EPICS QT Framework

1.1.8

Generated by Doxygen 1.7.4

Mon Mar 26 2012 16:09:13

Contents

Chapter 1

QCa - EPICS aware Qt Widgets

- QCa is a layered software framework for accessing EPICS data using Channel Access on a range of platforms.
- The QCa 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 QCa at several levels. QCa includes Qt plugin libraries, EPICS aware widgets, data formatting classes, and classes for accessing raw EPICS data in a Qt friendly way.
- QCa also includes an application - ASgui - 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.
- QCa handles much of the complexities of Channel Access including initiating and managing a channel. Applications using QCa can interact with Channel Access using Qt based classes and data types. Channel Access updates are delivered using Qt’s signals and slots mechanism.

1.1 Documentation

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

1.2 License

epicsqt is distributed under the terms of the GNU General Public License.

1.3 Platforms

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

1.4 Screenshots

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

Screenshots are only available in the HTML docs.

1.5 Downloads

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

For getting a development snapshot from the SVN repository:

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

1.6 Installation

Read `QCa_GettingStarted.pdf` in the documentation for setting up an environment for building or using the epicsqt framework.

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

Symbolic links to the resultant library libQCaPlugin.so will need to set up according to how it is to be used - see QCa_GettingStarted.pdf for details.

Any Qt specific queries? start at `Qt developer documentation`

1.7 Support

Visit the sourceforge epicsqt support page for assistance.

1.8 Related Projects

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

1.9 Credits:

Authors:

Andrew Rhyder, Anthony Owen, Glenn Jackson

Project admin:

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

Chapter 2

GNU General Public License

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

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

See the GNU General Public License for more details.

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

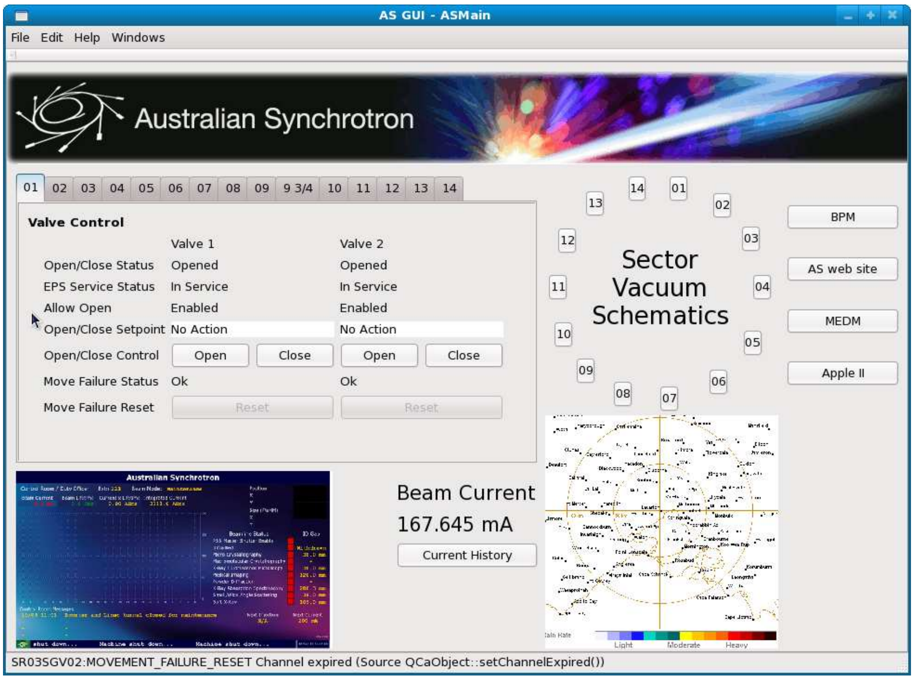
If not, see "http://www.gnu.org/licenses/

Chapter 3

INSTALL

Chapter 4

ASgui screen shots



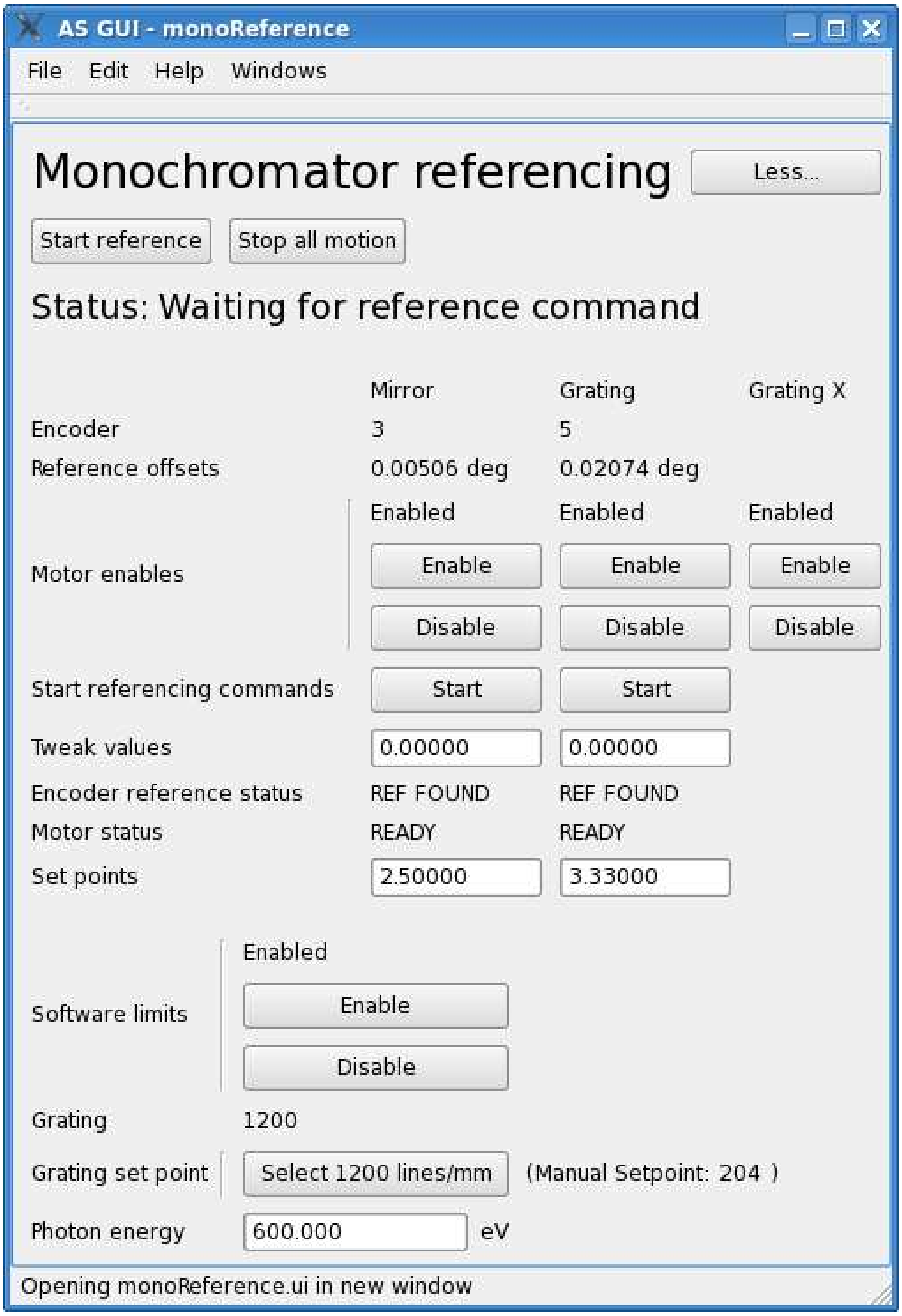
Valve Control

	Valve 1	Valve 2
Open/Close Status	Opened	Opened
EPS Service Status	In Service	In Service
Allow Open	Enabled	Enabled
Open/Close Setpoint	No Action	No Action
Open/Close Control	Open Close	Open Close
Move Failure Status	Ok	Ok
Move Failure Reset	Reset	Reset

Beam Current
167.645 mA

SR03SGV02:MOVEMENT_FAILURE_RESET Channel expired (Source QCaObject::setChannelExpired())

Figure 4.1: Australian Synchrotron mock up



Monochromator referencing

Status: Waiting for reference command

	Mirror	Grating	Grating X
Encoder	3	5	
Reference offsets	0.00506 deg	0.02074 deg	
Motor enables	Enabled	Enabled	Enabled
Start referencing commands	Start	Start	
Tweak values	0.00000	0.00000	
Encoder reference status	REF FOUND	REF FOUND	
Motor status	READY	READY	
Set points	2.50000	3.33000	

Software limits: Enabled

Grating: 1200

Grating set point: Select 1200 lines/mm (Manual Setpoint: 204)

Photon energy: 600.000 eV

Opening monoReference.ui in new window

Figure 4.2: Monochromator referencing

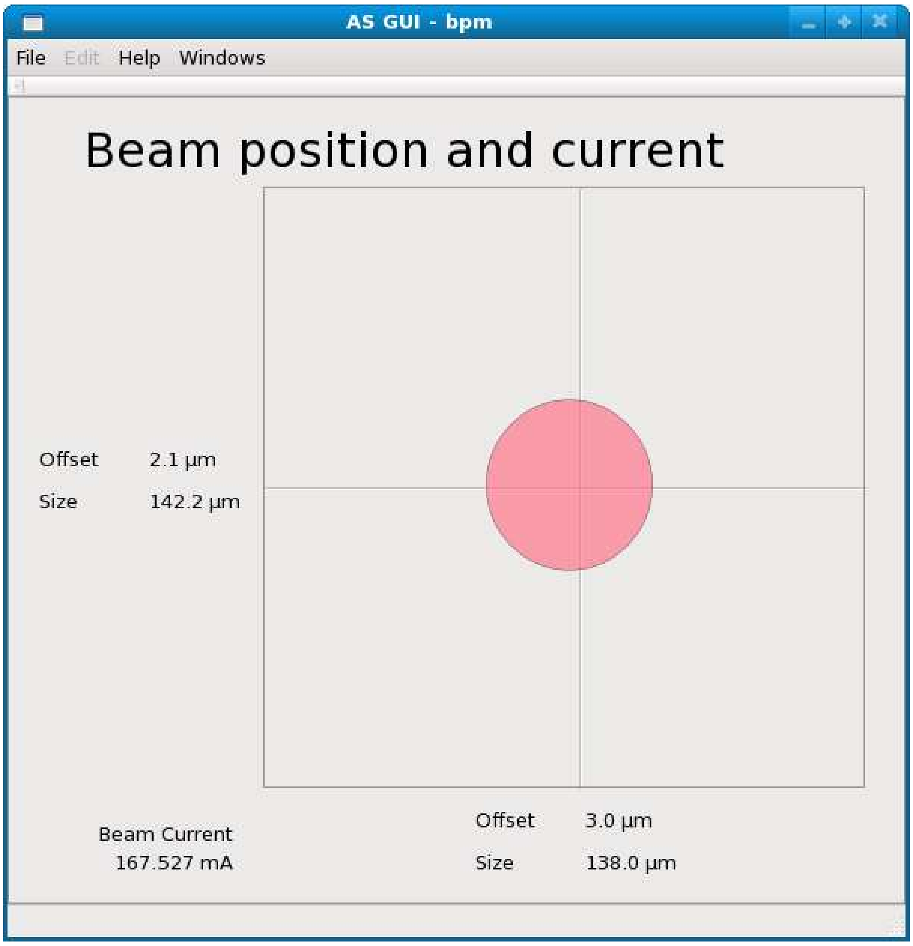
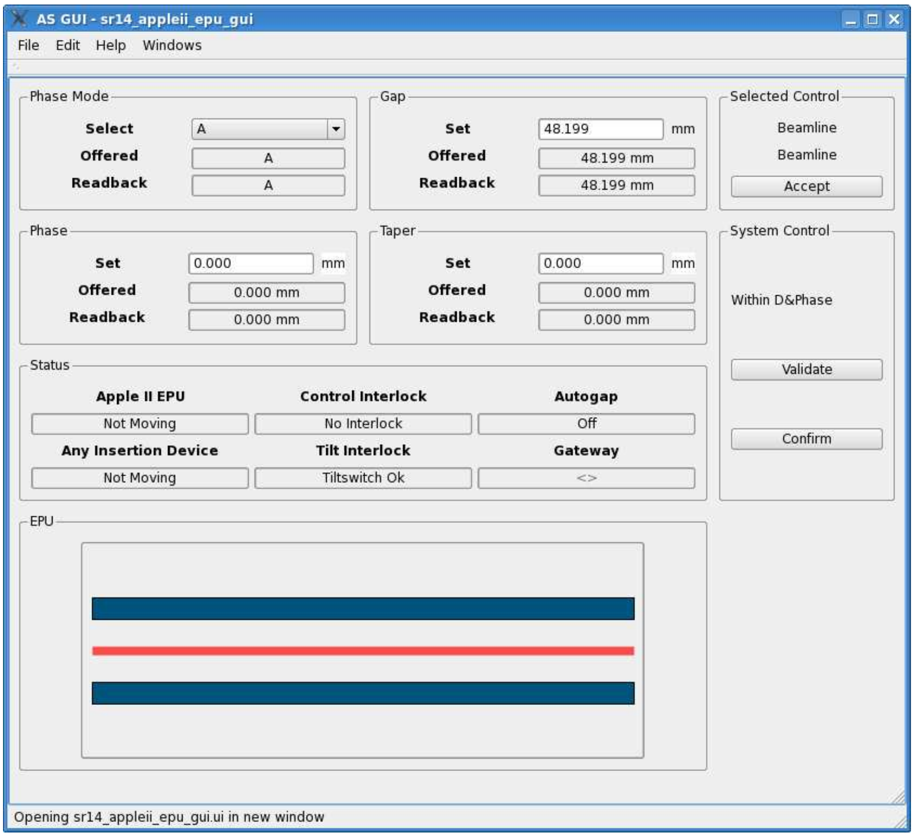


Figure 4.3: Beam position monitor



Phase Mode

Select	A
Offered	A
Readback	A

Gap

Set	48.199 mm
Offered	48.199 mm
Readback	48.199 mm

Selected Control

Beamline

Beamline

Accept

Phase

Set	0.000 mm
Offered	0.000 mm
Readback	0.000 mm

Taper

Set	0.000 mm
Offered	0.000 mm
Readback	0.000 mm

System Control

Within D&Phase

Validate

Confirm

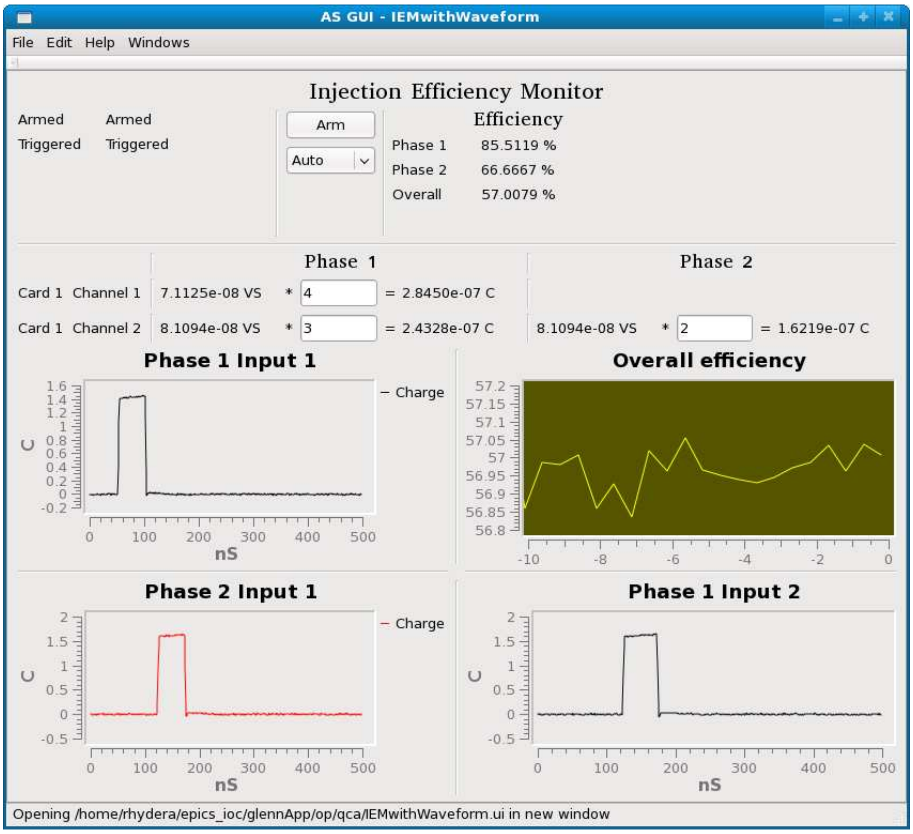
Status

Apple II EPU	Control Interlock	Autogap
Not Moving	No Interlock	Off
Any Insertion Device	**Tilt Interlock**	**Gateway**
Not Moving	Tiltswitch Ok	<>

EPU

Opening sr14_appleii_epu_gui.ui in new window

Figure 4.4: Insertion device



Injection Efficiency Monitor

Armed Armed
Triggered Triggered

Efficiency

Phase 1	85.5119 %
Phase 2	66.6667 %
Overall	57.0079 %

	Phase 1			Phase 2		
Card 1 Channel 1	7.1125e-08 VS	* 4	= 2.8450e-07 C			
Card 1 Channel 2	8.1094e-08 VS	* 3	= 2.4328e-07 C	8.1094e-08 VS	* 2	= 1.6219e-07 C

