameValueMaps **getPvNameValueMap** () const
- void **actionConnect** (QObject *actionCompleteObject, const char *actionCompleteSlot, const char *actionInCompleteSlot)
- void **extractPVData** ()
- void **applyPVData** ()
- void **readArchiveData** (const QCaDateTime &dateTime)
- void **abortAction** ()
- int **leafCount** () const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPvLoadSaveItem.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo

9.230 QEPvLoadSaveModel Class Reference

`#include <QEPvLoadSaveModel.h>`

Signals

- void **reportActionComplete** (const QEPvLoadSaveItem *item, QEPvLoadSaveCommon::ActionKinds, bool)
- void **reportActionInComplete** (const QEPvLoadSaveItem *item, QEPvLoadSaveCommon::ActionKinds action)

Public Member Functions

- **QEPvLoadSaveModel** (QTreeView *treeView, QEPvLoadSave *parent)
- QVariant **data** (const QModelIndex &index, int role) const
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
- QModelIndex **index** (int row, int column, const QModelIndex &parent=QModelIndex()) const
- QModelIndex **parent** (const QModelIndex &child) const
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const
- Qt::ItemFlags **flags** (const QModelIndex &index) const
- bool **setData** (const QModelIndex &index, const QVariant &value, int role=Qt::EditRole)
- bool **setHeaderData** (int section, Qt::Orientation orientation, const QVariant &value, int role=Qt::EditRole)
- bool **insertRows** (int position, int rows, const QModelIndex &parent=QModelIndex())
- bool **removeRows** (int position, int rows, const QModelIndex &parent=QModelIndex())
- void **setHeading** (const QString &heading)
- QString **getHeading** () const
- void **setupModelData** (QEPvLoadSaveItem *rootItem, const QString &heading)
- void **modelUpdated** ()
- void **itemUpdated** (const QEPvLoadSaveItem *item, const QEPvLoadSaveCommon::ColumnKinds kind)
- bool **addItemToModel** (QEPvLoadSaveItem *item, QEPvLoadSaveItem *parentItem)
- bool **removeItemFromModel** (QEPvLoadSaveItem *item)
- bool **mergeItemInToModel** (QEPvLoadSaveItem *item)
- void **extractPVData** ()
- void **applyPVData** ()
- void **readArchiveData** (const QCaDateTime &dateTime)
- void **abortAction** ()

- int **leafCount** () const
- QEPvLoadSaveCommon::PvNameValueMaps **getPvNameValueMap** () const
- QEPvLoadSaveItem * **getRootItem** ()
- QEPvLoadSaveItem * **getSelectedItem** ()
- QModelIndex **getRootIndex** ()
- QEPvLoadSaveItem * **indexToItem** (const QModelIndex &index) const

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)

9.230.1 Detailed Description

This class is based on the TreeModel example specified in: http://qt-project.org/doc/qt-4.8/itemviews-editabletreemodel.html

Note on naming: the example's root item that provide header info is refered to the core item. It only ever has one child which is the visible (on the QTreeView) which is the user root item, named "ROOT".

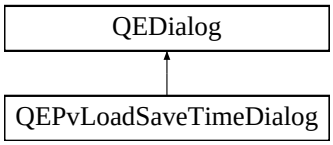
Note: we only re-size the number of rows (children). The number of columns is fixed.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPvLoadSaveModel.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLo

9.231 QEPvLoadSaveTimeDialog Class Reference

Inheritance diagram for QEPvLoadSaveTimeDialog::



Public Member Functions

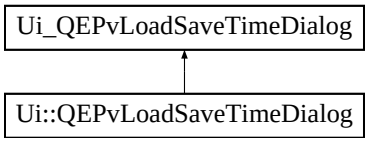
- **QEPvLoadSaveTimeDialog** (QWidget ∗parent=0)
- void **setMaximumDateTime** (const QDateTime &datetime)
- void **setDateTime** (const QDateTime &datetime)
- QDateTime **getDateTime** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveTimeI
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveTimeI

9.232 Ui::QEPvLoadSaveTimeDialog Class Reference

Inheritance diagram for Ui::QEPvLoadSaveTimeDialog::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveTimeDialog.h

9.233 QEPvLoadSaveUtilities Class Reference

Static Public Member Functions

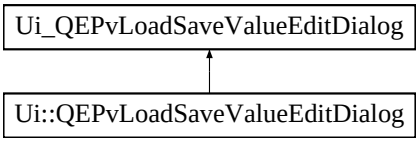
- static QEPvLoadSaveItem * **readTree** (const QString &filename, const QString &macroString)
- static bool **writeTree** (const QString &filename, const QEPvLoadSaveItem *root)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveUtiliti
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveUtiliti

9.234 Ui::QEPvLoadSaveValueEditDialog Class Reference

Inheritance diagram for Ui::QEPvLoadSaveValueEditDialog::

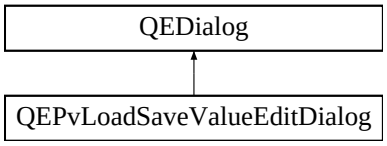


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveValueEditDialog.h

9.235 QEPvLoadSaveValueEditDialog Class Reference

Inheritance diagram for QEPvLoadSaveValueEditDialog::



Public Member Functions

- **QEPvLoadSaveValueEditDialog** (QWidget *parent=0)
- void **setPvName** (const QString &pvName)
- void **setValue** (const QVariant &valueList)
- QVariant **getValue** () const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveValue
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvLoadSave/QEPvLoadSaveValue

9.236 QEPVNameLists Class Reference

Public Member Functions

- void **prependOrMoveToFirst** (const QString &item)
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Static Public Member Functions

- static void **constructor** ()

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.237 QEPvNameSearch Class Reference

`#include <QEPvNameSearch.h>`

Public Member Functions

- **QEPvNameSearch** (const QEPvNameSearch &other)
- **QEPvNameSearch** (const QStringList &pvNameList)
- void **setPvNameList** (const QStringList &pvNameList)
- void **addPvNameList** (const QStringList &pvNameList)
- QStringList **getAllPvNames** () const
- QStringList **getMatchingPvNames** (const QRegExp &regExp, const bool exactMatch) const
- QStringList **getMatchingPvNames** (const QString &str, const Qt::CaseSensitivity cs) const

9.237.1 Detailed Description

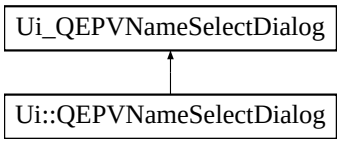
Provides a basic name search capability. Extracted from archiver manager in ordger to provide a more flexibility, i.e. sets of PV names which can be source from any source.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPvNameSearch.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPvNameSearch.cpp

9.238 Ui::QEPVNameSelectDialog Class Reference

Inheritance diagram for Ui::QEPVNameSelectDialog::

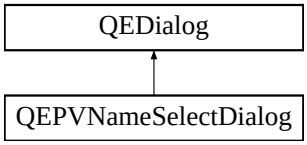


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPVNameSelectDialog.h

9.239 QEPVNameSelectDialog Class Reference

Inheritance diagram for QEPVNameSelectDialog::



Public Member Functions

- **QEPVNameSelectDialog** (QWidget ∗parent=0)
- void **setPvName** (QString pvNameIn)
- QString **getPvName** ()

Protected Member Functions

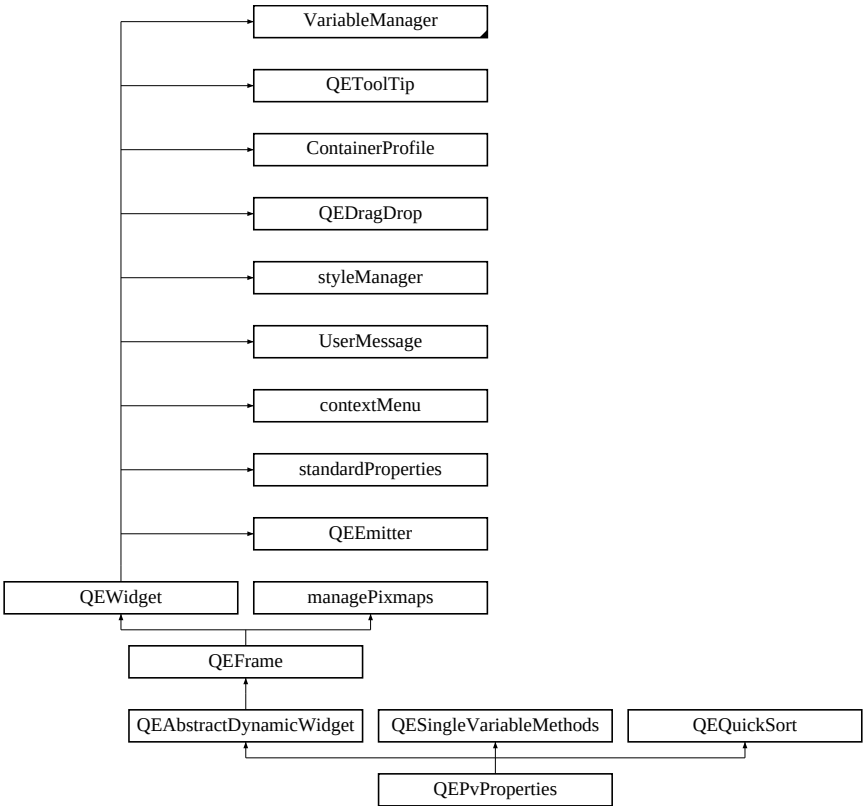
- void **closeEvent** (QCloseEvent ∗e)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPVNameSelectDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEPVNameSelectDialog.cpp

9.240 QEPvProperties Class Reference

Inheritance diagram for QEPvProperties::



Public Types

- enum **OwnContextMenuOptions** {

  PVPROP_NONE = QEAbstractDynamicWidget::ADWCM_SUB_CLASS_-WIDGETS_START_HERE, **PVPROP_SORT_FIELD_NAMES**, **PVPROP_-RESET_FIELD_NAMES**, **PVPROP_PROCESS_RECORD**,

  PVPROP_SUB_CLASS_WIDGETS_START_HERE }

Signals

- void **setCurrentBoxIndex** (int index)

Public Member Functions

- **QEPvProperties** (QWidget ∗parent=0)

- **QEPvProperties** (const QString &variableName, QWidget *parent=0)
- QSize **sizeHint** () const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()

Protected Member Functions

- void **resizeEvent** (QResizeEvent *event)
- QMenu * **buildContextMenu** ()
- void **contextMenuTriggered** (int selectedItemNum)
- bool **itemLessThan** (const int a, const int b, QObject *context=NULL) const
- void **swapItems** (const int a, const int b, QObject *context=NULL)
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void saveConfiguration (PersistanceManager *pm)
- void restoreConfiguration (PersistanceManager *pm, restorePhases restorePhase)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **enableEditPvChanged** ()

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex

9.240.1 Member Function Documentation

9.240.1.1 void QEPvProperties::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.240.1.2 void QEPvProperties::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

9.240.2 Property Documentation

9.240.2.1 int QEPvProperties::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.240.2.2 int QEPvProperties::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.240.2.3 QString QEPvProperties::variable [read, write]

EPICS variable name (CA PV)

9.240.2.4 QString QEPvProperties::variableSubstitutions [read, write]

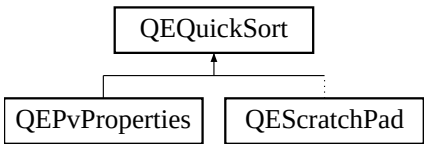
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPro
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEPvProperties.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPro

9.241 QEQuickSort Class Reference

`#include <QEQuickSort.h>`Inheritance diagram for QEQuickSort::



Public Member Functions

- void **sort** (const int first, const int last, QObject ∗context=NULL)

Protected Member Functions

- virtual bool **itemLessThan** (const int a, const int b, QObject ∗context) const =0
- virtual void **swapItems** (const int a, const int b, QObject ∗context)=0

9.241.1 Detailed Description

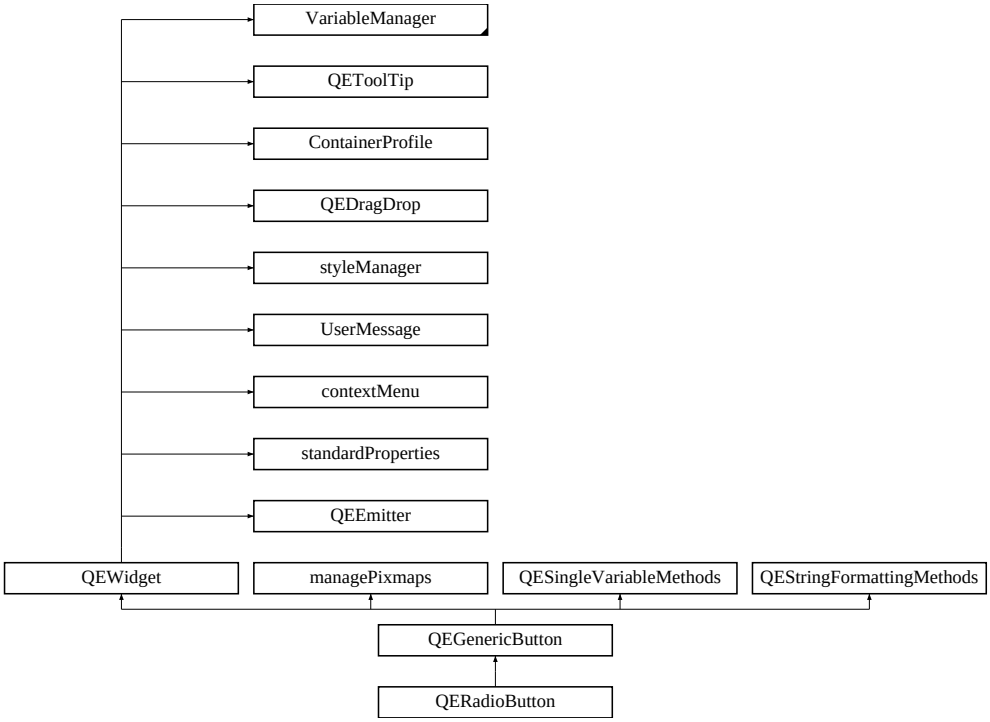
This base class provides the means to add quick sort functionality to a class. Alas the algorithms qSort API does not accept a class instance LessThan function, so we roll out own. The items to be sorted must be indexable via consecutive integers.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEQuickSort.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEQuickSort.cpp

9.242 QERadioButton Class Reference

Inheritance diagram for QERadioButton::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }
- enum Formats {

  Default = QEStringFormatting::FORMAT_DEFAULT, Floating = QEStringFormatting::FORMAT_FLOATING, Integer = QEStringFormatting::FORMAT_INTEGER, UnsignedInteger = QEStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QEStringFormatting::FORMAT_TIME, LocalEnumeration = QEStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum Separators { NoSeparator = QEStringFormatting::SEPARATOR_-NONE, Comma = QEStringFormatting::SEPARATOR_COMMA, Underscore = QEStringFormatting::SEPARATOR_UNDERSCORE, Space = QEStringFormatting::SEPARATOR_SPACE }

- enum Notations { Fixed = QEStringFormatting::NOTATION_FIXED, Scientific = QEStringFormatting::NOTATION_SCIENTIFIC, Automatic = QEStringFormatting::NOTATION_AUTOMATIC }
- enum ArrayActions { Append = QEStringFormatting::APPEND, Ascii = QEStringFormatting::ASCII, Index = QEStringFormatting::INDEX }
- enum UpdateOptions {

  Text = QEGenericButton::UPDATE_TEXT, Icon = QEGenericButton::UPDATE_ICON, TextAndIcon = QEGenericButton::UPDATE_TEXT_AND_ICON, State = QEGenericButton::UPDATE_STATE,

  TextAndState = QEGenericButton::UPDATE_TEXT_AND_STATE, IconAndState = QEGenericButton::UPDATE_ICON_AND_STATE, TextIconAndState = QEGenericButton::UPDATE_TEXT_ICON_AND_STATE }

  User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

- enum ProgramStartupOptionNames { None = applicationLauncher::PSO_NONE, Terminal = applicationLauncher::PSO_TERMINAL, LogOutput = applicationLauncher::PSO_LOGOUTPUT, StdOutput = applicationLauncher::PSO_STDOUTPUT }
- enum CreationOptionNames {

  Open = QEActionRequests::OptionOpen, NewTab = QEActionRequests::OptionNewTab, NewWindow = QEActionRequests::OptionNewWindow, DockTop = QEActionRequests::OptionTopDockWindow,

  DockBottom = QEActionRequests::OptionBottomDockWindow, DockLeft = QEActionRequests::OptionLeftDockWindow, DockRight = QEActionRequests::OptionRightDockWindow, DockTopTabbed = QEActionRequests::OptionTopDockWindowTabbed,

  DockBottomTabbed = QEActionRequests::OptionBottomDockWindowTabbed, DockLeftTabbed = QEActionRequests::OptionLeftDockWindowTabbed, DockRightTabbed = QEActionRequests::OptionRightDockWindowTabbed, DockFloating = QEActionRequests::OptionFloatingDockWindow }

  Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Public Slots

- void requestAction (const QEActionRequests &request)
- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void requestResend ()

  Internal use only. Used when changing a property value to force a re-display to reflect the new property value.

- void newGui (const QEActionRequests &request)

  Internal use only. Request a new GUI is created. Typically, this is caught by the QEGui application.

- void pressed (int value)
- void released (int value)
- void clicked (int value)
- void programCompleted ()

  Program started by button has compelted.

Public Member Functions

- QERadioButton (QWidget *parent=0)
- QERadioButton (const QString &variableName, QWidget *parent=0)
- void writeNow ()
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setFormatProperty (Formats format)

  Access function for format property - refer to format property for details.

- Formats getFormatProperty ()

  Access function for format property - refer to format property for details.

- void setSeparatorProperty (const Separators notation)

  Access function for separator property - refer to separator property for details.

- Separators getSeparatorProperty () const

  Access function for separator property - refer to separator property for details.

- void setNotationProperty (Notations notation)

  Access function for notation property - refer to notation property for details.

- Notations getNotationProperty ()

  Access function for notation property - refer to notation property for details.

- void setArrayActionProperty (ArrayActions arrayAction)

  Access function for arrayAction property - refer to arrayAction property for details.

- ArrayActions getArrayActionProperty ()

  Access function for arrayAction property - refer to arrayAction property for details.

Properties

- QString variable
- QString variableSubstitutions
- int arrayIndex
- bool subscribe
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle

- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- int precision
- bool useDbPrecision
- bool leadingZero
- bool trailingZeros
- bool addUnits
- bool forceSign
- QString localEnumeration
- Formats format
- int radix
- Separators separator
- Notations notation
- ArrayActions arrayAction
- QEWidgetProperties::DisabledRecordPolicy disabledRecordPolicy
- Qt::Alignment alignment
- UpdateOptions updateOption
- QPixmap pixmap0
- QPixmap pixmap1
- QPixmap pixmap2
- QPixmap pixmap3
- QPixmap pixmap4
- QPixmap pixmap5
- QPixmap pixmap6
- QPixmap pixmap7
- QString password
- bool confirmAction
- QString confirmText
- bool writeOnPress
- bool writeOnRelease
- bool writeOnClick
- QString pressText
- QString releaseText
- QString clickText
- QString clickCheckedText
- QString labelText
- QString program
- QStringList arguments
- ProgramStartupOptionNames programStartupOption
- QString guiFile
- CreationOptionNames creationOption
- QString prioritySubstitutions
- QString customisationName

9.242.1 Member Enumeration Documentation

9.242.1.1 enum QERadioButton::ArrayActions

User friendly enumerations for arrayAction property - refer to QEStringFormatting::arrayActions for details.

Enumerator:

Append Refer to QEStringFormatting::APPEND for details.

Ascii Refer to QEStringFormatting::ASCII for details.

Index Refer to QEStringFormatting::INDEX for details.

9.242.1.2 enum QERadioButton::CreationOptionNames

Creation options. Used to indicate how to present a GUI when requesting a new GUI be created. Open a new window, open a new tab, or replace the current window.

Enumerator:

Open Replace the current GUI with the new GUI.

NewTab Open new GUI in a new tab.

NewWindow Open new GUI in a new window.

DockTop Open new GUI in a top dock window.

DockBottom Open new GUI in a bottom dock window.

DockLeft Open new GUI in a left dock window.

DockRight Open new GUI in a right dock window.

DockTopTabbed Open new GUI in a top dock window (tabbed with any existing dock in that area).

DockBottomTabbed Open new GUI in a bottom dock window (tabbed with any existing dock in that area).

DockLeftTabbed Open new GUI in a left dock window (tabbed with any existing dock in that area).

DockRightTabbed Open new GUI in a right dock window (tabbed with any existing dock in that area).

DockFloating Open new GUI in a floating dock window.

9.242.1.3 enum QERadioButton::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.242.1.4 enum QERadioButton::Formats

User friendly enumerations for format property - refer to QEStringFormatting::formats for details.

Enumerator:

Default Format as best appropriate for the data type.

Floating Format as a floating point number.

Integer Format as an integer.

UnsignedInteger Format as an unsigned integer.

Time Format as a time.

LocalEnumeration Format as a selection from the localEnumeration property.

9.242.1.5 enum QERadioButton::Notations

User friendly enumerations for notation property - refer to QEStringFormatting::notations for details.

Enumerator:

Fixed Refer to QEStringFormatting::NOTATION_FIXED for details.

Scientific Refer to QEStringFormatting::NOTATION_SCIENTIFIC for details.

Automatic Refer to QEStringFormatting::NOTATION_AUTOMATIC for details.

9.242.1.6 enum QERadioButton::ProgramStartupOptionNames

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

Enumerator:

None Just run the program.

Terminal Run the program in a termainal (in Windows a command interpreter will also be started, so the program may be a built-in command like 'dir').

LogOutput Run the program, and log the output in the QE message system.

StdOutput Run the program, and send doutput to standard output and standard error.

9.242.1.7 enum QERadioButton::Separators

User friendly enumerations for seprator property - refer to QEStringFormatting::formats for details.

Enumerator:

NoSeparator Use no separator.

Comma Use ',' as separator.

Underscore Use '_' as separator.

Space Use ' ' as separator.

9.242.1.8 enum QERadioButton::UpdateOptions

User friendly enumerations for updateOption property - refer to QEGenericButton::updateOptions for details.

Enumerator:

Text Data updates will update the button text.

Icon Data updates will update the button icon.

TextAndIcon Data updates will update the button text and icon.

State Data updates will update the button state (checked or unchecked).

TextAndState Data updates will update the button text and state.

IconAndState Data updates will update the button icon and state.

TextIconAndState Data updates will update the button - the lot.

9.242.1.9 enum QERadioButton::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.242.2 Constructor & Destructor Documentation

9.242.2.1 QERadioButton::QERadioButton (QWidget * *parent* = 0)

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.242.2.2 QERadioButton::QERadioButton (const QString & *variableName*, QWidget * *parent* = 0)

Create with a variable. A connection is automatically established. If macro substitutions are required, create without a variable and set the variable and macro substitutions after creation.

9.242.3 Member Function Documentation

9.242.3.1 void QERadioButton::clicked (int *value*) [signal]

Button has been Clicked. The value emitted is the integer interpretation of the clickText property (or the clickCheckedText property if the button was checked)

9.242.3.2 void QERadioButton::dbValueChanged () [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.

9.242.3.3 void QERadioButton::pressed (int *value*) [signal]

Button has been Pressed. The value emitted is the integer interpretation of the pressText property

9.242.3.4 void QERadioButton::released (int *value*) [signal]

Button has been Released The value emitted is the integer interpretation of the releaseText property

9.242.3.5 void QERadioButton::requestAction (const QEActionRequests & *request*) [inline, slot]

Default slot used to create a new GUI if there is no slot indicated in the ContainerProfile class. This slot is typically used when the button is pressed within the Designer preview window to allow the operation of the button to be tested. If an application does not specify a slot to use for creating new windows (through the ContainerProfile class) a window will still be created through this slot, but it will not respect the window creation options or any other window related application constraints. For example, the QEGui application does provide a slot for creating new GUIs in the ContainerProfile class which respects the creation options, knows how to add tabs in the application, and extend the application's window menu in the menu bar.

9.242.3.6 void QERadioButton::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.242.3.7 void QERadioButton::writeNow () [inline, virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.242.4 Property Documentation

9.242.4.1 bool QERadioButton::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.242.4.2 Qt::Alignment QERadioButton::alignment [read, write]

Set the buttons text alignment. Left justification is particularly useful when displaying quickly changing numeric data updates.

9.242.4.3 bool QERadioButton::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEDragDrop.

9.242.4.4 QStringList QERadioButton::arguments [read, write]

Arguments for program specified in the 'program' property.

9.242.4.5 ArrayActions QERadioButton::arrayAction [read, write]

Text formatting option for array data. Default is ASCII. Options are:

- ASCII - treat array as a single text string. For example an array of three characters 'a' 'b' 'c' will be formatted as 'abc'.
- APPEND - treat array as an array of numbers and format a string containing them all with a space between each. For example, an array of three numbers 10, 11 and 12 will be formatted as '10 11 12'.

- INDEX - Extract a single item from the array. The item is then formatted as any other non array data would be. The item selected is determined by the arrayIndex property. For example, if arrayIndex property is 1, an array of three numbers 10, 11 and 12 will be formatted as '11'.

9.242.4.6 int QERadioButton::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.242.4.7 QString QERadioButton::clickCheckedText [read, write]

Text used to compare with text written or read to determine if push button should be marked as checked. Note, must be an exact match following formatting of data updates. When writing values, the 'pressText', 'ReleaseText', or 'clickedtext' must match this property to cause the button to be checked when the write occurs.

Good example: formatting set to diaplay a data value of '1' as 'On', clickCheckedText is 'On', clickText is 'On'. In this example, the push button will be checked when a data update occurs with a value of 1 or when the button is clicked.

Bad example: formatting set to diaplay a data value of '1' as 'On', clickCheckedText is 'On', clickText is '1'. In this example, the push button will be checked when a data update occurs with a value of 1 but, although a valid value will be written when clicked, the button will not be checked when clicked as '1' is not the same as 'On'.

Reimplemented from QEGenericButton.

9.242.4.8 QString QERadioButton::clickText [read, write]

Value written when user clicks button if 'writeOnClick' property is true

Reimplemented from QEGenericButton.

9.242.4.9 bool QERadioButton::confirmAction [read, write]

If true, a dialog will be presented asking the user to confirm if the button action should be carried out

9.242.4.10 QString QERadioButton::confirmText [read, write]

Text used to confirm acion if confirmation dialog is presented

Reimplemented from QEGenericButton.

9.242.4.11 CreationOptionNames QERadioButton::creationOption [read, write]

Creation options when opening a new GUI. Open a new window, open a new tab, or replace the current window. the creation option is supplied when the button generates a newGui signal. Application code connected to this signal should honour this request if possible. When used within the QEGui application, the QEGui application creates a new window, new tab, or replaces the current window as appropriate.

Reimplemented from QEGenericButton.

9.242.4.12 QString QERadioButton::customisationName [read, write]

Window customisation name. This name will be used to select a set of window customisations including menu items and tool bar buttons. Applications such as QEGui can load .xml files containing named sets of window customisations. This property is used to select a set loaded from these files. The selected set of customisations will be applied to the main window containing the new GUI. Customisations are not applied if the GUI is opened as a dock.

Reimplemented from QEGenericButton.

9.242.4.13 QString QERadioButton::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.242.4.14 QEWidgetProperties::DisabledRecordPolicy QERadioButton::disabledRecordPolicy [read, write]

Set the widget's disabled record policy, i.e. the action to be taken when the under lying record is disabled, i.e. when the assiociated record's DISA and DISV field values are equal. Note: this is only applicable IOC process variables. When the policy is ignore, then no special action is taken. This is the default policy. When the policy is grayout, the widget is style is set as if disconnected when the record is disabled.

Reimplemented from QEGenericButton.

9.242.4.15 bool QERadioButton::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.242.4.16 DisplayAlarmStateOptions QERadioButton::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.242.4.17 bool QERadioButton::forceSign [read, write]

If false (default), no "+" sign, when true always add a sign.

9.242.4.18 Formats QERadioButton::format [read, write]

Format to apply to data. Default is 'Default' in which case the data type supplied with the data determines how the data is formatted. For all other options, an attempt is made to format the data as requested (whatever its native form).

9.242.4.19 QString QERadioButton::guiFile [read, write]

File name of GUI to be presented on button click. File name can be absolute, relative to the path of the QEform in which the QEPushButton is located, relative to the any path in the path list published in the ContainerProfile class, or relative to the current path. See QEWidget::openQEFile() in QEWidget.cpp for details.

9.242.4.20 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.

9.242.4.21 QString QERadioButton::labelText [read, write]

Button label text (prior to substitution). Macro substitutions will be applied to this text and the result will be set as the button text. Used when data updates are not being represented in the button text. IF NOT LEFT EMPTY, THIS TEXT WILL TAKE PRIORITY OVER THE PUSH BUTTON 'text' PROPERTY! For example, a button in a sub form may have a 'labelText' property of 'Turn Pump On'. When the sub form is used twice in a main form with substitutions PUMPNUM=1 and PUMPNUM=2 respectively, the two identical buttons in the sub forms will have the labels 'Turn Pump 1 On' and 'Turn Pump 2 On' respectively.

Reimplemented from QEGenericButton.

9.242.4.22 bool QERadioButton::leadingZero [read, write]

If true (default), always add a leading zero when formatting numbers.

9.242.4.23 QString QERadioButton::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.242.4.24 Notations QERadioButton::notation [read, write]

Notation used for numerical formatting. Default is fixed.

9.242.4.25 QString QERadioButton::password [read, write]

Password user will need to enter before any action is taken

Reimplemented from QEGenericButton.