Opening /home/rhydera/epics_ioc/glennApp/op/qca/IEMwithWaveform.ui in new window

Figure 4.5: Injection efficiency monitor

Chapter 5

other applications using epicsqt widgets

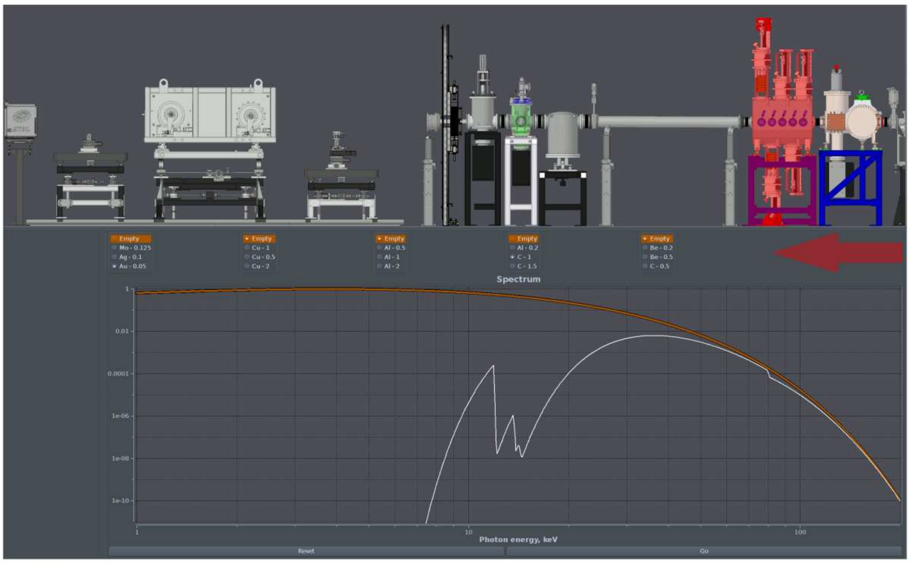
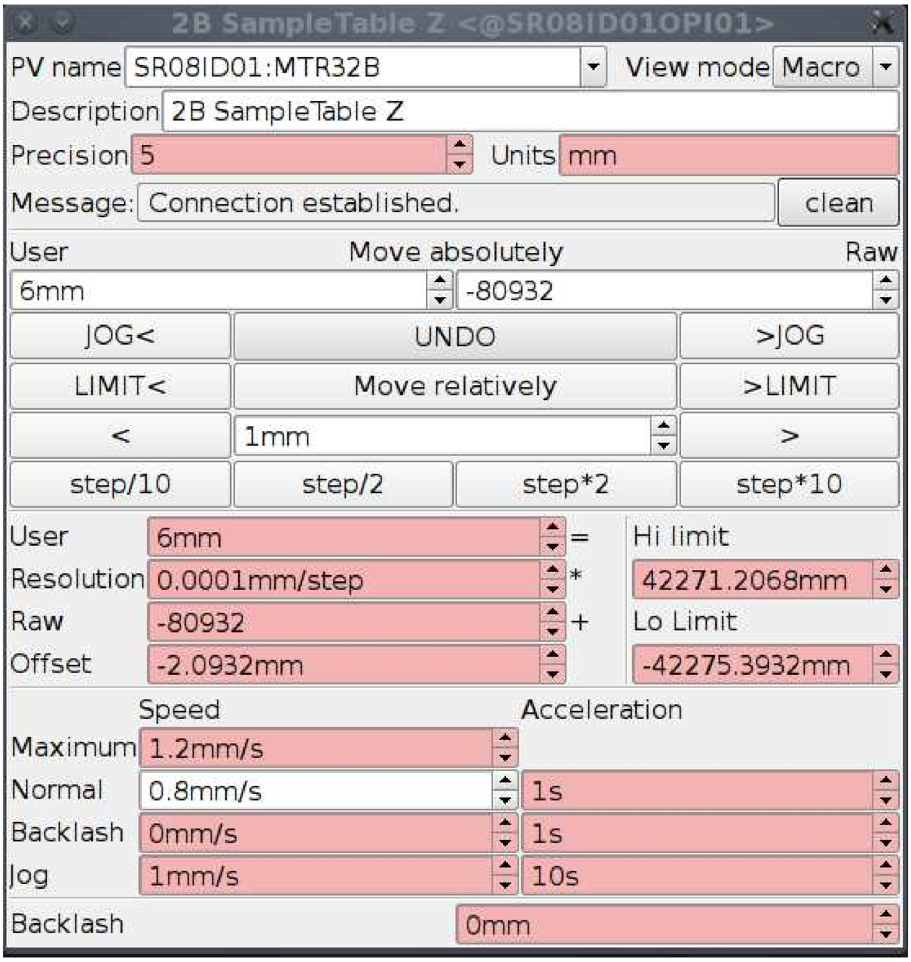


Figure 5.1: Medical Imaging beamline



2B SampleTable Z <@SR08ID01OPI01>

PV name SR08ID01:MTR32B View mode Macro

Description 2B SampleTable Z

Precision 5 Units mm

Message: Connection established. clean

User Move absolutely Raw

6mm -80932

User 6mm =

Resolution 0.0001mm/step *

Raw -80932 +

Offset -2.0932mm

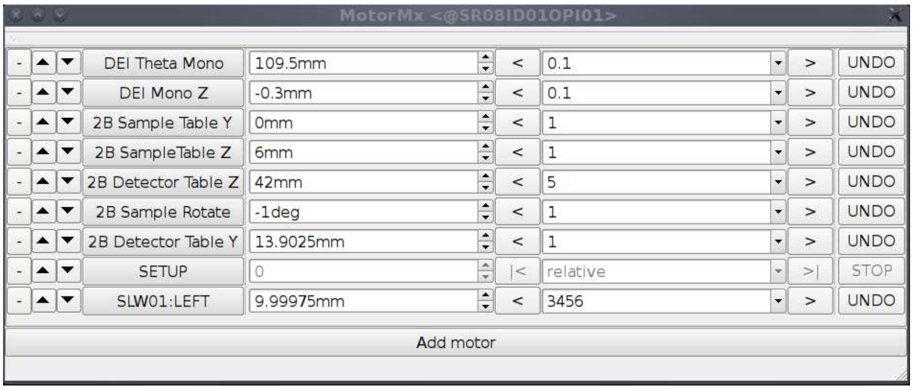
Hi limit 42271.2068mm

Lo Limit -42275.3932mm

	Speed	Acceleration
Maximum	1.2mm/s	
Normal	0.8mm/s	1s
Backlash	0mm/s	1s
Jog	1mm/s	10s

Backlash 0mm

Figure 5.2: Motor controller



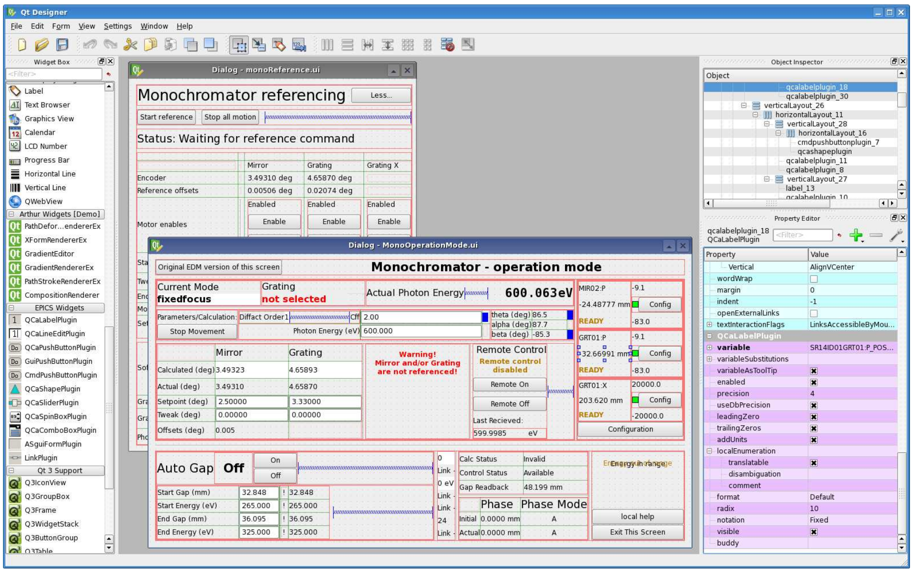
MotorMx <@SR08ID01OPI01>

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

Figure 5.3: Motor controller

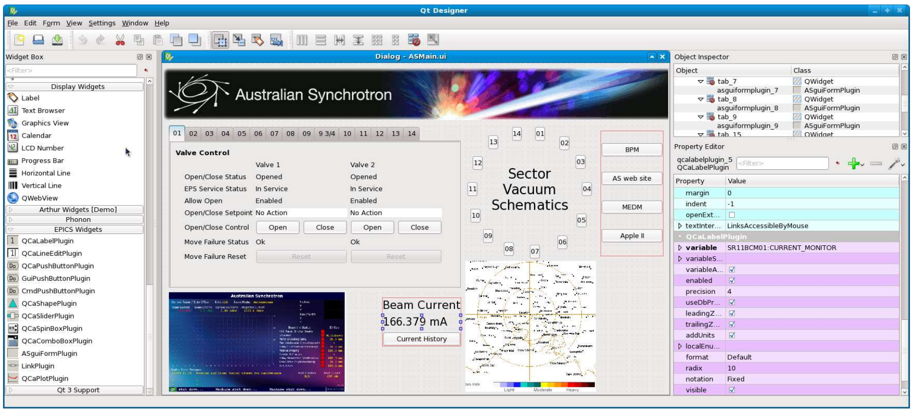
Chapter 6

Qt Designer



Property	Value
Vertical	AlignVCenter
wordWrap	□
margin	0
indent	-1
openExternalLinks	□
textInteractionFlags	LinksAccessibleByMou...
QCaLabelPlugin	
variable	SR14ID01GRT01:P_POS...
variableSubstitutions	
variableAsToolTip	☒
enabled	☒
precision	4
useDbPrecision	☒
leadingZero	☒
trailingZeros	☒
addUnits	☒
localEnumeration	
translatable	☒
disambiguation	
comment	
format	Default
radix	10
notation	Fixed
visible	☒
buddy	

Figure 6.1: Editing multiple GUIs



Valve Control

	Valve 1		Valve 2	
Open/Close Status	Opened		Opened	
EPS Service Status	In Service		In Service	
Allow Open	Enabled		Enabled	
Open/Close Setpoint	No Action		No Action	
Open/Close Control	Open	Close	Open	Close
Move Failure Status	Ok		Ok	
Move Failure Reset	Reset		Reset	

Beam Current
166.379 mA
Current History

Object	Class
tab_7	QWidget
asguiformplugin_7	ASguiFormPlugin
tab_8	QWidget
asguiformplugin_8	ASguiFormPlugin
tab_9	QWidget
asguiformplugin_9	ASguiFormPlugin
tab_15	QWidget

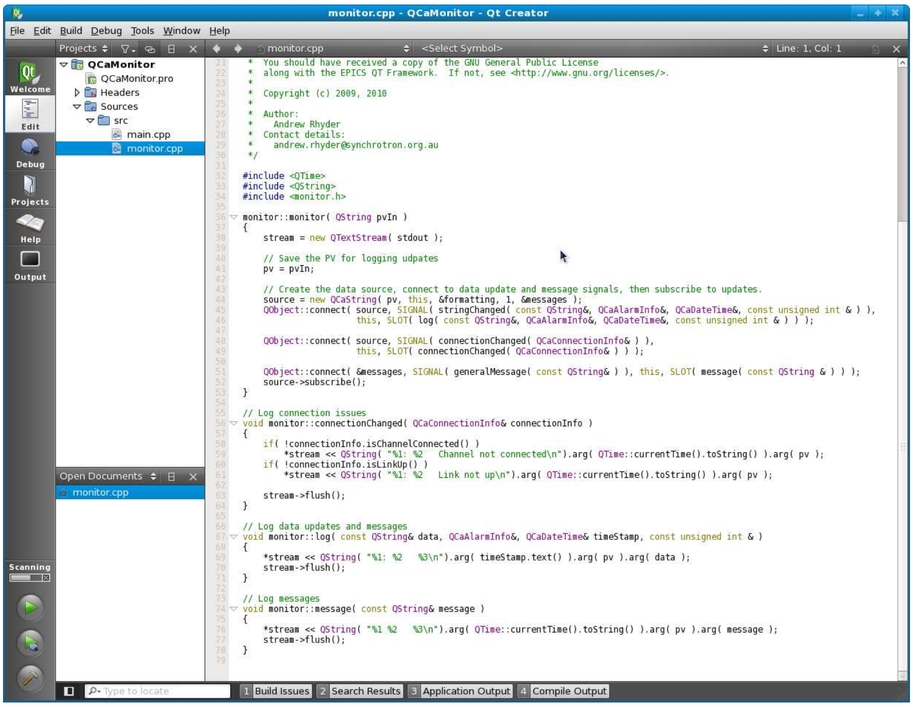
qcalabelplugin_5
QCaLabelPlugin

Property	Value
margin	0
indent	-1
openExt...	☐
textInter...	LinksAccessibleByMouse
QCaLabelPlugin	
variable	SR11BCM01:CURRENT_MONITOR
variableS...	
variableA...	☑
enabled	☑
precision	4
useDbPr...	☑
leadingZ...	☑
trailingZ...	☑
addUnits	☑
localEnu...	
format	Default
radix	10
notation	Fixed
visible	☑

Figure 6.2: Editing a GUI

Chapter 7

Qt Creator



```
 *  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/>.
 *
 *  Copyright (c) 2009, 2010
 *
 *  Author:
 *    Andrew Rhyder
 *  Contact details:
 *    andrew.rhyder@synchrotron.org.au
 */

#include <QTime>
#include <QString>
#include <monitor.h>

monitor::monitor( QString pvIn )
{
    stream = new QTextStream( stdout );

    // Save the PV for logging udpates
    pv = pvIn;

    // Create the data source, connect to data update and message signals, then subscribe to updates.
    source = new QCaString( pv, this, &formatting, 1, &messages );
    QObject::connect( source, SIGNAL( stringChanged( const QString&, QCaAlarmInfo&, QCaDateTime&, const unsigned int & ) ),
                      this, SLOT( log( const QString&, QCaAlarmInfo&, QCaDateTime&, const unsigned int & ) ) );

    QObject::connect( source, SIGNAL( connectionChanged( QCaConnectionInfo& ) ),
                      this, SLOT( connectionChanged( QCaConnectionInfo& ) ) );

    QObject::connect( &messages, SIGNAL( generalMessage( const QString& ) ), this, SLOT( message( const QString & ) ) );
    source->subscribe();
}

// Log connection issues
void monitor::connectionChanged( QCaConnectionInfo& connectionInfo )
{
    if( !connectionInfo.isChannelConnected() )
        *stream << QString( "%1: %2   Channel not connected\n").arg( QTime::currentTime().toString() ).arg( pv );
    if( !connectionInfo.isLinkUp() )
        *stream << QString( "%1: %2   Link not up\n").arg( QTime::currentTime().toString() ).arg( pv );

    stream->flush();
}

// Log data updates and messages
void monitor::log( const QString& data, QCaAlarmInfo&, QCaDateTime& timeStamp, const unsigned int & )
{
    *stream << QString( "%1: %2   %3\n").arg( timeStamp.text() ).arg( pv ).arg( data );
    stream->flush();
}

// Log messages
void monitor::message( const QString& message )
{
    *stream << QString( "%1 %2   %3\n").arg( QTime::currentTime().toString() ).arg( pv ).arg( message );
    stream->flush();
}
```

Figure 7.1: Application using epicsqt data source classes

Chapter 8

Namespace Index

8.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 9

Class Index

9.1 Class Hierarchy

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

Chapter 10

Class Index

10.1 Class List

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

Chapter 11

Namespace Documentation

11.1 qcaobject Namespace Reference

Classes

- class QCaObject

11.1.1 Detailed Description

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

Last checked in by:

Author:

rhydera

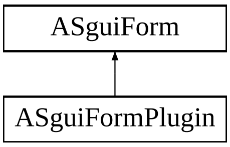
Contact details: andrew.rhyder@synchrotron.org.au.au 800 Blackburn Road, Clayton, Victoria 3168, Australia.

Chapter 12

Class Documentation

12.1 ASguiForm Class Reference

Inheritance diagram for ASguiForm:



Public Types

- enum **creationOptions** { **CREATION_OPTION_OPEN**, **CREATION_OPTION_NEW_TAB**, **CREATION_OPTION_NEW_WINDOW** }

Public Slots

- void **requestEnabled** (const bool &state)
- bool readUiFile ()
- void **onGeneralMessage** (QString message)
- void **onStatusMessage** (QString message)
- void **onErrorMessage** (QString message)
- void **onWarningMessage** (QString message)
- void **launchGui** (QString guiName, ASguiForm::creationOptions createOption)

Public Member Functions

- ASguiForm (QWidget ∗parent=0)
- ASguiForm (const QString &uifileNameIn, QWidget ∗parent=0)

- void commonInit (const bool alertIfUINoFoundIn)

  Common construction.
- virtual ~ASguiForm ()

  Destructor.
- QString **getASGuiTitle** ()
- QString **getGuiFileName** ()
- void setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setUiFileName** (QString uiFile)
- QString **getUiFileName** ()
- void **setHandleGuiLaunchRequests** (bool handleGuiLaunchRequests)
- bool **getHandleGuiLaunchRequests** ()

Protected Member Functions

- void setVariableNameSubstitutions (QString variableNameSubstitutionsIn)

  Set the variable name substitutions used by all QCa widgets wihtin the form.