9.242.4.26 QPixmap QERadioButton::pixmap0 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 0

9.242.4.27 QPixmap QERadioButton::pixmap1 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 1

9.242.4.28 QPixmap QERadioButton::pixmap2 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 2

9.242.4.29 QPixmap QERadioButton::pixmap3 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 3

9.242.4.30 QPixmap QERadioButton::pixmap4 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 4

9.242.4.31 QPixmap QERadioButton::pixmap5 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 5

9.242.4.32 QPixmap QERadioButton::pixmap6 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 6

9.242.4.33 QPixmap QERadioButton::pixmap7 [read, write]

Pixmap to display if updateOption is Icon or TextAndIcon and data value translates to an index of 7

9.242.4.34 int QERadioButton::precision [read, write]

Precision used when formatting floating point numbers. The default is 4. This is only used if useDbPrecision is false.

9.242.4.35 QString QERadioButton::pressText [read, write]

Value written when user presses button if 'writeOnPress' property is true

Reimplemented from QEGenericButton.

9.242.4.36 QString QERadioButton::prioritySubstitutions [read, write]

Overriding macro substitutions. These macro substitions take precedence over any existing macro substitutions defined by the variableSubstitutions property, any parent forms, or the application containing the button. These macro substitutions are particularly usefull when the button's function is to reload the same form but with different macro substitutions. The variableSubstitutions property cannot be used for this since, although they are added to the list of macro substittions applied to the new form, they are appended to the list and the existing macro substitutions take precedence.

Reimplemented from QEGenericButton.

9.242.4.37 QString QERadioButton::program [read, write]

Program to run when the button is clicked. No attempt to run a program is made if this property is empty. Example: firefox

9.242.4.38 ProgramStartupOptionNames QERadioButton::programStartupOption [read, write]

Startup options. Just run the command, run the command within a terminal, or display the output in QE message system.

9.242.4.39 int QERadioButton::radix [read, write]

Base used for when formatting integers. Default is 10 (duh!)

9.242.4.40 QString QERadioButton::releaseText [read, write]

Value written when user releases button if 'writeOnRelease' property is true

Reimplemented from QEGenericButton.

9.242.4.41 Separators QERadioButton::separator [read, write]

Seperators used for interger and fixed point formatting. Default is None.

9.242.4.42 QString QERadioButton::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.242.4.43 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 VariableManager.

9.242.4.44 bool QERadioButton::trailingZeros [read, write]

If true (default), always remove any trailing zeros when formatting numbers.

9.242.4.45 UpdateOptions QERadioButton::updateOption [read, write]

Update options (text, pixmap, both, or state (checked or unchecked)

Reimplemented from QEGenericButton.

9.242.4.46 bool QERadioButton::useDbPrecision [read, write]

If true (default), format floating point numbers using the precision supplied with the data. If false, the precision property is used.

9.242.4.47 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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.242.4.48 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.

9.242.4.49 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.

9.242.4.50 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.

9.242.4.51 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'.

9.242.4.52 QString QERadioButton::variable [read, write]

EPICS variable name (CA PV)

9.242.4.53 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.

9.242.4.54 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 are also used for other purposes.

9.242.4.55 bool QERadioButton::visible [read, write]

Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

9.242.4.56 bool QERadioButton::writeOnClick [read, write]

If true, the 'clickText' property is written when the button is clicked. Default is true

Reimplemented from QEGenericButton.

9.242.4.57 bool QERadioButton::writeOnPress [read, write]

If true, the 'pressText' property is written when the button is pressed. Default is false

Reimplemented from QEGenericButton.

9.242.4.58 bool QERadioButton::writeOnRelease [read, write]

If true, the 'releaseText' property is written when the button is released. Default is false

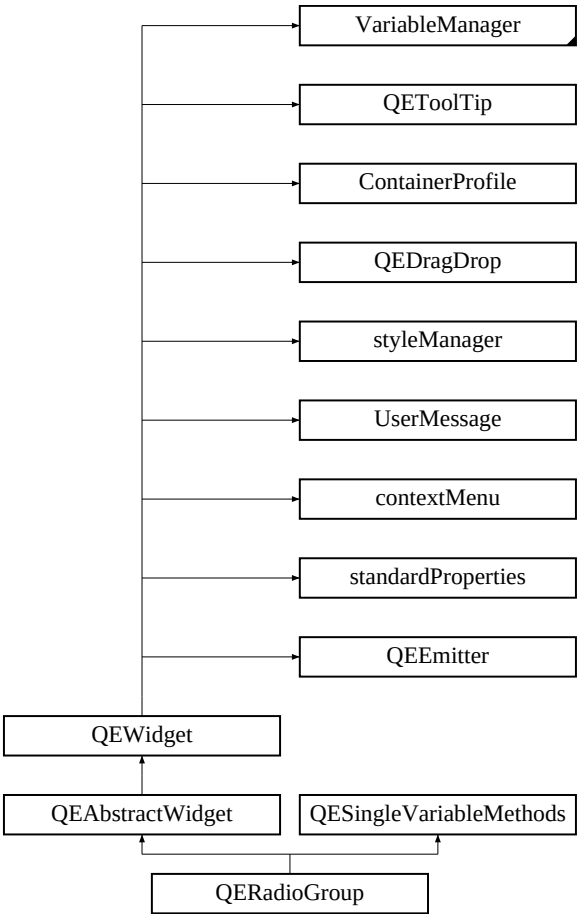
Reimplemented from QEGenericButton.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QERadioButto
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QERadioButton.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEButton/QERadioButto

9.243 QERadioGroup Class Reference

Inheritance diagram for QERadioGroup::



Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)

- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- QERadioGroup (QWidget *parent=0)
- QERadioGroup (const QString &variableName, QWidget *parent=0)
- QERadioGroup (const QString &title, const QString &variableName, QWidget *parent=0)
- virtual ~QERadioGroup ()

  Destruction.

- int **getCurrentIndex** () const
- void setVariableNameSubstitutionsProperty (const QString &substitutions)
- void **setSubstitutedTitleProperty** (const QString &substitutedTitle)
- QString **getSubstitutedTitleProperty** () const
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** () const
- void **setLocalEnumerations** (const QString &localEnumerations)
- QString **getLocalEnumerations** () const

Protected Member Functions

- QSize **sizeHint** () const
- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- QString substitutedTitle
- QString **title**
- int columns
- int spacing
- bool useDbEnumerations
- QString localEnumeration
- QRadioGroup::ButtonStyles buttonStyle
- QRadioGroup::ButtonOrders buttonOrder

9.243.1 Constructor & Destructor Documentation

9.243.1.1 QERadioGroup::QERadioGroup (QWidget * *parent* = 0) [explicit]

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.243.1.2 QERadioGroup::QERadioGroup (const QString & *variableName*, QWidget * *parent* = 0) [explicit]

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.

9.243.1.3 QERadioGroup::QERadioGroup (const QString & *title*, const QString & *variableName*, QWidget * *parent* = 0) [explicit]

Create with a group title and 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.

9.243.2 Member Function Documentation

9.243.2.1 void QERadioGroup::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.243.2.2 void QERadioGroup::dbValueChanged () [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.

9.243.2.3 void QERadioGroup::setVariableNameSubstitutionsProperty (const QString & *substitutions*)

Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

Reimplemented from QESingleVariableMethods.

9.243.3 Property Documentation

9.243.3.1 int QERadioGroup::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.243.3.2 QRadioGroup::ButtonOrders QERadioGroup::buttonOrder [read, write]

Allows selection of buttom order (rowMajor (default) or colMajor)

9.243.3.3 QRadioGroup::ButtonStyles QERadioGroup::buttonStyle [read, write]

Allows selection of buttom style (Radio (default) or Push)

9.243.3.4 int QERadioGroup::columns [read, write]

Number of colums - defaults to two.

9.243.3.5 int QERadioGroup::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.243.3.6 QString QERadioGroup::localEnumeration [read, write]

Enumrations values used when useDbEnumerations is false.

9.243.3.7 int QERadioGroup::spacing [read, write]

Interay layout margins and spacing - defaults to 4.

9.243.3.8 QString QERadioGroup::substitutedTitle [read, write]

Group box title text to be substituted. This text will be copied to the group box title text after applying any macro substitutions from the variableSubstitutions property. The former is depricated and included for backward compatabilty only and not presented on designer.

9.243.3.9 bool QERadioGroup::useDbEnumerations [read, write]

Use database enumerations - defaults to true. False implies use of local enumeration.

9.243.3.10 QString QERadioGroup::variable [read, write]

EPICS variable name (CA PV)

9.243.3.11 QString QERadioGroup::variableSubstitutions [read, write]

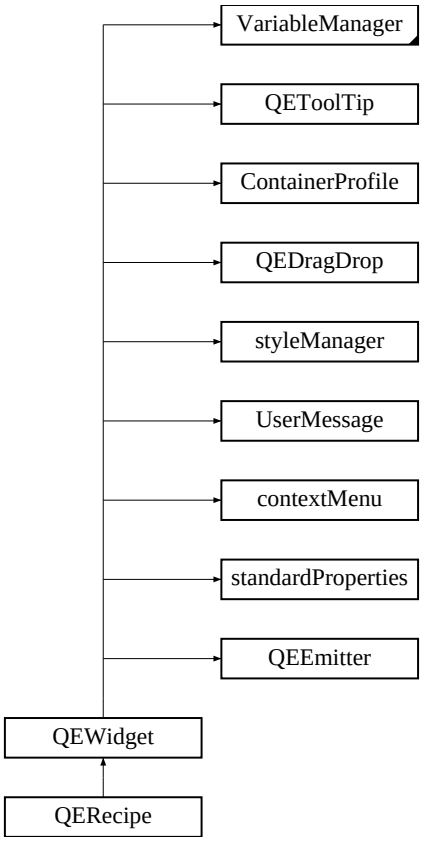
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERadioGroup/QERadioGroup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QERadioGroup.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERadioGroup/QERadioGroup.cpp

9.244 QERecipe Class Reference

Inheritance diagram for QERecipe::



Public Types

- enum **configurationTypesProperty** { **File**, **Text** }
- enum **optionsLayoutProperty** { **Top**, **Bottom**, **Left**, **Right** }
- enum **userTypesProperty** { **User** = userLevelTypes::USERLEVEL_USER, **Scientist** = userLevelTypes::USERLEVEL_SCIENTIST, **Engineer** = userLevelTypes::USERLEVEL_ENGINEER }

Public Member Functions

- **QERecipe** (QWidget ∗pParent=0)
- void **setRecipeDescription** (QString pValue)
- QString **getRecipeDescription** ()
- void **setShowRecipeList** (bool pValue)
- bool **getShowRecipeList** ()

- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowApply** (bool pValue)
- bool **getShowApply** ()
- void **setShowRead** (bool pValue)
- bool **getShowRead** ()
- void **setShowFields** (bool pValue)
- bool **getShowFields** ()
- void **setConfigurationType** (int pValue)
- int **getConfigurationType** ()
- void **setConfigurationFile** (QString pValue)
- QString **getConfigurationFile** ()
- void **setRecipeFile** (QString pValue)
- QString **getRecipeFile** ()
- void **setConfigurationText** (QString pValue)
- QString **getConfigurationText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **setCurrentUserType** (int pValue)
- int **getCurrentUserType** ()
- bool **saveRecipeList** ()
- void **refreshRecipeList** ()
- void **refreshButton** ()
- void **userLevelChanged** (userLevelTypes::userLevels pValue)
- void **setConfigurationTypeProperty** (configurationTypesProperty pConfigurationType)
- configurationTypesProperty **getConfigurationTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- void **setCurrentUserTypeProperty** (userTypesProperty pUserType)
- userTypesProperty **getCurrentUserTypeProperty** ()

Protected Attributes

- QLabel * **qLabelRecipeDescription**
- QComboBox * **qComboBoxRecipeList**
- QPushButton * **qPushButtonNew**
- QPushButton * **qPushButtonSave**
- QPushButton * **qPushButtonDelete**
- QPushButton * **qPushButtonApply**
- QPushButton * **qPushButtonRead**
- QEConfiguredLayout * **qEConfiguredLayoutRecipeFields**

- QDomDocument **document**
- QString **recipeFile**
- QString **filename**
- int **optionsLayout**
- int **currentUserType**

Properties

- QString **recipeDescription**
- bool **showRecipeList**
- bool **showNew**
- bool **showSave**
- bool **showDelete**
- bool **showApply**
- bool **showRead**
- bool **showFields**
- configurationTypesProperty **configurationType**
- QString **configurationFile**
- QString **configurationText**
- optionsLayoutProperty **optionsLayout**
- userTypesProperty **currentUserType**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERecipe/QERecipe.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERecipe/QERecipe.cpp

9.245 QERecordFieldName Class Reference

Static Public Member Functions

- static QString **recordName** (const QString &pvName)
- static QString **fieldName** (const QString &pvName)
- static QString **fieldPvName** (const QString &pvName, const QString &field)
- static QString **rtypePvName** (const QString &pvName)
- static bool **pvNameIsValid** (const QString &pvName)
- static bool **extractPvName** (const QString &item, QString &pvName)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QERecordFieldName.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QERecordFieldName.cpp

9.246 QERecordSpec Class Reference

Public Member Functions

- **QERecordSpec** (const QString recordType)
- QString **getRecordType** () const
- QString **getFieldName** (const int index) const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPro
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPro

9.247 QERecordSpecList Class Reference

Public Member Functions

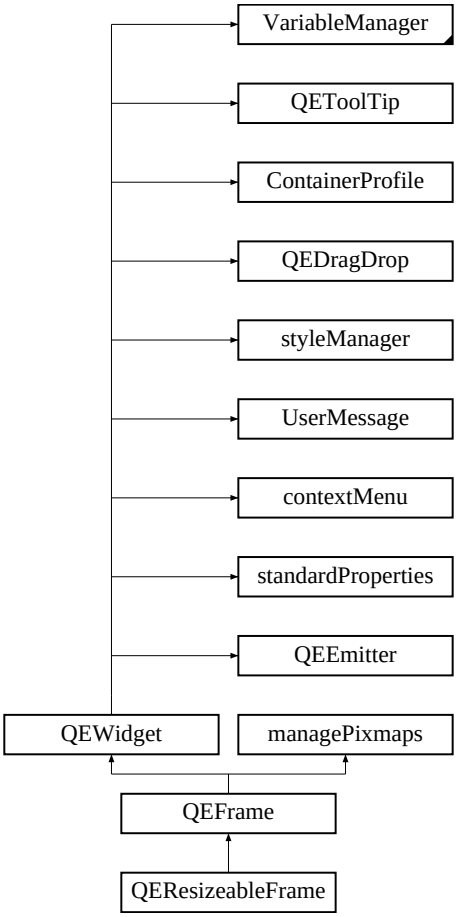
- QERecordSpec * **find** (const QString recordType) const
- void **appendOrReplace** (QERecordSpec *recordSpec)
- bool **processRecordSpecFile** (const QString &filename)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPropertiesUtilit
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPvProperties/QEPvPropertiesUtilit

9.248 QEResizeableFrame Class Reference

`#include <QEResizeableFrame.h>`Inheritance diagram for QEResizeableFrame::



Public Types

- enum **GrabbingEdges** { **TopEdge**, **LeftEdge**, **BottomEdge**, **RightEdge** }

Public Member Functions

- **QEResizeableFrame** (QWidget ∗parent=0)
- QEResizeableFrame (GrabbingEdges grabbingEdge, int minimum, int maximum, QWidget ∗parent=0)

  Construct widget specifying min and max allowed heights.

- QWidget ∗ widget () const

Returns a ref to the resizeable frame's widget, or 0 if there is none.

- void setWidget (QWidget ∗widget)
- QWidget ∗ takeWidget ()

  Removes the resizeable frame's widget, and passes ownership management of the widget to the caller.

- void setGrabberToolTip (const QString &tip)

  Set the tool tip for the internal grabber object.

- void setAllowedMinimum (const int minimum)

  (Re)set allowed limits.

- int **getAllowedMinimum** () const
- void **setAllowedMaximum** (const int maximum)
- int **getAllowedMaximum** () const
- void **setGrabbingEdge** (const GrabbingEdges edge)
- GrabbingEdges **getGrabbingEdge** () const

Protected Member Functions

- bool **eventFilter** (QObject ∗obj, QEvent ∗event)

Properties

- GrabbingEdges grabbingEdge
- int allowedMinimum
- int allowedMaximum

9.248.1 Detailed Description

The QEResizeableFrame provides a frame capable of holding another widget together with a grabber widget that allows the frame to be re-sized, and hence contained widget to be resized. The class currently only supports vertical or horizontal resizing, but not both.

9.248.2 Member Function Documentation

9.248.2.1 void QEResizeableFrame::setWidget (QWidget ∗ *widget*)

Sets the resizeable frame's widget. The widget becomes a child of the resizeable frame, and will be destroyed when the resizeable frame is deleted or when a new widget is set. Any existing widget is deleted - use takeWidget first if needs be.

9.248.3 Property Documentation

9.248.3.1 int QEResizeableFrame::allowedMaximum [read, write]

Set the maximium allowd size (defaults to 100).

9.248.3.2 int QEResizeableFrame::allowedMinimum [read, write]

Set the minimium allowd size (defaults to 10).

9.248.3.3 GrabbingEdges QEResizeableFrame::grabbingEdge [read, write]

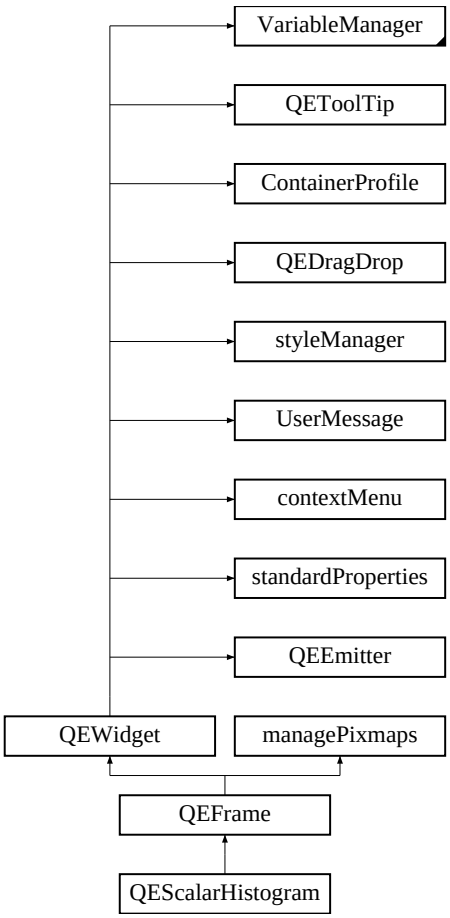
Nominated edge for the grabbing location. Defaults to BottomEdge.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEResizeableFrame/QER
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEResizeableFrame/QER

9.249 QEScalarHistogram Class Reference

Inheritance diagram for QEScalarHistogram::



Public Types

- enum ScaleModes { Manual, Auto, OperationalRange }

Signals

- void **mouseIndexChanged** (const int index)
- void **mouseIndexPressed** (const int index, const Qt::MouseButton button)

Public Member Functions

- **QEScalarHistogram** (QWidget ∗parent=0)

- void **setScaleMode** (const ScaleModes scaleMode)
- ScaleModes **getScaleMode** () const

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant v)
- void **addPvName** (const QString &pvName)

Properties

- QString variableSubstitutions
- bool **autoBarGapWidths**
- int **barWidth**
- int **gap**
- int **margin**
- ScaleModes **scaleMode**
- double **minimum**
- double **maximum**
- double **baseLine**
- bool **drawAxies**
- bool **showScale**
- bool **showGrid**
- bool **logScale**
- QColor **backgroundColour**
- QColor **secondBgColour**
- int **secondBgSize**
- bool **showSecondBg**
- QColor **barColour**
- bool **drawBorder**
- Qt::Orientation **orientation**
- QString variable1

  EPICS variable names (CA PV).

- QString **variable2**

- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- QString **variable7**
- QString **variable8**
- QString **variable9**
- QString **variable10**
- QString **variable11**
- QString **variable12**
- QString **variable13**
- QString **variable14**
- QString **variable15**
- QString **variable16**
- QString **variable17**
- QString **variable18**
- QString **variable19**
- QString **variable20**
- QString **variable21**
- QString **variable22**
- QString **variable23**
- QString **variable24**
- QString **variable25**
- QString **variable26**
- QString **variable27**
- QString **variable28**
- QString **variable29**
- QString **variable30**
- QString **variable31**
- QString **variable32**
- QString **variable33**
- QString **variable34**
- QString **variable35**
- QString **variable36**
- QString **variable37**
- QString **variable38**
- QString **variable39**
- QString **variable40**
- QString **variable41**
- QString **variable42**
- QString **variable43**
- QString **variable44**
- QString **variable45**
- QString **variable46**
- QString **variable47**
- QString **variable48**

- QString **variable49**
- QString **variable50**
- QString **variable51**
- QString **variable52**
- QString **variable53**
- QString **variable54**
- QString **variable55**
- QString **variable56**
- QString **variable57**
- QString **variable58**
- QString **variable59**
- QString **variable60**
- QString **variable61**
- QString **variable62**
- QString **variable63**
- QString **variable64**
- QString **variable65**
- QString **variable66**
- QString **variable67**
- QString **variable68**
- QString **variable69**
- QString **variable70**
- QString **variable71**
- QString **variable72**
- QString **variable73**
- QString **variable74**
- QString **variable75**
- QString **variable76**
- QString **variable77**
- QString **variable78**
- QString **variable79**
- QString **variable80**
- QString **variable81**
- QString **variable82**
- QString **variable83**
- QString **variable84**
- QString **variable85**
- QString **variable86**
- QString **variable87**
- QString **variable88**
- QString **variable89**
- QString **variable90**
- QString **variable91**
- QString **variable92**
- QString **variable93**
- QString **variable94**

- QString **variable95**
- QString **variable96**
- QString **variable97**
- QString **variable98**
- QString **variable99**
- QString **variable100**
- QString **variable101**
- QString **variable102**
- QString **variable103**
- QString **variable104**
- QString **variable105**
- QString **variable106**
- QString **variable107**
- QString **variable108**
- QString **variable109**
- QString **variable110**
- QString **variable111**
- QString **variable112**
- QString **variable113**
- QString **variable114**
- QString **variable115**
- QString **variable116**
- QString **variable117**
- QString **variable118**
- QString **variable119**
- QString **variable120**

9.249.1 Member Enumeration Documentation

9.249.1.1 enum QEScalarHistogram::ScaleModes

Enumerator:

Manual Use property minimum/maximum to scale histogram.

Auto Dynamically scale based on minimum/maximum displayed value.

OperationalRange Use process variable operational range (LOPR/HOPR).

9.249.2 Property Documentation

9.249.2.1 QString QEScalarHistogram::variableSubstitutions `[read, write]`

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEScalarH
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEScalarHistogram.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEScalarH

9.250 QEScaling Class Reference

`#include <QEScaling.h>`

Static Public Member Functions

- static void setScaling (const int m, const int d)
- static void getScaling (int &m, int &d)
- static void setFontScaling (const int fm, const int fd)
- static void getFontScaling (int &fm, int &fd)
- static void applyToWidget (QWidget ∗widget)
- static void rescaleWidget (QWidget ∗widget, const int rm, const int rd)
- static void **rescaleWidget** (QWidget ∗widget, const double newScale)
- static void getWidgetScaling (QWidget ∗widget, int &m, int &d)
- static int scale (const int v)
- static int scaleFont (const int v)

  Font specific scale function.

- static void applyToPoint (QPoint &point)
- static QString scaleStyleSheet (const QString &input)

9.250.1 Detailed Description

This class provide the means to scale a Qt application and/or an individual widget. When scaled, the size related attributes including size, min sizee, max size and font size are scaled (note: different from the standard Qt layout streching). While not perfect, this can be useful and acceptable.

The scaling is defined using a rational number specifed by two integers (m, d). The first (m) integer is the multiplier and the second (d) integer is the divisor. For example, if m = 4 and d = 5, then this specifies an 80%; and if m = 5 and d = 4, this specifies that a 125% scaling is required.

Additonal font Scaling, above and beyond the overall scaling, may also be specified.

Scaling is deemed to be application wide, hence all scaling data (and functions) are static.

9.250.2 Member Function Documentation

9.250.2.1 void QEScaling::applyToPoint (QPoint & *point*) `[static]`

Scales a point.

9.250.2.2 void QEScaling::applyToWidget (QWidget * *widget*) [static]

Adjust the geometry and font scaling of the widget and all child widgets by the defined global application scaling parameters (m, d). Unless m and d different, no scaling occurs. The function tree walks the hiearchy of widgets paranted by the specified widget. This function is idempotent.

9.250.2.3 void QEScaling::getFontScaling (int & *fm*, int & *fd*) [static]

Extract currently applied font scaling - allows widgets to perform widget class specific scaling functionality.

9.250.2.4 void QEScaling::getScaling (int & *m*, int & *d*) [static]

Extract currently applied scaling - allows widgets to perform widget class specific scaling functionality.

9.250.2.5 void QEScaling::getWidgetScaling (QWidget * *widget*, int & *m*, int & *d*) [static]

Get the current widget scaling. This may not be the global application scaling if rescaleWidget has been called, but does include the global scaling.

9.250.2.6 void QEScaling::rescaleWidget (QWidget * *widget*, const int *rm*, const int *rd*) [static]

Applies the specified scaling to the nominated widget and all child widgets. This is in addition to the application wide scaling defined by setScaling. The scaling is limited to 0.1 to 400.0 (10% to 400%)

9.250.2.7 static int QEScaling::scale (const int *v*) [inline, static]

Conveniance functions for widget specific 'scaleBy' functions.

Scales a single value. Note: all other scaling functions take a object by reference and modify that object. Only this function returns a scaled value.

General scale function.

9.250.2.8 QString QEScaling::scaleStyleSheet (const QString & *input*) [static]

Scales styleSheet - looks for "... <number>px ..." or "... <number>pt ..." and scales the number.

9.250.2.9 void QEScaling::setFontScaling (const int *fm*, const int *fd*) [static]

Set application wide additional font scaling values. The final font scaling is this (m $*$ fm) / (d $*$ fd) Both values default to 1, which is a null scaling. Only valid scaling values (fm $>$ 0, fd $>$ 0) are accepted.

9.250.2.10 void QEScaling::setScaling (const int *m*, const int *d*) [static]

Set application wide general gui scaling values. Both values default to 1, which is a null scaling. Only valid scaling values (m $>$ 0, d $>$ 0) are accepted.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEScaling.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEScaling.cpp

9.251 QEScanTimers Class Reference

Public Types

- enum **ScanRates** {
  VerySlow, **Slow**, **Medium**, **Fast**,
  VeryFast }

Signals

- void **flipFlopVerySlow** (const bool)
- void **flipFlopSlow** (const bool)
- void **flipFlopMedium** (const bool)
- void **flipFlopFast** (const bool)
- void **flipFlopVeryFast** (const bool)

Static Public Member Functions

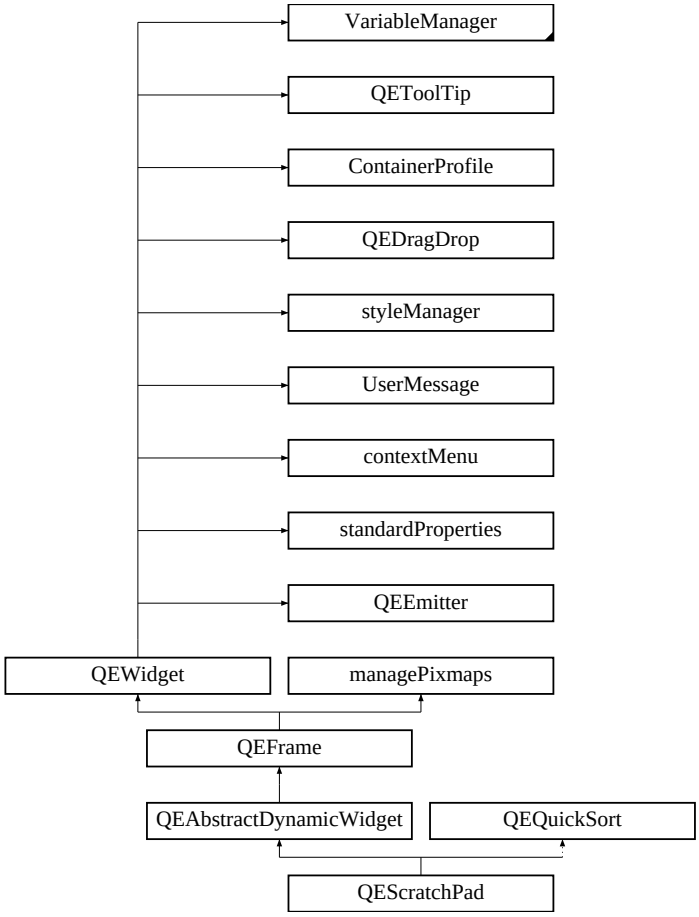
- static bool **attach** (QObject ∗target, const char ∗member, const ScanRates scanRate)
- static void **detach** (QObject ∗target, const char ∗member)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEScanTimers.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEScanTimers.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEScanTimers.cpp

9.252 QEScratchPad Class Reference

`#include <QEScratchPad.h>`Inheritance diagram for QEScratchPad::



Classes

- class **DataSets**

Public Slots

- void **setSelection** (int value)
- void **setPvNameSet** (const QStringList &pvNameSet)

Signals

- void **selectionChanged** (int value)
- void **pvNameSetChanged** (const QStringList &nameSet)

Public Member Functions

- **QEScratchPad** (QWidget *parent=0)
- QSize **sizeHint** () const
- void **setPvName** (const int slot, const QString &pvName)
- QString **getPvName** (const int slot) const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()
- int **getSelection** () const
- QStringList **getPvNameSet** () const

Static Public Attributes

- static const int **NUMBER_OF_ITEMS** = 72

Protected Member Functions

- void **resizeEvent** (QResizeEvent *event)
- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- QMenu * **buildContextMenu** ()
- void **contextMenuTriggered** (int selectedItemNum)
- void **enableEditPvChanged** ()
- bool **itemLessThan** (const int a, const int b, QObject *context=NULL) const
- void **swapItems** (const int a, const int b, QObject *context=NULL)
- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dragMoveEvent** (QDragMoveEvent *event)
- void **dropEvent** (QDropEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- void saveConfiguration (PersistanceManager *pm)
- void restoreConfiguration (PersistanceManager *pm, restorePhases restorePhase)
- void **showEvent** (QShowEvent *event)
- void **keyPressEvent** (QKeyEvent *event)
- bool **eventFilter** (QObject *obj, QEvent *event)
- int **findSlot** (QObject *obj) const

9.252.1 Detailed Description

This class provides a flexible scratch pad form, to which any Process Variable may be added. It displays the PV Name, the Description, i.e. the content of the DESC field together with the value of PV itself.