Protected Attributes

- QString **uiFileName**
- bool **handleGuiLaunchRequests**

12.1.1 Constructor & Destructor Documentation

12.1.1.1 ASguiForm::ASguiForm (QWidget ∗ *parent* = 0)

This class is used as a container for QCa widgets.

It adds any variable name macro substitutions to the current environment profile, creates a form widget and reads a UI file which can contain QCa widgets.

As QCa widgets are created, they note the current environment profile, including variable name substitutions. QCa widgets also register themselves with this class so this class can activate them once they are fully created. QCa widgets can't activate themselves. The Qt form loader creates each widget and calls the appropriate property functions to set it up. The widget itself does not know what properties are going to be set and when they have all been set. For this reason the QCa widgets don't know when to request CA data. Both variable name properties and variable name substitution properties must be set up to request data and other properties may need to be set up before udates can be used.

This class can be used directly (within a GUI application) as the top level form, or as a base class for the ASguiFormPlugin class. The ASguiFormPlugin plugin class is used to store a gui form in a UI file. Constructor. No UI file is read. uiFileName must be set and then readUiFile() called after construction

12.1.1.2 ASguiForm::ASguiForm (const QString & *uiFileNameIn,* QWidget ∗ *parent* = 0)

Constructor. UI filename is supplied and UI file is read as part of construction.

12.1.2 Member Function Documentation

12.1.2.1 bool ASguiForm::readUiFile ( ) [slot]

!!??? currently not working. profile not set up yet so user messages can't be displayed this early perhaps

Ensure no other files are being monitored (belt and braces)

Monitor the opened file

Extract the file name part used for the window title

Any QCa widgets that have just been created need to be activated. Note, this is only required when QCa widgets are not loaded within a form and not directly by 'designer'. When loaded directly by 'designer' they are activated (a CA connection is established) as soon as either the variable name or variable name substitution properties are set

Present the gui to the user

12.1.2.2 void ASguiForm::setVariableNameAndSubstitutions (QString *variableNameIn,* QString *variableNameSubstitutionsIn,* unsigned int *variableIndex*)

Set new variable name substitutions

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

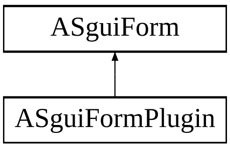
- epicsqt/ca_framework/widgets/include/ASguiForm.h
- epicsqt/ca_framework/widgets/src/ASguiForm.cpp

12.2 ASguiFormPlugin Class Reference

Wrapper around ASguiForm to make it a Qt Designer Plugin.

`#include <ASguiFormPlugin.h>`

Inheritance diagram for ASguiFormPlugin:



Public Member Functions

- ASguiFormPlugin (QWidget ∗parent=0)

  Constructors.
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()

Properties

- QString uiFile
- QString **variableSubstitutions**
- bool **handleGuiLaunchRequests**

12.2.1 Detailed Description

Wrapper around ASguiForm to make it a Qt Designer Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.2.2 Constructor & Destructor Documentation

12.2.2.1 ASguiFormPlugin::ASguiFormPlugin (QWidget ∗ *parent* = 0)

Constructors.

Constructor.

Set up a connection to recieve variable name property changes (actually only substitutions are used) The variable name property manager class only delivers an updated variable name (or substition) after the user has stopped typing

12.2.3 Property Documentation

12.2.3.1 QString ASguiFormPlugin::uiFile `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString uiFileName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/ASguiFormPlugin.h
- epicsqt/ca_framework/plugins/src/ASguiFormPlugin.cpp

12.3 ASguiFormPluginManager Class Reference

ASgui form Plugin for Qt Designer.

`#include <ASguiFormPluginManager.h>`

Public Member Functions

- **ASguiFormPluginManager** (QObject ∗parent=0)
- bool **isContainer** () const
- bool **isInitialized** () const
- QIcon icon () const

  Icon for widget. Used by Qt Designer in widget list.
- QString group () const

  Name of group Qt Designer will add widget to.
- QString **includeFile** () const
- QString name () const

  Name for widget. Used by Qt Designer in widget list.
- QString toolTip () const

  Tool tip for widget. Used by Qt Designer in widget list.
- QString **whatsThis** () const
- QWidget ∗ createWidget (QWidget ∗parent)

  Widget factory. Creates a ASguiForm widget.
- void **initialize** (QDesignerFormEditorInterface ∗core)

12.3.1 Detailed Description

ASgui form Plugin for Qt Designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

<andrew.rhyder>

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

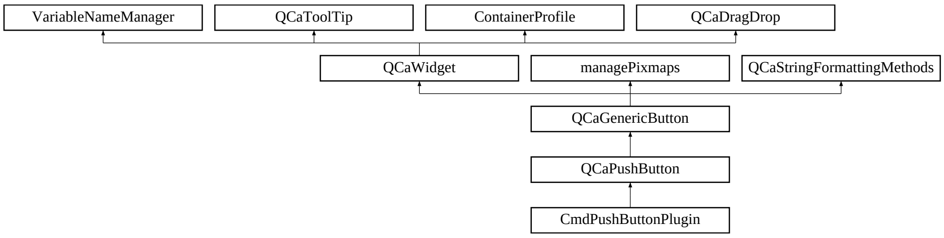
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/plugins/include/ASguiFormPluginManager.h
- epicsqt/ca_framework/plugins/src/ASguiFormPluginManager.cpp

12.4 CmdPushButtonPlugin Class Reference

Command Push Button Widget Plugin.

Inheritance diagram for CmdPushButtonPlugin:



Public Member Functions

- CmdPushButtonPlugin (QWidget ∗parent=0)

  Constructors.

Properties

- QString program

  Text to write on a button click when the button moves to the checked position.
- QStringList arguments

  Program to run.
- QString variableSubstitutions

12.4.1 Detailed Description

Command Push Button Widget Plugin.

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

Last checked in by:

Author:

rhydera

12.4.2 Constructor & Destructor Documentation

12.4.2.1 CmdPushButtonPlugin::CmdPushButtonPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

12.4.3 Property Documentation

12.4.3.1 QString CmdPushButtonPlugin::variableSubstitutions `[read, write]`

Variable substitutions Example: SECTOR=01 will result in any occurance of $SECTOR in variable name being replaced with 01. Note, unlike most objects with a variable name substitions, changes to this property do not take immediate effect, so the normal code which waits for the user to finish typing, then propogates the substitutions is not required. the substitutions are simply used next time the button is pushed.

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

- epicsqt/ca_framework/plugins/include/CmdPushButtonPlugin.h
- epicsqt/ca_framework/plugins/src/CmdPushButtonPlugin.cpp

12.5 CmdPushButtonPluginManager Class Reference

Command Push Button Widget Plugin Manager for designer.

`#include <CmdPushButtonPluginManager.h>`

Public Member Functions

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

12.5.1 Detailed Description

Command Push Button Widget Plugin Manager for designer.

???

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

Last checked in by:

Author:

rhydera

12.5.2 Constructor & Destructor Documentation

12.5.2.1 CmdPushButtonPluginManager::CmdPushButtonPluginManager (QObject ∗ *parent* = 0)

???

12.5.3 Member Function Documentation

12.5.3.1 QWidget ∗ CmdPushButtonPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLabel widget.

12.5.3.2 QString CmdPushButtonPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.5.3.3 QIcon CmdPushButtonPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.5.3.4 QString CmdPushButtonPluginManager::includeFile ( ) const

???

???

12.5.3.5 void CmdPushButtonPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.5.3.6 bool CmdPushButtonPluginManager::isContainer ( ) const

???

12.5.3.7 bool CmdPushButtonPluginManager::isInitialized ( ) const

???

12.5.3.8 QString CmdPushButtonPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.5.3.9 QString CmdPushButtonPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.5.3.10 QString CmdPushButtonPluginManager::whatsThis ( ) const

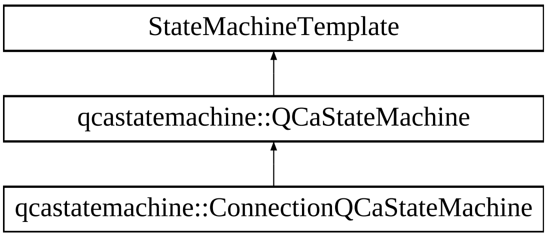
???

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

- epicsqt/ca_framework/plugins/include/CmdPushButtonPluginManager.h
- epicsqt/ca_framework/plugins/src/CmdPushButtonPluginManager.cpp

12.6 qcastatemachine::ConnectionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ConnectionQCaStateMachine:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::ConnectionQCaStateMachine

Public Member Functions

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

12.6.1 Constructor & Destructor Documentation

12.6.1.1 ConnectionQCaStateMachine::ConnectionQCaStateMachine (void ∗ *parent*)

Startup for the Connection statemachine.

12.6.2 Member Function Documentation

12.6.2.1 bool ConnectionQCaStateMachine::process (int *requestedState*) [virtual]

Process the Connection statemachine.

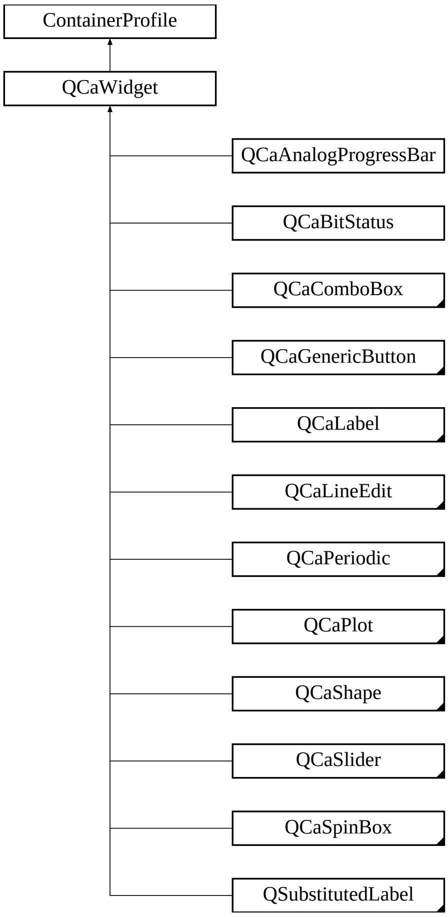
Implements qcastatemachine::QCaStateMachine.

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

12.7 ContainerProfile Class Reference

Inheritance diagram for ContainerProfile:



Public Member Functions

- void **takeLocalCopy** ()
- void setupProfile (QObject ∗statusMessageConsumerIn, QObject ∗errorMessageConsumerIn, QObject ∗warningMessageConsumerIn, QObject ∗guiLaunchConsumerIn, QString pathIn, QString parentPathIn, QString macroSubstitutionsIn)
- void setupLocalProfile (QObject ∗statusMessageConsumerIn, QObject ∗errorMessageConsumerIn, QObject ∗warningMessageConsumerIn, QObject ∗guiLaunchConsumerIn, QString pathIn, QString parentPathIn, QString macroSubstitutionsIn)
- void updateConsumers (QObject ∗statusMessageConsumerIn, QObject ∗errorMessageConsumerIn, QObject ∗warningMessageConsumerIn, QObject ∗guiLaunchConsumerIn)
- QObject ∗ replaceGuiLaunchConsumer (QObject ∗newGuiLaunchConsumerIn)
- void addMacroSubstitutions (QString macroSubstitutionsIn)
- void removeMacroSubstitutions ()
- QObject ∗ getStatusMessageConsumer ()
- QObject ∗ getWarningMessageConsumer ()
- QObject ∗ getErrorMessageConsumer ()
- QObject ∗ getGuiLaunchConsumer ()

- QString getPath ()
- QString getParentPath ()
- void setPublishedParentPath (QString publishedParentPathIn)
- QString getMacroSubstitutions ()
- bool isProfileDefined ()
- void addContainedWidget (QCaWidget ∗containedWidget)
- QCaWidget ∗ getNextContainedWidget ()
- void releaseProfile ()
- void publishOwnProfile ()

12.7.1 Member Function Documentation

12.7.1.1 void ContainerProfile::addContainedWidget (QCaWidget ∗ *containedWidget*)

Add a QCa widget to the list of QCa widgets created under the currently published profile. This provides the application with a list of its QCa widgets without having to trawl through the widget hierarchy looking for them. Note, in some applications the application may know exactly what QCa widgets have been created, but if the application has loaded a .ui file unrelated to the application development (for example, a user created control GUI), then the application will not know how many, if any, QCa widgets it owns.

12.7.1.2 void ContainerProfile::addMacroSubstitutions (QString *macroSubstitutionsIn*)

Extend the macro substitutions currently being used by all new QCaWidgets. This is used when a form is created. This allow a form to pass on macro substitutions to the QCa widgets it contains. Since it adds to the end of the existing macro substitutions, any substitutions already added by the originating container or higher forms take precedence.

12.7.1.3 QObject ∗ ContainerProfile::getErrorMessageConsumer ( )

Return the object to emit error message signals to. If NULL, there is no object available.

12.7.1.4 QObject ∗ ContainerProfile::getGuiLaunchConsumer ( )

Return the object to emit GUI launch request signals to. If NULL, there is no object available.

12.7.1.5 QString ContainerProfile::getMacroSubstitutions ( )

Return the object to emit warning message signals to. If NULL, there is no object available.

12.7.1.6 QCaWidget ∗ ContainerProfile::getNextContainedWidget ( )

Return the next QCa widget from the list of QCa widgets built using addContainedWidget(). Note, this is destructive to the list. It is fine if the application only needs to get the widgets from the list once, such as when activating QCa widgets after creating a form.

12.7.1.7 QString ContainerProfile::getParentPath ( )

Return the current object path to use for file operations.

12.7.1.8 QString ContainerProfile::getPath ( )

Return the application path to use for file operations.

12.7.1.9 QObject ∗ ContainerProfile::getStatusMessageConsumer ( )

Return the object to emit status message signals to. If NULL, there is no object available.

12.7.1.10 QObject ∗ ContainerProfile::getWarningMessageConsumer ( )

Return the object to emit warning message signals to. If NULL, there is no object available.

12.7.1.11 bool ContainerProfile::isProfileDefined ( )

Return the flag indicating true if a profile is currently being published.

12.7.1.12 void ContainerProfile::publishOwnProfile ( )

Set the published profile to whatever is saved in our local copy

12.7.1.13 void ContainerProfile::releaseProfile ( )

Clears any profile context. Must be called by any code that calls setupProfile() once the profile should no longer be used

12.7.1.14 void ContainerProfile::removeMacroSubstitutions ( )

Reduce the macro substitutions currently being used by all new QCaWidgets. This is used after a form is created. Any macro substitutions passed on by the form being created are no longer relevent.

12.7.1.15 QObject ∗ ContainerProfile::replaceGuiLaunchConsumer (QObject ∗ *newGuiLaunchConsumerIn*)

Update just the published signal consumer object that is used to launch new GUIs. The previous object is returned so it can be reinstated later.

12.7.1.16 void ContainerProfile::setPublishedParentPath (QString *publishedParentPathIn*)

Set the current object path to use for file operations.

12.7.1.17 void ContainerProfile::setupLocalProfile (QObject ∗ *statusMessageConsumerIn*, QObject ∗ *errorMessageConsumerIn*, QObject ∗ *warningMessageConsumerIn*, QObject ∗ *guiLaunchConsumerIn*, QString *pathIn*, QString *parentPathIn*, QString *macroSubstitutionsIn*)

Set up the local profile only (don't refer to any published profile) This is used when a QCa widget needs a profile, but none has been published. A default local profile can be set up using this method. The local profile can then be made public if required by calling publishOwnProfile()

12.7.1.18 void ContainerProfile::setupProfile (QObject ∗ *statusMessageConsumerIn*, QObject ∗ *errorMessageConsumerIn*, QObject ∗ *warningMessageConsumerIn*, QObject ∗ *guiLaunchConsumerIn*, QString *pathIn*, QString *parentPathIn*, QString *macroSubstitutionsIn*)

Setup the environmental profile prior to creating some QCa widgets. The new widgets will use this profile to determine their external environment.

This method locks access to the envionmental profile. ReleaseProfile() must be called to release the lock once all QCa widgets have been created.

12.7.1.19 void ContainerProfile::updateConsumers (QObject ∗ *statusMessageConsumerIn*, QObject ∗ *errorMessageConsumerIn*, QObject ∗ *warningMessageConsumerIn*, QObject ∗ *guiLaunchConsumerIn*)

Update published signal consumer objects. This is used if the signal consumer objects were not available when the profile was first set up, or if the objects are changing

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

- epicsqt/ca_framework/plugins/include/ContainerProfile.h
- epicsqt/ca_framework/plugins/src/ContainerProfile.cpp

12.8 QCaPeriodic::elementInfoStruct Struct Reference

Public Attributes

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

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

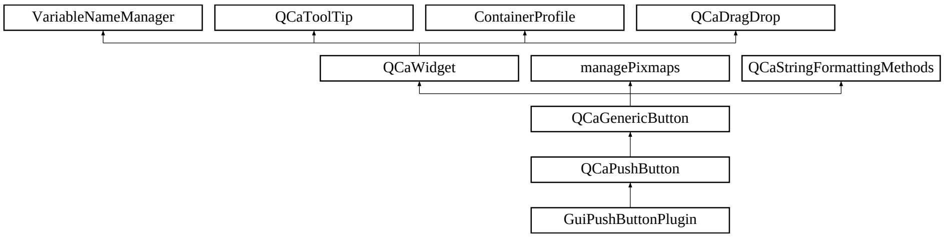
- epicsqt/ca_framework/widgets/include/QCaPeriodic.h

12.9 GuiPushButtonPlugin Class Reference

Launch GUI Push Button Widget Plugin.

```
#include <GuiPushButtonPlugin.h>
```

Inheritance diagram for GuiPushButtonPlugin:



Public Types

- enum **CreationOptionNames** { **Open** = ASguiForm::CREATION_OPTION_OPEN, **NewTab** = ASguiForm::CREATION_OPTION_NEW_TAB, **NewWindow** = ASguiForm::CREATION_-OPTION_NEW_WINDOW }

Public Member Functions

- GuiPushButtonPlugin (QWidget ∗parent=0)

  Constructors.
- void **setCreationOptionProperty** (CreationOptionNames creationOptionIn)
- CreationOptionNames **getCreationOptionProperty** ()

Properties

- QString **guiFile**
- CreationOptionNames creationOption

  GUI file name to launch.
- QString **variableSubstitutions**

12.9.1 Detailed Description

Launch GUI Push Button Widget Plugin.

Version

Revision:

#6

Date

DateTime:

2010/09/06 11:58:56

Author

Last checked in by:

Author:

rhydera

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

Last checked in by:

Author:

rhydera

12.9.2 Constructor & Destructor Documentation

12.9.2.1 GuiPushButtonPlugin::GuiPushButtonPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

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

- epicsqt/ca_framework/plugins/include/GuiPushButtonPlugin.h
- epicsqt/ca_framework/plugins/src/GuiPushButtonPlugin.cpp

12.10 GuiPushButtonPluginManager Class Reference

Launch GUI Push Button Widget Plugin Manager for designer.

`#include <GuiPushButtonPluginManager.h>`

Public Member Functions

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

12.10.1 Detailed Description

Launch GUI Push Button Widget Plugin Manager for designer.

???

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

Last checked in by:

Author:

rhydera

12.10.2 Constructor & Destructor Documentation

12.10.2.1 GuiPushButtonPluginManager::GuiPushButtonPluginManager (QObject ∗ *parent* = 0)

???

12.10.3 Member Function Documentation

12.10.3.1 QWidget ∗ GuiPushButtonPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLabel widget.

12.10.3.2 QString GuiPushButtonPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.10.3.3 QIcon GuiPushButtonPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.10.3.4 QString GuiPushButtonPluginManager::includeFile ( ) const

???

???

12.10.3.5 void GuiPushButtonPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.10.3.6 bool GuiPushButtonPluginManager::isContainer ( ) const

???

12.10.3.7 bool GuiPushButtonPluginManager::isInitialized ( ) const

???

12.10.3.8 QString GuiPushButtonPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.10.3.9 QString GuiPushButtonPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.10.3.10 QString GuiPushButtonPluginManager::whatsThis ( ) const

???

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

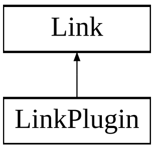
- epicsqt/ca_framework/plugins/include/GuiPushButtonPluginManager.h
- epicsqt/ca_framework/plugins/src/GuiPushButtonPluginManager.cpp

12.11 Link Class Reference

Manage updating tool tip with variable name, alarm state and connected state.

`#include <Link.h>`

Inheritance diagram for Link:



Public Types

- enum **conditions** {
  CONDITION_EQ, **CONDITION_NE**, **CONDITION_GT**, **CONDITION_GE**,
  CONDITION_LT, **CONDITION_LE** }

Public Slots

- void **in** (const bool &in)
- void **in** (const qlonglong &in)
- void **in** (const double &in)
- void **in** (const QString &in)
- void **autoFillBackground** (const bool &enable)

Signals

- void **out** (const bool &out)
- void **out** (const qlonglong &out)
- void **out** (const double &out)
- void **out** (const QString &out)

Public Member Functions

- Link (QWidget ∗parent=0)
- void **setCondition** (conditions conditionIn)
- conditions **getCondition** ()
- void **setComparisonValue** (QString comparisonValue)
- QString **getComparisonValue** ()
- void **setSignalTrue** (bool signalTrue)
- bool **getSignalTrue** ()
- void **setSignalFalse** (bool signalFalse)
- bool **getSignalFalse** ()
- void **setOutTrueValue** (QString outTrueValue)
- QString **getOutTrueValue** ()
- void **setOutFalseValue** (QString outFalseValue)
- QString **getOutFalseValue** ()
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()

Protected Attributes

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

12.11.1 Detailed Description

Manage updating tool tip with variable name, alarm state and connected state.

Link Widget.

Version

Revision:

#8

Date

DateTime:

2010/09/06 13:16:04

Author

andrew.rhyder

Version

Revision:

#7

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.11.2 Constructor & Destructor Documentation

12.11.2.1 Link::Link (QWidget * *parent* = 0)

Constructor with no initialisation

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

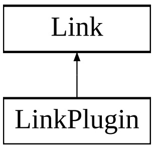
- epicsqt/ca_framework/widgets/include/Link.h
- epicsqt/ca_framework/widgets/src/Link.cpp

12.12 LinkPlugin Class Reference

Link Widget Plugin for designer.

```
#include <LinkPlugin.h>
```

Inheritance diagram for LinkPlugin:



Public Types

- enum **ConditionNames** {

  Equal = Link::CONDITION_EQ, **NotEqual** = Link::CONDITION_NE, **GreaterThan** = Link::CONDITION_GT, **GreaterThanOrEqual** = Link::CONDITION_GE,

  LessThan = Link::CONDITION_LT, **LessThanOrEqual** = Link::CONDITION_LE }

Public Member Functions

- LinkPlugin (QWidget ∗parent=0)

  Constructors.
- void **setConditionProperty** (ConditionNames condition)
- ConditionNames **getConditionProperty** ()

Properties

- ConditionNames condition

  Qt Designer Properties - condition.
- QString **comparisonValue**
- bool **signalTrue**
- bool **signalFalse**
- QString **outTrueValue**
- QString **outFalseValue**

12.12.1 Detailed Description

Link Widget Plugin for designer.

Version

Revision:

#6

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.12.2 Constructor & Destructor Documentation

12.12.2.1 LinkPlugin::LinkPlugin (QWidget ∗ *parent* = `0`)

Constructors.

???

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

- epicsqt/ca_framework/plugins/include/LinkPlugin.h
- epicsqt/ca_framework/plugins/src/LinkPlugin.cpp

12.13 LinkPluginManager Class Reference

Link Widget Plugin Manager for designer.

`#include <LinkPluginManager.h>`

Public Member Functions

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

12.13.1 Detailed Description

Link Widget Plugin Manager for designer.

CA Label Widget Plugin Manager for designer.

???

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.13.2 Constructor & Destructor Documentation

12.13.2.1 LinkPluginManager::LinkPluginManager (QObject ∗ *parent* = 0)

???

12.13.3 Member Function Documentation

12.13.3.1 QWidget ∗ LinkPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a Link widget.

12.13.3.2 QString LinkPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.13.3.3 QIcon LinkPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.13.3.4 QString LinkPluginManager::includeFile ( ) const

???

???

12.13.3.5 void LinkPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.13.3.6 bool LinkPluginManager::isContainer ( ) const

???

12.13.3.7 bool LinkPluginManager::isInitialized ( ) const

???

12.13.3.8 QString LinkPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.13.3.9 QString LinkPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.13.3.10 QString LinkPluginManager::whatsThis ( ) const

???

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

- epicsqt/ca_framework/plugins/include/LinkPluginManager.h
- epicsqt/ca_framework/plugins/src/LinkPluginManager.cpp

12.14 localEnumerationItem Class Reference

Public Types

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

Public Attributes

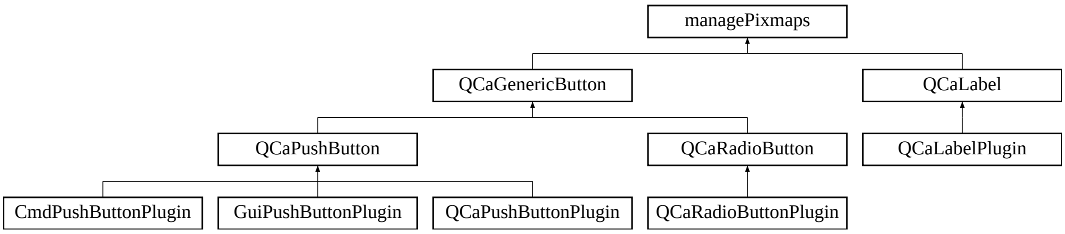
- double **dValue**
- QString **sValue**
- operations **op**
- QString **text**

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

- epicsqt/ca_framework/data/include/QCaStringFormatting.h

12.15 managePixmaps Class Reference

Inheritance diagram for managePixmaps:



Public Member Functions

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

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

- epicsqt/ca_framework/widgets/include/managePixmaps.h
- epicsqt/ca_framework/widgets/src/managePixmaps.cpp

12.16 PeriodicDialog Class Reference

Public Member Functions

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

Protected Member Functions

- void **changeEvent** (QEvent ∗e)

12.16.1 Constructor & Destructor Documentation

12.16.1.1 PeriodicDialog::PeriodicDialog (QWidget ∗ *parent* = 0)

This class manages the presenation of a periodic table and the selection of an element in the table by a user. It is used by the QCaPeriodic widget. The QCaPeriodic widget appears as a push button displaying an element name. When the QCaPeriodic widget is pressed this dialog is presented with the focus on the button in the periodic table for the current element. When a button for an element in the periodic table is pressed the element symbol is saved ready for recall by the QCaPeriodic widget.

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

- epicsqt/ca_framework/widgets/include/PeriodicDialog.h
- epicsqt/ca_framework/widgets/src/PeriodicDialog.cpp

12.17 PeriodicElementSetupForm Class Reference

Public Member Functions

- PeriodicElementSetupForm (QWidget ∗parent=0)

12.17.1 Constructor & Destructor Documentation

12.17.1.1 PeriodicElementSetupForm::PeriodicElementSetupForm (QWidget ∗ *parent* = 0) [explicit]

This class manages a each of the element sub forms in the dialog for setting up a QCaPeriodicPlugin in Qt designer.

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

- epicsqt/ca_framework/plugins/include/PeriodicElementSetupForm.h
- epicsqt/ca_framework/plugins/src/PeriodicElementSetupForm.cpp

12.18 PeriodicSetupDialog Class Reference

Public Member Functions

- PeriodicSetupDialog (QWidget ∗parent=0)

12.18.1 Constructor & Destructor Documentation

12.18.1.1 PeriodicSetupDialog::PeriodicSetupDialog (QWidget ∗ *parent* = 0) [explicit]

This class manages a dialog for setting up a QCaPeriodicPlugin in Qt designer.

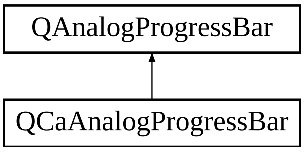
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/plugins/include/PeriodicSetupDialog.h
- epicsqt/ca_framework/plugins/src/PeriodicSetupDialog.cpp

12.19 QAnalogProgressBar Class Reference

```
#include <QAnalogProgressBar.h>
```

Inheritance diagram for QAnalogProgressBar:



Public Slots

- void **setAnalogRange** (const double analogMinimumIn, const double analogMaximumIn)
- void **setAnalogMinimum** (const double analogMinimumIn)
- void **setAnalogMaximum** (const double analogMaximumIn)
- void **setAnalogValue** (const double analogValueIn)

Public Member Functions

- QAnalogProgressBar (QWidget ∗parent=0)

  Constructor.
- double **getAnalogMinimum** ()
- double **getAnalogMaximum** ()
- double **getAnalogValue** ()

Protected Member Functions

- virtual QString **getSprintfFormat** ()

Properties

- double **analogMinimum**
- double **analogMaximum**
- double **analogValue**

12.19.1 Detailed Description

This class inherits directly from QProgressBar and provides an analog view of the progress bar widget including analogous analog functions and properties:

QProgressBar QAnalogProgressBar

setRange() setAnalogRange() setMinimum() setAnalogMinimum() getMinimum() getAnalogMinimum() setMaximum() setAnalogMaximum() getMaximum() getAnalogMaximum() setValue() setAnalogValue() getValue() getAnalogValue()

Minimum analogMinimum Maximum analogMaximum Value analogValue

The class sets the range of of the underlying QProgressBar to 0 .. 10000. It it considered unlikely, in practice, that an AnalogProgressBar wll stretch more than 10000 pixels given that current screen technology is limited to 2000 pixels.

This class inherited from rather than containing a QProgressBar, and as such cannot hide and does not override the integer related base class functions and properties. Therefore they should not be used.

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

- epicsqt/ca_framework/widgets/include/QAnalogProgressBar.h
- epicsqt/ca_framework/widgets/src/QAnalogProgressBar.cpp

12.20 QAnalogProgressBarPluginManager Class Reference

```
#include <QAnalogProgressBarPluginManager.h>
```

Public Member Functions

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

12.20.1 Detailed Description

???

12.20.2 Constructor & Destructor Documentation

12.20.2.1 QAnalogProgressBarPluginManager::QAnalogProgressBarPluginManager (QObject ∗ *parent* = 0)

???

12.20.3 Member Function Documentation

12.20.3.1 QWidget ∗ QAnalogProgressBarPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QAnalogProgressBar widget.

12.20.3.2 QString QAnalogProgressBarPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.20.3.3 QIcon QAnalogProgressBarPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.20.3.4 QString QAnalogProgressBarPluginManager::includeFile ( ) const

???

???

12.20.3.5 void QAnalogProgressBarPluginManager::initialize (QDesignerFormEditorInterface $*$ *core*)

???

12.20.3.6 bool QAnalogProgressBarPluginManager::isContainer ( ) const

???

12.20.3.7 bool QAnalogProgressBarPluginManager::isInitialized ( ) const

???

12.20.3.8 QString QAnalogProgressBarPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.20.3.9 QString QAnalogProgressBarPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.20.3.10 QString QAnalogProgressBarPluginManager::whatsThis ( ) const

???

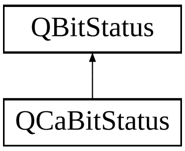
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/plugins/include/QAnalogProgressBarPluginManager.h
- epicsqt/ca_framework/plugins/src/QAnalogProgressBarPluginManager.cpp

12.21 QBitStatus Class Reference

CA Shape Widget.

Inheritance diagram for QBitStatus:



Public Types

- enum **Orientations** { **LSB_On_Right**, **LSB_On_Bottom**, **LSB_On_Left**, **LSB_-On_Top** }

Public Slots

- void **setValue** (const long 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 **setDrawBorder** (const bool value)
- bool **getDrawBorder** ()
- void **setNumberOfBits** (const int value)
- int **getNumberOfBits** ()
- void **setShift** (const int value)
- int **getShift** ()
- void **setIsValid** (const bool value)
- bool **getIsValid** ()
- void **setOrientation** (const enum Orientations value)
- enum Orientations **getOrientation** ()
- long **getValue** ()

Properties

- int **value**
- int **numberOfBits**
- int **shift**
- QColor **boarderColour**
- QColor **invalidColour**
- QColor **onColour**
- QColor **offColour**
- bool **drawBorder**
- bool **isValid**
- Orientations **Orientation**

12.21.1 Detailed Description

CA Shape Widget.

Version

Revision:

#3

Date

DateTime:

2011/06/12 15:07:03

Author

andrew.starritt

12.21.2 Constructor & Destructor Documentation

12.21.2.1 QBitStatus::QBitStatus (QWidget ∗ *parent* = 0)

12.21.3 Member Function Documentation

12.21.3.1 QSize QBitStatus::sizeHint ( ) const [virtual]

--- Define default size for this widget class.

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

- epicsqt/ca_framework/widgets/include/QBitStatus.h
- epicsqt/ca_framework/widgets/src/QBitStatus.cpp

12.22 QBitStatusPluginManager Class Reference

CA ProgressBar Widget Plugin Manager for designer.

`#include <QBitStatusPluginManager.h>`

Public Member Functions

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

12.22.1 Detailed Description

CA ProgressBar Widget Plugin Manager for designer.

BitStatus Widget Plugin Manager for designer.

???

Version

Revision:

#1

Date

DateTime:

2011/06/11 16:35:50

Author

andrew.starritt

12.22.2 Constructor & Destructor Documentation

12.22.2.1 QBitStatusPluginManager::QBitStatusPluginManager (QObject ∗ *parent* = 0)

???

12.22.3 Member Function Documentation

12.22.3.1 QWidget ∗ QBitStatusPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QBitStatus widget.

12.22.3.2 QString QBitStatusPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.22.3.3 QIcon QBitStatusPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.22.3.4 QString QBitStatusPluginManager::includeFile ( ) const

???

???

12.22.3.5 void QBitStatusPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.22.3.6 bool QBitStatusPluginManager::isContainer ( ) const

???

12.22.3.7 bool QBitStatusPluginManager::isInitialized ( ) const

???

12.22.3.8 QString QBitStatusPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.22.3.9 QString QBitStatusPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.22.3.10 QString QBitStatusPluginManager::whatsThis ( ) const

???