This class is a direct re-implementation of TScratch_Pad_Form out of the Delphi OPI framework.

9.252.2 Member Function Documentation

9.252.2.1 void QEScratchPad::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.252.2.2 void QEScratchPad::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScratchPad/QEScratchPad.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEScratchPad.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScratchPad/QEScratchPad.cpp

9.253 QEScratchPadMenu Class Reference

Public Types

- enum **ContextMenuOptions** {

  SCRATCHPAD_NONE = QEAbstractDynamicWidget::ADWCM_SUB_-CLASS_WIDGETS_START_HERE, **SCRATCHPAD_SORT_PV_NAMES**, **SCRATCHPAD_CLEAR_ALL**, **SCRATCHPAD_ADD_PV_NAME**,

  SCRATCHPAD_PASTE_PV_NAME, **SCRATCHPAD_EDIT_PV_NAME**, **SCRATCHPAD_DATA_CLEAR** }

Signals

- void **contextMenuSelected** (const int, const QEScratchPad-Menu::ContextMenuOptions)

Public Member Functions

- **QEScratchPadMenu** (const int slot, QWidget ∗parent=0)
- void **setActionChecked** (const ContextMenuOptions option, const bool visible)
- void **setActionEnabled** (const ContextMenuOptions option, const bool visible)
- void **setActionVisible** (const ContextMenuOptions option, const bool visible)
- void **setIsInUse** (const bool isInUse)

Static Public Attributes

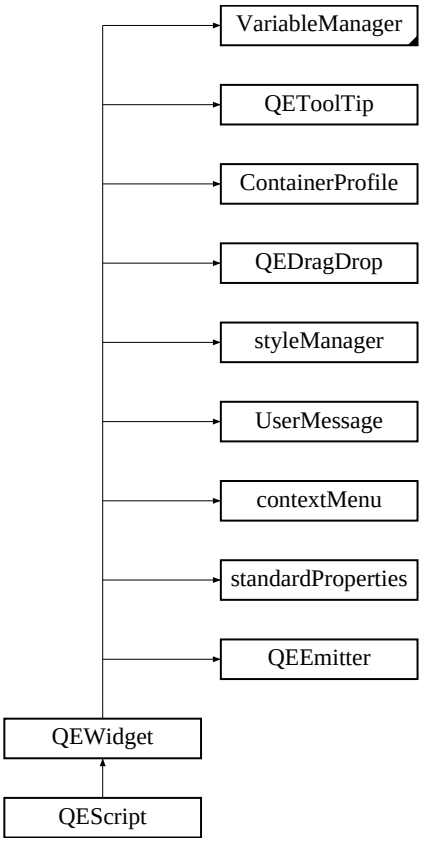
- static const ContextMenuOptions **ContextMenuItemFirst** = SCRATCHPAD_-ADD_PV_NAME
- static const ContextMenuOptions **ContextMenuItemLast** = SCRATCHPAD_-DATA_CLEAR
- static const int **NumberContextMenuItems** = ContextMenuItemLast - ContextMenuItemFirst + 1

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScratchPad/QEScratch
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEScratchPadMenu.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScratchPad/QEScratch

9.254 QEScript Class Reference

`#include <QEScript.h>`Inheritance diagram for QEScript::



Public Types

- enum **scriptTypesProperty** { **File**, **Text** }
- enum **optionsLayoutProperty** { **Top**, **Bottom**, **Left**, **Right** }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Signals

- void **selected** (QString pFilename)

Public Member Functions

- **QEScript** (QWidget ∗pParent=0)
- void **setShowScriptList** (bool pValue)
- bool **getShowScriptList** ()
- void **setShowNew** (bool pValue)
- bool **getShowNew** ()
- void **setShowSave** (bool pValue)
- bool **getShowSave** ()
- void **setShowDelete** (bool pValue)
- bool **getShowDelete** ()
- void **setShowExecute** (bool pValue)
- bool **getShowExecute** ()
- void **setShowAbort** (bool pValue)
- bool **getShowAbort** ()
- void **setEditableTable** (bool pValue)
- bool **getEditableTable** ()
- void **setShowTable** (bool pValue)
- bool **getShowTable** ()
- void **setShowTableControl** (bool pValue)
- bool **getShowTableControl** ()
- void **setShowColumnNumber** (bool pValue)
- bool **getShowColumnNumber** ()
- void **setShowColumnEnable** (bool pValue)
- bool **getShowColumnEnable** ()
- void **setShowColumnProgram** (bool pValue)
- bool **getShowColumnProgram** ()
- void **setShowColumnParameters** (bool pValue)
- bool **getShowColumnParameters** ()
- void **setShowColumnWorkingDirectory** (bool pValue)
- bool **getShowColumnWorkingDirectory** ()
- void **setShowColumnTimeout** (bool pValue)
- bool **getShowColumnTimeout** ()
- void **setShowColumnStop** (bool pValue)
- bool **getShowColumnStop** ()
- void **setShowColumnLog** (bool pValue)
- bool **getShowColumnLog** ()
- void **setScriptType** (int pValue)
- int **getScriptType** ()
- void **setScriptFile** (QString pValue)
- QString **getScriptFile** ()
- void **setScriptText** (QString pValue)

- QString **getScriptText** ()
- void **setScriptDefault** (QString pValue)
- QString **getScriptDefault** ()
- void **setExecuteText** (QString pValue)
- QString **getExecuteText** ()
- void **setOptionsLayout** (int pValue)
- int **getOptionsLayout** ()
- void **insertRow** (bool pEnable, QString pProgram, QString pParameter, QString pWorkingDirectory, int pTimeOut, bool pStop, bool pLog)
- bool **saveScriptList** ()
- void **refreshScriptList** ()
- void **refreshWidgets** ()
- void **setScriptTypeProperty** (scriptTypesProperty pScriptType)
- scriptTypesProperty **getScriptTypeProperty** ()
- void **setOptionsLayoutProperty** (optionsLayoutProperty pOptionsLayout)
- optionsLayoutProperty **getOptionsLayoutProperty** ()
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Protected Attributes

- QComboBox * **qComboBoxScriptList**
- QPushButton * **qPushButtonNew**
- QPushButton * **qPushButtonSave**
- QPushButton * **qPushButtonDelete**

- QPushButton * **qPushButtonExecute**
- QPushButton * **qPushButtonAbort**
- QPushButton * **qPushButtonAdd**
- QPushButton * **qPushButtonRemove**
- QPushButton * **qPushButtonUp**
- QPushButton * **qPushButtonDown**
- QPushButton * **qPushButtonCopy**
- QPushButton * **qPushButtonPaste**
- _QTableWidgetScript * **qTableWidgetScript**
- QString scriptFile

  Define the file where to save the scripts (if not defined then the scripts will be saved in a file named "QEScript.xml").

- QString scriptText

  Define the XML text that contains the scripts.

- QString scriptDefault

  Define the script (previously saved by the user) that will be load as the default script when the widget starts.

- int **scriptType**
- int **optionsLayout**
- QDomDocument **document**
- QString **filename**
- QList< _CopyPaste * > **copyPasteList**
- bool editableTable

  Enable/disable table edition.

- bool **isExecuting**

Properties

- bool showScriptList

  Show/hide combobox that contains the list of existing scripts created by the user.

- bool showNew

  Show/hide button to reset (initialize) the table that contains the sequence of programs to be executed.

- bool showSave

  Show/hide button to save/overwrite a new/existing script.

- bool showDelete

  Show/hide button to delete an existing script.

- bool showExecute

Show/hide button to execute a sequence of programs.

- bool showAbort

  Show/hide button to abort the execution of a sequence of programs.

- bool showTable

  Show/hide table that contains a sequence of programs to be executed.

- bool showTableControl

  Show/hide the controls of the table that contains a sequence of programs to be executed.

- bool showColumnNumber

  Show/hide the column '#' that displays the sequential number of programs.

- bool showColumnEnable

  Show/hide the column 'Enable' that enables the execution of programs.

- bool showColumnProgram

  Show/hide the column 'Program' that contains the external programs to be executed.

- bool showColumnParameters

  Show/hide the column 'Parameters' that contains the parameters that are passed to external programs to be executed.

- bool showColumnWorkingDirectory

  Show/hide the column 'Directory' that defines the working directory to be used when external programs are executed.

- bool showColumnTimeout

  Show/hide the column 'Timeout' that defines a time out period in seconds (if equal to 0 then the program runs until it finishes; otherwise if greater than 0 then the program will only run during this amount of seconds and will be aborted beyond this time).

- bool showColumnStop

  Show/hide the column 'Stop' that enables stopping the execution of subsequent programs when the current one exited with an error code different from 0.

- bool showColumnLog

  Show/hide the column 'Log' that enables the generation of log messages (these messages may be displayed using the QELog widget).

- scriptTypesProperty scriptType

  Select if the scripts are to be loaded/saved from an XML file or from an XML text.

- QString executeText

  Define the caption of the button responsible for starting the execution of external programs (if not defined then the caption will be "Execute").

- optionsLayoutProperty optionsLayout

  Change the order of the widgets. Valid orders are: TOP, BOTTOM, LEFT and RIG.

- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption

9.254.1 Detailed Description

This class is a EPICS aware widget. The QEScript widget allows the user to define a certain sequence of external programs to be executed. This sequence may be saved, modified or loaded for future usage.

9.254.2 Member Enumeration Documentation

9.254.2.1 enum QEScript::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.254.2.2 enum QEScript::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.254.3 Member Function Documentation

9.254.3.1 void QEScript::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.254.4 Property Documentation

9.254.4.1 bool QEScript::allowDrop [read, write]

Allow drag/drops operations to this widget. Default is false. Any dropped text will be used as a new variable name.

Reimplemented from QEDragDrop.

9.254.4.2 QString QEScript::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.254.4.3 bool QEScript::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.254.4.4 DisplayAlarmStateOptions QEScript::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.254.4.5 unsigned QEScript::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

9.254.4.6 QString QEScript::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.254.4.7 UserLevels QEScript::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.254.4.8 QString QEScript::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.254.4.9 QString QEScript::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.254.4.10 QString QEScript::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string

will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.254.4.11 UserLevels QEScript::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or 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'.

9.254.4.12 bool QEScript::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from QEToolTip.

9.254.4.13 bool QEScript::visible [read, write]

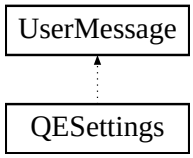
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEScript.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEScript/QEScript.cpp

9.255 QESettings Class Reference

`#include <QESettings.h>`Inheritance diagram for QESettings::



Public Member Functions

- **QESettings** (const QString &fileName)
- bool **isDefined** () const
- QString **getSettingFileName** () const
- QStringList **groupKeys** (const QString &group)
- QVariant **getValue** (const QString &key, const QVariant &defaultValue)
- bool **getBool** (const QString &key, const bool defaultValue)
- QString **getString** (const QString &key, const QString &defaultValue)
- int **getInt** (const QString &key, const int defaultValue)
- double **getFloat** (const QString &key, const double defaultValue)
- QString **getFilename** (const QString &key, const QString &defaultValue)
- QESettings * **getSettings** (const QString &key)

Static Public Member Functions

- static QESettings * **getConfigurationParameters** (const QString &section)

9.255.1 Detailed Description

This class provides access to user defined settings. The setting file is expected to conform to the QSettings::IniFormat, i.e. like a windows style .ini file.

The types of data that my be extracted from an option are bool, QString, int and double; These are accessed by the associated getXxxx () functions where Xxxx is one of Bool, String, Int and Float.

Each of these function takes a default value of the appropriate type which is returned to the caller if the value is not defined or, as in the case of numerical values, is ill-defined.

Numeric validity is defined by QString::toInt () and QString::toDouble () Do note that QString::toInt () does not accept 0x... hexadecimal or 0... octal numbers.

The class also provides a number of addironal conveniance functions that are described below.

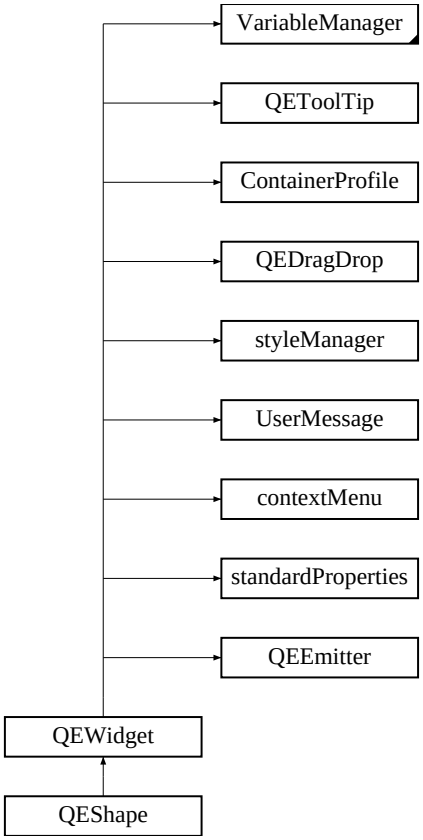
This class is one of several provided to access configuration data such as from environment variables, command line options and settings. They all provide a similar API.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_-parameters/QESettings.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/adaptation_-parameters/QESettings.cpp

9.256 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** }
- enum animationOptions {

  Width, **Height**, **X**, **Y**,

  Transperency, **Rotation**, **ColourHue**, **ColourSaturation**,

  ColourValue, **ColourIndex**, **Penwidth** }
- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }

- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Signals

- void dbValueChanged1 (const qlonglong &out)
- void dbValueChanged2 (const qlonglong &out)
- void dbValueChanged3 (const qlonglong &out)
- void dbValueChanged4 (const qlonglong &out)
- void dbValueChanged5 (const qlonglong &out)
- void dbValueChanged6 (const qlonglong &out)

Public Member Functions

- QEShape (QWidget *parent=0)
- QEShape (const QString &variableName, QWidget *parent=0)
- void scaleBy (const int m, const int d)

  Scale the widgets my m/d.

- void setAnimation (animationOptions animation, const int index)

  Access function for animation' properties - refer to animation' properties for details.

- animationOptions getAnimation (const int index)

  Access function for animation' properties - refer to animation' properties for details.

- void setScale (const double scale, const int index)

  Access function for scale' properties - refer to scale' properties for details.

- double getScale (const int index)

  Access function for scale' properties - refer to scale' properties for details.

- void setOffset (const double offset, const int index)

  Access function for offset' properties - refer to offset' properties for details.

- double getOffset (const int index)

  Access function for offset' properties - refer to offset' properties for details.

- void setBorder (const bool border)

  Access function for border' properties - refer to border' properties for details.

- bool getBorder ()

  Access function for border' properties - refer to border' properties for details.

- void setFill (const bool fill)

  Access function for fill' properties - refer to fill' properties for details.

- bool getFill ()

  Access function for fill' properties - refer to fill' properties for details.

- void setShape (shapeOptions shape)

  Access function for shape' properties - refer to shape' properties for details.

- shapeOptions getShape ()

  Access function for shape' properties - refer to shape' properties for details.

- void setNumPoints (const unsigned int numPoints)

  Access function for number of points' properties - refer to number of points' properties for details.

- unsigned int getNumPoints ()

  Access function for number of points' properties - refer to number of points' properties for details.

- void setOriginTranslation (const QPoint originTranslation)

  Access function for origin translation' properties - refer to origin translation' properties for details.

- QPoint getOriginTranslation ()

  Access function for origin translation' properties - refer to origin translation' properties for details.

- void setPoint (const QPoint point, const int index)

  Access function for point' properties - refer to point' properties for details.

- QPoint getPoint (const int index)

  Access function for point' properties - refer to point' properties for details.

- void setColor (const QColor color, const int index)

  Access function for colour' properties - refer to colour' properties for details.

- QColor getColor (const int index)

  Access function for colour' properties - refer to colour' properties for details.

- void setDrawBorder (const bool drawBorder)

  Access function for draw border' properties - refer to draw border' properties for details.

- bool getDrawBorder ()

  Access function for draw border' properties - refer to draw border' properties for details.

- void setLineWidth (const unsigned int lineWidth)

  Access function for line width' properties - refer to line width' properties for details.

- unsigned int getLineWidth ()

  Access function for line width' properties - refer to line width' properties for details.

- void setStartAngle (const double startAngle)

  Access function for start angle' properties - refer to start angle' properties for details.

- double getStartAngle ()

  Access function for start angle' properties - refer to start angle' properties for details.

- void setRotation (const double rotation)

  Access function for rotation' properties - refer to rotation' properties for details.

- double getRotation ()

  Access function for rotation' properties - refer to rotation' properties for details.

- void setArcLength (const double arcLength)

  Access function for arc length' properties - refer to arc length' properties for details.

- double getArcLength ()

  Access function for arc length' properties - refer to arc length' properties for details.

- void setVariableNameSubstitutionsProperty (QString variableNameSubstitutions)

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- QString getVariableNameSubstitutionsProperty ()

  Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Properties

- QString variable1
- QString variable2
- QString variable3
- QString variable4
- QString variable5
- QString variable6
- QString variableSubstitutions
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- animationOptions animation1
- animationOptions animation2
- animationOptions animation3
- animationOptions animation4
- animationOptions animation5
- animationOptions animation6
- double scale1

  Scale factor applied to data from the 1st variable before it is used to animate the shape.

- double scale2
- double scale3
- double scale4
- double scale5
- double scale6
- double offset1
- double offset2
- double offset3
- double offset4
- double offset5
- double offset6
- QPoint point1
- QPoint point2
- QPoint point3
- QPoint point4
- QPoint point5
- QPoint point6
- QPoint point7
- QPoint point8
- QPoint point9
- QPoint point10
- QColor color1
- QColor color2
- QColor color3
- QColor color4
- QColor color5
- QColor color6
- QColor color7
- QColor color8
- QColor color9
- QColor color10

9.256.1 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 tighly integrated with the base class QEWidget which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.256.2 Member Enumeration Documentation

9.256.2.1 enum QEShape::animationOptions

Options for how a variable will animate the shape.

9.256.2.2 enum QEShape::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.256.2.3 enum QEShape::shapeOptions

Options for the type of shape.

9.256.2.4 enum QEShape::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.256.3 Constructor & Destructor Documentation

9.256.3.1 QEShape::QEShape (QWidget * *parent* = 0)

Create without a variable. Use setVariableNameProperty() and setSubstitutionsProperty() to define a variable and, optionally, macro substitutions later.

9.256.3.2 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.

9.256.4 Member Function Documentation

9.256.4.1 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.

9.256.4.2 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.

9.256.4.3 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.

9.256.4.4 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.

9.256.4.5 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.

9.256.4.6 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.

9.256.4.7 void QEShape::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.256.5 Property Documentation

9.256.5.1 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.

9.256.5.2 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.

9.256.5.3 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.

9.256.5.4 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.

9.256.5.5 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.

9.256.5.6 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.

9.256.5.7 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.

9.256.5.8 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'.

9.256.5.9 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'.

9.256.5.10 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'.

9.256.5.11 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'.

9.256.5.12 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'.

9.256.5.13 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'.

9.256.5.14 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'.

9.256.5.15 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'.

9.256.5.16 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'.

9.256.5.17 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'.

9.256.5.18 QString QEShape::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.256.5.19 bool QEShape::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.256.5.20 DisplayAlarmStateOptions QEShape::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.256.5.21 unsigned QEShape::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

The number of points to use when drawing shapes that are defined by a variable number of points, such as polyline, polygon, path, and series of points.

Sets the width of the pen. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path

9.256.5.22 double QEShape::offset1 [read, write]

Offset applied to data from the 1st variable before it is used to animate the shape

9.256.5.23 double QEShape::offset2 [read, write]

Offset applied to data from the 2nd variable before it is used to animate the shape

9.256.5.24 double QEShape::offset3 [read, write]

Offset applied to data from the 3rd variable before it is used to animate the shape

9.256.5.25 double QEShape::offset4 [read, write]

Offset applied to data from the 4th variable before it is used to animate the shape

9.256.5.26 double QEShape::offset5 [read, write]

Offset applied to data from the 5th variable before it is used to animate the shape

9.256.5.27 double QEShape::offset6 [read, write]

Offset applied to data from the 6th variable before it is used to animate the shape

9.256.5.28 QPoint QEShape::point1 [read, write]

1st coordinate used when drawing the shape. Used for the following shapes: Line, Points, Polyline, Polygon, Rect, RoundedRect, Ellipse, Arc, Chord, Pie, Path, Text, Pixmap

9.256.5.29 QPoint QEShape::point10 [read, write]

10th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.30 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

9.256.5.31 QPoint QEShape::point3 [read, write]

3rd coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.32 QPoint QEShape::point4 [read, write]

4th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.33 QPoint QEShape::point5 [read, write]

5th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.34 QPoint QEShape::point6 [read, write]

6th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.35 QPoint QEShape::point7 [read, write]

7th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.36 QPoint QEShape::point8 [read, write]

8th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.37 QPoint QEShape::point9 [read, write]

9th coordinate used when drawing the shape. Used for the following shapes: Points, Polyline, Polygon, Path

9.256.5.38 double QEShape::scale2 [read, write]

Scale factor applied to data from the 2nd variable before it is used to animate the shape

9.256.5.39 double QEShape::scale3 [read, write]

Scale factor applied to data from the 3rd variable before it is used to animate the shape

9.256.5.40 double QEShape::scale4 [read, write]

Scale factor applied to data from the 4th variable before it is used to animate the shape

9.256.5.41 double QEShape::scale5 [read, write]

Scale factor applied to data from the 5th variable before it is used to animate the shape

9.256.5.42 double QEShape::scale6 [read, write]

Scale factor applied to data from the 6th variable before it is used to animate the shape

9.256.5.43 QString QEShape::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.256.5.44 UserLevels QEShape::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.256.5.45 QString QEShape::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.256.5.46 QString QEShape::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.256.5.47 QString QEShape::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.256.5.48 UserLevels QEShape::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or 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'.

9.256.5.49 QString QEShape::variable1 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale1 and offset1 then the attribute selected for animation is selected by the property animation1.

9.256.5.50 QString QEShape::variable2 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale2 and offset2 then the attribute selected for animation is selected by the property animation2.

9.256.5.51 QString QEShape::variable3 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale3 and offset3 then the attribute selected for animation is selected by the property animation3.

9.256.5.52 QString QEShape::variable4 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale4 and offset4 then the attribute selected for animation is selected by the property animation4.

9.256.5.53 QString QEShape::variable5 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale5 and offset5 then the attribute selected for animation is selected by the property animation5.

9.256.5.54 QString QEShape::variable6 [read, write]

EPICS variable name (CA PV). This variable is read and used to animate an attribute of the shape. The value read is first scaled and offset by properties scale6 and offset6 then the attribute selected for animation is selected by the property animation6.

9.256.5.55 bool QEShape::variableAsToolTip [read, write]

Use the variable as the tool tip. Default is true. Tool tip property will be overwritten by the variable name.

Reimplemented from QEToolTip.

9.256.5.56 QString QEShape::variableSubstitutions [read, write]

Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

9.256.5.57 bool QEShape::visible [read, write]

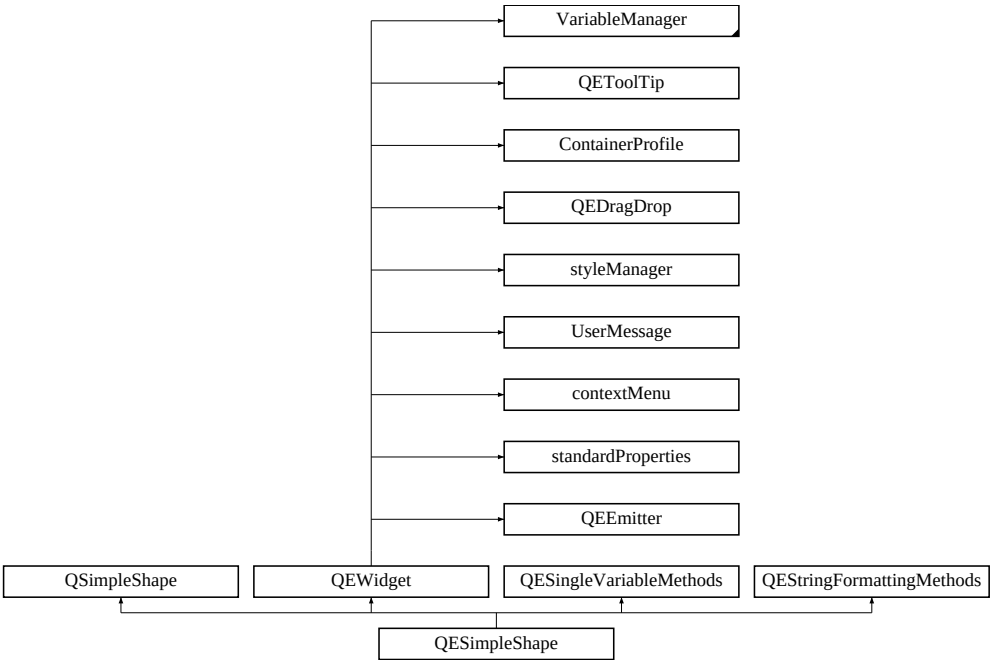
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEShape/QEShape.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEShape.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEShape/QEShape.cpp

9.257 QESimpleShape Class Reference

`#include <QESimpleShape.h>`Inheritance diagram for QESimpleShape::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)

- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- **QESimpleShape** (QWidget *parent=0)
- **QESimpleShape** (const QString &variableName, QWidget *parent=0)
- void setVariableNameSubstitutionsProperty (const QString &substitutions)
- void **setEdgeVariableNameProperty** (const QString &variableName)
- QString **getEdgeVariableNameProperty** () const
- void **setEdgeElementsRequired** (const int elementsRequired)
- int **getEdgeElementsRequired** () const
- void **setEdgeArrayIndex** (const int arrayIndex)
- int **getEdgeArrayIndex** () const
- DisplayAlarmStateOptions getEdgeAlarmStateOptionProperty () const

  Access function for edgeAlarmStateOption property - refer to edgeAlarmStateOption property for details.

- void setEdgeAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for edgeAlarmStateOption property - refer to edgeAlarmStateOption property for details.

Protected Member Functions

- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant v)
- QString **getItemText** ()
- QColor getItemColour ()

  returns "" unless overriden.

- void **stringFormattingChange** ()

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- QString **edgeVariable**
- int **edgeElementsRequired**
- int **edgeArrayIndex**
- DisplayAlarmStateOptions **edgeAlarmStateOption**
- bool addUnits
- QString localEnumeration
- int value

  Shape value - range 0 to 15 - default is zero.

- bool isActive

  When inActive, the widgers is grayed-out, as if disabled. The default is true.

- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet

- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption

9.257.1 Detailed Description

This class is an EPICS aware simple shape.

The shape is selected at design time using the shape property - refer to Shapes for details, or may be set by the setShape function.

The colour is based on either the value of or the alarm state of a single PV.

When the (standard) displayAlarmState property is false, the PV value (modulo 16) is used to select one of the colours defined by the colour0 to colour15 properties.

When the displayAlarmState property is true, the colour is based on the alarm state, i.e. green when no alarm, yellow for minor alarm etc.

This class extends QSimpleShape in order to make it EPICS aware.

9.257.2 Member Enumeration Documentation

9.257.2.1 enum QESimpleShape::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.257.2.2 enum QESimpleShape::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.

9.257.3 Member Function Documentation

9.257.3.1 void QESimpleShape::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.257.3.2 void QESimpleShape::dbValueChanged () [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.

9.257.3.3 void QESimpleShape::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.257.3.4 void QESimpleShape::setVariableNameSubstitutionsProperty (const QString & *substitutions*)

Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

Reimplemented from QESingleVariableMethods.

9.257.4 Property Documentation

9.257.4.1 bool QESimpleShape::addUnits [read, write]

If true (default), add engineering units supplied with the data.

9.257.4.2 bool QESimpleShape::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.

9.257.4.3 int QESimpleShape::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.257.4.4 QString QESimpleShape::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.257.4.5 bool QESimpleShape::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.257.4.6 DisplayAlarmStateOptions QESimpleShape::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.257.4.7 int QESimpleShape::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.257.4.8 unsigned QESimpleShape::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.

9.257.4.9 QString QESimpleShape::localEnumeration [read, write]

An enumeration list used to data values. Used only when the formatting option is 'local enumeration'. Value is converted to an integer and used to select a string from this list.

Format is:

[[<|<=|=|!=|>=|>]value1|*] : string1 , [[<|<=|=|!=|>=|>]value2|*] : string2 , [[<|<=|=|!=|>=|>]value3|*] : string3 , ...

Where: < Less than <= Less than or equal = Equal (default if no operator specified) >= Greather than or equal > Greater than Always match (used to specify default text)

Values may be numeric or textual Values do not have to be in any order, but first match wins Values may be quoted Strings may be quoted Consecutive values do not have to be present. Operator is assumed to be equality if not present. White space is ignored except within quoted strings.

may be included in a string to indicate a line break

Examples are:

0:Off,1:On 0 : "Pump Running", 1 : "Pump not running" 0:"", 1:"Warning!\nAlarm" <2:"Value is less than two", =2:"Value is equal to two", >2:"Value is grater than 2" 3:"Beamline Available", *:"" "Pump Off":"OH NO!, the pump is OFF!","Pump On":"It's OK, the pump is on"

The data value is converted to a string if no enumeration for that value is available. For example, if the local enumeration is '0:off,1:on', and a value of 10 is processed, the text generated is '10'. If a blank string is required, this should be explicit. for example, '0:off,1:on,10:""'

A range of numbers can be covered by a pair of values as in the following example: >=4:"Between 4 and 8",<=8:"Between 4 and 8"

9.257.4.10 QString QESimpleShape::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.257.4.11 UserLevels QESimpleShape::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 accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.257.4.12 QString QESimpleShape::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.