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

- epicsqt/ca_framework/plugins/include/QBitStatusPluginManager.h
- epicsqt/ca_framework/plugins/src/QBitStatusPluginManager.cpp

12.23 QCaAlarmInfo Class Reference

Public Member Functions

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

Static Public Member Functions

- static QCAALARMINFO_SEVERITY getInvalidSeverity ()

12.23.1 Constructor & Destructor Documentation

12.23.1.1 QCaAlarmInfo::QCaAlarmInfo ( )

Construct an empty instance. By default there is no alarm present.

12.23.1.2 QCaAlarmInfo::QCaAlarmInfo (unsigned short *statusIn*, unsigned short *severityIn*)

Construct an instance given an alarm state and severity

12.23.2 Member Function Documentation

12.23.2.1 QCAALARMINFO_SEVERITY QCaAlarmInfo::getInvalidSeverity ( ) [static]

Return a severity that will not match any valid severity

12.23.2.2 QCAALARMINFO_SEVERITY QCaAlarmInfo::getSeverity ( )

Return the severity The caller is only expected to compare this to

12.23.2.3 bool QCaAlarmInfo::isInAlarm ( )

Return true if there is an alarm

12.23.2.4 bool QCaAlarmInfo::isInvalid ( )

Return true if there is an invalid alarm

12.23.2.5 bool QCaAlarmInfo::isMajor ( )

Return true if there is a major alarm

12.23.2.6 bool QCaAlarmInfo::isMinor ( )

Return true if there is a minor alarm

12.23.2.7 QString QCaAlarmInfo::severityName ( )

Return a string identifying the alarm severity

12.23.2.8 QString QCaAlarmInfo::statusName ( )

Return a string identifying the alarm state

12.23.2.9 QString QCaAlarmInfo::style ( )

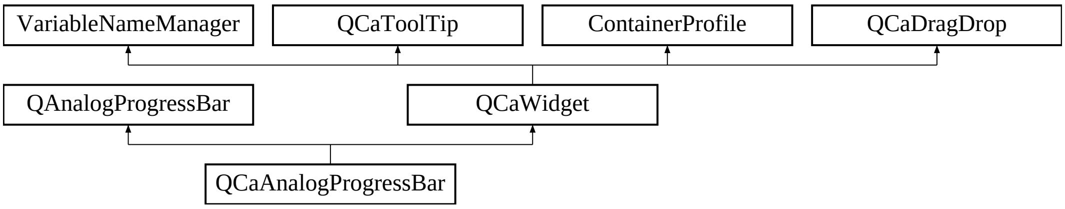
Return a style string to update the widget's look to reflect the current alarm state

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

- epicsqt/ca_framework/data/include/QCaAlarmInfo.h
- epicsqt/ca_framework/data/src/QCaAlarmInfo.cpp

12.24 QCaAnalogProgressBar Class Reference

Inheritance diagram for QCaAnalogProgressBar:



Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged** (const double &out)

Public Member Functions

- QCaAnalogProgressBar (QWidget ∗parent=0)
- QCaAnalogProgressBar (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setUseDbPrecision** (bool useDbPrecisionIn)
- bool **getUseDbPrecision** ()
- void **setUseDbDisplayLimits** (bool useDbDisplayLimitsIn)
- bool **getUseDbDisplayLimits** ()
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()

Protected Member Functions

- virtual QString getSprintfFormat ()
- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaFloatingFormatting **floatingFormatting**
- bool **localEnabled**

Properties

- QString variable
- QString **variableSubstitutions**
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- bool useDbDisplayLimits

  Display properties.
- bool **visible**

12.24.1 Constructor & Destructor Documentation

12.24.1.1 QCaAnalogProgressBar::QCaAnalogProgressBar (QWidget ∗ *parent* = 0)

This class is a CA aware analog progress bar widget based on the analog progress bar widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

--- Constructor with no initialisation

12.24.1.2 QCaAnalogProgressBar::QCaAnalogProgressBar (const QString & *variableNameIn*, QWidget ∗ *parent* = 0)

--- Constructor with known variable

12.24.2 Member Function Documentation

12.24.2.1 void QCaAnalogProgressBar::establishConnection (unsigned int *variableIndex*) [protected, virtual]

--- Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.24.2.2 QString QCaAnalogProgressBar::getSprintfFormat ( ) [protected, virtual]

--- Provide suffix, i.e. EGU if appropriate

Reimplemented from QAnalogProgressBar.

12.24.2.3 bool QCaAnalogProgressBar::isEnabled ( ) const

-- Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.24.2.4 void QCaAnalogProgressBar::requestEnabled (const bool & *state*) [slot]

-- Slot similar to default widget setEnabled, but will use our own setEnabled which will allow alarm states to override current enabled state.

12.24.2.5 void QCaAnalogProgressBar::setEnabled (bool *state*)

-- Override the default widget setEnabled to allow alarm states to override current enabled state.

Note the new ’enabled’ state

Set the enabled state of the widget only if connected

12.24.2.6 void QCaAnalogProgressBar::setVariableNameAndSubstitutions (QString *variableNameIn,* QString *variableNameSubstitutionsIn,* unsigned int *variableIndex*)

TODO a WTF comment

12.24.3 Property Documentation

12.24.3.1 QString QCaAnalogProgressBar::variable [read, write]

Note, a property macro in the form ’Q_PROPERTY(QString variableName READ ...’ doesn’t work. A property name ending with ’Name’ results in some sort of string a variable being displayed, but will only accept alphanumeric and won’t generate callbacks on change.

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

- epicsqt/ca_framework/widgets/include/QCaAnalogProgressBar.h
- epicsqt/ca_framework/widgets/src/QCaAnalogProgressBar.cpp

12.25 QCaAnalogProgressBarPluginManager Class Reference

```
#include <QCaAnalogProgressBarPluginManager.h>
```

Public Member Functions

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

12.25.1 Detailed Description

???

12.25.2 Constructor & Destructor Documentation

12.25.2.1 QCaAnalogProgressBarPluginManager::QCaAnalogProgressBarPluginManager (QObject ∗ *parent* = 0)

???

12.25.3 Member Function Documentation

12.25.3.1 QWidget ∗ QCaAnalogProgressBarPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaAnalogProgressBar widget.

12.25.3.2 QString QCaAnalogProgressBarPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.25.3.3 QIcon QCaAnalogProgressBarPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.25.3.4 QString QCaAnalogProgressBarPluginManager::includeFile ( ) const

???

???

12.25.3.5 void QCaAnalogProgressBarPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.25.3.6 bool QCaAnalogProgressBarPluginManager::isContainer ( ) const

???

12.25.3.7 bool QCaAnalogProgressBarPluginManager::isInitialized ( ) const

???

12.25.3.8 QString QCaAnalogProgressBarPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.25.3.9 QString QCaAnalogProgressBarPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.25.3.10 QString QCaAnalogProgressBarPluginManager::whatsThis ( ) const

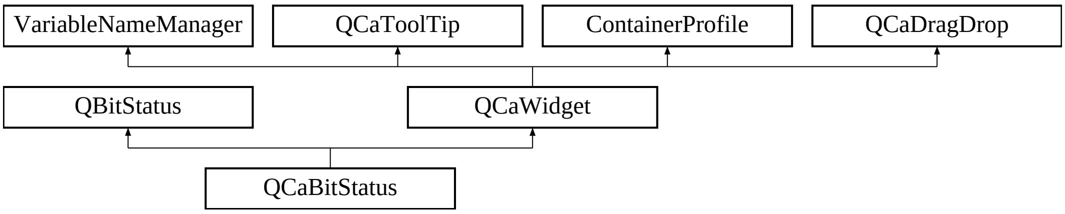
???

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

- epicsqt/ca_framework/plugins/include/QCaAnalogProgressBarPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaAnalogProgressBarPluginManager.cpp

12.26 QCaBitStatus Class Reference

Inheritance diagram for QCaBitStatus:



Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged** (const long &out)

Public Member Functions

- QCaBitStatus (QWidget ∗parent=0)

  #endif
- QCaBitStatus (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void setVariableNameAndSubstitutions (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaIntegerFormatting **integerFormatting**
- bool **localEnabled**

Properties

- QString variable

  #ifdef PLUGIN_APP
- QString **variableSubstitutions**
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- bool **visible**

12.26.1 Constructor & Destructor Documentation

12.26.1.1 QCaBitStatus::QCaBitStatus (QWidget ∗ *parent* = 0)

#endif

This class is a CA aware Bit Status widget based on the BitStatus widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

--- Constructor with no initialisation

12.26.1.2 QCaBitStatus::QCaBitStatus (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

--- Constructor with known variable

12.26.2 Member Function Documentation

12.26.2.1 void QCaBitStatus::establishConnection (unsigned int *variableIndex*) [protected, virtual]

--- Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.26.2.2 bool QCaBitStatus::isEnabled ( ) const

--- Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.26.2.3 void QCaBitStatus::requestEnabled (const bool & *state*) [slot]

-- Slot similar to default widget setEnabled, but will use our own setEnabled which will allow alarm states to override current enabled state.

12.26.2.4 void QCaBitStatus::setEnabled (bool *state*)

-- Override the default widget setEnabled to allow alarm states to override current enabled state.

Note the new 'enabled' state

Set the enabled state of the widget only if connected

12.26.2.5 void QCaBitStatus::setVariableNameAndSubstitutions (QString *variableNameIn,* QString *variableNameSubstitutionsIn,* unsigned int *variableIndex*)

TODO a WTF comment

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

- epicsqt/ca_framework/widgets/include/QCaBitStatus.h
- epicsqt/ca_framework/widgets/src/QCaBitStatus.cpp

12.27 QCaBitStatusPluginManager Class Reference

CA ProgressBar Widget Plugin Manager for designer.

`#include <QCaBitStatusPluginManager.h>`

Public Member Functions

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

12.27.1 Detailed Description

CA ProgressBar Widget Plugin Manager for designer.

???

Version

Revision:

#1

Date

DateTime:

2011/06/11 21:36:00

Author

andrew.starritt

12.27.2 Constructor & Destructor Documentation

12.27.2.1 QCaBitStatusPluginManager::QCaBitStatusPluginManager (QObject ∗ *parent* = 0)

???

12.27.3 Member Function Documentation

12.27.3.1 QWidget ∗ QCaBitStatusPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaBitStatus widget.

12.27.3.2 QString QCaBitStatusPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.27.3.3 QIcon QCaBitStatusPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.27.3.4 QString QCaBitStatusPluginManager::includeFile ( ) const

???

???

12.27.3.5 void QCaBitStatusPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.27.3.6 bool QCaBitStatusPluginManager::isContainer ( ) const

???

12.27.3.7 bool QCaBitStatusPluginManager::isInitialized ( ) const

???

12.27.3.8 QString QCaBitStatusPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.27.3.9 QString QCaBitStatusPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.27.3.10 QString QCaBitStatusPluginManager::whatsThis ( ) const

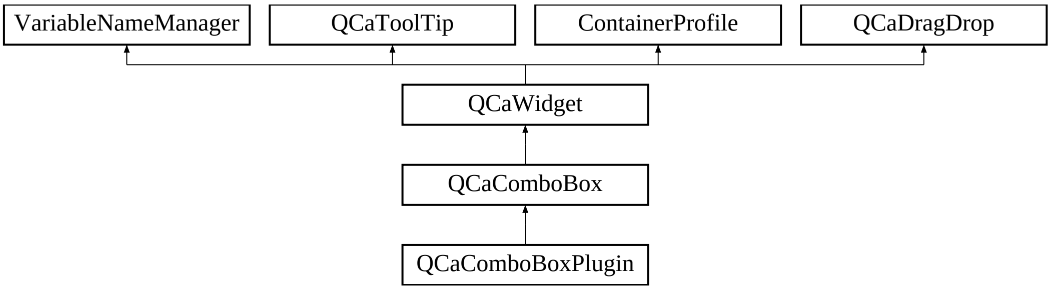
???

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

- epicsqt/ca_framework/plugins/include/QCaBitStatusPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaBitStatusPluginManager.cpp

12.28 QCaComboBox Class Reference

Inheritance diagram for QCaComboBox:



Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged** (const qlonglong &out)

Public Member Functions

- QCaComboBox (QWidget ∗parent=0)
- QCaComboBox (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setUseDbEnumerations** (bool useDbEnumerations)
- bool **getUseDbEnumerations** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaIntegerFormatting **integerFormatting**
- bool **useDbEnumerations**
- bool **localEnabled**
- bool **allowDrop**

12.28.1 Constructor & Destructor Documentation

12.28.1.1 QCaComboBox::QCaComboBox (QWidget ∗ *parent* = 0)

This class is a CA aware combo box widget based on the Qt combo box widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Construct a combo box with no variable specified yet

12.28.1.2 QCaComboBox::QCaComboBox (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Construct a combo box with a variable specified

12.28.2 Member Function Documentation

12.28.2.1 void QCaComboBox::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.28.2.2 bool QCaComboBox::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.28.2.3 void QCaComboBox::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

12.28.2.4 void QCaComboBox::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

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

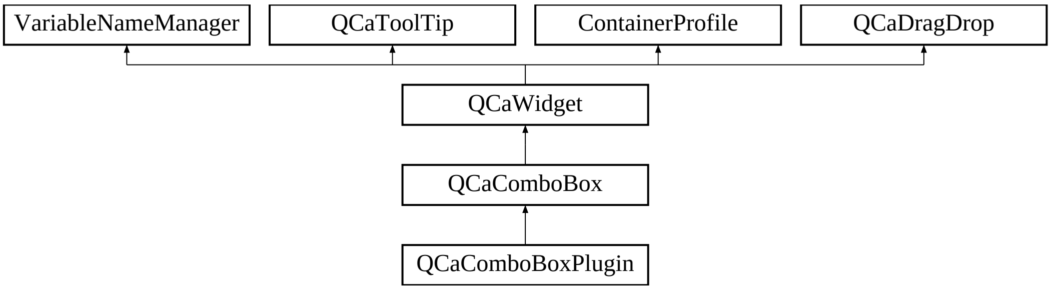
- epicsqt/ca_framework/widgets/include/QCaComboBox.h

- epicsqt/ca_framework/widgets/src/QCaComboBox.cpp

12.29 QCaComboBoxPlugin Class Reference

CA Combobox Widget Plugin.

Inheritance diagram for QCaComboBoxPlugin:



Public Member Functions

- QCaComboBoxPlugin (QWidget *parent=0)
- **QCaComboBoxPlugin** (QString variableName, QWidget *parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()

Properties

- QString **variable**
- QString **variableSubstitutions**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- bool **useDbEnumerations**

12.29.1 Detailed Description

CA Combobox Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.29.2 Constructor & Destructor Documentation

12.29.2.1 QCaComboBoxPlugin::QCaComboBoxPlugin (QWidget ∗ *parent* = 0)

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

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

- epicsqt/ca_framework/plugins/include/QCaComboBoxPlugin.h
- epicsqt/ca_framework/plugins/src/QCaComboBoxPlugin.cpp

12.30 QCaComboBoxPluginManager Class Reference

CA Combobox Widget Plugin Manager for designer.

```
#include <QCaComboBoxPluginManager.h>
```

Public Member Functions

- QCaComboBoxPluginManager (QObject ∗parent=0)
- bool isContainer () const
- bool isInitialized () const

- QIcon icon () const
- QString group () const
- QString includeFile () const
- QString name () const
- QString toolTip () const
- QString whatsThis () const
- QWidget ∗ createWidget (QWidget ∗parent)
- void initialize (QDesignerFormEditorInterface ∗core)

12.30.1 Detailed Description

CA Combobox Widget Plugin Manager for designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.30.2 Constructor & Destructor Documentation

12.30.2.1 QCaComboBoxPluginManager::QCaComboBoxPluginManager (QObject ∗ *parent* = 0)

???

12.30.3 Member Function Documentation

12.30.3.1 QWidget ∗ QCaComboBoxPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaComboBoxPlugin widget.

12.30.3.2 QString QCaComboBoxPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.30.3.3 QIcon QCaComboBoxPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.30.3.4 QString QCaComboBoxPluginManager::includeFile ( ) const

???

???

12.30.3.5 void QCaComboBoxPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.30.3.6 bool QCaComboBoxPluginManager::isContainer ( ) const

???

12.30.3.7 bool QCaComboBoxPluginManager::isInitialized ( ) const

???

12.30.3.8 QString QCaComboBoxPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.30.3.9 QString QCaComboBoxPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.30.3.10 QString QCaComboBoxPluginManager::whatsThis ( ) const

???

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

- epicsqt/ca_framework/plugins/include/QCaComboBoxPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaComboBoxPluginManager.cpp

12.31 QCaConnectionInfo Class Reference

Public Member Functions

- QCaConnectionInfo ()

- QCaConnectionInfo (unsigned short channelStateIn, unsigned short linkStateIn)
- bool isChannelConnected ()
- bool isLinkUp ()

12.31.1 Constructor & Destructor Documentation

12.31.1.1 QCaConnectionInfo::QCaConnectionInfo ( )

Construct an empty instance. By default the channel and link states are unknown

12.31.1.2 QCaConnectionInfo::QCaConnectionInfo (unsigned short *channelStateIn*, unsigned short *linkStateIn*)

Construct an instance given a channel and link state

12.31.2 Member Function Documentation

12.31.2.1 bool QCaConnectionInfo::isChannelConnected ( )

Return true if the channel is connected

12.31.2.2 bool QCaConnectionInfo::isLinkUp ( )

Return true if the link is up

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

- epicsqt/ca_framework/data/include/QCaConnectionInfo.h
- epicsqt/ca_framework/data/src/QCaConnectionInfo.cpp

12.32 QCaDatetime Class Reference

CA alarm info manager.