9.257.4.13 QString QESimpleShape::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.

9.257.4.14 QString QESimpleShape::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.

9.257.4.15 UserLevels QESimpleShape::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'.

9.257.4.16 QString QESimpleShape::variable [read, write]

EPICS variable name (CA PV)

9.257.4.17 bool QESimpleShape::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.

9.257.4.18 QString QESimpleShape::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.

9.257.4.19 bool QESimpleShape::visible [read, write]

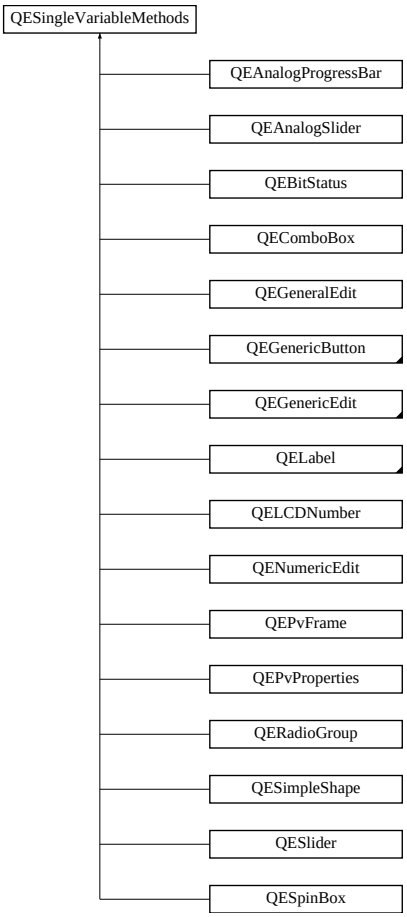
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESimpleShape/QESimp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QESimpleShape.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESimpleShape/QESimp

9.258 QESingleVariableMethods Class Reference

Inheritance diagram for QESingleVariableMethods::



Public Member Functions

- **QESingleVariableMethods** (QEWidget *owner, unsigned int variableIndex=0)

- void setVariableNameProperty (const QString &variableName)
- QString getVariableNameProperty () const
- void setVariableNameSubstitutionsProperty (const QString &substitutions)
- QString getVariableNameSubstitutionsProperty () const
- void setElementsRequired (const int elementsRequired)
- int getElementsRequired () const
- void setArrayIndex (const int arrayIndex)
- int getArrayIndex () const
- void connectNewVariableNameProperty (const char *useNameSlot)

- void **setSingleVariableQCaProperties** (qcaobject::QCaObject ∗qca)
- Q_DECL_DEPRECATED void **setQCaArrayIndex** (qcaobject::QCaObject ∗qca)

9.258.1 Member Function Documentation

9.258.1.1 void QESingleVariableMethods::connectNewVariableNameProperty (const char ∗ *useNameSlot*)

Connects internal variable name property manager's newVariableNameProperty signal to the specified slot.

Example: this->connectNewVariableNameProperty (SLOT (useNewVariableNameProperty (QString, QString, unsigned int));

9.258.1.2 int QESingleVariableMethods::getArrayIndex () const

Property access function for arrayIndex property. Array element to access if variable is an array variable. Defaults to 0, i.e. first element.

9.258.1.3 int QESingleVariableMethods::getElementsRequired () const

Property access function to get elementsRequired property. Limits the the numer of elements requested to the given value.

9.258.1.4 QString QESingleVariableMethods::getVariableNameProperty () const

Property access function for variable property. This has special behaviour to work well within designer.

9.258.1.5 QString QESingleVariableMethods::getVariableNameSubstitutionsProperty () const

Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

9.258.1.6 void QESingleVariableMethods::setArrayIndex (const int *arrayIndex*)

Property access function for arrayIndex property. Array element to access if variable is an array variable. Defaults to 0, i.e. first element. arrayIndex value is restricted to be >= 0

If the assocated qcaobject::QCaObject exists then calls its setArrayIndex function and then requests that the object resend last data. The function adjusts the elementsRequired property value if necessary.

Reimplemented in QEGeneralEdit.

9.258.1.7 void QESingleVariableMethods::setElementsRequired (const int *elementsRequired*)

Property access function to set elementsRequired property. Limits the the numer of elements requested to the given value. Defaults to 0, which means no limit is applied to the subscription. The function adjusts the arrayIndex property value if necessary.

9.258.1.8 void QESingleVariableMethods::setVariableNameProperty (const QString & *variableName*)

Property access function for variable property. This has special behaviour to work well within designer.

9.258.1.9 void QESingleVariableMethods::setVariableNameSubstitutionsProperty (const QString & *substitutions*)

Property access function for variableSubstitutions property. This has special behaviour to work well within designer.

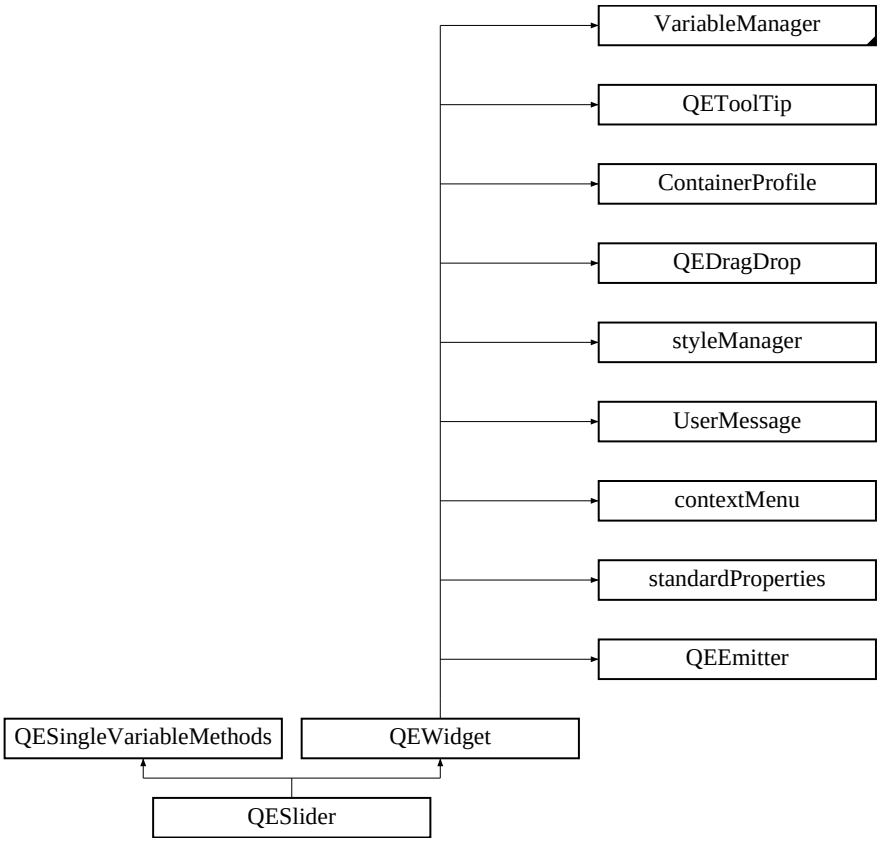
Reimplemented in QEAnalogSlider, QEPushButton, QERadioGroup, and QESimpleShape.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QESingleVariableMethods
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QESingleVariableMethods

9.259 QESlider Class Reference

Inheritance diagram for QESlider::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)

Public Member Functions

- **QESlider** (QWidget *parent=0)
- **QESlider** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChange)
- bool **getWriteOnChange** () const
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** () const
- void **setScale** (double scaleIn)
- double **getScale** () const
- void **setOffset** (double offsetIn)
- double **getOffset** () const
- void **setAllowFocusUpdate** (bool allowFocusUpdate)
- bool **getAllowFocusUpdate** () const
- void writeNow ()
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Protected Member Functions

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool writeOnChange

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool subscribe
- bool allowFocusUpdate
- bool variableAsToolTip
- bool allowDrop
- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled

- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- double **value**
- int **sliderPosition**

9.259.1 Member Enumeration Documentation

9.259.1.1 enum QESlider::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.259.1.2 enum QESlider::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.259.2 Member Function Documentation

9.259.2.1 void QESlider::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.259.2.2 void QESlider::dbValueChanged () [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.

9.259.2.3 void QESlider::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.259.2.4 void QESlider::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.259.3 Member Data Documentation

9.259.3.1 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).

9.259.4 Property Documentation

9.259.4.1 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.

9.259.4.2 bool QESlider::allowFocusUpdate [read, write]

Allow updated while widget has focus - defaults to false

9.259.4.3 int QESlider::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.259.4.4 QString QESlider::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.259.4.5 bool QESlider::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.259.4.6 DisplayAlarmStateOptions QESlider::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.259.4.7 int QESlider::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.259.4.8 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.

9.259.4.9 QString QESlider::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.259.4.10 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 VariableManager.

9.259.4.11 UserLevels QESlider::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.259.4.12 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.

9.259.4.13 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.

9.259.4.14 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.

9.259.4.15 UserLevels QESlider::userLevelVisibility [read, write]

Lowest user level at which the widget is visible. Default is 'User'. Used when designing GUIs that display more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always visible should be visible at 'User'. Widgets that are only used by scientists managing the facility should be visible at 'Scientist'. Widgets that are only used by engineers maintaining the facility should be visible at 'Engineer'.

9.259.4.16 QString QESlider::variable [read, write]

EPICS variable name (CA PV)

9.259.4.17 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.

9.259.4.18 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 are also used for other purposes.

9.259.4.19 bool QESlider::visible [read, write]

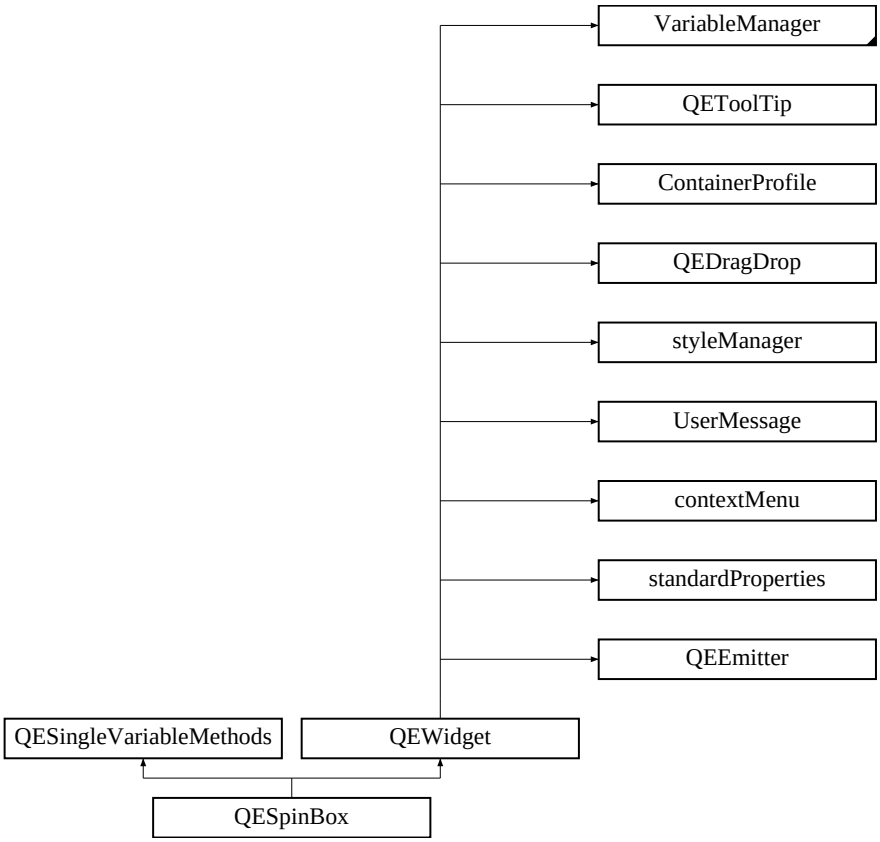
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESlider/QESlider.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QESlider.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESlider/QESlider.cpp

9.260 QESpinBox Class Reference

Inheritance diagram for QESpinBox::



Public Types

- enum UserLevels { User = userLevelTypes::USERLEVEL_USER, Scientist = userLevelTypes::USERLEVEL_SCIENTIST, Engineer = userLevelTypes::USERLEVEL_ENGINEER }
- enum DisplayAlarmStateOptions { Never = standardProperties::DISPLAY_-ALARM_STATE_NEVER, Always = standardProperties::DISPLAY_-ALARM_STATE_ALWAYS, WhenInAlarm = standardProperties::DISPLAY_-ALARM_STATE_WHEN_IN_ALARM }

Public Slots

- void setDefaultStyle (const QString &style)

  Update the default style applied to this widget.

- void setManagedVisible (bool v)

Signals

- void dbValueChanged ()
- void **dbValueChanged** (const QString &out)
- void **dbValueChanged** (const int &out)
- void **dbValueChanged** (const long &out)
- void **dbValueChanged** (const qlonglong &out)
- void **dbValueChanged** (const double &out)
- void **dbValueChanged** (const bool &out)
- void dbConnectionChanged (const bool &isConnected)
- void userChange (const QString &oldValue, const QString &newValue, const QString &lastValue)

  Internal use only. Used by QEConfiguredLayout to be notified when one of its widgets has written something.

Public Member Functions

- **QESpinBox** (QWidget *parent=0)
- **QESpinBox** (const QString &variableName, QWidget *parent=0)
- void **setWriteOnChange** (bool writeOnChangeIn)
- bool **getWriteOnChange** () const
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** () const
- void **setAddUnitsAsSuffix** (bool addUnitsAsSuffixIn)
- bool **getAddUnitsAsSuffix** () const
- void **setUseDbPrecisionForDecimals** (bool useDbPrecisionForDecimalIn)
- bool **getUseDbPrecisionForDecimals** () const
- void **setAllowFocusUpdate** (bool allowFocusUpdate)
- bool **getAllowFocusUpdate** () const
- void writeNow ()
- UserLevels getUserLevelVisibilityProperty ()

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- void setUserLevelVisibilityProperty (UserLevels level)

  Access function for userLevelVisibility property - refer to userLevelVisibility property for details.

- UserLevels getUserLevelEnabledProperty ()

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- void setUserLevelEnabledProperty (UserLevels level)

  Access function for userLevelEnabled property - refer to userLevelEnabled property for details.

- DisplayAlarmStateOptions getDisplayAlarmStateOptionProperty ()

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

- void setDisplayAlarmStateOptionProperty (DisplayAlarmStateOptions option)

  Access function for displayAlarmStateOption property - refer to displayAlarmStateOption property for details.

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant s)
- QMenu * getDefaultContextMenu ()

  Flag indicating this widget is running inside Qt's 'designer'.

Protected Attributes

- QEFloatingFormatting **floatingFormatting**
- bool **writeOnChange**
- bool **addUnitsAsSuffix**
- bool **useDbPrecisionForDecimal**

Properties

- QString variable
- QString variableSubstitutions
- int elementsRequired
- int arrayIndex
- bool variableAsToolTip
- bool allowDrop

- bool visible
- unsigned int
- QString styleSheet
- QString defaultStyle
- QString userLevelUserStyle
- QString userLevelScientistStyle
- QString userLevelEngineerStyle
- UserLevels userLevelVisibility
- UserLevels userLevelEnabled
- bool displayAlarmState
- DisplayAlarmStateOptions displayAlarmStateOption
- bool subscribe
- bool allowFocusUpdate
- bool **useDbPrecision**
- bool **addUnits**
- double **value**

9.260.1 Member Enumeration Documentation

9.260.1.1 enum QESpinBox::DisplayAlarmStateOptions

User friendly enumerations for displayAlarmStateOption property - refer to displayAlarmStateOption property and displayAlarmStateOptions enumeration for details.

Enumerator:

Never Refer to DISPLAY_ALARM_STATE_NEVER for details.

Always Refer to DISPLAY_ALARM_STATE_ALWAYS for details.

WhenInAlarm Refer to DISPLAY_ALARM_STATE_WHEN_IN_ALARM for details.

9.260.1.2 enum QESpinBox::UserLevels

User friendly enumerations for userLevelVisibility and userLevelEnabled properties - refer to userLevelVisibility and userLevelEnabled properties and userLevel enumeration for details.

Enumerator:

User Refer to USERLEVEL_USER for details.

Scientist Refer to USERLEVEL_SCIENTIST for details.

Engineer Refer to USERLEVEL_ENGINEER for details.

9.260.2 Member Function Documentation

9.260.2.1 void QESpinBox::dbConnectionChanged (const bool & *isConnected*) [signal]

Sent when the widget state updated following a channel connection change Applied to provary varible.

9.260.2.2 void QESpinBox::dbValueChanged () [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.

9.260.2.3 void QESpinBox::setManagedVisible (bool *v*) [inline, slot]

Slot to set the visibility of a QE widget, taking into account the user level. Widget will be hidden if hidden by a call this slot, by will only be made visible by a calll to this slot if the user level allows.

9.260.2.4 void QESpinBox::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

Reimplemented from VariableManager.

9.260.3 Property Documentation

9.260.3.1 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.

9.260.3.2 bool QESpinBox::allowFocusUpdate [read, write]

Allow updated while widget has focus - defaults to false

9.260.3.3 int QESpinBox::arrayIndex [read, write]

Index used to select a single item of data for processing. The default is 0.

Reimplemented from QESingleVariableMethods.

9.260.3.4 QString QESpinBox::defaultStyle [read, write]

Style Sheet string to be applied before, i.e. lower priority than, any other style, e.g. alarm style and/or user level style. Default is an empty string.

9.260.3.5 bool QESpinBox::displayAlarmState [read, write]

DEPRECATED. USE displayAlarmStateOption INSTEAD. If set (default) widget will indicate the alarm state of any variable data it is displaying. If clear widget will never indicate the alarm state of any variable data it is displaying. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

Reimplemented from standardProperties.

9.260.3.6 DisplayAlarmStateOptions QESpinBox::displayAlarmStateOption [read, write]

If 'Always' (default) widget will indicate the alarm state of any variable data it is displaying, including 'No Alarm' If 'Never' widget will never indicate the alarm state of any variable data it is displaying. If 'WhenInAlarm' widget only indicate the alarm state of any variable data it is displaying if it is 'in alarm'. Typically the background colour is set to indicate the alarm state. Note, this property is included in the set of standard properties as it applies to most widgets. It will do nothing for widgets that don't display data.

9.260.3.7 int QESpinBox::elementsRequired [read, write]

The number of elements required to be subscribed for from the PV host (IOC). The default is 0 which means subscribed for all elements. Note: changing this value causes the unsubscribe/re-subscribe just as if the variable name changed.

Reimplemented from QESingleVariableMethods.

9.260.3.8 unsigned QESpinBox::int [read, write]

Set the ID used by the message filtering system. Default is zero. Widgets or applications that use messages from the framework have the option of filtering on this ID. For example, by using a unique message source ID a QELog widget may be set up to only log messages from a select set of widgets.

9.260.3.9 QString QESpinBox::styleSheet [read, write]

Hide style sheet from designer as style calculation by the styleManager and not directly setable per se. This also stops transient styles being saved to the ui file.

9.260.3.10 bool QESpinBox::subscribe [read, write]

Sets if this widget subscribes for data updates and displays current data. Default is 'true' (subscribes for and displays data updates)

Reimplemented from VariableManager.

9.260.3.11 UserLevels QESpinBox::userLevelEnabled [read, write]

Lowest user level at which the widget is enabled. Default is 'User'. Used when designing GUIs that allow access to more and more detail according to the user mode. The user mode is set application wide through the QELogin widget, or programatically through setUserLevel() Widgets that are always accessable should be visible at 'User'. Widgets that are only accessable to scientists managing the facility should be visible at 'Scientist'. Widgets that are only accessable to engineers maintaining the facility should be visible at 'Engineer'.

9.260.3.12 QString QESpinBox::userLevelEngineerStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.260.3.13 QString QESpinBox::userLevelScientistStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.260.3.14 QString QESpinBox::userLevelUserStyle [read, write]

Style Sheet string to be applied when the widget is displayed in 'User' mode. Default is an empty string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red' This Style Sheet string will be applied by the styleManager class. Refer to the styleManager class for details about how this Style Sheet string will be merged with any pre-existing Style Sheet string and any Style Sheet strings generated during the display of data.

9.260.3.15 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'.

9.260.3.16 QString QESpinBox::variable [read, write]

EPICS variable name (CA PV)

9.260.3.17 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.

9.260.3.18 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.

9.260.3.19 bool QESpinBox::visible [read, write]

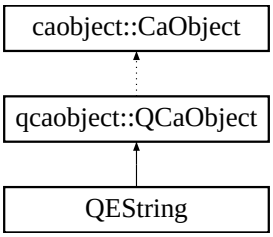
Display the widget. Default is true. Setting this property false is usefull if widget is only used to provide a signal - for example, when supplying data to a QELink widget. Note, when false the widget will still be visible in Qt Designer.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESpinBox/QESpinBox.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QESpinBox.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESpinBox/QESpinBox.cpp

9.261 QEString Class Reference

Inheritance diagram for QEString::



Public Slots

- void **writeString** (const QString &data)
- void **writeStringElement** (const QString &data)
- void **writeString** (const QVector< QString > &data)

Signals

- void **stringConnectionChanged** (QCaConnectionInfo &connectionInfo, const unsigned int &variableIndex)
- void **stringChanged** (const QString &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **stringArrayChanged** (const QVector< QString > &values, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- **QEString** (QString recordName, QObject *eventObject, QEStringFormatting *stringFormattingIn, unsigned int variableIndexIn)
- **QEString** (QString recordName, QObject *eventObject, QEStringFormatting *stringFormattingIn, unsigned int variableIndexIn, UserMessage *userMessageIn)
- bool **writeString** (const QString &data, QString &message)
- bool **writeStringElement** (const QString &data, QString &message)
- bool **writeString** (const QVector< QString > &data, QString &message)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEString.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEString.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEString.cpp

9.262 QEStringFormatting 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 separators { SEPARATOR_NONE = 0, SEPARATOR_COMMA, SEPARATOR_UNDERSCORE, SEPARATOR_SPACE }
- enum arrayActions { APPEND, ASCII, INDEX }

Public Member Functions

- QString **formatString** (const QVariant &value, int arrayIndex=0) const
- QVariant **formatValue** (const QString &text, bool &ok)
- QVariant **formatValue** (const QVector< QString > &text, bool &ok)
- void **setDbEgu** (QString egu)
- void **setDbEnumerations** (QStringList enumerations)
- void **setDbPrecision** (unsigned int dbPrecisionIn)
- void **setPrecision** (int precision)
- void **setUseDbPrecision** (bool useDbPrecision)
- void **setLeadingZero** (bool leadingZero)
- void **setTrailingZeros** (bool trailingZeros)
- void **setForceSign** (bool forceSign)
- void **setFormat** (formats format)
- void **setSeparator** (const separators separator)
- void **setRadix** (const int radix)
- void **setNotation** (notations notation)
- void **setArrayAction** (arrayActions arrayActionIn)
- void **setAddUnits** (bool addUnits)
- void **setLocalEnumeration** (QString localEnumerationIn)
- int **getPrecision** () const
- bool **getUseDbPrecision** () const
- bool **getLeadingZero** () const
- bool **getTrailingZeros** () const
- bool **getForceSign** () const
- formats **getFormat** () const
- separators **getSeparator** () const
- unsigned int **getRadix** () const
- notations **getNotation** () const
- arrayActions **getArrayAction** () const
- bool **getAddUnits** () const
- QString **getLocalEnumeration** () const
- QELocalEnumeration **getLocalEnumerationObject** () const

9.262.1 Member Enumeration Documentation

9.262.1.1 enum QEStringFormatting::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.

9.262.1.2 enum QEStringFormatting::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.

9.262.1.3 enum QEStringFormatting::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.

9.262.1.4 enum QEStringFormatting::separators

separators Defines the digit 'thousands' separator to be used.

Enumerator:

SEPARATOR_NONE Use no separator, e.g. 123456.123456789.

SEPARATOR_COMMA Use ',' as separator, e.g. 123,456.123,456,789.

SEPARATOR_UNDERSCORE Use '_' as separator, e.g. 123_456.123_456_-789.

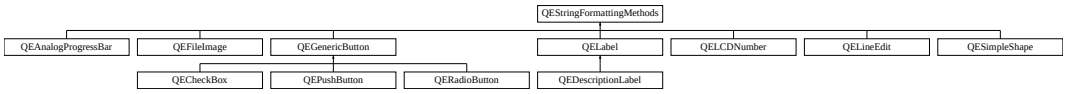
SEPARATOR_SPACE Use ' ' as separator, e.g. 123 456.123 456 789.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEStringFormatting.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QEStringFormatting.cpp

9.263 QEStringFormattingMethods Class Reference

Inheritance diagram for QEStringFormattingMethods::



Public Member Functions

- virtual void **stringFormattingChange** ()=0
- void **setPrecision** (int precision)
- int **getPrecision** () const
- void **setUseDbPrecision** (bool useDbPrecision)
- bool **getUseDbPrecision** () const
- void **setLeadingZero** (bool leadingZero)
- bool **getLeadingZero** () const
- void **setTrailingZeros** (bool trailingZeros)
- bool **getTrailingZeros** () const
- void **setForceSign** (bool forceSign)
- bool **getForceSign** () const
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** () const
- void **setLocalEnumeration** (QString localEnumeration)
- QString **getLocalEnumeration** () const
- void **setFormat** (QEStringFormatting::formats format)
- QEStringFormatting::formats **getFormat** () const
- void **setSeparator** (const QEStringFormatting::separators separator)
- QEStringFormatting::separators **getSeparator** () const
- void **setRadix** (const int radix)
- unsigned int **getRadix** () const
- void **setNotation** (QEStringFormatting::notations notation)
- QEStringFormatting::notations **getNotation** () const
- void **setArrayAction** (QEStringFormatting::arrayActions arrayAction)
- QEStringFormatting::arrayActions **getArrayAction** () const
- QELocalEnumeration **getLocalEnumerationObject** () const

Protected Attributes

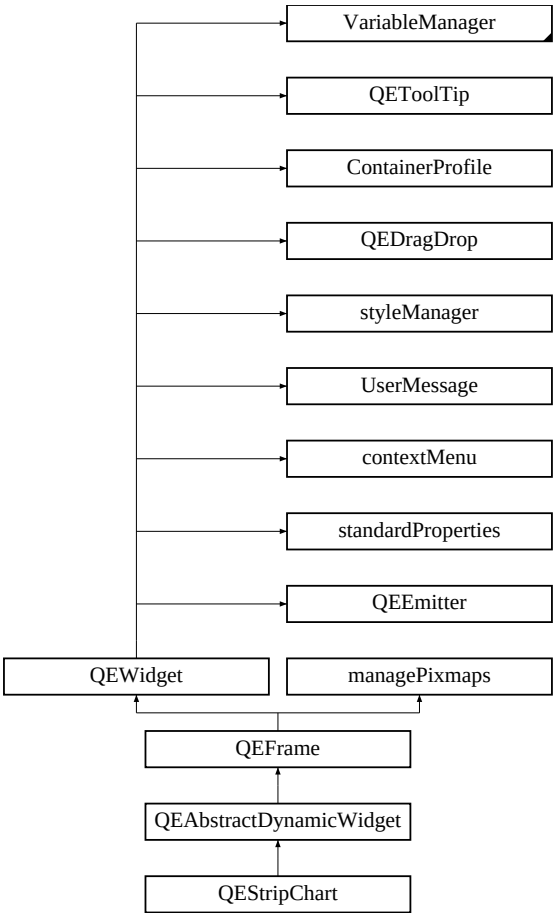
- QEStringFormatting **stringFormatting**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEStringForm
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEStringForm

9.264 QEStripChart Class Reference

Inheritance diagram for QEStripChart::



Public Types

- enum **PropertyChartYRanges** { **manual** = QEStripChartNames::manual, **dynamic** = QEStripChartNames::dynamic }
- enum **Constants** { **NUMBER_OF_PVS** = 16 }

Public Slots

- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)

Public Member Functions

- **QEStripChart** (QWidget *parent=0)
- QSize **sizeHint** () const
- QDateTime **getStartDateTime** () const
- QDateTime **getEndDateTime** () const
- void **setEndDateTime** (QDateTime endDateTimeIn)
- int **getDuration** () const
- void **setDuration** (int durationIn)
- double **getYMinimum** () const
- void **setYMinimum** (const double yMinimumIn)
- double **getYMaximum** () const
- void **setYMaximum** (const double yMaximumIn)
- void **setYRange** (const double yMinimumIn, const double yMaximumIn)
- void **setPvName** (const int slot, const QString &pvName)
- QString **getPvName** (const int slot) const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()
- PropertyChartYRanges **getYRangeMode** () const
- void **setYRangeMode** (const PropertyChartYRanges scale)
- QEStripChartNames::VideoModes **getVideoMode** () const
- QEStripChartNames::YScaleModes **getYScaleMode** () const
- void **updateItemMenu** (const int slot, QAction *action, const bool inUseMenu)

Protected Member Functions

- bool **eventFilter** (QObject *obj, QEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void saveConfiguration (PersistanceManager *pm)
- void restoreConfiguration (PersistanceManager *pm, restorePhases restorePhase)
- void **addToPredefinedList** (const QString &pvName)
- QStringList **getPredefinedPVNameList** () const
- QString **getPredefinedItem** (int i) const
- void **setReplotIsRequired** ()
- void **evaluateAllowDrop** ()

Properties

- int **duration**
- double **yMinimum**
- double **yMaximum**
- QEStripChartNames::VideoModes **videoMode**
- PropertyChartYRanges **chartRange**
- QEStripChartNames::YScaleModes **scaleMode**
- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- QString **variable7**
- QString **variable8**
- QString **variable9**
- QString **variable10**
- QString **variable11**
- QString **variable12**
- QString **variable13**
- QString **variable14**
- QString **variable15**
- QString **variable16**
- QString variableSubstitutions
- QColor **colour1**
- QColor **colour2**
- QColor **colour3**
- QColor **colour4**
- QColor **colour5**
- QColor **colour6**
- QColor **colour7**
- QColor **colour8**
- QColor **colour9**
- QColor **colour10**
- QColor **colour11**
- QColor **colour12**
- QColor **colour13**
- QColor **colour14**
- QColor **colour15**

Friends

- class QEStripChartItem

9.264.1 Member Function Documentation

9.264.1.1 void QEStripChart::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.264.1.2 void QEStripChart::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

9.264.2 Property Documentation

9.264.2.1 QString QEStripChart::variableSubstitutions [read, write]

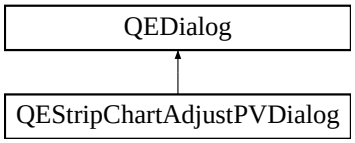
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'SAMPLE=SAM1, NAME = "Ref foil"' These substitutions are applied to all the variable names.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.265 QEStripChartAdjustPVDialog Class Reference

Inheritance diagram for QEStripChartAdjustPVDialog::



Public Member Functions

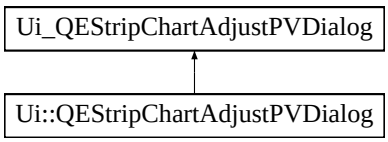
- **QEStripChartAdjustPVDialog** (QWidget *parent=0)
- void **setValueScaling** (const ValueScaling &valueScale)
- ValueScaling **getValueScaling** () const
- void **setSupport** (const double min, const double max, const QEDisplayRanges &loprHopr, const QEDisplayRanges &plotted, const QEDisplayRanges &buffered)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartAdjustPV
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartAdjustPV

9.266 Ui::QEStripChartAdjustPVDialog Class Reference

Inheritance diagram for Ui::QEStripChartAdjustPVDialog::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartAdjustPVDialog.h

9.267 QEStripChartContextMenu Class Reference

Signals

- void **contextMenuSelected** (const QEStripChartNames::ContextMenuOptions)

Public Member Functions

- QEStripChartContextMenu (bool inUse, QWidget *parent=0)
- void **setPredefinedNames** (const QStringList &pvList)
- void **setUseReceiveTime** (const bool useReceiveTime)
- void **setArchiveReadHow** (const QEArchiveInterface::How how)
- void **setLineDrawMode** (const QEStripChartNames::LineDrawModes mode)
- void **setLinePlotMode** (const QEStripChartNames::LinePlotModes mode)

9.267.1 Constructor & Destructor Documentation

9.267.1.1 QEStripChartContextMenu::QEStripChartContextMenu (bool *inUse*, QWidget * *parent* = 0) [explicit]

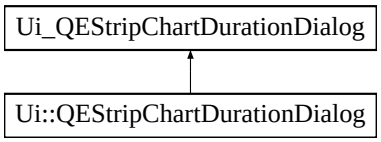
Construct strip chart item context menu. This menu item creates all required sub menu items. inUse set true for an inuse slot, i.e. already has a PV allocated. inUse set false for an empty slot.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartContextM
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEStripChartContextMenu.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartContextM

9.268 Ui::QEStripChartDurationDialog Class Reference

Inheritance diagram for Ui::QEStripChartDurationDialog::

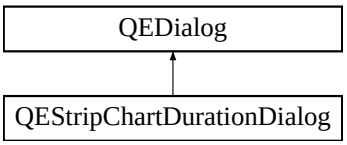


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartDurationDialog.h

9.269 QEStripChartDurationDialog Class Reference

Inheritance diagram for QEStripChartDurationDialog::



Public Member Functions

- **QEStripChartDurationDialog** (QWidget ∗parent=0)
- void **setDuration** (int secs)
- int **getDuration** () const

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartDuration
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartDuration

9.270 QEStripChartItem Class Reference

Public Types

- typedef double **CalcInputs** [QEStripChart::NUMBER_OF_PVS]

Public Slots

- void **setColour** (const QColor &colour)

Signals

- void **requestAction** (const QEActionRequests &)

Public Member Functions

- **QEStripChartItem** (QEStripChart ∗chart, const int slot, QWidget ∗parent)
- bool **isInUse** () const
- bool **isPvData** () const
- bool **isCalculation** () const
- void **setPvName** (const QString &pvName, const QString &substitutions)
- QString **getPvName** () const
- QString **getEgu** () const
- bool **isScaled** () const
- QPointF **dataPointToReal** (const QCaDataPoint &point) const
- bool **getUseReceiveTime** () const
- QEArchiveInterface::How **getArchiveReadHow** () const
- QEStripChartNames::LineDrawModes **getLineDrawMode** () const
- QEStripChartNames::LinePlotModes **getLinePlotMode** () const
- QColor **getColour** ()
- QEDisplayRanges **getLoprHopr** (bool doScale)
- QEDisplayRanges **getDisplayedMinMax** (bool doScale)
- QEDisplayRanges **getBufferedMinMax** (bool doScale)
- QCaDataPointList **determinePlotPoints** ()
- void **readArchive** ()
- void **normalise** ()
- void **plotData** ()
- double **getCurrentValue** (bool &okay) const
- void **calculateAndUpdate** (const QCaDateTime &datetime, const CalcInputs values)
- const QCaDataPoint ∗ **findNearestPoint** (const QCaDateTime &searchTime) const
- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)
- void **updateMenu** (QAction ∗action, const bool inUseMenu)

Public Attributes

- QCaVariableNamePropertyManager **pvNameProperyManager**

Protected Member Functions

- bool **eventFilter** (QObject ∗obj, QEvent ∗event)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartItem.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEStripChartItem.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartItem.cpp

9.271 QEStripChartNames Class Reference

Public Types

- enum **ChartTimeModes** { **tmRealTime**, **tmPaused**, **tmHistorical** }
- enum **ChartYRanges** {

  manual, **operatingRange**, **plotted**, **buffered**,

  dynamic, **normalised** }
- enum **PlayModes** {

  play, **pause**, **forward**, **backward**,

  selectTimes }
- enum **StateModes** { **previous**, **next** }
- enum **VideoModes** { **normal**, **reverse** }
- enum **YScaleModes** { **linear**, **log** }
- enum **LineDrawModes** { **ldmHide**, **ldmRegular**, **ldmBold** }
- enum **LinePlotModes** { **lpmRectangular**, **lpmSmooth** }
- enum **ContextMenuOptions** {

  SCCM_NONE = QEAbstractDynamicWidget::ADWCM_SUB_CLASS_-WIDGETS_START_HERE, **SCCM_READ_ARCHIVE**, **SCCM_SCALE_-CHART_AUTO**, **SCCM_SCALE_CHART_PLOTTED**,

  SCCM_SCALE_CHART_BUFFERED, **SCCM_SCALE_PV_RESET**, **SCCM_SCALE_PV_GENERAL**, **SCCM_SCALE_PV_AUTO**,

  SCCM_SCALE_PV_PLOTTED, **SCCM_SCALE_PV_BUFFERED**, **SCCM_SCALE_PV_CENTRE**, **SCCM_PLOT_RECTANGULAR**,

  SCCM_PLOT_SMOOTH, **SCCM_PLOT_SERVER_TIME**, **SCCM_-PLOT_CLIENT_TIME**, **SCCM_ARCH_LINEAR**,

  SCCM_ARCH_PLOTBIN, **SCCM_ARCH_RAW**, **SCCM_ARCH_SHEET**, **SCCM_ARCH_AVERAGED**,

  SCCM_LINE_HIDE, **SCCM_LINE_REGULAR**, **SCCM_LINE_BOLD**, **SCCM_LINE_COLOUR**,

  SCCM_PV_EDIT_NAME, **SCCM_ADD_TO_PREDEFINED**, **SCCM_-PV_WRITE_TRACE**, **SCCM_PV_STATS**,

  SCCM_PV_CLEAR, **SCCM_PV_ADD_NAME**, **SCCM_PV_PASTE_-NAME**, **SCCM_PREDEFINED_01**,

  SCCM_PREDEFINED_02, **SCCM_PREDEFINED_03**, **SCCM_-PREDEFINED_04**, **SCCM_PREDEFINED_05**,

  SCCM_PREDEFINED_06, **SCCM_PREDEFINED_07**, **SCCM_-PREDEFINED_08**, **SCCM_PREDEFINED_09**,

  SCCM_PREDEFINED_10 }

Static Public Member Functions

- static QString **chartTimeModeStatus** (const ChartTimeModes mode)
- static QString **chartYRangeStatus** (const ChartYRanges yRange)

Static Public Attributes

- static const ContextMenuOptions **ContextMenuItemFirst** = SCCM_READ_-ARCHIVE
- static const ContextMenuOptions **ContextMenuItemLast** = SCCM_-PREDEFINED_10
- static const int **NumberPrefefinedItems** = (SCCM_PREDEFINED_10 - SCCM_PREDEFINED_01 + 1)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartNames.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartNames.cp

9.272 QEStripChartPushButtonSpecifications Struct Reference

Public Attributes

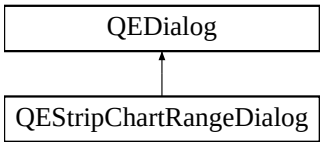
- int **gap**
- int **width**
- int **value**
- bool **isIcon**
- const QString **captionOrIcon**
- const QString **toolTip**
- const char * **member**

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.273 QEStripChartRangeDialog Class Reference

Inheritance diagram for QEStripChartRangeDialog::



Public Member Functions

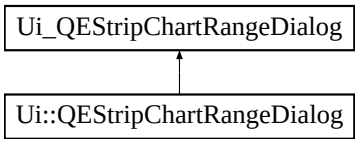
- **QEStripChartRangeDialog** (QWidget ∗parent=0)
- void **setRange** (const double min, const double max)
- double **getMinimum** ()
- double **getMaximum** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartRangeDia
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartRangeDia

9.274 Ui::QEStripChartRangeDialog Class Reference

Inheritance diagram for Ui::QEStripChartRangeDialog::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartRangeDialog.h

9.275 QEStripChartState Class Reference

Public Member Functions

- void **saveConfiguration** (PMElement &parentElement)
- void **restoreConfiguration** (PMElement &parentElement)

Public Attributes

- bool **isNormalVideo**
- QEStripChartNames::ChartTimeModes **chartTimeMode**
- QEStripChartNames::YScaleModes **yScaleMode**
- QEStripChartNames::ChartYRanges **chartYScale**
- double **yMinimum**
- double **yMaximum**
- int **duration**
- Qt::TimeSpec **timeZoneSpec**
- QDateTime **endDateTime**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartState.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartState.cpp

9.276 QEStripChartStateList Class Reference

Public Member Functions

- void **clear** ()
- void **push** (const QEStripChartState &state)
- bool **prev** (QEStripChartState &state)
- bool **next** (QEStripChartState &state)
- bool **prevAvailable** ()
- bool **nextAvailable** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.277 QEStripChartStatistics Class Reference

Public Member Functions

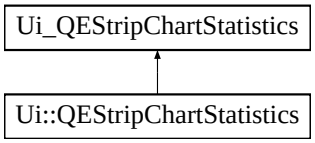
- **QEStripChartStatistics** (const QString &pvName, const QString &egu, const QCaDataPointList &dataList, QEStripChartItem ∗owner, QWidget ∗parent=0)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartStatistics.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripChartStatistics.c

9.278 Ui::QEStripChartStatistics Class Reference

Inheritance diagram for Ui::QEStripChartStatistics::

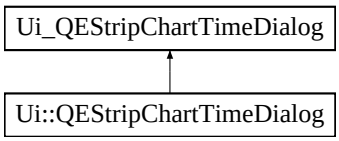


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartStatistics.h

9.279 Ui::QEStripChartTimeDialog Class Reference

Inheritance diagram for Ui::QEStripChartTimeDialog::

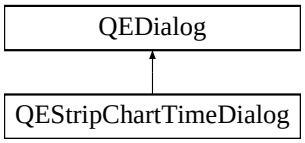


The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartTimeDialog.h

9.280 QEStripChartTimeDialog Class Reference

Inheritance diagram for QEStripChartTimeDialog::



Public Member Functions

- **QEStripChartTimeDialog** (QWidget ∗parent=0)
- void **setMaximumDateTime** (QDateTime datetime)
- void **setStartDateTime** (QDateTime datetime)
- QDateTime **getStartDateTime** ()
- void **setEndDateTime** (QDateTime datetime)
- QDateTime **getEndDateTime** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.281 QEStripChartToolBar Class Reference

This class holds all the StripChart tool bar widgets.

`#include <QEStripChartToolBar.h>`

Classes

- class OwnTabWidget

Signals

- void **stateSelected** (const QEStripChartNames::StateModes mode)
- void **videoModeSelected** (const QEStripChartNames::VideoModes mode)
- void **yScaleModeSelected** (const QEStripChartNames::YScaleModes mode)
- void **yRangeSelected** (const QEStripChartNames::ChartYRanges scale)
- void **durationSelected** (const int seconds)
- void **selectDuration** ()
- void **timeZoneSelected** (const Qt::TimeSpec timeSpec)
- void **playModeSelected** (const QEStripChartNames::PlayModes mode)
- void **readArchiveSelected** ()
- void **loadSelected** ()
- void **saveAsSelected** ()
- void **loadSelectedFile** (const QString &filename)

Public Member Functions

- **QEStripChartToolBar** (QWidget *parent=0)
- void **setYRangeStatus** (const QEStripChartNames::ChartYRanges yRange)
- void **setTimeStatus** (const QString &timeStatus)
- void **setDurationStatus** (const QString &durationStatus)
- void **setNOARStatus** (const int noar)
- void **setTimeModeStatus** (const QEStripChartNames::ChartTimeModes timeMode)
- void **setStateSelectionEnabled** (const QEStripChartNames::StateModes mode, const bool enabled)
- void **setTimeRefs** (const QDateTime &t1, const QDateTime &t2)
- void **setValue1Refs** (const double v1, const double v2)
- void **setValue2Refs** (const double v1, const double v2)

Static Public Member Functions

- static int **designHeight** ()

Friends

- class **OwnTabWidget**

9.281.1 Detailed Description

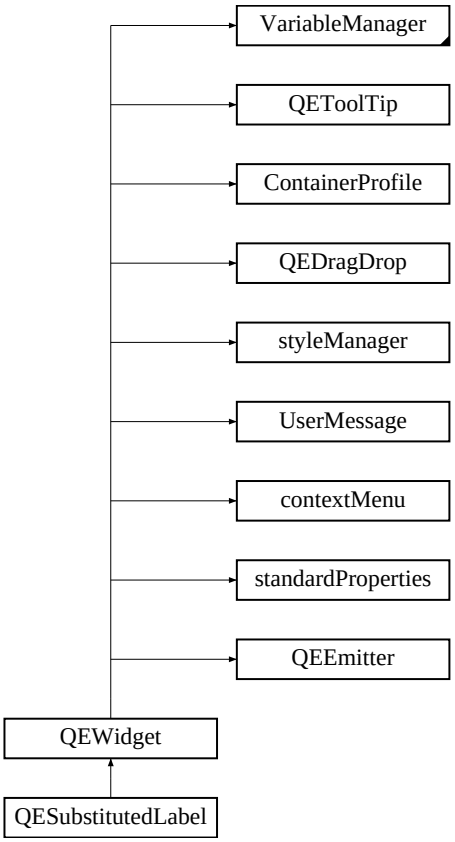
This class holds all the StripChart tool bar widgets.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEStripChartToolBar.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.282 QESubstitutedLabel Class Reference

Inheritance diagram for QESubstitutedLabel::



Public Member Functions

- **QESubstitutedLabel** (QWidget *parent=0)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- void **setSubstitutionsProperty** (QString macroSubstitutionsIn)
- QString **getSubstitutionsProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

Protected Attributes

- QString labelText

Properties

- QString textSubstitutions

9.282.1 Member Data Documentation

9.282.1.1 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

9.282.2 Property Documentation

9.282.2.1 QString QESubstitutedLabel::textSubstitutions [read, write]

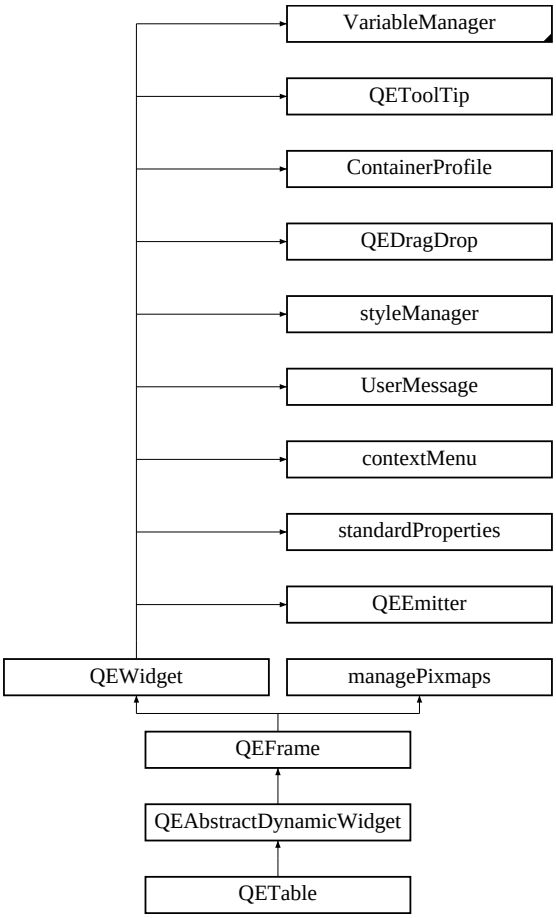
Text substitutions. These substitutions are applied to the 'labelText' property prior to copying it to the label text.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESubstitutedLabel/QES
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESubstitutedLabel/QES

9.283 QETable Class Reference

`#include <QETable.h>`Inheritance diagram for QETable::



Classes

- class **DataSets**

Public Slots

- void **setVariableName** (const int, const QString &)
- void **setTitles** (const QStringList &titles)
- void **setTitle** (const int slot, const QString &title)
- void **setSelection** (int value)
- void **setPvNameSet** (const QStringList &pvNameSet)
- void **setTableEntry** (const int slot, const QString &pvName, const QString &title)

Signals

- void **selectionChanged** (int value)
- void **pvNameSetChanged** (const QStringList &pvNameSet)
- void **titlesChanged** (const QStringList &pvNameSet)
- void dbValueChanged (const QVector< double > &out)

Public Member Functions

- QETable (QWidget *parent=0)
- virtual ~QETable ()

  Destruction.

- QString **getVariableName** (const int) const
- void **setSubstitutions** (const QString &substitutions)
- QString **getSubstitutions** () const
- void **setColumnWidthMinimum** (const int columnWidthMinimum)
- int **getColumnWidthMinimum** () const
- void **setDisplayMaximum** (const int displayMaximum)
- int **getDisplayMaximum** () const
- void **setOrientation** (const Qt::Orientation orientation)
- Qt::Orientation **getOrientation** () const
- int **addPvName** (const QString &pvName)
- void **clearAllPvNames** ()
- QStringList **getTitles** () const
- int **getSelection** () const
- QStringList **getPvNameSet** () const

Protected Types

- enum **OwnContextMenuOptions** { **CM_HORIZONTAL_TABLE** = ADWCM_SUB_CLASS_WIDGETS_START_HERE, **CM_VERTICAL_TABLE** }

Protected Member Functions

- QSize **sizeHint** () const
- bool **eventFilter** (QObject *watched, QEvent *event)
- void **resizeEvent** (QResizeEvent *event)
- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- QMenu * **buildContextMenu** ()
- void **contextMenuTriggered** (int selectedItemNum)
- void **mousePressEvent** (QMouseEvent *event)
- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void saveConfiguration (PersistanceManager *pm)
- void restoreConfiguration (PersistanceManager *pm, restorePhases restorePhase)

Properties

- QString variableName1

  EPICS variable names (CA PV).

- QString **variableName2**
- QString **variableName3**
- QString **variableName4**
- QString **variableName5**
- QString **variableName6**
- QString **variableName7**
- QString **variableName8**
- QString **variableName9**
- QString **variableName10**
- QString **variableName11**
- QString **variableName12**
- QString **variableName13**
- QString **variableName14**
- QString **variableName15**
- QString **variableName16**
- QString **variableName17**
- QString **variableName18**
- QString **variableName19**
- QString **variableName20**
- QString variableSubstitutions
- QStringList titles
- int colWidthMinimum
- int displayMaximum
- Qt::Orientation orientation
- bool showGrid
- Qt::PenStyle gridStyle

9.283.1 Detailed Description

This class provides an EPICS aware table widget which is capable of displaying up to 20 array PVs in tabular form. When in the default vertical orientation each column displays a consecutive element from an array EPICS variable.

When in horizonal mode, the table and functionality is transposed.

It is tighly integrated, via the QEAbstractWidget, with the base class QEWidget class which provides generic support such as macro substitutions, drag/drop, and standard properties. QEAbstractWidget provides all standard properties.

9.283.2 Constructor & Destructor Documentation

9.283.2.1 QETable::QETable (QWidget * *parent* = 0) [explicit]

Create without a variable(s). Use setVariableName functions.

9.283.3 Member Function Documentation

9.283.3.1 void QETable::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. Note: this widget emits the numeric enumeration value as opposed to the associated text.

9.283.3.2 void QETable::restoreConfiguration (PersistanceManager *, restorePhases) [protected, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented from QEWidget.

9.283.3.3 void QETable::saveConfiguration (PersistanceManager *) [protected, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented from QEWidget.

9.283.4 Property Documentation

9.283.4.1 int QETable::colWidthMinimum [read, write]

Specified the minimum allow column width. The widget will shrink/expand the width of each column to as to exactly fit the with of the widget. However, columns will not shrink to less than the value provided by this property. Defaults to 80.

9.283.4.2 int QETable::displayMaximum [read, write]

The maximum number of array elements that will be displayed irrespective of the number of elements that the EPICS variable contains. Defaults to 4096.

9.283.4.3 Qt::PenStyle QETable::gridStyle [read, write]

Sets table grid style. Defaults to SolidLine.

9.283.4.4 Qt::Orientation QETable::orientation [read, write]

Determines if the variable values are displayed in rows (orientation is horizontal) or in columns (orientation is vertical). The default is vertical.

9.283.4.5 bool QETable::showGrid [read, write]

Controls if table grid is displayed. Default to true.

9.283.4.6 QStringList QETable::titles [read, write]

Allows specification of tables titles. If blank, the default, then out-of-the-box QTableWidget heading are used, i.e. 1, 2, etc. If "<>" is specified, then this is replaced by the PV name. This is particulary useful when PV na,es are specifed dynamically or by substitution.

9.283.4.7 QString QETable::variableSubstitutions [read, write]

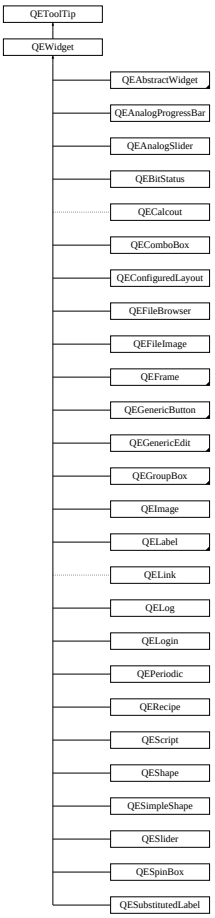
Macro substitutions. The default is no substitutions. The format is NAME1=VALUE1[,] NAME2=VALUE2... Values may be quoted strings. For example, 'PUMP=PMP3, NAME = "My Pump"' These substitutions are applied to variable names for all QE widgets. In some widgets are are also used for other purposes.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QETable/QETable.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QETable.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QETable/QETable.cpp

9.284 QEToolTip Class Reference

`#include <QEToolTip.h>`Inheritance diagram for QEToolTip::



Public Member Functions

- **QEToolTip** (QWidget ∗ownerIn)
- void **setNumberToolTipVariables** (const unsigned int number)
- void **updateToolTipVariable** (const QString &variable, const unsigned int variableIndex)
- void **updateToolTipAlarm** (const QString &alarm, const unsigned int variableIndex)
- void **updateToolTipConnection** (bool connection, const unsigned int variableIndex=0)
- void **updateToolTipCustom** (const QString &custom)
- void **setVariableAsToolTip** (const bool variableAsToolTip)
- bool **getVariableAsToolTip** () const

Friends

- class QEToolTipSingleton

9.284.1 Detailed Description

QE widget tool tips may be set to display the names of the process variables supplying data to the widget and the alarm state and connectino status of those variables. The QE widget may also set some custom text to be displayed along with this information. The QEToolTip class manages building and setting the QE widget tool tips when this functino is required.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEToolTip.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEToolTip.cpp

9.285 QEToolTipSingleton Class Reference

Friends

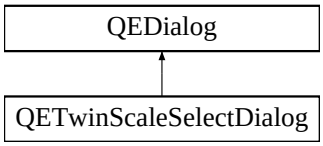
- class QEToolTip

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEToolTip.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEToolTip.cp

9.286 QETwinScaleSelectDialog Class Reference

Inheritance diagram for QETwinScaleSelectDialog::



Classes

- struct ScaleLimit

Public Member Functions

- **QETwinScaleSelectDialog** (const QString &windowTitle, const QString &scaleOneName, const QString &scaleTwoName, QWidget *parent=0)
- void **setActiveMap** (const ScaleLimit &scaleOne, const ScaleLimit &scaleTwo)
- void **getActiveMap** (ScaleLimit &scaleOne, ScaleLimit &scaleTwo)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QETwinScaleSelectDialog.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QETwinScaleSelectDialog.cpp

9.287 QEUtilities Class Reference

Static Public Member Functions

- static QColor fontColour (const QColor &backgroundColour)
- static QString colourToStyle (const QColor &backgroundColour, const QColor &foregroundColour)
- static QString colourToStyle (const QColor &backgroundColour)
- static QString offBackgroundStyle ()
- static QColor **darkColour** (const QColor &lightColour)
- static QColor **blandColour** (const QColor &vibrantColour)
- static int roundToInt (const double x, bool *ok=NULL)
- static QString intervalToString (const double time, const int precision, const bool showDays)
- static int getTimeZoneOffset (const QDateTime &atTime)
- static QString getTimeZoneTLA (const Qt::TimeSpec timeSpec, const QDateTime &atTime)
- static QString getTimeZoneTLA (const QDateTime &atTime)
- static QString enumToString (const QObject &object, const QString &enumTypeName, const int enumValue)
- static QString enumToString (const QMetaObject &mo, const QString &enumTypeName, const int enumValue)
- static int stringToEnum (const QObject &object, const QString &enumTypeName, const QString &enumImage, bool *ok=0)
- static int stringToEnum (const QMetaObject &mo, const QString &enumTypeName, const QString &enumImage, bool *ok=0)
- static QWidget * findWidget (QWidget *parent, const QString &className)
- static QStringList variantToStringList (const QVariant &v)
- static void listPVNames (QWidget *rootWidget, const QString &targetFile, const QString &comment)
- static QString dirName (const QString &pathName)
- static void debugWidgetHierarchy (const QWidget *root, const int instance=0, const int level=0)

9.287.1 Member Function Documentation

9.287.1.1 QString QEUtilities::colourToStyle (const QColor & *backgroundColour*) [static]

Converts a given colour to associated background-color style, and sets foreground (font) colour to black or white accordingly.

9.287.1.2 QString QEUtilities::colourToStyle (const QColor & *backgroundColour*, const QColor & *foregroundColour*) [static]

Converts a given colours to the associated style.

9.287.1.3 void QEUtilities::debugWidgetHierarchy (const QWidget * *root*, const int *instance* = 0, const int *level* = 0) [static]

Testing only - outputs the widget hierarchy to std out.

9.287.1.4 QString QEUtilities::dirName (const QString & *pathName*) [static]

Returns the directory name as QString of the given pathname. This is essentially like the Linux dirname command. This conveniance functions hides the faffing with QFileInfo/QDir

9.287.1.5 QString QEUtilities::enumToString (const QMetaObject & *mo*, const QString & *enumTypeName*, const int *enumValue*) [static]

This function returns the image, as a QString, of a enumeration value (cast as an integer). An invalid enumeration value returns a null string.

Note: This functions rely on the meta object compiler (moc) generated code. To use this functions, the enumeration type must be declared within a Q_OBJECT QObject and the enum type qualifier with the Q_ENUMS, e.g. Q_ENUMS (Shapes).

9.287.1.6 QString QEUtilities::enumToString (const QObject & *object*, const QString & *enumTypeName*, const int *enumValue*) [static]

This function returns the image, as a QString, of a enumeration value (cast as an integer). An invalid enumeration value returns a null string.

Note: This functions rely on the meta object compiler (moc) generated code. To use this functions, the enumeration type must be declared within a Q_OBJECT QObject and the enum type qualifier with the Q_ENUMS, e.g. Q_ENUMS (Shapes).

9.287.1.7 QWidget * QEUtilities::findWidget (QWidget * *parent*, const QString & *className*) [static]

This function tree walks the given parent looking a widget of the specified class name. The function returns the first found or NULL.

9.287.1.8 QColor QEUtilities::fontColour (const QColor & *backgroundColour*) [static]

For the specified bacground colour, returns font colour (black or white) with a suitable contrast. Currently based on rgb values, and ignores alpha.

9.287.1.9 int QEUtilities::getTimeZoneOffset (const QDateTime & *atTime*) [static]

Get the local time zone offset (in seconds) for the nominated time. This is not fixed for the location and will depend on if the the time is a standard time or a daylight savings/summer time.

9.287.1.10 QString QEUtilities::getTimeZoneTLA (const QDateTime & *atTime*) [static]

Overloaded function that uses the timeSpec assocaited with atTime.

9.287.1.11 QString QEUtilities::getTimeZoneTLA (const Qt::TimeSpec *timeSpec*, const QDateTime & *atTime*) [static]

Extract the local time zone three letter acronym for the time. If the timeSpec patameter is Qt::UTC, then this simply returns "UTC". However, if the timeSpec patameter is Qt::LocalTime, then extract the TLA application of the time in question (and NOT for the current time). E.g. for the UK this would be GMT or BST, for Victoria Australia EST or EST.