12.32.1 Detailed Description

CA alarm info manager.

CA Date Time manager.

CA connection info manager.

Version

Revision:

#6

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#4

Date

DateTime:

2010/08/30 16:37:08

Author

andrew.rhyder

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

- epicsqt/ca_framework/data/src/QCaAlarmInfo.cpp

12.33 QCaDateTime Class Reference

Public Member Functions

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

Public Attributes

- unsigned long **nSec**

12.33.1 Constructor & Destructor Documentation

12.33.1.1 QCaDateTime::QCaDateTime ( )

Construct an empty QCa date time

12.33.1.2 QCaDateTime::QCaDateTime (QDateTime *dt*)

Construct a QCa date time set to the same date/time as a conventional QDateTime

12.33.1.3 QCaDateTime::QCaDateTime (unsigned long *seconds,* unsigned long *nanoseconds*)

Construct a QCa date time set to the same date/time as an EPICS time stamp

12.33.2 Member Function Documentation

12.33.2.1 double QCaDateTime::floating (QDateTime *base*)

Returns a double which represents the date and time in seconds (to mS resolution) to the base time

12.33.2.2 void QCaDateTime::operator= (const **QCaDateTime** & *other*)

Copy a QCaDateTime from another

12.33.2.3 QString QCaDateTime::text ( )

Returns a string which represents the date and time

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

- epicsqt/ca_framework/data/include/QCaDateTime.h
- epicsqt/ca_framework/data/src/QCaDateTime.cpp

12.34 QCaDesignerPlugin Class Reference

Collection of CA plugins for QT Designer.

12.34.1 Detailed Description

Collection of CA plugins for QT Designer.

Version

Revision:

#6

Date

DateTime:

2010/09/06 13:29:44

Author

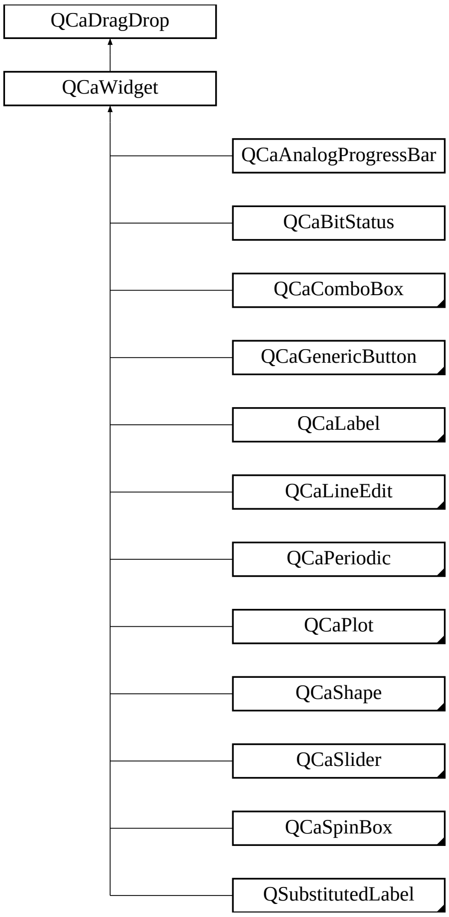
andrew.rhyder

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

- epicsqt/ca_framework/plugins/src/QCaDesignerPlugin.cpp

12.35 QCaDragDrop Class Reference

Inheritance diagram for QCaDragDrop:



Public Member Functions

- QCaDragDrop (QWidget ∗ownerIn)

Protected Member Functions

- void **qcaDragEnterEvent** (QDragEnterEvent ∗event)
- void **qcaDropEvent** (QDropEvent ∗event)
- void **qcaMousePressEvent** (QMouseEvent ∗event)
- virtual void **setDropText** (QString)
- virtual QString **getDropText** ()

12.35.1 Constructor & Destructor Documentation

12.35.1.1 QCaDragDrop::QCaDragDrop (QWidget ∗ *ownerIn*)

This class adds common drag/drop support to all QCa widgets if required.

This class manages the common drag and drop functions leaving the actual QCa widget class to supply and accept the data transfered (typically a PV name).

All QCa widgets (eg, QCaLabel, QCaLineEdit) have an instance of this class as they based on QCaWidget which itself uses this class as a base class.

To implement any sort of drag/drop a QCa widget (like any other widget) must implement the following virtual functions of its base QWidget: dragEnterEvent() dropEvent() mousePressEvent()

To make use of the common QCa drag drop support provided by this class, the above functions can be defined to simply call the equivelent drag/drop functions defined in this class as follows: void dragEnterEvent(QDragEnterEvent ∗event) { qcaDragEnterEvent(event); } void dropEvent(QDropEvent ∗event) { qcaDropEvent(event); } void mousePressEvent(QMouseEvent ∗event) { qcaMousePressEvent(event); }

To allow this class to obtain text for dragging, or deliver text dropped, the QCa widget also needs to implement the following functions defined in this class: setDropText() getDropText()

Typically, the text dragged and dropped is the underlying PV

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

- epicsqt/ca_framework/widgets/include/QCaDragDrop.h
- epicsqt/ca_framework/widgets/src/QCaDragDrop.cpp

12.36 QCaEventFilter Class Reference

Manage CA alarm and severity information.

`#include <QCaEventFilter.h>`

Public Member Functions

- void addFilter (QObject ∗objectIn)

  Add a fitler to an object.
- void deleteFilter (QObject ∗objectIn)

  Remove an event filter from an object.
- bool eventFilter (QObject ∗watched, QEvent ∗e)

  Event filter added to a QObject.

12.36.1 Detailed Description

Manage CA alarm and severity information.

CA event filter for QT.

CA Date Time manager.

Manage CA connection information.

This class allows instances of QCaObject to recieve events posted to them by foriegn CA threads. Each event recipient creates an instance of this class. This class, however, manages only a single event filter for all instances of itself.

Version

Revision:

#5

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#4

Date

DateTime:

2010/08/30 16:37:08

Author

andrew.rhyder

12.36.2 Member Function Documentation

12.36.2.1 void QCaEventFilter::addFilter (QObject ∗ *eventObject*)

Add a fitler to an object.

Add the event filter to an object.

Protect access to the list of objects

Check if the filter is already present. If so, just increase the reference count

If the filter was not present (known because the search reached the end of the list) then add the object to the list of objects using the filter and add the filter to the object.

Add the object to the list

Add the filter

12.36.2.2 void QCaEventFilter::deleteFilter (QObject ∗ *eventObject*)

Remove an event filter from an object.

Remove the event filter from an object.

Protect access to the list of objects

Search for the QObject the filter is to be removed from.

Get the next item and check for a match

Item matched. Reduce the count of QCaObjects relying on this filter and remove the filter if there are no more QCaObject requiring the filter

If the item containing the object was not found, log an error

??? log an error better than this

12.36.2.3 bool QCaEventFilter::eventFilter (QObject ∗ *watched,* QEvent ∗ *e*)

Event filter added to a QObject.

Filter events for a QObject, processing events posted by a QCaObject. This is an overloaded function of QObject As the originating QCaObject is referenced in the event, events can be processed back within the originating QCaObject using the QCaObject::processEvent() method.

The event is our update event. Pass it back to the originating QCaObject for processing if still required

The event has been dealt with. Indicate it requires no further processing (return true)

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

- epicsqt/ca_framework/data/include/QCaEventFilter.h
- epicsqt/ca_framework/data/src/QCaEventFilter.cpp

12.37 QCaEventItem Class Reference

Simple container for a list QCaEventUpdate instances.

```
#include <QCaEventUpdate.h>
```

Public Member Functions

- **QCaEventItem** (QCaEventUpdate ∗newEvent)

Public Attributes

- QCaEventUpdate ∗ **event**

12.37.1 Detailed Description

Simple container for a list QCaEventUpdate instances.

This class is simply a reference to an update event (an instance of class QCaEventUpdate) This class exists so that a list of references to outstanding events can be maintained.

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

- epicsqt/ca_framework/data/include/QCaEventUpdate.h

12.38 QCaEventUpdate Class Reference

Packages CA callback data as a Qt event.

```
#include <QCaEventUpdate.h>
```

Public Member Functions

- QCaEventUpdate (qcaobject::QCaObject ∗emitterObjectIn, long newReason, void ∗newDataPtr)

Public Attributes

- bool **acceptThisEvent**
- qcaobject::QCaObject ∗ emitterObject

  If true, the originator of this event has been deleted. Ignore the event.
- long reason

  The object to process the data and emit a signal.
- void ∗ dataPtr

  Callback reason. Actually of type caobject::callback_reasons.

Static Public Attributes

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

  The Qt event type for update events. This is allocated by Qt once at application startup.

12.38.1 Detailed Description

Packages CA callback data as a Qt event.

Packages data and status generated by a CA callback as a Qt event.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

This class packages data and status generated by a CA callback as a Qt event. When data or status updates must be passed from a foreign CA thread to a Qt thread an instance of this class is created, populated with the CA information and posted through the event mechanism to a Qt thread.

Version

Revision:

#5

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.38.2 Constructor & Destructor Documentation

12.38.2.1 QCaEventUpdate::QCaEventUpdate (qcaobject::QCaObject ∗ *emitterObjectIn*, long *newReason*, void ∗ *newDataPtr*)

Create an event holding a CA update

Set up the event mechanism

Populate the update

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

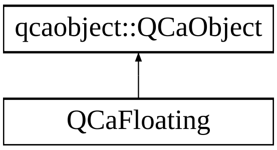
- epicsqt/ca_framework/data/include/QCaEventUpdate.h
- epicsqt/ca_framework/data/src/QCaEventUpdate.cpp

12.39 QCaFloating Class Reference

Floating specific wrapper for QCaObject.

`#include <QCaFloating.h>`

Inheritance diagram for QCaFloating:



Public Slots

- void writeFloating (const double &data)

Signals

- 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

- QCaFloating (QString recordName, QObject ∗eventObject, QCaFloatingFormatting ∗floatingFormattingIn, unsigned int variableIndexIn)
- **QCaFloating** (QString recordName, QObject ∗eventObject, QCaFloatingFormatting ∗floatingFormattingIn, unsigned int variableIndexIn, UserMessage ∗userMessageIn)

12.39.1 Detailed Description

Floating specific wrapper for QCaObject.

Version

Revision:

#5

Date

DateTime:

2010/09/06 13:16:04

Author

andrew.rhyder

Version

Revision:

#4

Date

DateTime:

2010/08/30 16:37:08

Author

andrew.rhyder

12.39.2 Constructor & Destructor Documentation

12.39.2.1 QCaFloating::QCaFloating (QString *recordName,* QObject ∗ *eventObject,* QCaFloatingFormatting ∗ *floatingFormattingIn,* unsigned int *variableIndexIn*)

???

12.39.3 Member Function Documentation

12.39.3.1 void QCaFloating::writeFloating (const double & *data*) [slot]

Take a new floating value and write it to the database. The type of data formatted (text, floating, integer, etc) will be determined by the record data type, How the floating is parsed will be determined by the floating formatting. For example, floating to string may require always including a sign.

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

- epicsqt/ca_framework/data/include/QCaFloating.h
- epicsqt/ca_framework/data/src/QCaFloating.cpp

12.40 QCaFloatingFormatting Class Reference

Provides textual formatting for QCaFloating data.

`#include <QCaFloatingFormatting.h>`

Public Types

- enum **formats** {

  FORMAT_e = 'e', **FORMAT_E** = 'E', **FORMAT_f** = 'f', **FORMAT_g** = 'g',

  FORMAT_G = 'G' }

Public Member Functions

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

12.40.1 Detailed Description

Provides textual formatting for QCaFloating data.

Version

Revision:

#4

Date

DateTime:

2010/08/30 16:37:08

Author

andrew.rhyder

12.40.2 Constructor & Destructor Documentation

12.40.2.1 QCaFloatingFormatting::QCaFloatingFormatting ( )

???

Default formatting properties.

12.40.3 Member Function Documentation

12.40.3.1 double QCaFloatingFormatting::formatFloating (const QVariant & *value*)

Generate an floating point number given a value, using formatting defined within this class. The value may be an array of variants or a single variant

12.40.3.2 QVector< double > QCaFloatingFormatting::formatFloatingArray (const QVariant & *value*)

Generate an floating point number array given a value, using formatting defined within this class.

12.40.3.3 QVariant QCaFloatingFormatting::formatValue (const double & *floatingValue,* generic::generic_types *valueType*)

Generate a value given a floating point number, using formatting defined within this class. The formatting mainly applies if formatting as a string. For example, was is the number base? should a sign always be included? are leading zeros requried? The formatting could include properties related to other types. For example, generate an error if attempting to convert a negative floating point number to an unsigned integer.

12.40.3.4 int QCaFloatingFormatting::getFormat ( )

Get the format. Relevent when formatting the floating point number as a string.

12.40.3.5 unsigned int QCaFloatingFormatting::getPrecision ( )

Get the precision. Relevent when formatting the floating point number as a string.

12.40.3.6 void QCaFloatingFormatting::setFormat (formats *formatIn*)

Set the format. Relevent when formatting the floating point number as a string.

12.40.3.7 void QCaFloatingFormatting::setPrecision (unsigned int *precisionIn*)

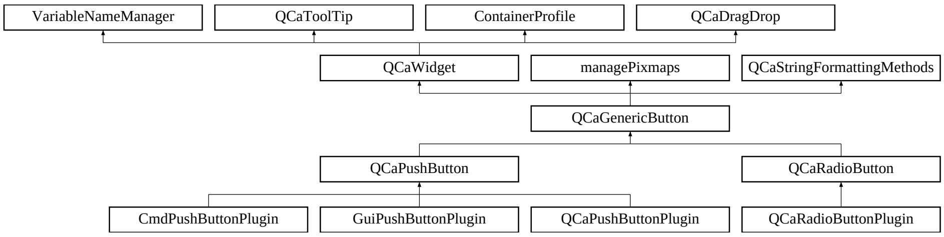
Set the precision. Relevent when formatting the floating point number as a string.

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

- epicsqt/ca_framework/data/include/QCaFloatingFormatting.h
- epicsqt/ca_framework/data/src/QCaFloatingFormatting.cpp

12.41 QCaGenericButton Class Reference

Inheritance diagram for QCaGenericButton:



Public Types

- enum **updateOptions** { **UPDATE_TEXT**, **UPDATE_ICON**, **UPDATE_TEXT_AND_-ICON**, **UPDATE_STATE** }

Public Member Functions

- QCaGenericButton (QWidget ∗owner)
- bool isEnabled () const
- void setGenericEnabled (const bool &state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()

- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setUpdateOption** (updateOptions updateOptionIn)
- updateOptions **getUpdateOption** ()
- void **setTextAlignment** (Qt::Alignment alignment)
- Qt::Alignment **getTextAlignment** ()
- void **setPassword** (QString password)
- QString **getPassword** ()
- void **setWriteOnPress** (bool writeOnPress)
- bool **getWriteOnPress** ()
- void **setWriteOnRelease** (bool writeOnRelease)
- bool **getWriteOnRelease** ()
- void **setWriteOnClick** (bool writeOnClick)
- bool **getWriteOnClick** ()
- void **setPressText** (QString pressText)
- QString **getPressText** ()
- void **setReleaseText** (QString releaseTextIn)
- QString **getReleaseText** ()
- void **setClickText** (QString clickTextIn)
- QString **getClickText** ()
- void **setClickCheckedText** (QString clickCheckedTextIn)
- QString **getClickCheckedText** ()
- void **setProgram** (QString program)
- QString **getProgram** ()
- void **setArguments** (QStringList arguments)
- QStringList **getArguments** ()
- void **setGuiName** (QString guiName)
- QString **getGuiName** ()
- void **setCreationOption** (ASguiForm::creationOptions creationOption)
- ASguiForm::creationOptions **getCreationOption** ()
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- void **onGeneralMessage** (QString message)

Protected Member Functions

- void connectionChanged (QCaConnectionInfo &connectionInfo)
- void setGenericButtonText (const QString &text, QCaAlarmInfo &alarmInfo, QCaDateTime &, const unsigned int &variableIndex)
- void userPressed ()
- void userReleased ()
- void userClicked (bool checked)
- void **launchGui** (QString guiName, ASguiForm::creationOptions creationOption)

- virtual updateOptions **getDefaultUpdateOption** ()=0
- void establishConnection (unsigned int variableIndex)
- bool **getIsConnected** ()
- void setup ()

Protected Attributes

- Qt::Alignment **textAlignment**
- QString **password**
- bool **writeOnPress**
- bool **writeOnRelease**
- bool **writeOnClick**
- QString **releaseText**
- QString pressText

  Text to write on a button release.
- QString clickText

  Text to write on a button press.
- QString clickCheckedText

  Text to write on a button click.
- QString program

  Text to write on a button click when the button moves to the checked position.
- QStringList arguments

  Program to run.
- QString guiName

  Program arguments.
- ASguiForm::creationOptions creationOption

  GUI file name to launch.
- bool **localEnabled**
- updateOptions **updateOption**
- bool **allowDrop**
- QString **labelText**

12.41.1 Constructor & Destructor Documentation

12.41.1.1 QCaGenericButton::QCaGenericButton (QWidget ∗ *owner*)

This class is a CA aware push button widget based on the Qt push button widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

12.41.2 Member Function Documentation

12.41.2.1 void QCaGenericButton::connectionChanged (QCaConnectionInfo & *connectionInfo*) [protected]

Act on a connection change. Change how the label looks and change the tool tip This is the slot used to recieve connection updates from a QCaObject based class.