9.287.1.12 QString QEUtilities::intervalToString (const double *time*, const int *precision*, const bool *showDays*) [static]

Converts a time interval (in seconds) to format +d hh:mm:ss.fff

9.287.1.13 void QEUtilities::listPVNames (QWidget * *rootWidget*, const QString & *targetFile*, const QString & *comment*) [static]

Performs deep search and writes all PV names to nominated target file.

9.287.1.14 QString QEUtilities::offBackgroundStyle () [static]

Provides a slightly lighter colour than the standard form background colour. This is particularly useful for QLabel based widgets.

9.287.1.15 int QEUtilities::roundToInt (const double *x*, bool * *ok* = NULL) [static]

Rounds a float to the nearest integer. Not all compilers support lround. If x outside of integer range, the result is set to the min or max allowed integer value and the ok parameter (if defined) set false if x out of range.

9.287.1.16 int QEUtilities::stringToEnum (const QMetaObject & *mo*, const QString & *enumTypeName*, const QString & *enumImage*, bool * *ok* = 0) [static]

This function returns the enumeration value given an enumeration image. The caller must cast the result to the appropriate enumeration type. The image must be exact match including case. The only tolerance allowed for is that the image is trimmed. An invalid image cause this function to return -1. However, -1 is an uncommon but not invalid enumeration value. If -1 might be a valid value, then the caller should specify the and check the ok parameter.

Note: This functions rely on the meta object compiler (moc) generated code. To use this functions, the enumeration type must be declared within a Q_OBJECT QObject and the enum type qualifier with the Q_ENUMS, e.g. Q_ENUMS (Shapes).

9.287.1.17 int QEUtilities::stringToEnum (const QObject & *object*, const QString & *enumTypeName*, const QString & *enumImage*, bool * *ok* = 0) [static]

This function returns the enumeration value given an enumeration image. The caller must cast the result to the appropriate enumeration type. The image must be exact match including case. The only tolerance allowed for is that the image is trimmed. An invalid image cause this function to return -1. However, -1 is an uncommon but not invalid enumeration value. If -1 might be a valid value, then the caller should specify the and check the ok parameter.

Note: This functions rely on the meta object compiler (moc) generated code. To use this functions, the enumeration type must be declared within a Q_OBJECT QObject and the enum type qualifier with the Q_ENUMS, e.g. Q_ENUMS (Shapes).

9.287.1.18 QStringList QEUtilities::variantToStringList (const QVariant & *v*) [static]

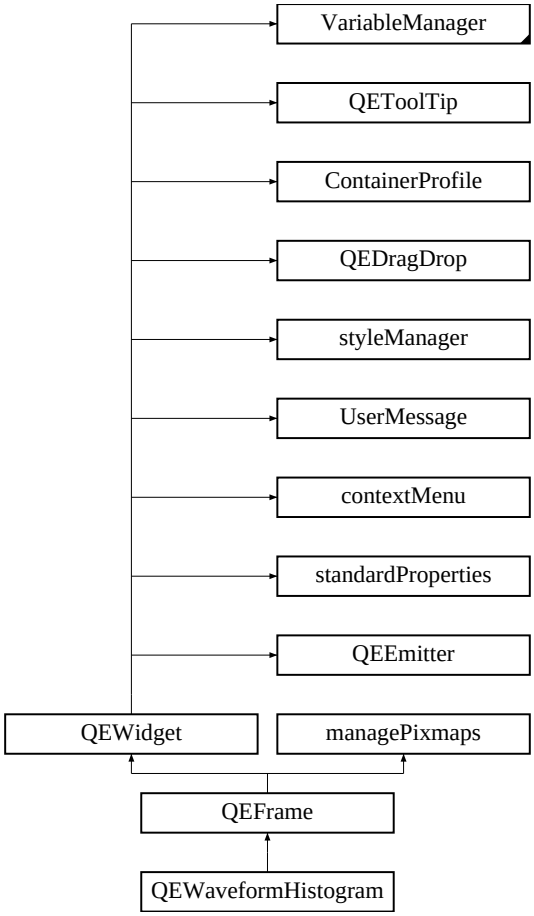
Converts QVariant to a QStringList, where each string item is further split using white space as delimiter. This function was principally designed for use within the paste/drop functions of widgets that support multiple PVs.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QECommon.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QECommon.cpp

9.288 QEWaveformHistogram Class Reference

`#include <QEWaveformHistogram.h>`Inheritance diagram for QEWaveformHistogram::



Public Types

- enum ScaleModes { Manual, Auto, OperationalRange }

Signals

- void **mouseIndexChanged** (const int index)
- void **mouseIndexPressed** (const int index, const Qt::MouseButton button)
- void dbConnectionChanged (const bool &isConnected)

  Sent when the widget state updated following a channel connection change.

- void dbValueChanged ()

Sent when the widget is updated following a data change.

Public Member Functions

- **QEWaveformHistogram** (QWidget *parent=0)
- void **setScaleMode** (const ScaleModes scaleMode)
- ScaleModes **getScaleMode** () const

Protected Member Functions

- qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- void **dragEnterEvent** (QDragEnterEvent *event)
- void **dropEvent** (QDropEvent *event)
- void **mousePressEvent** (QMouseEvent *event)
- QString **copyVariable** ()
- QVariant **copyData** ()
- void **paste** (QVariant v)
- void **setPvName** (const QString &pvName)

Properties

- QString **variable**
- QString **variableSubstitutions**
- bool **autoBarGapWidths**
- int **barWidth**
- int **gap**
- int **margin**
- ScaleModes **scaleMode**
- double **minimum**
- double **maximum**
- double **baseLine**
- bool **drawAxies**
- bool **showScale**
- bool **showGrid**
- bool **logScale**
- QColor **backgroundColour**
- QColor **secondBgColour**
- int **secondBgSize**
- bool **showSecondBg**

- QColor **barColour**
- bool **drawBorder**
- Qt::Orientation **orientation**

9.288.1 Detailed Description

The QEWaveformHistogram class is a EPICS aware histogram widget. The value of, i.e. the length of each bar of the histogram is controlled by the corresponding element of an array, e.g. waveform, process variable. When the variable is defined (connected), the bar lengths are updated, and optionally the bar colours set to reflect the variable's severity status. The bar is 'grayed-out' when the variable is disconnected (although the bars retains their last known values/lengths). The histogram nature of the this widget is provided by a QEHistogram widget. The QEWaveformHistogram widget is tighly integrated with the base class QEWidget, via QEFrame, which provides generic support such as macro substitutions, drag/drop, and standard properties.

9.288.2 Member Enumeration Documentation

9.288.2.1 enum QEWaveformHistogram::ScaleModes

Enumerator:

Manual Use property minimum/maximum to scale histogram.

Auto Dynamically scale based on minimum/maximum displayed value.

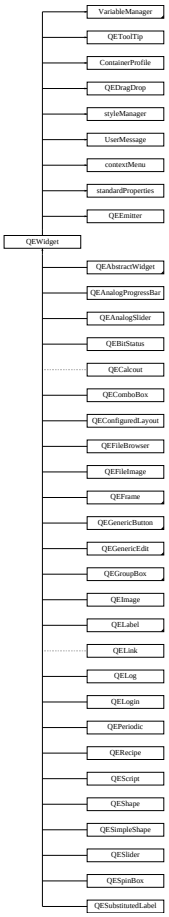
OperationalRange Use process variable operational range (LOPR/HOPR).

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEWavefo
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QEWaveformHistogram.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEHistogram/QEWavefo

9.289 QEWidget Class Reference

`#include <QEWidget.h>`Inheritance diagram for QEWidget::



Public Types

- enum restorePhases { **APPLICATION** = SaveRestoreSignal::RESTORE_APPLICATION, **FRAMEWORK** = SaveRestoreSignal::RESTORE_QEFRAMEWORK }

  Restore phases. When a widget's persistant data is restored, the restore occurs in two phases.

- typedef QList< unsigned int > ControlVariableIndicesSet

Public Member Functions

- QEWidget (QWidget *ownerIn)

Constructor.

- virtual ~QEWidget ()

  Destructor.

- unsigned int getMessageSourceId ()
- void setMessageSourceId (unsigned int messageSourceId)
- void processConnectionInfo (bool isConnected, const unsigned int variableIndex=0)
- void processAlarmInfo (QCaAlarmInfo &alarmInfo, const unsigned int variableIndex=0)
- virtual void setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void reestablishConnection (unsigned int variableIndex)
- QFile * openQEFile (QString name, QFile::OpenModeFlag mode)
- QString defaultFileLocation ()
- QString getFrameworkVersion ()
- virtual void saveConfiguration (PersistanceManager *)
- virtual void restoreConfiguration (PersistanceManager *, restorePhases)
- virtual void scaleBy (const int, const int)
- QWidget * getQWidget ()
- const QList< QCaInfo > getQCaInfo ()
- virtual QMenu * getDefaultContextMenu ()

  Flag indicating this widget is running inside Qt's 'designer'.

- virtual int getUserPrecision ()

  Return the Qt default context menu to add to the QE context menu.

- virtual int getUserAlarmMin ()

  Return the widgets precision property if any (otherwise zero).

- virtual int getUserAlarmMax ()

  Return the widget alarm minimum if any (otherwise zero).

- virtual bool getAlarmSensitive ()

  Return the widget alarm maximum if any (otherwise zero).

Static Public Member Functions

- static QColor getColor (QCaAlarmInfo &alarmInfo, const int saturation)
- static QFile * findQEFile (QString name, ContainerProfile *profile)
- static QFile * findQEFile (QString name)
- static void doAction (QWidget *searchPoint, QString widgetName, QString action, QStringList arguments, bool initialise, QAction *originator)
- static bool **inDesigner** ()

Protected Member Functions

- qcaobject::QCaObject * createConnection (unsigned int variableIndex)

  Create a CA connection. Use default subscribe option. Return a QCaObject if successfull.

- qcaobject::QCaObject * createConnection (unsigned int variableIndex, const bool do_subscribe)

  Create a CA connection. Use specified subscribe option. Return a QCaObject if successfull.

- QString persistantName (QString prefix) const

  Returns a string that will not change between runs of the application (given the same configuration).

- virtual void actionRequest (QString, QStringList, bool, QAction *)

  Perform a named action.

- void setDrop (QVariant drop)

  Default get drop action.

- QVariant getDrop ()

  Default set drop action.

- void **setControlPV** (const unsigned int variableIndex)
- void setControlPVs (const ControlVariableIndicesSet &variableIndexList)

  Nominate a single variable index as the control variable index.

- ControlVariableIndicesSet getControlPVs () const

  Nominate a variable number of variable indicis as the control variable indies list.

- void **setAccessCursorStyle** ()
- void startGui (const QEActionRequests &request)

  Update cursor style if all control variable indices are write inhibited.

9.289.1 Detailed Description

This class is used as a base for all CA aware wigets, 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 VariableManager, VariableNameManager and QEToolTip.

In particular, the VariableManager class manages QCaObject classes that stream updates to the CA aware widgets. But the VariableManager class, however, doesn't know

how to format the data, or how the updates will be used. To resolve this, the VariableManager class asks a parent class (such as QELabel) to create the QCaObject class in what ever flavour it wants, by calling the virtual function createQcaItem. 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 VariableManager calling its 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 contining plugin definitions. After loading the plugin widgets, code in the QEForm class calls the activate() function in the VariableManager base of this class (QEWiget). 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 immedietly calls (through VariableManager::createVariable) 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.

9.289.2 Member Typedef Documentation

9.289.2.1 typedef QList<unsigned int> QEWidget::ControlVariableIndicesSet

Defines a list or set of variable indices. Used to indicate those variables that are considered control variables, and thus have their cursor updated to the forbidden cursor if/when write access is denied.

9.289.3 Member Function Documentation

9.289.3.1 QString QEWidget::defaultFileLocation ()

Returns the default location to create files. Use this to create files in a consistant location

9.289.3.2 void QEWidget::doAction (QWidget * *searchPoint*, QString *widgetName*, QString *action*, QStringList *arguments*, bool *initialise*, QAction * *originator*) [static]

Find a QE widget and request an action. The widget hierarchy under a supplied widget is searched for a QE widget with a given name. If found the QE widget will attecjpt to carry out the requested action which consists of an action string and an argument list. This method allows an application to initiate QE widget activity. The QEGui application uses this mechanism when providing custom menus defined in XML files. The method returns true if the named widget was found. (The action was not nessesarily performed, or even recognised by the widget)

9.289.3.3 QFile * QEWidget::findQEFile (QString *name*) [static]

Static method that looks for a file in a standard set of locations and assumes a current published profile. Returns a pointer to a QFile which is the caller's responsibility to delete, or NULL if the file was not found.

9.289.3.4 QFile * QEWidget::findQEFile (QString *name*, ContainerProfile * *profile*) [static]

Static method that looks for a file in a standard set of locations Returns a pointer to a QFile which is the caller's responsibility to delete, or NULL if the file was not found.

9.289.3.5 QColor QEWidget::getColor (QCaAlarmInfo & *alarmInfo*, const int *saturation*) [static]

Static method to 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

9.289.3.6 QString QEWidget::getFrameworkVersion ()

Returns the QE framework that built this instance of the widget. On windows, the QEFramework DLL may be loaded twice with potentially different versions of it.

9.289.3.7 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.

9.289.3.8 const QList< QCaInfo > QEWidget::getQCaInfo ()

Return information about the data sources for this widget

9.289.3.9 QWidget * QEWidget::getQWidget ()

Get the QWidget that the parent of this QEWidget instance is based on. For example, the parent of a QEWidget might be a QELabel, which is based on QLabel which is based on QWidget.

9.289.3.10 QFile * QEWidget::openQEFile (QString *name*, QFile::OpenModeFlag *mode*)

Looks for a file in a standard set of locations (and opens the file)

9.289.3.11 void QEWidget::processAlarmInfo (QCaAlarmInfo & *alarmInfo*, const unsigned int *variableIndex* = 0)

This convenience function updates the alarm tool tip, and alarm status style if the displayAlarmState property is set to true - assumes the widget uses standard properties. This function is perhaps most usefull for single-variable widgets.

9.289.3.12 void QEWidget::processConnectionInfo (bool *isConnected*, const unsigned int *variableIndex* = 0)

This convenience function updates the connection style and also clears/re-initialises the last saved severity and display alarm info state, as used by processAlarmInfo.

9.289.3.13 void QEWidget::reestablishConnection (unsigned int *variableIndex*)

Essentially like setVariableNameAndSubstitutions, but without resetting the variable name or substitutions. This function calls establishConnection.

9.289.3.14 virtual void QEWidget::restoreConfiguration (PersistanceManager *, restorePhases) [inline, virtual]

Service a request to restore the QE widget's configuration. A QE widget recover any configuration details from the PersistanceManager. For example, a QEStripChart may restore the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method. This is called twice with an incrementing restorePhase. Most widgets will miss the first call as they don't exist yet (they are created as part of the first phase)

Reimplemented in QECorrelation, QEPlotter, QEPvProperties, QEScratchPad, QEStripChart, and QETable.

9.289.3.15 virtual void QEWidget::saveConfiguration (PersistanceManager *) [inline, virtual]

Service a request to save the QE widget's current configuration. A widget may save any configuration details through the PersistanceManager. For example, a QEStripChart may save the variables being plotted. Many QE widgets do not have any persistant data requirements and do not implement this method.

Reimplemented in QECorrelation, QEPlotter, QEPvProperties, QEScratchPad, QEStripChart, and QETable.

9.289.3.16 virtual void QEWidget::scaleBy (const *int*, const *int*) [inline, virtual]

Any QEWidget that requires additional scaling, i.e. above and beyond the standard scaling applied to size, minimum size, maximum size and font size, may override this function in order to perform any bespoke scaling need by the widget (for example see QEShape). The scaling is defined using a rational number specifed by two integers (m, d). The first (m) parameter is the multiplier and the second (d) parameter is the divisor. For example, if m = 4 and d = 5, then an 80% scaling should be applied. And if m = 5 and d = 4, and a 125% scaling is required.

Reimplemented in QEShape.

9.289.3.17 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.

9.289.3.18 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 suitible in most cases.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidget.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidget.cpp

9.290 QEWidgetProperties Class Reference

`#include <QEWidgetProperties.h>`

Public Types

- enum DisabledRecordPolicy { ignore = 0, grayout }

Public Member Functions

- **QEWidgetProperties** (QObject *parent=0)

9.290.1 Detailed Description

This class defines common widget property types.

This class is not intended to be inherited by any other class. The use of the class is to provide a name space and allow moc to generate the asociated meta data.

9.290.2 Member Enumeration Documentation

9.290.2.1 enum QEWidgetProperties::DisabledRecordPolicy

Specifies how a (control) widget should behave if/when the underlying record becomes disabled (i.e. DISA and DISV fields become equal).

Enumerator:

ignore No change in appearance - the default.

grayout Grayed out apperance, i.e. same as when the widgest PV is disconnected.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidgetProperties.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidgetProperties.cpp

9.291 QNumericEdit Class Reference

`#include <QNumericEdit.h>`

Public Types

- enum Notations { Fixed, Scientific }

Public Slots

- void **setValue** (const double value)
- void **setValue** (const int value)

Signals

- void **valueChanged** (const double value)
- void **valueChanged** (const int value)
- void **returnPressed** ()
- void **editingFinished** ()

Public Member Functions

- QNumericEdit (QWidget *parent=0)

  Construction.

- virtual ~QNumericEdit ()

  Destruction.

- double **getValue** () const
- void **setPrefix** (const QString &prefix)
- QString **getPrefix** () const
- void **setSuffix** (const QString &suffix)
- QString **getSuffix** () const
- QString **getCleanText** () const
- void **setLeadingZeros** (const int value)
- int **getLeadingZeros** () const
- void **setPrecision** (const int value)
- int **getPrecision** () const
- void **setNotation** (const Notations notation)
- Notations **getNotation** () const
- void **setMinimum** (const double value)
- double **getMinimum** () const
- void **setMaximum** (const double value)
- double **getMaximum** () const

- void **setRadix** (const QEFixedPointRadix::Radicies value)
- QEFixedPointRadix::Radicies **getRadix** () const
- void **setSeparator** (const QEFixedPointRadix::Separators value)
- QEFixedPointRadix::Separators **getSeparator** () const

Protected Member Functions

- void **focusInEvent** (QFocusEvent ∗event)
- bool **eventFilter** (QObject ∗obj, QEvent ∗event)

Protected Attributes

- QEFixedPointRadix **fpr**

Properties

- bool frame
- QString suffix
- QString prefix
- Qt::Alignment alignment
- QString cleanText
- Notations notation
- QEFixedPointRadix::Radicies radix

  Specify radix, default is Decimal.

- QEFixedPointRadix::Separators separator

  Specify digit 'thousands' separator character, default is none.

- int leadingZeros
- int precision
- double minimum

  Specify the mimimum allowed value.

- double maximum

  Specify the maximum allowed value.

- double value

  Specify the value after min/max.

Friends

- class QENumericEdit

9.291.1 Detailed Description

This class provides a non EPICS aware numeric edit widget, somewhat akin to QDoubleSpin. However QNumericEdit provides a number of advantages over QDoubleSpin: a/ the former is restricted to a single spin value, where as QNumericEdit allows any digit to be selected and used as the spin value; b/ the former is restricted to decimal representation where as QNumericEdit allows for hexadecimal, octal and binary represntations; c/ this QNumericEdit allow allows a 'thousands' seperator character to be specified; and d/ by setting the precision to 0, it effectively becems akin to a QSpinBox.

9.291.2 Member Enumeration Documentation

9.291.2.1 enum QNumericEdit::Notations

User friendly enumerations for notation property.

Enumerator:

Fixed Fixed point: +ddd.ddd.

Scientific Scienctific: +d.ddddde+nn.

9.291.3 Property Documentation

9.291.3.1 Qt::Alignment QNumericEdit::alignment `[read, write]`

This property holds the alignment of the numeric edit. Both horizontal and vertical alignment is allowed here, Qt::AlignJustify will map to Qt::AlignLeft. By default, this property contains a combination of Qt::AlignRight and Qt::AlignVCenter.

9.291.3.2 QString QNumericEdit::cleanText `[read]`

This property holds the displayed text. Not a property available to designer. It exludes any prefix/suffix.

9.291.3.3 bool QNumericEdit::frame `[read, write]`

This property holds whether the numeric edit draws itself with a frame. If enabled (the default) the numeric edit draws itself inside a frame, otherwise the line edit draws itself without any frame.

9.291.3.4 int QNumericEdit::leadingZeros `[read, write]`

Speficies the number of leading zeros. Strictly speaking, this should be an unsigned int, but designer int properties editor much 'nicer'.

9.291.3.5 Notations QNumericEdit::notation [read, write]

Notation used for formatting/editing. Default is fixed.

9.291.3.6 int QNumericEdit::precision [read, write]

Precision used for the display and editing of numbers. The default is 4. Strictly speaking, this should be an unsigned int, but designer int properties editor much 'nicer'.

9.291.3.7 QString QNumericEdit::prefix [read, write]

This property holds any fixed text (default is "") displayed before the numeric value.

9.291.3.8 QString QNumericEdit::suffix [read, write]

This property holds any fixed text (default is "") displayed after the numeric value.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QENumericEdit/QNumericEdit.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QNumericEdit.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QENumericEdit/QNumericEdit.cpp

9.292 QRadioGroup Class Reference

Public Types

- enum ButtonStyles { Radio, **Push** }
- enum ButtonOrders { rowMajor, **colMajor** }

  Enumrations values used to select the button order.

Public Slots

- void **setValue** (const int value)

Signals

- void **valueChanged** (const int value)

Public Member Functions

- QRadioGroup (QWidget *parent=0)
- QRadioGroup (const QString &title, QWidget *parent=0)
- virtual ~QRadioGroup ()

  Destruction.

- int **getMaximumButtons** () const
- int **getValue** () const
- void **setStrings** (const QStringList &strings)
- QStringList **getStrings** () const
- void **setColumns** (int columns)
- int **getColumns** () const
- void **setSpacing** (int spacing)
- int **getSpacing** () const
- void **setButtonStyle** (const ButtonStyles buttonStyle)
- ButtonStyles **getButtonStyle** () const
- void **setButtonOrder** (const ButtonOrders buttonOrder)
- ButtonOrders **getButtonOrder** () const

Protected Member Functions

- QSize **sizeHint** () const
- bool **eventFilter** (QObject *watched, QEvent *event)

Properties

- QRadioGroup::ButtonStyles **buttonStyle**
- QRadioGroup::ButtonOrders **buttonOrder**
- int **columns**
- int **spacing**
- QStringList **strings**
- int **value**

9.292.1 Member Enumeration Documentation

9.292.1.1 enum QRadioGroup::ButtonOrders

Enumrations values used to select the button order.

Enumerator:

rowMajor row by row button order - the default

9.292.1.2 enum QRadioGroup::ButtonStyles

Enumrations values used to select the button style. Whereas check box buttons can/do work, this option not provided as check boxes are not assoicated with the radio button, i.e. one and only one selected, paradigm.

Enumerator:

Radio Use radio buttons - the default.

9.292.2 Constructor & Destructor Documentation

9.292.2.1 QRadioGroup::QRadioGroup (QWidget * *parent* = 0) [explicit]

Create with default title.

9.292.2.2 QRadioGroup::QRadioGroup (const QString & *title*, QWidget * *parent* = 0) [explicit]

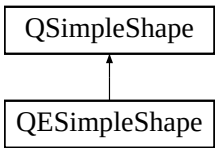
Create with a group title.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERadioGroup/QRadioGroup.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_QRadioGroup.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QERadioGroup/QRadioGroup.cpp

9.293 QSimpleShape Class Reference

`#include <QSimpleShape.h>`Inheritance diagram for QSimpleShape::



Public Types

- enum **Shapes** {
  circle, **ellipse**, **rectangle**, **roundRectangle**,
  roundSquare, **square**, **triangleUp**, **triangleDown**,
  triangleLeft, **triangleRight**, **triangleTopRight**, **triangleBottomRight**,
  triangleBottomLeft, **triangleTopLeft**, **diamond**, **equalDiamond**,
  arrowUp, **arrowDown**, **arrowLeft**, **arrowRight**,
  crossHorizontal, **crossVertical**, **hexagon**, **octogon**,
  snakeHorizontal, **snakeVertical** }
- enum TextFormats { FixedText, StateSet, PvText, **LocalEnumeration** }

Public Slots

- void **setValue** (const bool value)
- void **setValue** (const int value)

Public Member Functions

- QSimpleShape (QWidget *parent=0)
- virtual ~QSimpleShape ()

  Destruction.

- int **getValue** () const
- void **setModulus** (const int value)
- int **getModulus** () const
- void **setStateSet** (const QStringList &stateSet)
- QStringList **getStateSet** () const
- void **setShape** (const Shapes value)
- Shapes **getShape** () const
- void **setEdgeWidth** (const int value)
- int **getEdgeWidth** () const
- void **setSemiCycles** (const int value)

- int **getSemiCycles** () const
- void **setPercentSize** (const int value)
- int **getPercentSize** () const
- void **setEdgeColour** (const QColor value)
- QColor **getEdgeColour** () const
- void **setTextFormat** (const TextFormats value)
- TextFormats **getTextFormat** () const
- void **setFixedText** (const QString &value)
- QString **getFixedText** () const
- void **setFlashRate** (const QEScanTimers::ScanRates)
- QEScanTimers::ScanRates **getFlashRate** () const
- void **setFlashOffColour** (const QColor colour)
- QColor **getFlashOffColour** () const
- void **setIsActive** (const bool value)
- bool **getIsActive** () const
- void **setColourProperty** (const int slot, const QColor color)
- QColor **getColourProperty** (const int slot) const
- void **setFlashProperty** (int slot, const bool isFlashing)
- bool **getFlashProperty** (int slot) const

Protected Member Functions

- virtual QString **getItemText** ()
- virtual QColor getItemColour ()

  returns "" unless overriden.

Properties

- Shapes shape
- int edgeWidth
- int semiCycles
- int percentSize
- int value

  Shape value - range 0 to 15 - default is zero.

- int modulus

  Shape value modulus - range 2 to 16 - default is 16.

- TextFormats format
- QString fixedText

  User specified text - defaults to null string.

- QStringList stateSet

  Text values used when format is LocalEnumeration.

- QEScanTimers::ScanRates flashRate
- bool isActive

  When inActive, the widgers is grayed-out, as if disabled. The default is true.

- QColor edgeColour

  Edge color - default is black.

- QColor flashOffColour

  Flash offColour - default is clear.

- QColor colour0

  Value (modulo modulus) used to select colour.

- QColor **colour1**
- QColor **colour2**
- QColor **colour3**
- QColor **colour4**
- QColor **colour5**
- QColor **colour6**
- QColor **colour7**
- QColor **colour8**
- QColor **colour9**
- QColor **colour10**
- QColor **colour11**
- QColor **colour12**
- QColor **colour13**
- QColor **colour14**
- QColor **colour15**
- bool flash0
- bool **flash1**
- bool **flash2**
- bool **flash3**
- bool **flash4**
- bool **flash5**
- bool **flash6**
- bool **flash7**
- bool **flash8**
- bool **flash9**
- bool **flash10**
- bool **flash11**
- bool **flash12**
- bool **flash13**
- bool **flash14**
- bool **flash15**

9.293.1 Detailed Description

This class is an basic simple shape widget. It has been refactored from QESimpleShape and will become the bases for QESimpleShape.

9.293.2 Member Enumeration Documentation

9.293.2.1 enum QSimpleShape::TextFormats

Enumerator:

FixedText Use user specified fixed text (default).

StateSet Use one of the stae Set values.

PvText Use EPICS value agumented with units if selected.

9.293.3 Constructor & Destructor Documentation

9.293.3.1 QSimpleShape::QSimpleShape (QWidget * *parent* = 0)

Construction

9.293.4 Property Documentation

9.293.4.1 int QSimpleShape::edgeWidth [read, write]

Edge width - range 0 to 20 - default is 1. If edge width set to 0, then shape colour used for edge/boarder colour.

9.293.4.2 bool QSimpleShape::flash0 [read, write]

When the widget's state coresponds to N and flashN is set true, the widget will alternate its normal colour with the specified flashOffColour.

9.293.4.3 QEScanTimers::ScanRates QSimpleShape::flashRate [read, write]

Flash rate. The default value is Medium, i.e. 1Hz flash rate.

9.293.4.4 TextFormats QSimpleShape::format [read, write]

Nominated text format

9.293.4.5 int QSimpleShape::percentSize [read, write]

Breadth of serpentine line as a percentage of widget height or width depending on wheather horizontal or vertical orientatiion. Range 1 to 50, default 10% Only applies to serpentine items.

9.293.4.6 int QSimpleShape::semiCycles [read, write]

Number of semi-cycles - range 1 to 30 - default is 8. Only applies to serpentine items.

9.293.4.7 Shapes QSimpleShape::shape [read, write]

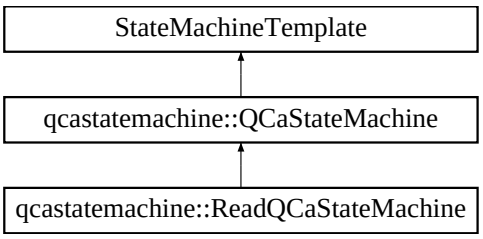
Nominated shape

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESimpleShape/QSimple
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QESimpleShape/QSimple

9.294 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.cpp

9.295 QE::WorkerManager::ReallyPrivate Class Reference

Public Attributes

- WorkerThread * **threadList** [MAXIMUM_THREADS]
- bool **workerComplete** [MAXIMUM_THREADS]

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.cpp

9.296 recording Class Reference

Signals

- void **byteArrayChanged** (const QByteArray &value, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)
- void **playingBack** (bool playing)

Public Member Functions

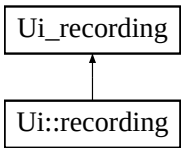
- **recording** (QWidget ∗parent=0)
- bool **isRecording** ()
- void **recordImage** (QByteArray image, unsigned long dataSize, QCaAlarmInfo &alarmInfo, QCaDateTime &time)
- void **nextFrameDue** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/recording.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_recording.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/recording.cpp

9.297 Ui::recording Class Reference

Inheritance diagram for Ui::recording::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_recording.h

9.298 PersistanceManager::ResourceLocker Class Reference

Public Member Functions

- **ResourceLocker** (const QString &)
- bool **tryLock** (int=0)

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceManager.cpp

9.299 QEArchiveInterface::ResponseValues Struct Reference

Public Attributes

- QString **pvName**
- double **displayLow**
- double **displayHigh**
- int **precision**
- QString **units**
- unsigned int **elementCount**
- QCaDataPointList **dataPoints**

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveInterface.h

9.300 imageDisplayProperties::rgbPixel Struct Reference

Public Attributes

- unsigned char **p** [4]

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/brightnessContrast.h

9.301 SaveRestoreSignal Class Reference

Public Types

- enum **saveRestoreOptions** { **SAVE**, **RESTORE_APPLICATION**, **RESTORE_QEFRAMEWORK** }

Signals

- void **saveRestore** (SaveRestoreSignal::saveRestoreOptions option)

Public Member Functions

- void **setOwner** (PersistanceManager ∗ownerIn)
- void save ()
- void restore ()

9.301.1 Member Function Documentation

9.301.1.1 void SaveRestoreSignal::restore ()

!! signal must be blocking

9.301.1.2 void SaveRestoreSignal::save ()

!! signal must be blocking

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_persistanceManager.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/persistanceMa

9.302 QETwinScaleSelectDialog::ScaleLimit Struct Reference

Public Attributes

- double **min**
- double **max**

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QETwinScaleSelectDialog.h

9.303 screenSelectDialog Class Reference

Public Types

- enum **screens** { **PRIMARY_SCREEN** = -3, **THIS_SCREEN** = -2, **ALL_SCREENS** = -1 }

Public Member Functions

- **screenSelectDialog** (int numScreens, QWidget ∗parent=0)
- int **getScreenNum** ()

Static Public Member Functions

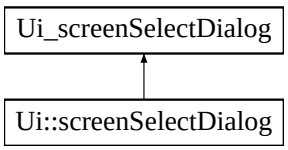
- static bool **getFullscreenGeometry** (QWidget ∗target, QRect &geom)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/screenSelectDi
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/screenSelectDi

9.304 Ui::screenSelectDialog Class Reference

Inheritance diagram for Ui::screenSelectDialog::



The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_screenSelectDialog.h

9.305 selectMenu Class Reference

Public Member Functions

- **selectMenu** (QWidget ∗parent=0)
- imageContextMenu::imageContextMenuOptions **getSelectOption** (const QPoint &pos)
- void **enable** (imageContextMenu::imageContextMenuOptions option, bool state)
- bool **isEnabled** (imageContextMenu::imageContextMenuOptions option)
- void **setChecked** (const int mode)
- void **setItemText** (imageContextMenu::imageContextMenuOptions option, QString title)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/selectMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/selectMenu.cpp

9.306 signalSlotHandler Class Reference

Public Slots

- void **saveRestore** (SaveRestoreSignal::saveRestoreOptions option)

Public Member Functions

- void **setOwner** (QEWidget ∗ownerIn)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidget.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/QEWidget.cpp

9.307 SourceSpec Class Reference

Public Attributes

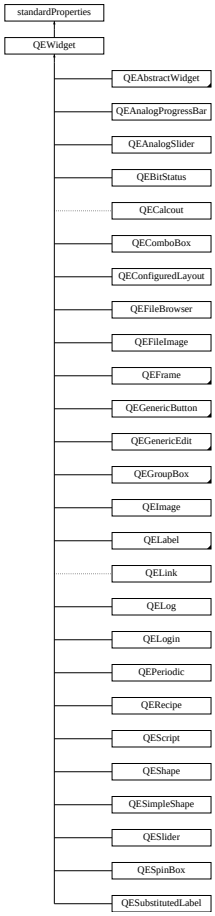
- QEArchiveInterface * **interface**
- QHash< int, KeyTimeSpec > **keyToTimeSpecLookUp**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.h

9.308 standardProperties Class Reference

Inheritance diagram for standardProperties::



Public Types

- enum **displayAlarmStateOptions** { **DISPLAY_ALARM_STATE_NEVER**, **DISPLAY_ALARM_STATE_ALWAYS**, **DISPLAY_ALARM_STATE_-WHEN_IN_ALARM** }

Public Member Functions

- **standardProperties** (QWidget ∗ownerIn)
- userLevelTypes::userLevels **getUserLevelVisibility** ()
- void **setUserLevelVisibility** (userLevelTypes::userLevels level)
- userLevelTypes::userLevels **getUserLevelEnabled** ()
- void **setUserLevelEnabled** (userLevelTypes::userLevels level)

- bool **getApplicationEnabled** () const
- void **setApplicationEnabled** (bool state)
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **setDisplayAlarmState** (bool displayAlarmStateIn)
- bool **getDisplayAlarmState** ()
- void **setDisplayAlarmStateOption** (displayAlarmStateOptions displayAlarmStateIn)
- displayAlarmStateOptions **getDisplayAlarmStateOption** ()

Protected Member Functions

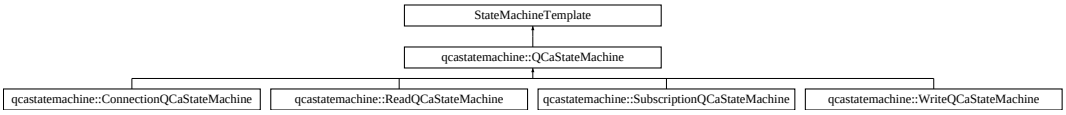
- void **checkVisibilityEnabledLevel** (userLevelTypes::userLevels level)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/standardPrope
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/standardPrope

9.309 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h

9.310 QEArchiveAccess::Status Struct Reference

Public Attributes

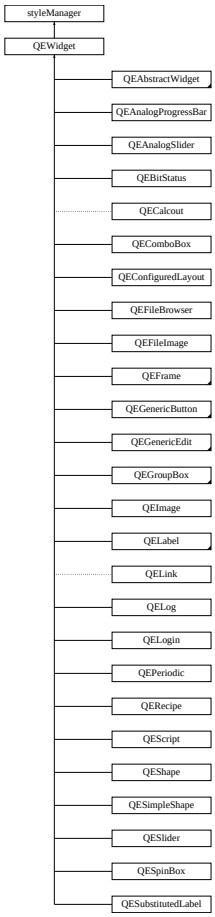
- QString **hostName**
- int **portNumber**
- QString **endPoint**
- QEArchiveInterface::States **state**
- int **available**
- int **read**
- int **numberPVs**
- int **pending**

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveAccess.h

9.311 styleManager Class Reference

`#include <styleManager.h>`Inheritance diagram for styleManager::



Public Member Functions

- **styleManager** (QWidget *ownerIn)
- void setStyleDefault (QString style)
- QString getStyleDefault () const
- void setStyleUser (QString style)
- QString getStyleUser () const
- void setStyleScientist (QString style)
- QString getStyleScientist () const
- void setStyleEngineer (QString style)
- QString getStyleEngineer () const
- void updateDataStyle (QString style)
- void updateStatusStyle (QString style)

- void updatePropertyStyle (QString style)

  For example, a style string is used to set QE button text alignment.

- void updateConnectionStyle (bool connected)
- void styleUserLevelChanged (userLevelTypes::userLevels levelIn)

Friends

- class changeEventFilter

9.311.1 Detailed Description

This class adds common style support to all QE widgets if required.

Standard properties for all QE widgets specify a style to be applied for user, scientist, and engineer mode. Also QE widgets can specify data or status related style changes.

The syntax for all Style Sheet strings used by this class is the standard Qt Style Sheet syntax. For example, 'background-color: red'. Refer to Qt Style Sheets Reference for full details.

Note, as well as the large number of defined Style Sheet properties, the Style Sheet syntax allows setting any widget property using the 'qproperty' property. For example 'qproperty-geometry:rect(10 10 100 100);' Caution, any amount of weird behaviour can be effected using Style Sheet strings. Without carefull consideration they should only be used for simple visualisation effects such as altering the backgrouond color.

All QE widgets (eg, QELabel, QELineEdit) have an instance of this class as they based on QEWidget which itself uses this class as a base class.

To use the functionality provided by this class, QE widgets must include the following:

- Q_PROPERTY(QString styleSheet READ styleSheet WRITE setStyleSheet DESIGNABLE false)
- Q_PROPERTY(QString defaultStyle READ getStyleDefault WRITE setStyleDefault)
- Q_PROPERTY(QString userLevelUserStyle READ getStyleUser WRITE setStyleUser)
- Q_PROPERTY(QString userLevelScientistStyle READ getStyleScientist WRITE setStyleScientist)
- Q_PROPERTY(QString userLevelEngineerStyle READ getStyleEngineer WRITE setStyleEngineer)

To use this class to manage style changes related to data and status the QE widget must include the following:

- On presenation of new data call updateDataStyle() with the Style Sheet string related to the new data (if any).

- On change of data status, call updateStatusStyle() with the Style Sheet string related to the new status (if any).

Note, this class notes the initial style when instantiated and uses that style as the base style for all style changes. This means any style changes not performed through this class will be lost the next time this class changes the style.

Note, the stylesheet built by this class is not actually applied if the widget being managed is disabled. Instead it is noted and applied if and when the widget becomed enabled. Changes that affect the style will still cause a regeneration of the style while the widget is disabled, but the updated style will not be applied until the widget is enabled.

9.311.2 Member Function Documentation

9.311.2.1 QString styleManager::getStyleDefault () const

Get the default Style Sheet string.

9.311.2.2 QString styleManager::getStyleEngineer () const

Get the Style Sheet string to be applied when the widget is displayed in 'Engineer' mode.

9.311.2.3 QString styleManager::getStyleScientist () const

Get the Style Sheet string to be applied when the widget is displayed in 'Scientist' mode.

9.311.2.4 QString styleManager::getStyleUser () const

Get the Style Sheet string to be applied when the widget is displayed in 'User' mode.

9.311.2.5 void styleManager::setStyleDefault (QString *style*)

Set the default Style Sheet string. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'.

9.311.2.6 void styleManager::setStyleEngineer (QString *style*)

Set the Style Sheet string to be applied when the widget is displayed in 'Engineer' mode. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'

9.311.2.7 void styleManager::setStyleScientist (QString *style*)

Set the Style Sheet string to be applied when the widget is displayed in 'Scientist' mode. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'

9.311.2.8 void styleManager::setStyleUser (QString *style*)

Set the Style Sheet string to be applied when the widget is displayed in 'User' mode. The syntax is the standard Qt Style Sheet syntax. For example, 'background-color: red'.

9.311.2.9 void styleManager::styleUserLevelChanged (userLevelTypes::userLevels *levelIn*)

Set the current user level.

9.311.2.10 void styleManager::updateConnectionStyle (bool *connected*)

Set the Style Sheet string to be applied to reflect the current connection state (connected or disconnected) of the current data. For example, a disconnected value may be greyed out.

9.311.2.11 void styleManager::updateDataStyle (QString *style*)

Set the Style Sheet string to be applied to reflect an aspect of the current data. For example, a value over a high limit may be displayed in red.

9.311.2.12 void styleManager::updatePropertyStyle (QString *style*)

For example, a style string is used to set QE button text alignment. Set the Style Sheet string to be applied to implement a widget property.

9.311.2.13 void styleManager::updateStatusStyle (QString *style*)

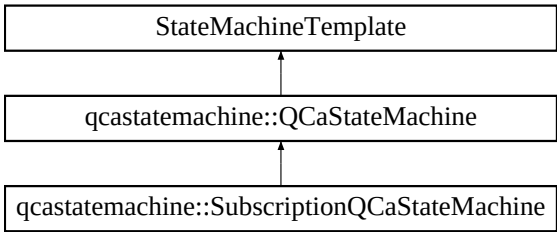
Set the Style Sheet string to be applied to reflect an aspect of the current status. For example, invalid data may be displayed with a white background.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/styleManager.
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/styleManager.

9.312 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.cpp

9.313 QEPlot::Trace Class Reference

Public Attributes

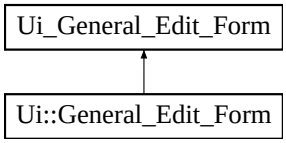
- QVector< QCaDateTime > **timeStamps**
- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve * **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**
- bool **hasCurrentPoint**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPlot/QEPlot.cpp

9.314 Ui_General_Edit_Form Class Reference

Inheritance diagram for Ui_General_Edit_Form::



Public Member Functions

- void **setupUi** (QWidget ∗General_Edit_Form)
- void **retranslateUi** (QWidget ∗General_Edit_Form)

Public Attributes

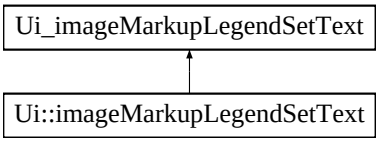
- QVBoxLayout ∗ **verticalLayout**
- QFrame ∗ **frame**
- QVBoxLayout ∗ **verticalLayout_2**
- QLabel ∗ **pvNameLabel**
- QELabel ∗ **valueLabel**
- QWidget ∗ **enumerationEditPanel**
- QHBoxLayout ∗ **horizontalLayout_3**
- QERadioGroup ∗ **radioGroupWidget**
- QFrame ∗ **numericEditPanel**
- QHBoxLayout ∗ **horizontalLayout_5**
- QENumericEdit ∗ **numericEditWidget**
- QPushButton ∗ **numericEditApplyButton**
- QLabel ∗ **label**
- QNumericEdit ∗ **zerosEdit**
- QLabel ∗ **label_2**
- QNumericEdit ∗ **precisionEdit**
- QWidget ∗ **stringEditPanel**
- QHBoxLayout ∗ **horizontalLayout_4**
- QELineEdit ∗ **stringEditWidget**
- QPushButton ∗ **stringEditApplyButton**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEGeneralEdit.h

9.315 Ui_imageMarkupLegendSetText Class Reference

Inheritance diagram for Ui_imageMarkupLegendSetText::



Public Member Functions

- void **setupUi** (QDialog ∗imageMarkupLegendSetText)
- void **retranslateUi** (QDialog ∗imageMarkupLegendSetText)

Public Attributes

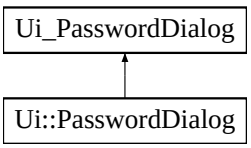
- QVBoxLayout ∗ **verticalLayout**
- QHBoxLayout ∗ **horizontalLayout**
- QLabel ∗ **label**
- QLineEdit ∗ **lineEditLegend**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_imageMarkupLegendSetText.h

9.316 Ui_PasswordDialog Class Reference

Inheritance diagram for Ui_PasswordDialog::



Public Member Functions

- void **setupUi** (QDialog ∗PasswordDialog)
- void **retranslateUi** (QDialog ∗PasswordDialog)

Public Attributes

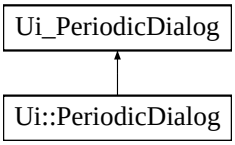
- QVBoxLayout ∗ **verticalLayout**
- QGridLayout ∗ **gridLayout**
- QLineEdit ∗ **lineEditUser**
- QLabel ∗ **label**
- QLabel ∗ **label_2**
- QLineEdit ∗ **lineEditScientist**
- QLabel ∗ **label_3**
- QLineEdit ∗ **lineEditEngineer**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_PasswordDialog.h

9.317 Ui_PeriodicDialog Class Reference

Inheritance diagram for Ui_PeriodicDialog::



Public Member Functions

- void **setupUi** (QDialog ∗PeriodicDialog)
- void **retranslateUi** (QDialog ∗PeriodicDialog)

Public Attributes

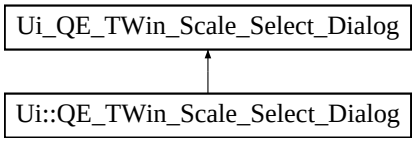
- QVBoxLayout ∗ **verticalLayout**
- QGridLayout ∗ **periodicGridLayout**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_PeriodicDialog.h

9.318 Ui_QE_TWin_Scale_Select_Dialog Class Reference

Inheritance diagram for Ui_QE_TWin_Scale_Select_Dialog::



Public Member Functions

- void **setupUi** (QWidget ∗QE_TWin_Scale_Select_Dialog)
- void **retranslateUi** (QWidget ∗QE_TWin_Scale_Select_Dialog)

Public Attributes

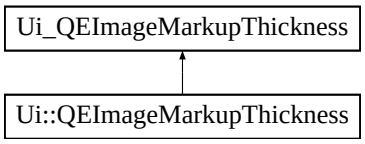
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_2**
- QLineEdit ∗ **scaleTwoMaxEdit**
- QLabel ∗ **scaleTwoNameLabel**
- QLineEdit ∗ **scaleTwoMinEdit**
- QLabel ∗ **scaleOneNameLabel**
- QLineEdit ∗ **scaleOneMinEdit**
- QLineEdit ∗ **scaleOneMaxEdit**
- QDialogButtonBox ∗ **buttonBox**
- QLabel ∗ **label**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QETwinScaleSelectDialog.h

9.319 Ui_QEImageMarkupThickness Class Reference

Inheritance diagram for Ui_QEImageMarkupThickness::



Public Member Functions

- void **setupUi** (QDialog ∗QEImageMarkupThickness)
- void **retranslateUi** (QDialog ∗QEImageMarkupThickness)

Public Attributes

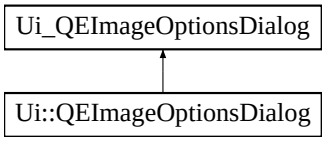
- QVBoxLayout ∗ **verticalLayout**
- QSpinBox ∗ **spinBox**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEImageMarkupThickness.h

9.320 Ui_QEImageOptionsDialog Class Reference

Inheritance diagram for Ui_QEImageOptionsDialog::



Public Member Functions

- void **setupUi** (QDialog ∗QEImageOptionsDialog)
- void **retranslateUi** (QDialog ∗QEImageOptionsDialog)

Public Attributes

- QVBoxLayout ∗ **verticalLayout_3**
- QHBoxLayout ∗ **horizontalLayout**
- QGroupBox ∗ **groupBox**
- QVBoxLayout ∗ **verticalLayout**
- QCheckBox ∗ **checkBoxVerticalProfile1**
- QCheckBox ∗ **checkBoxVerticalProfile2**
- QCheckBox ∗ **checkBoxVerticalProfile3**
- QCheckBox ∗ **checkBoxVerticalProfile4**
- QCheckBox ∗ **checkBoxVerticalProfile5**
- QCheckBox ∗ **checkBoxHorizontalProfile1**
- QCheckBox ∗ **checkBoxHorizontalProfile2**
- QCheckBox ∗ **checkBoxHorizontalProfile3**
- QCheckBox ∗ **checkBoxHorizontalProfile4**
- QCheckBox ∗ **checkBoxHorizontalProfile5**
- QCheckBox ∗ **checkBoxArbitraryProfile**
- QCheckBox ∗ **checkBoxArea1Selection**
- QCheckBox ∗ **checkBoxArea2Selection**
- QCheckBox ∗ **checkBoxArea3Selection**
- QCheckBox ∗ **checkBoxArea4Selection**
- QCheckBox ∗ **checkBoxTarget**
- QSpacerItem ∗ **verticalSpacer_2**
- QCheckBox ∗ **checkBoxBeam**
- QGroupBox ∗ **groupBox_2**
- QVBoxLayout ∗ **verticalLayout_2**
- QCheckBox ∗ **checkBoxButtonBar**
- QCheckBox ∗ **checkBoxBrightnessContrast**
- QCheckBox ∗ **checkBoxRecorder**
- QCheckBox ∗ **checkBoxTime**

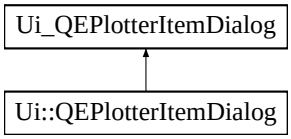
- QCheckBox * **checkBoxInfo**
- QSpacerItem * **verticalSpacer**
- QDialogButtonBox * **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEImageOptionsDialog.h

9.321 Ui_QEPlotterItemDialog Class Reference

Inheritance diagram for Ui_QEPlotterItemDialog::



Public Member Functions

- void **setupUi** (QDialog ∗QEPlotterItemDialog)
- void **retranslateUi** (QDialog ∗QEPlotterItemDialog)

Public Attributes

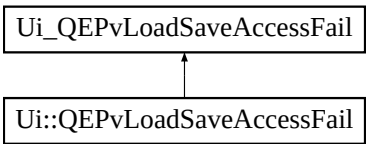
- QVBoxLayout ∗ **verticalLayout**
- QGridLayout ∗ **gridLayout**
- QLabel ∗ **label_2**
- QLineEdit ∗ **dataEdit**
- QLabel ∗ **label**
- QLabel ∗ **label_3**
- QLineEdit ∗ **aliasEdit**
- QLabel ∗ **label_5**
- QLabel ∗ **label_4**
- QLineEdit ∗ **sizeEdit**
- QLabel ∗ **label_6**
- QHBoxLayout ∗ **horizontalLayout**
- QSpacerItem ∗ **horizontalSpacer_2**
- QPushButton ∗ **clearButton**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPlotterItemDialog.h

9.322 Ui_QEPvLoadSaveAccessFail Class Reference

Inheritance diagram for Ui_QEPvLoadSaveAccessFail::



Public Member Functions

- void **setupUi** (QWidget ∗QEPvLoadSaveAccessFail)
- void **retranslateUi** (QWidget ∗QEPvLoadSaveAccessFail)

Public Attributes

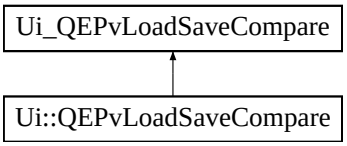
- QVBoxLayout ∗ **verticalLayout**
- QFrame ∗ **frame**
- QVBoxLayout ∗ **verticalLayout_2**
- QTextEdit ∗ **textEdit**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveAccessFail.h

9.323 Ui_QEPvLoadSaveCompare Class Reference

Inheritance diagram for Ui_QEPvLoadSaveCompare::



Public Member Functions

- void **setupUi** (QWidget ∗QEPvLoadSaveCompare)
- void **retranslateUi** (QWidget ∗QEPvLoadSaveCompare)

Public Attributes

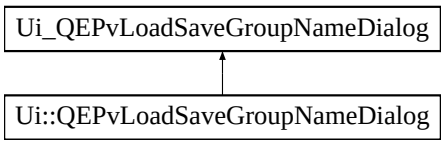
- QVBoxLayout ∗ **verticalLayout**
- QFrame ∗ **frame**
- QVBoxLayout ∗ **verticalLayout_2**
- QEHistogram ∗ **comparison**
- QAnalogSlider ∗ **scale_slider**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveCompare.h

9.324 Ui_QEPvLoadSaveGroupNameDialog Class Reference

Inheritance diagram for Ui_QEPvLoadSaveGroupNameDialog::



Public Member Functions

- void **setupUi** (QWidget ∗QEPvLoadSaveGroupNameDialog)
- void **retranslateUi** (QWidget ∗QEPvLoadSaveGroupNameDialog)

Public Attributes

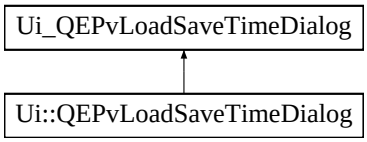
- QVBoxLayout ∗ **verticalLayout_2**
- QVBoxLayout ∗ **verticalLayout**
- QHBoxLayout ∗ **horizontalLayout_4**
- QLabel ∗ **label_2**
- QLineEdit ∗ **groupEdit**
- QHBoxLayout ∗ **horizontalLayout_2**
- QSpacerItem ∗ **horizontalSpacer_2**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveGroupNameDialog.h

9.325 Ui_QEPvLoadSaveTimeDialog Class Reference

Inheritance diagram for Ui_QEPvLoadSaveTimeDialog::



Public Member Functions

- void **setupUi** (QWidget ∗QEPvLoadSaveTimeDialog)
- void **retranslateUi** (QWidget ∗QEPvLoadSaveTimeDialog)

Public Attributes

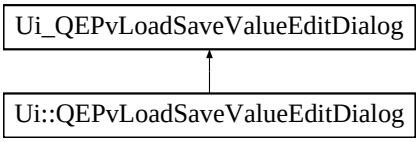
- QVBoxLayout ∗ **verticalLayout_2**
- QFrame ∗ **frame**
- QVBoxLayout ∗ **verticalLayout**
- QCalendarWidget ∗ **dateCalendar**
- QSlider ∗ **timeSlider**
- QTimeEdit ∗ **timeEdit**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveTimeDialog.h

9.326 Ui_QEPvLoadSaveValueEditDialog Class Reference

Inheritance diagram for Ui_QEPvLoadSaveValueEditDialog::



Public Member Functions

- void **setupUi** (QWidget *QEPvLoadSaveValueEditDialog)
- void **retranslateUi** (QWidget *QEPvLoadSaveValueEditDialog)

Public Attributes

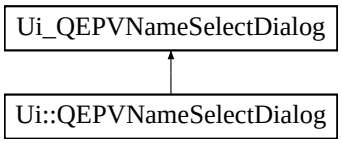
- QVBoxLayout * **verticalLayout_2**
- QVBoxLayout * **verticalLayout**
- QLabel * **nameLabel**
- QHBoxLayout * **horizontalLayout_3**
- QLabel * **fixed_label_3**
- QSpinBox * **elementIndexEdit**
- QSpacerItem * **horizontalSpacer_2**
- QLabel * **fixed_label_4**
- QSpinBox * **numberElementsEdit**
- QHBoxLayout * **horizontalLayout_2**
- QLabel * **fixed_label_2**
- QLineEdit * **valueEdit**
- QHBoxLayout * **horizontalLayout**
- QSpacerItem * **horizontalSpacer**
- QDialogButtonBox * **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPvLoadSaveValueEditDialog.h

9.327 Ui_QEPVNameSelectDialog Class Reference

Inheritance diagram for Ui_QEPVNameSelectDialog::



Public Member Functions

- void **setupUi** (QWidget *QEPVNameSelectDialog)
- void **retranslateUi** (QWidget *QEPVNameSelectDialog)

Public Attributes

- QVBoxLayout * **verticalLayout**
- QFrame * **frame_1**
- QHBoxLayout * **horizontalLayout**
- QLabel * **label_2**
- QLineEdit * **filterEdit**
- QPushButton * **helpButton**
- QFrame * **frame_2**
- QHBoxLayout * **horizontalLayout_2**
- QLabel * **label**
- QComboBox * **pvNameEdit**
- QFrame * **frame_3**
- QHBoxLayout * **horizontalLayout_3**
- QLabel * **label_4**
- QLabel * **matchCountLabel**
- QSpacerItem * **horizontalSpacer**
- QPushButton * **clearButton**
- QDialogButtonBox * **buttonBox**
- QFrame * **help_frame**
- QFrame * **frame_5**
- QGridLayout * **gridLayout**
- QLabel * **label_3**
- QLabel * **label_5**
- QLabel * **label_6**
- QLabel * **label_7**
- QLabel * **label_8**
- QLabel * **label_9**
- QLabel * **label_10**
- QLabel * **label_11**

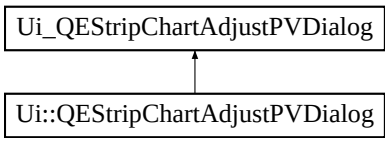
- QLabel * **label_12**
- QLabel * **label_13**
- QLabel * **label_14**
- QLabel * **label_15**
- QLabel * **label_16**
- QLabel * **label_17**
- QLabel * **label_18**
- QLabel * **label_19**
- QLabel * **label_20**
- QLabel * **label_22**
- QLabel * **label_21**
- QLabel * **label_23**
- QLabel * **label_24**
- QLabel * **label_25**
- QLabel * **label_26**
- QLabel * **label_27**
- QLabel * **label_28**
- QLabel * **label_29**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEPVNameSelectDialog.h

9.328 Ui_QEStripChartAdjustPVDialog Class Reference

Inheritance diagram for Ui_QEStripChartAdjustPVDialog::



Public Member Functions

- void **setupUi** (QWidget ∗QEStripChartAdjustPVDialog)
- void **retranslateUi** (QWidget ∗QEStripChartAdjustPVDialog)

Public Attributes

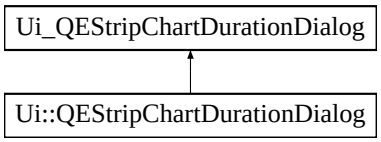
- QHBoxLayout ∗ **horizontalLayout_6**
- QVBoxLayout ∗ **verticalLayout**
- QLabel ∗ **title**
- QHBoxLayout ∗ **horizontalLayout**
- QLabel ∗ **label**
- QLineEdit ∗ **originEdit**
- QHBoxLayout ∗ **horizontalLayout_2**
- QLabel ∗ **label_2**
- QLineEdit ∗ **slopeEdit**
- QHBoxLayout ∗ **horizontalLayout_5**
- QLabel ∗ **label_3**
- QLineEdit ∗ **offsetEdit**
- QHBoxLayout ∗ **horizontalLayout_4**
- QSpacerItem ∗ **horizontalSpacer_2**
- QPushButton ∗ **loprHoprButton**
- QPushButton ∗ **plottedButton**
- QPushButton ∗ **bufferedButton**
- QHBoxLayout ∗ **horizontalLayout_3**
- QSpacerItem ∗ **horizontalSpacer**
- QPushButton ∗ **resetButton**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartAdjustPVDialog.h

9.329 Ui_QEStripChartDurationDialog Class Reference

Inheritance diagram for Ui_QEStripChartDurationDialog::



Public Member Functions

- void **setupUi** (QWidget ∗QEStripChartDurationDialog)
- void **retranslateUi** (QWidget ∗QEStripChartDurationDialog)

Public Attributes

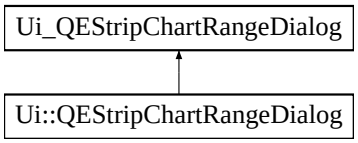
- QVBoxLayout ∗ **verticalLayout**
- QHBoxLayout ∗ **horizontalLayout_1**
- QSpinBox ∗ **spinBox**
- QTimeEdit ∗ **endTimeEdit**
- QHBoxLayout ∗ **horizontalLayout_2**
- QLabel ∗ **duration**
- QSpacerItem ∗ **horizontalSpacer**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartDurationDialog.h