If connected enabled the widget if required.

If disconnected always disable the widget.

12.41.2.2 void QCaGenericButton::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.41.2.3 bool QCaGenericButton::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Reimplemented in QCaPushButton, and QCaRadioButton.

12.41.2.4 void QCaGenericButton::setGenericButtonText (const QString & *text*, QCaAlarmInfo & *alarmInfo*, QCaDateTime & , const unsigned int & *variableIndex*) [protected]

Implement a slot to set the current text of the push button This is the slot used to recieve data updates from a QCaObject based class.

Signal a database value change to any Link widgets

Update the button state if required Display checked if text matches what is written when checked

Update the text if required

Update the icon if required

If in alarm, display as an alarm

12.41.2.5 void QCaGenericButton::setGenericEnabled (const bool & *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

12.41.2.6 void QCaGenericButton::setup ( ) [protected, virtual]

Setup common to all button constructors

Implements QCaWidget.

12.41.2.7 void QCaGenericButton::userClicked (bool *checked*) [protected]

Button click event.

Get the variable to write to

If a QCa object is present (if there is a variable to write to) and the object is set up to write when the user clicks the button then write the value

12.41.2.8 void QCaGenericButton::userPressed ( ) [protected]

Button press event.

Get the variable to write to

If a QCa object is present (if there is a variable to write to) and the object is set up to write when the user presses the button then write the value

12.41.2.9 void QCaGenericButton::userReleased ( ) [protected]

Button release event.

Get the variable to write to

If a QCa object is present (if there is a variable to write to) and the object is set up to write when the user releases the button then write the value

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

- epicsqt/ca_framework/widgets/include/QCaGenericButton.h
- epicsqt/ca_framework/widgets/src/QCaGenericButton.cpp

12.42 QCaInstalledFiltersListItem Class Reference

Public Member Functions

- **QCaInstalledFiltersListItem** (QObject ∗eventObjectIn)

Public Attributes

- QObject ∗ eventObject

  QObject that an event filter has been added to.

- long referenceCount

  Count of how many QCaObjects expect the event filter to be on the QObject.

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

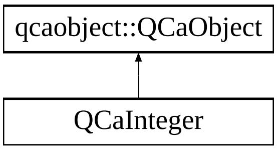
- epicsqt/ca_framework/data/include/QCaEventFilter.h

12.43 QCaInteger Class Reference

Integer specific wrapper for QCaObject.

`#include <QCaInteger.h>`

Inheritance diagram for QCaInteger:



Public Slots

- void writeInteger (const long &data)

Signals

- void **integerChanged** (const long &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- QCaInteger (QString recordName, QObject *eventObject, QCaIntegerFormatting *integerFormattingIn, unsigned int variableIndexIn)
- **QCaInteger** (QString recordName, QObject *eventObject, QCaIntegerFormatting *integerFormattingIn, unsigned int variableIndexIn, UserMessage *userMessageIn)

12.43.1 Detailed Description

Integer specific wrapper for QCaObject.

Version

Revision:

#6

Date

DateTime:

2010/09/06 13:16:04

Author

andrew.rhyder

Version

Revision:

#5

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.43.2 Constructor & Destructor Documentation

12.43.2.1 QCaInteger::QCaInteger (QString *recordName,* QObject ∗ *eventObject,* QCaIntegerFormatting ∗ *integerFormattingIn,* unsigned int *variableIndexIn*)

???

12.43.3 Member Function Documentation

12.43.3.1 void QCaInteger::writeInteger (const long & *data*) [slot]

Take a new integer value and write it to the database. The type of data formatted (text, floating, integer, etc) will be determined by the record data type, How the integer is parsed will be determined by the integer formatting. For example, integer to string may require always including a sign.

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

- epicsqt/ca_framework/data/include/QCaInteger.h
- epicsqt/ca_framework/data/src/QCaInteger.cpp

12.44 QCaIntegerFormatting Class Reference

Provides textual formatting for QCaInteger data.

```
#include <QCaIntegerFormatting.h>
```

Public Member Functions

- QCaIntegerFormatting ()
- long formatInteger (const QVariant &value)
- QVariant formatValue (const long &integerValue, generic::generic_types valueType)
- void setRadix (unsigned int radix)
- unsigned int **getPrecision** ()
- unsigned int getRadix ()

12.44.1 Detailed Description

Provides textual formatting for QCaInteger data.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.44.2 Constructor & Destructor Documentation

12.44.2.1 QCaIntegerFormatting::QCaIntegerFormatting ( )

???

Default formatting properties.

12.44.3 Member Function Documentation

12.44.3.1 long QCaIntegerFormatting::formatInteger (const QVariant & *value*)

Generate an integer given a value, using formatting defined within this class.

12.44.3.2 QVariant QCaIntegerFormatting::formatValue (const long & *integerValue,* generic::generic_types *valueType*)

Generate a value given an integer, using formatting defined within this class. The formatting mainly applies if formatting as a string. For example, was is the number base? should a sign always be included? are leading zeros requried? The formatting could include properties related to other types. For example, generate an error if attempting to convert a negative integer to an unsigned integer.

12.44.3.3 unsigned int QCaIntegerFormatting::getRadix ( )

Get the numerical base. See setRadix() for the use of 'radix'.

12.44.3.4 void QCaIntegerFormatting::setRadix (unsigned int *radixIn*)

Set the numer system base. Relevent when formatting the string as an interger. Any radix of 2 or more is accepted. Check the conversion code that uses this number to see what values are expected. At the time of writing (16/2/9) it is anticipated that floating point numbers will always be base 10 and integer numbers will be base 2, 8, 10, or 16. ??? if radix processing beocomes significant, create a radix class that can be used for both QCaIntegerFormatting and QCaStringFormatting.

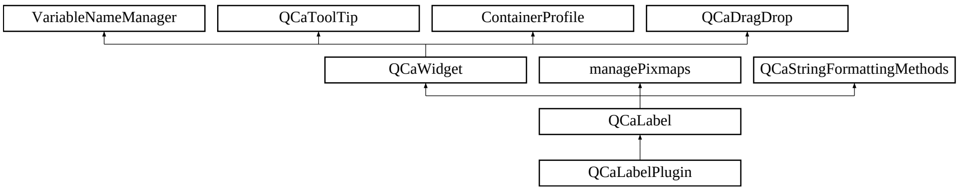
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/data/include/QCaIntegerFormatting.h
- epicsqt/ca_framework/data/src/QCaIntegerFormatting.cpp

12.45 QCaLabel Class Reference

CA Label Widget.

Inheritance diagram for QCaLabel:



Public Types

- enum **updateOptions** { **UPDATE_TEXT**, **UPDATE_PIXMAP** }

Public Slots

- void requestEnabled (const bool &state)

Signals

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

Public Member Functions

- QCaLabel (QWidget ∗parent=0)
- QCaLabel (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setUpdateOption** (updateOptions updateOptionIn)
- updateOptions **getUpdateOption** ()
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setRunVisible** (bool visibleIn)
- bool **getRunVisible** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- bool **caEnabled**
- bool **caVisible**
- updateOptions **updateOption**
- bool **allowDrop**

12.45.1 Detailed Description

CA Label Widget.

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.45.2 Constructor & Destructor Documentation

12.45.2.1 QCaLabel::QCaLabel (QWidget ∗ *parent* = 0)

This class is a CA aware label widget based on the Qt label widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.45.2.2 QCaLabel::QCaLabel (const QString & *variableNameIn*, QWidget ∗ *parent* = 0)

Constructor with known variable

12.45.3 Member Function Documentation

12.45.3.1 void QCaLabel::establishConnection (unsigned int *variableIndex*) `[protected, virtual]`

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.45.3.2 bool QCaLabel::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.45.3.3 void QCaLabel::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled, but will use our own setEnabled which will allow alarm states to override current enabled state

12.45.3.4 void QCaLabel::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

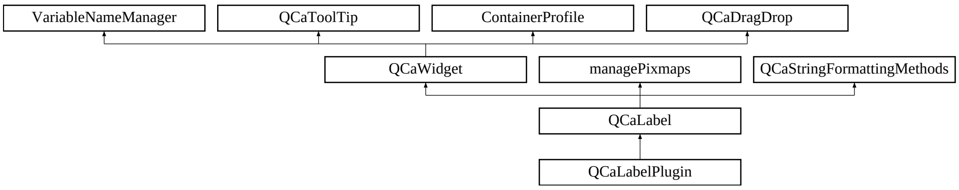
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaLabel.h
- epicsqt/ca_framework/widgets/src/QCaLabel.cpp

12.46 QCaLabelPlugin Class Reference

CA Label Widget Plugin for designer.

Inheritance diagram for QCaLabelPlugin:



Public Types

- enum **UpdateOptions** { **Text** = QCaLabel::UPDATE_TEXT, **Picture** = QCaLabel::UPDATE_PIXMAP }
- enum **Formats** {

  Default = QCaStringFormatting::FORMAT_DEFAULT, **Floating** = QCaStringFormatting::FORMAT_FLOATING, **Integer** = QCaStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QCaStringFormatting::FORMAT_UNSIGNEDINTEGER,

Time = QCaStringFormatting::FORMAT_TIME, **LocalEnumeration** = QCaStringFormatting::FORMAT_LOCAL_ENUMERATE }

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

Public Member Functions

- QCaLabelPlugin (QWidget ∗parent=0)

    Constructors.
- **QCaLabelPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setUpdateOptionProperty** (UpdateOptions updateOption)
- UpdateOptions **getUpdateOptionProperty** ()
- void **setPixmap0Property** (QPixmap pixmap)
- void **setPixmap1Property** (QPixmap pixmap)
- void **setPixmap2Property** (QPixmap pixmap)
- void **setPixmap3Property** (QPixmap pixmap)
- void **setPixmap4Property** (QPixmap pixmap)
- void **setPixmap5Property** (QPixmap pixmap)
- void **setPixmap6Property** (QPixmap pixmap)
- void **setPixmap7Property** (QPixmap pixmap)
- QPixmap **getPixmap0Property** ()
- QPixmap **getPixmap1Property** ()
- QPixmap **getPixmap2Property** ()
- QPixmap **getPixmap3Property** ()
- QPixmap **getPixmap4Property** ()
- QPixmap **getPixmap5Property** ()
- QPixmap **getPixmap6Property** ()
- QPixmap **getPixmap7Property** ()
- void **setFormatProperty** (Formats format)
- Formats **getFormatProperty** ()
- void **setNotationProperty** (Notations notation)
- Notations **getNotationProperty** ()
- void **setArrayActionProperty** (ArrayActions arrayAction)
- ArrayActions **getArrayActionProperty** ()

Properties

- QString variable
- QString **variableSubstitutions**
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- UpdateOptions updateOption

  Update options (text, pixmap, or both)
- QPixmap pixmap0

  Pixmaps.
- QPixmap **pixmap1**
- QPixmap **pixmap2**
- QPixmap **pixmap3**
- QPixmap **pixmap4**
- QPixmap **pixmap5**
- QPixmap **pixmap6**
- QPixmap **pixmap7**
- unsigned int

  String formatting properties.
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- QString **localEnumeration**
- Formats **format**
- Notations **notation**
- ArrayActions **arrayAction**
- bool **visible**

12.46.1 Detailed Description

CA Label Widget Plugin for designer.

Version

Revision:

#6

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.46.2 Constructor & Destructor Documentation

12.46.2.1 QCaLabelPlugin::QCaLabelPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.46.3 Property Documentation

12.46.3.1 QString QCaLabelPlugin::variable `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaLabelPlugin.h
- epicsqt/ca_framework/plugins/src/QCaLabelPlugin.cpp

12.47 QCaLabelPluginManager Class Reference

CA Label Widget Plugin Manager for designer.

`#include <QCaLabelPluginManager.h>`

Public Member Functions

- QCaLabelPluginManager (QObject ∗parent=0)
- bool isContainer () const
- bool isInitialized () const
- QIcon icon () const
- QString group () const
- QString includeFile () const

- QString name () const
- QString toolTip () const
- QString whatsThis () const
- QWidget ∗ createWidget (QWidget ∗parent)
- void initialize (QDesignerFormEditorInterface ∗core)

12.47.1 Detailed Description

CA Label Widget Plugin Manager for designer.

???

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.47.2 Constructor & Destructor Documentation

12.47.2.1 QCaLabelPluginManager::QCaLabelPluginManager (QObject ∗ *parent* = 0)

???

12.47.3 Member Function Documentation

12.47.3.1 QWidget ∗ QCaLabelPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLabelPlugin widget.

12.47.3.2 QString QCaLabelPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.47.3.3 QIcon QCaLabelPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.47.3.4 QString QCaLabelPluginManager::includeFile ( ) const

???

???

12.47.3.5 void QCaLabelPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.47.3.6 bool QCaLabelPluginManager::isContainer ( ) const

???

12.47.3.7 bool QCaLabelPluginManager::isInitialized ( ) const

???

12.47.3.8 QString QCaLabelPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.47.3.9 QString QCaLabelPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.47.3.10 QString QCaLabelPluginManager::whatsThis ( ) const

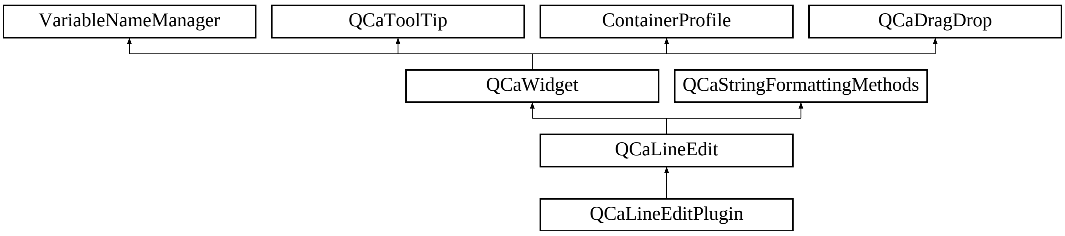
???

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

- epicsqt/ca_framework/plugins/include/QCaLabelPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaLabelPluginManager.cpp

12.48 QCaLineEdit Class Reference

Inheritance diagram for QCaLineEdit:



Public Slots

- void requestEnabled (const bool &state)

  Act on the user signaling text editing is complete (pressing return)

Signals

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

Public Member Functions

- QCaLineEdit (QWidget ∗parent=0)
- QCaLineEdit (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setWriteOnLoseFocus** (bool writeOnLoseFocus)
- bool **getWriteOnLoseFocus** ()
- void **setWriteOnEnter** (bool writeOnEnter)
- bool **getWriteOnEnter** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setConfirmWrite** (bool confirmWrite)
- bool **getConfirmWrite** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- bool **writeOnLoseFocus**
- bool writeOnEnter

  Write changed value to database when widget object loses focus (user moves from widget)
- bool localEnabled

  Write changed value to database when enter is pressed with focus on the widget.
- bool confirmWrite

  Override the default widget setEnabled to allow alarm states to override current enabled state.
- bool allowDrop

  Request confirmation before writing a value.

12.48.1 Constructor & Destructor Documentation

12.48.1.1 QCaLineEdit::QCaLineEdit (QWidget ∗ *parent* = 0)

This class is a CA aware line edit widget based on the Qt line edit widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.48.1.2 QCaLineEdit::QCaLineEdit (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Constructor with known variable

12.48.2 Member Function Documentation

12.48.2.1 void QCaLineEdit::establishConnection (unsigned int *variableIndex*) `[protected, virtual]`

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.48.2.2 bool QCaLineEdit::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.48.2.3 void QCaLineEdit::requestEnabled (const bool & *state*) `[slot]`

Act on the user signaling text editing is complete (pressing return)

Slot similar to default widget setEnabled, but will use our own setEnabled which will allow alarm states to override current enabled state

12.48.2.4 void QCaLineEdit::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

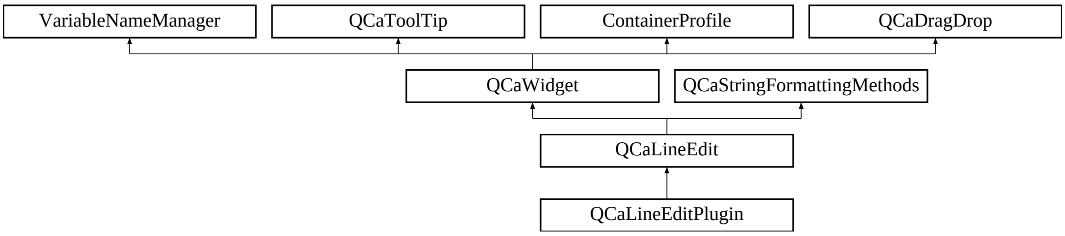
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaLineEdit.h
- epicsqt/ca_framework/widgets/src/QCaLineEdit.cpp

12.49 QCaLineEditPlugin Class Reference

CA Line Edit Widget Plugin.