9.330 Ui_QEStripChartRangeDialog Class Reference

Inheritance diagram for Ui_QEStripChartRangeDialog::



Public Member Functions

- void **setupUi** (QWidget ∗QEStripChartRangeDialog)
- void **retranslateUi** (QWidget ∗QEStripChartRangeDialog)

Public Attributes

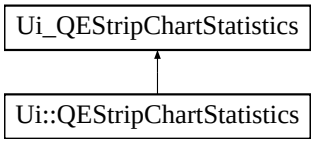
- QVBoxLayout ∗ **verticalLayout**
- QHBoxLayout ∗ **horizontalLayout**
- QLabel ∗ **label**
- QLabel ∗ **label_2**
- QHBoxLayout ∗ **horizontalLayout_2**
- QLineEdit ∗ **minimumEdit**
- QLineEdit ∗ **maximumEdit**
- QHBoxLayout ∗ **horizontalLayout_3**
- QSpacerItem ∗ **horizontalSpacer**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartRangeDialog.h

9.331 Ui_QEStripChartStatistics Class Reference

Inheritance diagram for Ui_QEStripChartStatistics::



Public Member Functions

- void **setupUi** (QWidget ∗QEStripChartStatistics)
- void **retranslateUi** (QWidget ∗QEStripChartStatistics)

Public Attributes

- QVBoxLayout ∗ **verticalLayout**
- QHBoxLayout ∗ **horizontalLayout**
- QLabel ∗ **label**
- QLabel ∗ **pvNameLabel**
- QHBoxLayout ∗ **horizontalLayout_2**
- QLabel ∗ **label_3**
- QLabel ∗ **startTimeLabel**
- QLabel ∗ **label_5**
- QLabel ∗ **endTimeLabel**
- QHBoxLayout ∗ **horizontalLayout_3**
- QLabel ∗ **label_7**
- QLabel ∗ **numberOfPointsLabel**
- QLabel ∗ **label_9**
- QLabel ∗ **validPointsLabel**
- QLabel ∗ **label_11**
- QLabel ∗ **durationLabel**
- QSpacerItem ∗ **verticalSpacer**
- QHBoxLayout ∗ **horizontalLayout_4**
- QLabel ∗ **label_13**
- QLabel ∗ **meanLabel**
- QSpacerItem ∗ **horizontalSpacer**
- QLabel ∗ **label_15**
- QLabel ∗ **meanRateOfChangeLabel**
- QHBoxLayout ∗ **horizontalLayout_5**
- QLabel ∗ **label_19**
- QLabel ∗ **minimumLabel**
- QSpacerItem ∗ **horizontalSpacer_2**
- QLabel ∗ **label_20**

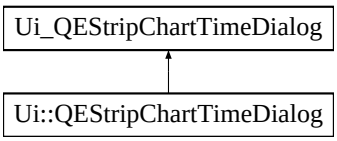
- QLabel * **standardDeviationLabel**
- QHBoxLayout * **horizontalLayout_6**
- QLabel * **label_23**
- QLabel * **maximumLabel**
- QSpacerItem * **horizontalSpacer_3**
- QLabel * **label_24**
- QLabel * **areaUnderCurveLabel**
- QHBoxLayout * **horizontalLayout_7**
- QLabel * **label_27**
- QLabel * **minMaxDiffLabel**
- QSpacerItem * **horizontalSpacer_4**
- QLabel * **label_28**
- QLabel * **firstLastDiffLabel**
- QHBoxLayout * **horizontalLayout_8**
- QSpacerItem * **horizontalSpacer_5**
- QPushButton * **updateButton**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartStatistics.h

9.332 Ui_QEStripChartTimeDialog Class Reference

Inheritance diagram for Ui_QEStripChartTimeDialog::



Public Member Functions

- void **setupUi** (QWidget *QEStripChartTimeDialog)
- void **retranslateUi** (QWidget *QEStripChartTimeDialog)

Public Attributes

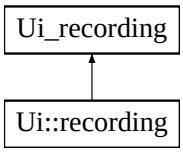
- QWidget * **layoutWidget**
- QVBoxLayout * **verticalLayout_3**
- QHBoxLayout * **horizontalLayout**
- QVBoxLayout * **verticalLayout**
- QLabel * **label**
- QCalendarWidget * **startDate**
- QSlider * **startTimeSlider**
- QTimeEdit * **startTimeEdit**
- QVBoxLayout * **verticalLayout_2**
- QLabel * **label_2**
- QCalendarWidget * **endDate**
- QSlider * **endTimeSlider**
- QTimeEdit * **endTimeEdit**
- QHBoxLayout * **horizontalLayout_2**
- QLabel * **duration**
- QSpacerItem * **horizontalSpacer**
- QDialogButtonBox * **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_QEStripChartTimeDialog.h

9.333 Ui_recording Class Reference

Inheritance diagram for Ui_recording::



Public Member Functions

- void **setupUi** (QWidget *recording)
- void **retranslateUi** (QWidget *recording)

Public Attributes

- QHBoxLayout * **horizontalLayout_4**
- QFrame * **frame**
- QVBoxLayout * **verticalLayout_5**
- QRadioButton * **radioButtonLive**
- QRadioButton * **radioButtonPlayback**
- QVBoxLayout * **verticalLayout_4**
- QGroupBox * **groupBoxLive**
- QHBoxLayout * **horizontalLayout_2**
- QPushButton * **pushButtonRecord**
- QPushButton * **pushButtonClear**
- QLabel * **labelImageCountRecord**
- QSpinBox * **spinBoxMaxImages**
- QVBoxLayout * **verticalLayout_2**
- QRadioButton * **radioButtonDiscardOldest**
- QRadioButton * **radioButtonStopAtLimit**
- QGroupBox * **groupBoxPlayback**
- QVBoxLayout * **verticalLayout_3**
- QHBoxLayout * **horizontalLayout_3**
- QPushButton * **pushButtonPlay**
- QDoubleSpinBox * **doubleSpinBoxPlaybackRate**
- QCheckBox * **checkBoxLoop**
- QHBoxLayout * **horizontalLayout**
- QPushButton * **pushButtonFirstImage**
- QPushButton * **pushButtonPreviousImage**
- QSlider * **horizontalSliderPosition**
- QPushButton * **pushButtonNextImage**
- QPushButton * **pushButtonLastImage**
- QLabel * **labelImageCountPlayback**

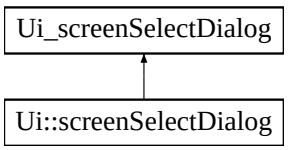
- QSpacerItem * **horizontalSpacer**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_recording.h

9.334 Ui_screenSelectDialog Class Reference

Inheritance diagram for Ui_screenSelectDialog::



Public Member Functions

- void **setupUi** (QDialog ∗screenSelectDialog)
- void **retranslateUi** (QDialog ∗screenSelectDialog)

Public Attributes

- QVBoxLayout ∗ **verticalLayout**
- QLabel ∗ **label**
- QComboBox ∗ **comboBox**
- QDialogButtonBox ∗ **buttonBox**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/ui/ui_screenSelectDialog.h

9.335 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.h

9.336 QEPeriodic::userInfoStructArray Struct Reference

Public Attributes

- userInfoStruct **array** [NUM_ELEMENTS]

The documentation for this struct was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEPeriodic/QEPeriodic.h

9.337 userLevelSignal Class Reference

Signals

- void userChanged (userLevelTypes::userLevels level)

  Internal use only. Send when the user level has changed.

Public Member Functions

- void **setLevel** (userLevelTypes::userLevels levelIn)
- userLevelTypes::userLevels **getLevel** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_ContainerProfile.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf

9.338 userLevelSlot Class Reference

Public Slots

- void **userChanged** (userLevelTypes::userLevels level)

Public Member Functions

- void **setOwner** (ContainerProfile *ownerIn)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.cpp

9.339 userLevelTypes Class Reference

Public Types

- enum userLevels { USERLEVEL_USER, USERLEVEL_SCIENTIST, USERLEVEL_ENGINEER }

9.339.1 Member Enumeration Documentation

9.339.1.1 enum userLevelTypes::userLevels

User levels set by widgets such as QELogin and used by many widgets to determine visibility, enabled state, and style.

Enumerator:

USERLEVEL_USER User level - least privilaged.

USERLEVEL_SCIENTIST User level - more privilaged than user, less than engineer.

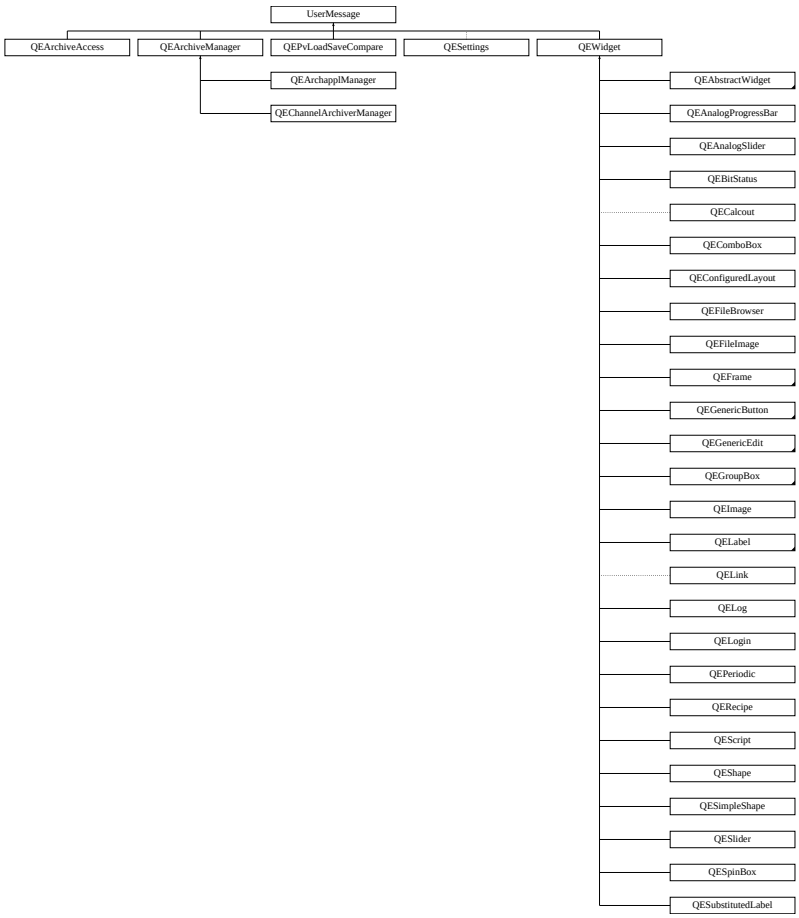
USERLEVEL_ENGINEER User level - most privilaged.

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProf

9.340 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 QEForm 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 () const

  Get the source ID (the ID set up by the GUI designer, usually matched to the source ID of logging widgets.

- unsigned int getFormId () const

  Get the form ID (the the same ID for all sibling widgets within an QEForm widget).

- message_filter_options getFormFilter () const

  Get the message filtering applied to the form ID.

- message_filter_options getSourceFilter () const

  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 () const

  Get the for ID of all widgets that are children of this widget.

- unsigned int getNextMessageFormId () const

  Generate a new form ID for all widgets in a new form.

- void sendMessage (QString message, message_types type=message_types(MESSAGE_TYPE_INFO))

  Send a message to the user.

- void sendMessage (QString message, QString source, message_types type=message_types(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 (typicaly logging widgets or application windows).

Friends

- class UserMessageSlot
- class UserMessageSignal

9.340.1 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 can base a class on the UserMessage class and set up filtering to receive all messages. An application with log messages for seperate 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.cpp

9.341 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 recieve 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.

9.341.1 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_UserMessage.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.

9.342 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.

9.342.1 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/UserMessage.cpp

9.343 ValueScaling Class Reference

Public Member Functions

- void **reset** ()
- void **assign** (const ValueScaling &s)
- void **set** (const double dIn, const double mIn, const double cIn)
- void **get** (double &dOut, double &mOut, double &cOut) const
- void **map** (const double fromLower, const double fromUpper, const double toLower, const double toUpper)
- bool **isScaled** () const
- double **value** (const double x) const
- QEDisplayRanges **value** (const QEDisplayRanges &x) const
- void **saveConfiguration** (PMElement &parentElement) const
- void **restoreConfiguration** (PMElement &parentElement)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEStripChart/QEStripCh

9.344 ValuesResponseContext Class Reference

Public Member Functions

- **ValuesResponseContext** (const QEArchiveAccess *archiveAccessIn, const QString &pvNameIn, QObject *userDataIn)

Public Attributes

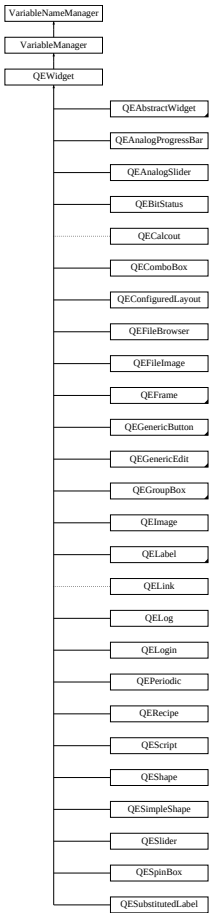
- const QEArchiveAccess * **archiveAccess**
- QString **pvName**
- QObject * **userData**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/archive/QEArchiveManager.cpp

9.345 VariableManager Class Reference

`#include <VariableManager.h>`Inheritance diagram for VariableManager::



Public Member Functions

- VariableManager ()

  Constructor.

- virtual ~VariableManager ()

  Destructor.

- void activate ()
- void deactivate ()
- qcaobject::QCaObject * getQcaItem (unsigned int variableIndex) const
- void readNow ()
- virtual void writeNow ()

- int * getDisconnectedCountRef () const
- int * getConnectedCountRef () const

Protected Member Functions

- void setNumVariables (unsigned int numVariablesIn)

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.

- unsigned int getNumVariables () const

  Get the number of variables streaming data updates to the widget.

- qcaobject::QCaObject * createVariable (unsigned int variableIndex, const bool do_subscribe)
- virtual qcaobject::QCaObject * createQcaItem (unsigned int variableIndex)

  Function to create a appropriate superclass of QCaObject to stream data updates.

- virtual void establishConnection (unsigned int variableIndex)

  Create a CA connection and initiates updates if required.

- virtual void activated ()

  Do any post-all-widgets-constructed, i.e. activated stuff.

- virtual void deactivated ()

  Do any post deactivated stuff.

- void deleteQcaItem (unsigned int variableIndex, bool disconnect)

  Delete a stream of CA updates.

Protected Attributes

- bool subscribe

  Flag if data updates should be requested (default value used by QEWidget).

9.345.1 Detailed Description

This class manages CA process variables for the QEWidget class. The QEWidget class provides a lot of generic support for all QE widgets (such as QELabel, QESpinBox, etc), including managing CA data sources. It manages CA data sources through this class.

Refer to the QEwidget class description for an overview of how QE widgets use QEWidget.

The VariableManager class manages QCaObject classes that stream updates to the CA aware widgets. But the VariableManager class, however, doesn't know how to format the data, or how the updates will be used. To resolve this, the VariableManager class asks a parent class (such as QELabel) to create the QCaObject class in what ever flavour it wants, by calling the virtual function createQcaItem. 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.

After construction, a CA aware widget is activated (starts updating) by VariableManager calling its 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 contining plugin definitions. After loading the plugin widgets, code in the QEForm class calls the activate() function in the VariableManager base of this class (QEWiget). 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 createVariable() function 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.

9.345.2 Member Function Documentation

9.345.2.1 void VariableManager::activate ()

Initiate updates. Called after all configuration is complete. Note: This function invokes the virtual activated() function.

9.345.2.2 qcaobject::QCaObject * VariableManager::createVariable (unsigned int *variableIndex*, const bool *do_subscribe*) [protected]

Create a CA connection. do_subscribe indicated if updates should be requested. Return a QCaObject if successfull.

9.345.2.3 void VariableManager::deactivate ()

Terminates updates. This has been provided for third party (non QEGui) applications using the framework. Note: This function invokes the virtual deactivated() function.

9.345.2.4 int * VariableManager::getConnectedCountRef () const

Return references to the current count of connections. The plugin library (and therefore the static connection and disconnection counts) can be mapped twice (on Windows at least). So it is no use just referencing these static variables from an application if the widgets of interest have been created by the UI Loader. This function can be called on any widget loaded by the UI loader and the reference returned can be used to get counts for all widgets loaded by the UI loader.

9.345.2.5 int * VariableManager::getDisconnectedCountRef () const

Return references to the current count of disconnections. The plugin library (and therefore the static connection and disconnection counts) can be mapped twice (on Windows at least). So it is no use just referencing these static variables from an application if the widgets of interest have been created by the UI Loader. This function can be called on any widget loaded by the UI loader and the reference returned can be used to get counts for all widgets loaded by the UI loader.

9.345.2.6 qcaobject::QCaObject * VariableManager::getQcaItem (unsigned int *variableIndex*) const

Return a reference to one of the qCaObjects used to stream CA updates

9.345.2.7 void VariableManager::readNow ()

Perform a single shot read on all variables (Usefull when not subscribing by default)

9.345.2.8 void VariableManager::writeNow () [virtual]

(Control widgets only - such as QELineEdit) Write the value now. Used when writeOnChange, writeOnEnter, etc are all false

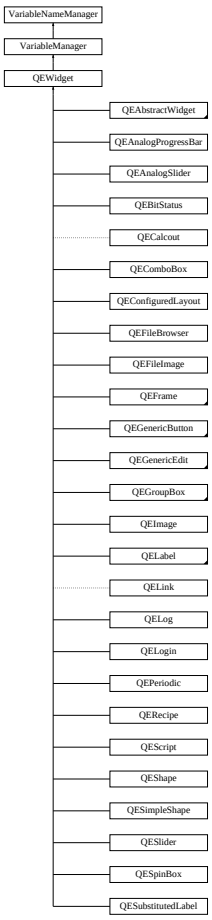
Reimplemented in QEAnalogSlider, QECheckBox, QEPushButton, QERadioButton, QEComboBox, QEGenericEdit, QENumericEdit, QESlider, and QESpinBox.

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/VariableMana
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/VariableMana

9.346 VariableNameManager Class Reference

Inheritance diagram for VariableNameManager::



Public Member Functions

- void **variableNameManagerInitialise** (unsigned int numVariables)
- int **getNumberVariables** () const
- QString **getOriginalVariableName** (unsigned int variableIndex) const
- QString **getVariableNameSubstitutions** () const
- QString **getSubstitutedVariableName** (unsigned int variableIndex) const
- void **setVariableNameSubstitutionsOverride** (const QString &substitutions)
- void **setVariableName** (const QString &variableName, unsigned int variableIndex)
- void **setVariableNameSubstitutions** (const QString &substitutions)
- QString substituteThis (const QString string) const

9.346.1 Member Function Documentation

9.346.1.1 QString VariableNameManager::substituteThis (const QString *string*) const

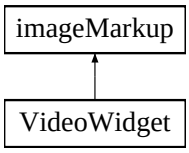
!! for efficiency, should this be done when substitutions are added or removed?? Build a list of keys and values...

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/VariableName
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/VariableName

9.347 VideoWidget Class Reference

Inheritance diagram for VideoWidget::



Signals

- void **userSelection** (imageMarkup::markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)
- void **zoomInOut** (int zoomAmount)
- void **currentPixelInfo** (QPoint pos)
- void **pan** (QPoint pos)
- void **redraw** ()

Public Member Functions

- **VideoWidget** (QWidget ∗parent=0)
- void **setNewImage** (QImage image, QCaDateTime &time)
- void **setPanning** (bool panningIn)
- bool **getPanning** ()
- QPoint **scalePoint** (QPoint pnt)
- int **scaleOrdinate** (int ord)
- QPoint **scaleImagePoint** (QPoint pnt)
- QRect **scaleImageRectangle** (QRect r)
- int **scaleImageOrdinate** (int ord)
- QImage **getImage** ()
- QSize **getImageSize** ()
- bool **hasCurrentImage** ()
- void **markupChange** ()

Protected Member Functions

- void **paintEvent** (QPaintEvent ∗)
- void **mousePressEvent** (QMouseEvent ∗event)
- void **mouseReleaseEvent** (QMouseEvent ∗event)
- void **mouseMoveEvent** (QMouseEvent ∗event)
- void **wheelEvent** (QWheelEvent ∗event)
- void **keyPressEvent** (QKeyEvent ∗event)
- void **markupChange** (QVector< QRect > &changedAreas)
- void **resizeEvent** (QResizeEvent ∗event)

- void **markupSetCursor** (QCursor cursor)
- void **markupAction** (markupIds mode, bool complete, bool clearing, QPoint point1, QPoint point2, unsigned int thickness)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/videowidget.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_videowidget.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/videowidget.cp

9.348 WidgetRef Class Reference

Public Member Functions

- **WidgetRef** (QEWidget *refIn)
- QEWidget * **getRef** ()

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEWidget/ContainerProfile.h

9.349 windowCreationListItem Class Reference

Public Member Functions

- **windowCreationListItem** (windowCreationListItem *item)

Public Attributes

- QString **uiFile**
- QString **macroSubstitutions**
- QString **customisationName**
- QEFormMapper::FormHandles **formHandle**
- QEActionRequests::Options **creationOption**
- bool **hidden**
- QString **title**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/QEActionRequests.cpp

9.350 windowCustomisation Class Reference

Public Member Functions

- **windowCustomisation** (const QString nameIn)
- void **addItem** (windowCustomisationMenuItem *menuItem, QStringList preMenuHierarchy=QStringList())
- void **addItem** (windowCustomisationButtonItem *button)
- QList< windowCustomisationMenuItem * > **getMenuItems** ()
- QList< windowCustomisationButtonItem * > **getButtons** ()
- QString **getName** ()

Static Public Member Functions

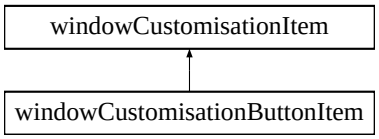
- static QEActionRequests::Options **translateCreationOption** (QString creationOption)

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cpp

9.351 windowCustomisationButtonItem Class Reference

Inheritance diagram for windowCustomisationButtonItem::



Public Member Functions

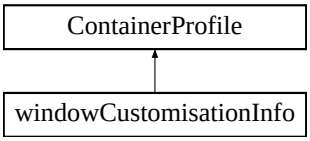
- **windowCustomisationButtonItem** (const QString buttonGroupIn, const QString buttonToolbarIn, const Qt::ToolBarArea buttonLocationIn, const QString buttonTextIn, const QString buttonIconIn, const QObject *launchRequestReceiver, const QList< windowCreationListItem > &windowsIn, const QString programIn, const QStringList argumentsIn)
- **windowCustomisationButtonItem** (const QString buttonGroupIn, const QString buttonToolbarIn, const Qt::ToolBarArea buttonLocationIn, const QString buttonTextIn, const QString buttonIconIn, const QString builtIn, const QString widgetNameIn)
- **windowCustomisationButtonItem** (const QString buttonGroupIn, const QString buttonToolbarIn, const Qt::ToolBarArea buttonLocationIn, const QString buttonTextIn, const QString buttonIconIn)
- **windowCustomisationButtonItem** (windowCustomisationButtonItem *buttonItem)
- QString **getButtonGroup** ()
- QString **getButtonToolbar** ()
- Qt::ToolBarArea **getButtonLocation** ()
- QString **getButtonText** ()
- QString **getButtonIcon** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cp

9.352 windowCustomisationInfo Class Reference

Inheritance diagram for windowCustomisationInfo::



Public Member Functions

- void **userLevelChangedGeneral** (userLevelTypes::userLevels)

Public Attributes

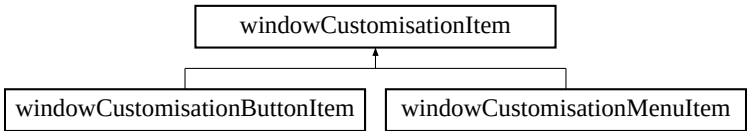
- QMap< QString, QMenu * > **placeholderMenus**
- QMap< QString, QMenu * > **menus**
- QMap< QString, QToolBar * > **toolbars**
- QList< windowCustomisationItem * > **items**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cpp

9.353 windowCustomisationItem Class Reference

Inheritance diagram for windowCustomisationItem::



Public Slots

- void **itemAction** ()

Signals

- void **newGui** (const QEActionRequests &request)

Public Member Functions

- **windowCustomisationItem** (const QObject *launchRequestReceiver, const QList< windowCreationListItem > &windowsIn, const QString programIn, const QStringList argumentsIn)
- **windowCustomisationItem** (windowCustomisationItem *item)
- **windowCustomisationItem** (const QString builtInActionIn)
- **windowCustomisationItem** (const QString builtInActionIn, const QString widgetNameIn)
- **windowCustomisationItem** (const QString dockTitleIn, bool unused)
- void **commonInit** ()
- QString **getProgram** ()
- QStringList **getArguments** ()
- QString **getBuiltInAction** ()
- QString **getDockTitle** ()
- QString **getGUITitle** ()
- bool **createsDocks** ()
- void **initialise** ()
- void **logItem** (customisationLog &log)
- void **addUserLevelAccess** (QDomElement element, customisationLog &log)
- void **setUserLevelState** (userLevelTypes::userLevels currentUserLevel)
- QAction * **getAction** ()

Protected Attributes

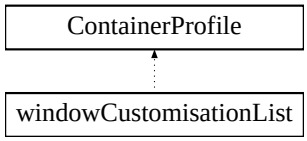
- ContainerProfile **profile**
- QAction ∗ **iAction**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cpp
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/O.linux-x86_64/moc/moc_windowCustomisation.cpp

9.354 windowCustomisationList Class Reference

Inheritance diagram for windowCustomisationList::



Public Types

- typedef QMap< QString, QDockWidget * > **dockMap**

Public Member Functions

- bool **loadCustomisation** (QString xmlFile)
- void **applyCustomisation** (QMainWindow *mw, QString customisationName, windowCustomisationInfo *customisationInfo, dockMap dockedComponents=dockMap())
- windowCustomisation * **getCustomisation** (QString name)
- void **initialise** (windowCustomisationInfo *customisationInfo)

Public Attributes

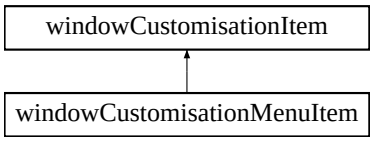
- customisationLog **log**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cp

9.355 windowCustomisationMenuItem Class Reference

Inheritance diagram for windowCustomisationMenuItem::



Public Types

- enum **menuObjectTypes** { **MENU_UNKNOWN**, **MENU_ITEM**, **MENU_-PLACEHOLDER**, **MENU_BUILT_IN** }

Public Member Functions

- **windowCustomisationMenuItem** (customisationLog &log, const QStringList menuHierarchyIn, const QString titleIn, const menuObjectTypes type, const bool separatorIn, const itemCheckInfo &checkInfoIn, const QObject *launchRequestReceiver, const QList< windowCreationListItem > &windowsIn, const QString programIn, const QStringList argumentsIn)
- **windowCustomisationMenuItem** (customisationLog &log, const QStringList menuHierarchyIn, const QString titleIn, const menuObjectTypes typeIn, const bool separatorIn, const itemCheckInfo &checkInfoIn)
- **windowCustomisationMenuItem** (customisationLog &log, const QStringList menuHierarchyIn, const QString titleIn, const menuObjectTypes typeIn, const bool separatorIn, const itemCheckInfo &checkInfoIn, const QString builtIn, const QString widgetNameIn)
- **windowCustomisationMenuItem** (customisationLog &log, const QStringList menuHierarchyIn, const QString titleIn, const menuObjectTypes typeIn, const bool separatorIn, const itemCheckInfo &checkInfoIn, const QString dockTitleIn)
- **windowCustomisationMenuItem** (windowCustomisationMenuItem *menuItem)
- QStringList **getMenuHierarchy** ()
- void **prependMenuHierarchy** (QStringList preMenuHierarchy)
- QString **getTitle** ()
- menuObjectTypes **getType** ()
- bool **hasSeparator** ()
- const itemCheckInfo & **getCheckInfo** ()

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/common/windowCustomisation.cpp

9.356 QE::Worker Class Reference

Signals

- void **processingComplete** (const QE::SequenceNumbers sequenceNumber, const QE::Counts instance)

Friends

- class **WorkerThread**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.cpp

9.357 QE::WorkerManager Class Reference

Classes

- class ReallyPrivate

Signals

- void **complete** (const QObject ∗workPackage)
- void **startProcessing** (const QE::SequenceNumbers sequenceNumber, QObject ∗workPackage)

Public Member Functions

- **WorkerManager** (const WorkerList &workForce, QObject ∗parent=0)
- void **process** (QObject ∗workPackage)
- Counts **getNumber** ()

Friends

- class **WorkerThread**

The documentation for this class was generated from the following files:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.cpp

9.358 QE::WorkerThread Class Reference

Public Member Functions

- **WorkerThread** (Worker ∗worker, Counts intance, Counts number, QObject ∗parent=0)

Protected Member Functions

- void **run** ()

Friends

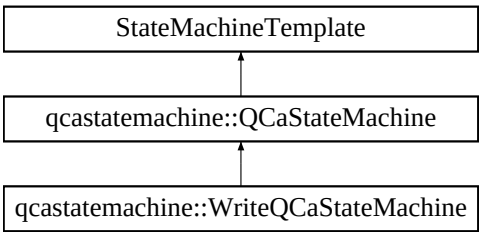
- class **WorkerManager**

The documentation for this class was generated from the following file:

- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/threads/QEWorkers.cpp

9.359 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/data/QCaStateMachine.cpp

9.360 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/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/zoomMenu.h
- /home/starritt/p4c/tec/gui/qeframework/trunk/qeframeworkSup/project/widgets/QEImage/zoomMenu.cpp

Index