Inheritance diagram for QCaLineEditPlugin:



Public Types

- enum **Formats** {

  Default = QCaStringFormatting::FORMAT_DEFAULT, **Floating** = QCaStringFormatting::FORMAT_FLOATING, **Integer** = QCaStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QCaStringFormatting::FORMAT_UNSIGNEDINTEGER,

Time = QCaStringFormatting::FORMAT_TIME }

- enum **Notations** { **Fixed** = QCaStringFormatting::NOTATION_FIXED, **Scientific** = QCaStringFormatting::NOTATION_SCIENTIFIC, **Automatic** = QCaStringFormatting::NOTATION_-AUTOMATIC }

Public Member Functions

- QCaLineEditPlugin (QWidget ∗parent=0)

  Constructors.
- **QCaLineEditPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setFormatProperty** (Formats format)
- Formats **getFormatProperty** ()
- void **setNotationProperty** (Notations notation)
- Notations **getNotationProperty** ()

Properties

- QString variable
- QString **variableSubstitutions**
- bool **writeOnLoseFocus**
- bool writeOnEnter

  Write changed value to database when widget object loses focus (user moves from widget)
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool allowDrop

  Request confirmation before writing a value.
- bool confirmWrite

  Override the default widget setEnabled to allow alarm states to override current enabled state.
- unsigned int

  String formatting properties.
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- Formats **format**
- Notations **notation**

12.49.1 Detailed Description

CA Line Edit Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.49.2 Constructor & Destructor Documentation

12.49.2.1 QCaLineEditPlugin::QCaLineEditPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.49.3 Property Documentation

12.49.3.1 QString QCaLineEditPlugin::variable `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaLineEditPlugin.h
- epicsqt/ca_framework/plugins/src/QCaLineEditPlugin.cpp

12.50 QCaLineEditPluginManager Class Reference

CA Line Edit Widget Plugin Manager for designer.

```
#include <QCaLineEditPluginManager.h>
```

Public Member Functions

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

12.50.1 Detailed Description

CA Line Edit Widget Plugin Manager for designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.50.2 Constructor & Destructor Documentation

12.50.2.1 QCaLineEditPluginManager::QCaLineEditPluginManager (QObject ∗ *parent* = 0)

???

12.50.3 Member Function Documentation

12.50.3.1 QWidget ∗ QCaLineEditPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLineEditPlugin widget.

12.50.3.2 QString QCaLineEditPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.50.3.3 QIcon QCaLineEditPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.50.3.4 QString QCaLineEditPluginManager::includeFile ( ) const

???

???

12.50.3.5 void QCaLineEditPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.50.3.6 bool QCaLineEditPluginManager::isContainer ( ) const

???

12.50.3.7 bool QCaLineEditPluginManager::isInitialized ( ) const

???

12.50.3.8 QString QCaLineEditPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.50.3.9 QString QCaLineEditPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.50.3.10 QString QCaLineEditPluginManager::whatsThis ( ) const

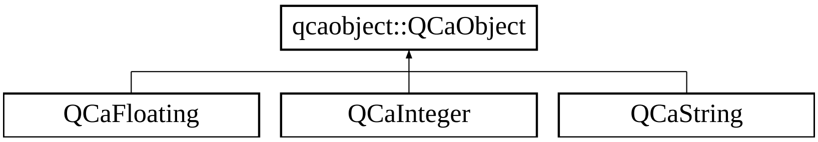
???

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

- epicsqt/ca_framework/plugins/include/QCaLineEditPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaLineEditPluginManager.cpp

12.51 qcaobject::QCaObject Class Reference

Inheritance diagram for qcaobject::QCaObject:



Public Slots

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

Signals

- void **dataChanged** (const QVariant &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp)
- void **connectionChanged** (QCaConnectionInfo &connectionInfo)

Public Member Functions

- QCaObject (const QString &recordName, QObject ∗eventObject)
- **QCaObject** (const QString &recordName, QObject ∗eventObject, UserMessage ∗userMessageIn)
- virtual ∼QCaObject ()
- bool subscribe ()
- bool singleShotRead ()
- bool dataTypeKnown ()
- bool createChannel ()

  State machine access functions.
- void deleteChannel ()
- bool createSubscription ()
- bool getChannel ()

- bool putChannel ()
- bool isChannelConnected ()
- void startConnectionTimer ()
- void stopConnectionTimer ()
- void setUserMessage (UserMessage ∗userMessageIn)
- void enableWriteCallbacks (bool enable)
- bool isWriteCallbacksEnabled ()
- QString getEgu ()
- QStringList getEnumerations ()
- unsigned int getPrecision ()
- double getDisplayLimitUpper ()
- double getDisplayLimitLower ()
- double getAlarmLimitUpper ()
- double getAlarmLimitLower ()
- double getWarningLimitUpper ()
- double getWarningLimitLower ()
- double getControlLimitUpper ()
- double getControlLimitLower ()

Static Public Member Functions

- static void processEventStatic (QCaEventUpdate ∗dataUpdateEvent)

Protected Member Functions

- generic::generic_types getDataType ()

12.51.1 Constructor & Destructor Documentation

12.51.1.1 QCaObject::QCaObject (const QString & *newRecordName,* QObject ∗ *newEventHandler*)

The event object can be any Qt object with an event queue. A filter will be inserted (and removed) by this class to catch events from this class and pass them back to this class for processing. In other words, the event object does not need to be set up in any way. It just need to have a suitable event loop running.

12.51.1.2 QCaObject::∼QCaObject ( ) [virtual]

Destructor. Remove the channel and ensure that any unprocessed events posted by this obect will be ignored when they pop out of the event queue. Also, remove the event filter if this is the last QCaObject.

12.51.2 Member Function Documentation

12.51.2.1 bool QCaObject::createChannel ( )

State machine access functions.

Create a channel

12.51.2.2 bool QCaObject::createSubscription ( )

Create a subscription

12.51.2.3 bool QCaObject::dataTypeKnown ( )

Returns true if the type of data being read or to be written is known. The data type will be unknown until a connection is established

12.51.2.4 void QCaObject::deleteChannel ( )

Delete a channel

12.51.2.5 void QCaObject::enableWriteCallbacks (bool *enable*)

Set if callbacks are required on write completion. (default is write with no callback) Note, this is not just for better write status, if affects the behaviour of the write as follows: When using write with callback, then record will finish processing before accepting next write. Writing with callback may be required when writing code that is tightly integrated with record processing and code nneds to know processing has completed. Writing with no callback is more desirable when a detachement from record processing is required, for example in a GUI after issuing a motor record move a motor stop command will take effect immedietly if writing without callback, but will only take affect after the move has finished if writing with callback.

12.51.2.6 double QCaObject::getAlarmLimitLower ( )

Return the alarm lower limit, if any

12.51.2.7 double QCaObject::getAlarmLimitUpper ( )

Return the alarm upper limit, if any

12.51.2.8 bool QCaObject::getChannel ( )

Read from a PV

12.51.2.9 double QCaObject::getControlLimitLower ( )

Return the control lower limit, if any

12.51.2.10 double QCaObject::getControlLimitUpper ( )

Return the control upper limit, if any

12.51.2.11 generic::generic_types QCaObject::getDataType ( ) [protected]

Returns the type of data being read or to be written. This will be Type::UNKNOWN if a connection has never been established.

12.51.2.12 double QCaObject::getDisplayLimitLower ( )

Return the display lower limit, if any

12.51.2.13 double QCaObject::getDisplayLimitUpper ( )

Return the display upper limit, if any

12.51.2.14 QString QCaObject::getEgu ( )

Return the engineering units, if any

12.51.2.15 QStringList QCaObject::getEnumerations ( )

Return the enumerations strings, if any

12.51.2.16 unsigned int QCaObject::getPrecision ( )

Return the precision, if any

12.51.2.17 double QCaObject::getWarningLimitLower ( )

Return the warning lower limit, if any

12.51.2.18 double QCaObject::getWarningLimitUpper ( )

Return the warning upper limit, if any

12.51.2.19 bool QCaObject::isChannelConnected ( )

Determine if the channel is currently connected

12.51.2.20 bool QCaObject::isWriteCallbacksEnabled ( )

Determine if callbacks are required on write completion.

12.51.2.21 void QCaObject::processEventStatic (QCaEventUpdate ∗ *dataUpdateEvent*) [static]

Process self generated events and only accept them if the originating QCaObject still exists. Note, this is a static method

12.51.2.22 bool QCaObject::putChannel ( )

Write to a PV

12.51.2.23 void QCaObject::resendLastData ( ) [slot]

Re-emit the last data emited, if any

12.51.2.24 void QCaObject::setUserMessage (UserMessage ∗ *userMessageIn*)

Setup the user message mechanism. After calling this method the QCaObject can report errors to the user.

12.51.2.25 bool QCaObject::singleShotRead ( )

Initiate a single shot read

12.51.2.26 void QCaObject::startConnectionTimer ( )

Wait one minute for a connection, then re-attempt the connection

12.51.2.27 void QCaObject::stopConnectionTimer ( )

Connection has been achieved within the expected time, stop the timer used to wait for a connection

12.51.2.28 bool QCaObject::subscribe ( )

Subcribe

12.51.2.29 bool QCaObject::writeData (const QVariant & *newData*) [slot]

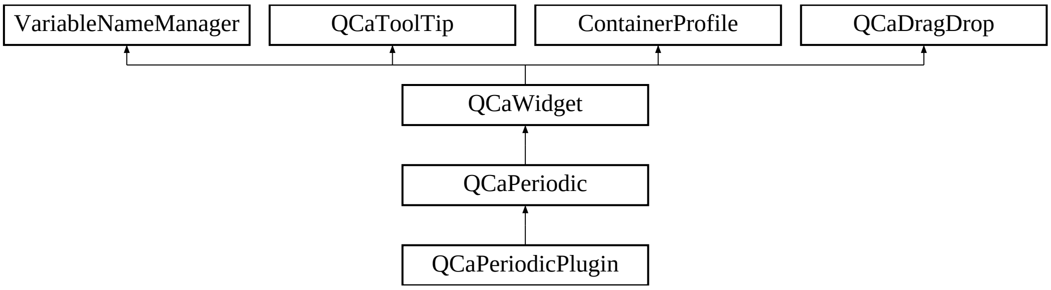
Write a data out to channel

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

- epicsqt/ca_framework/data/include/QCaObject.h
- epicsqt/ca_framework/data/src/QCaObject.cpp

12.52 QCaPeriodic Class Reference

Inheritance diagram for QCaPeriodic:



Classes

- struct elementInfoStruct
- struct userInfoStructArray

Public Types

- enum **variableTypes** {

  VARIABLE_TYPE_NUMBER, **VARIABLE_TYPE_ATOMIC_WEIGHT**, **VARIABLE_TYPE_MELTING_POINT**, **VARIABLE_TYPE_BOILING_POINT**,

  VARIABLE_TYPE_DENSITY, **VARIABLE_TYPE_GROUP**, **VARIABLE_TYPE_IONIZATION_ENERGY**, **VARIABLE_TYPE_USER_VALUE_1**,

  VARIABLE_TYPE_USER_VALUE_2 }
- enum **presentationOptions** { **PRESENTATION_BUTTON_AND_LABEL**, **PRESENTATION_BUTTON_ONLY**, **PRESENTATION_LABEL_ONLY** }

Public Slots

- void requestEnabled (const bool &state)

Signals

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

Public Member Functions

- QCaPeriodic (QWidget ∗parent=0)
- QCaPeriodic (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (const bool &state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setPresentationOption** (presentationOptions presentationOptionIn)
- presentationOptions **getPresentationOption** ()
- void **setVariableType1** (variableTypes variableType1In)
- variableTypes **getVariableType1** ()
- void **setVariableType2** (variableTypes variableType2In)
- variableTypes **getVariableType2** ()
- void **setVariableTolerance1** (double variableTolerance1In)
- double **getVariableTolerance1** ()
- void **setVariableTolerance2** (double variableTolerance2In)
- double **getVariableTolerance2** ()
- void **setUserInfo** (QString userInfo)
- QString **getUserInfo** ()

Public Attributes

- userInfoStruct **userInfo** [NUM_ELEMENTS]

Static Public Attributes

- static elementInfoStruct elementInfo [NUM_ELEMENTS]

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

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

12.52.1 Constructor & Destructor Documentation

12.52.1.1 QCaPeriodic::QCaPeriodic (QWidget ∗ *parent* = 0)

Constructor with no initialisation

12.52.1.2 QCaPeriodic::QCaPeriodic (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Constructor with known variable

12.52.2 Member Function Documentation

12.52.2.1 void QCaPeriodic::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.52.2.2 bool QCaPeriodic::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

12.52.2.3 void QCaPeriodic::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

12.52.2.4 void QCaPeriodic::setEnabled (const bool & *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

12.52.3 Member Data Documentation

12.52.3.1 QCaPeriodic::elementInfoStruct QCaPeriodic::elementInfo [static]

This class is a CA aware element selection widget based on the Qt push button widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

This control and display widget associates one or two values with an element. A typical use of this widget is to move a one or two axis element reference foil stage When the user presses the button an element selection dialog appears. The user selects an element and associated values written to variables. The values written may be one of several static values such as the atomic weight of the element selected, or they may be user defined values, such as a stage position value. If subscribing and the variables change, the updated values are matched to an element and the element is displayed. A user defined string may be emitted on element selection.

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

- epicsqt/ca_framework/widgets/include/QCaPeriodic.h
- epicsqt/ca_framework/widgets/src/QCaPeriodic.cpp

12.53 QCaPeriodicComponentData Class Reference

Public Attributes

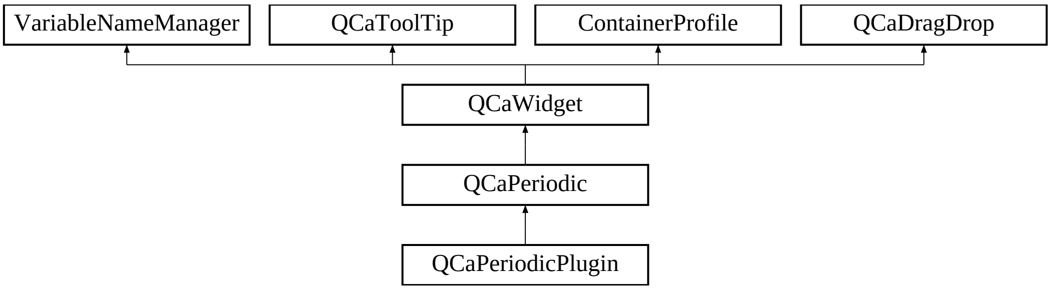
- unsigned int **variableIndex1**
- double **lastData1**
- bool **haveLastData1**
- unsigned int **variableIndex2**
- double **lastData2**
- bool **haveLastData2**

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

- epicsqt/ca_framework/widgets/include/QCaPeriodic.h

12.54 QCaPeriodicPlugin Class Reference

Inheritance diagram for QCaPeriodicPlugin:



Public Types

- enum **PresentationOptions** { **buttonAndLabel** = QCaPeriodic::PRESENTATION_BUTTON_AND_LABEL, **buttonOnly** = QCaPeriodic::PRESENTATION_BUTTON_ONLY, **labelOnly** = QCaPeriodic::PRESENTATION_LABEL_ONLY }
- enum **VariableTypes** {

  Number = QCaPeriodic::VARIABLE_TYPE_NUMBER, **atomicWeight** = QCaPeriodic::VARIABLE_TYPE_ATOMIC_WEIGHT, **meltingPoint** = QCaPeriodic::VARIABLE_TYPE_MELTING_POINT, **boilingPoint** = QCaPeriodic::VARIABLE_TYPE_BOILING_POINT,

  density = QCaPeriodic::VARIABLE_TYPE_DENSITY, **group** = QCaPeriodic::VARIABLE_TYPE_GROUP, **ionizationEnergy** = QCaPeriodic::VARIABLE_TYPE_IONIZATION_ENERGY, **userValue1** = QCaPeriodic::VARIABLE_TYPE_USER_VALUE_1,

  userValue2 = QCaPeriodic::VARIABLE_TYPE_USER_VALUE_2 }

Public Member Functions

- QCaPeriodicPlugin (QWidget ∗parent=0)

  Constructors.
- **QCaPeriodicPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableName1Property** (QString variable1Name)
- QString **getVariableName1Property** ()
- void **setVariableNameSubstitutions1Property** (QString variableNameSubstitutions1)
- QString **getVariableNameSubstitutions1Property** ()
- void **setVariableName2Property** (QString variable2Name)
- QString **getVariableName2Property** ()
- void **setVariableNameSubstitutions2Property** (QString variableNameSubstitutions2)
- QString **getVariableNameSubstitutions2Property** ()
- void **setVariableName3Property** (QString variable3Name)
- QString **getVariableName3Property** ()

- void **setVariableNameSubstitutions3Property** (QString variableNameSubstitutions3)
- QString **getVariableNameSubstitutions3Property** ()
- void **setVariableName4Property** (QString variable4Name)
- QString **getVariableName4Property** ()
- void **setVariableNameSubstitutions4Property** (QString variableNameSubstitutions4)
- QString **getVariableNameSubstitutions4Property** ()
- void **setPresentationOptionProperty** (PresentationOptions presentationOption)
- PresentationOptions **getPresentationOptionProperty** ()
- void **setVariableType1Property** (VariableTypes variableType)
- void **setVariableType2Property** (VariableTypes variableType)
- VariableTypes **getVariableType1Property** ()
- VariableTypes **getVariableType2Property** ()

Properties

- QString writeButtonVariable1
- QString **writeButtonVariableSubstitutions1**
- QString **writeButtonVariable2**
- QString **writeButtonVariableSubstitutions2**
- QString **readbackLabelVariable1**
- QString **readbackLabelVariableSubstitutions1**
- QString **readbackLabelVariable2**
- QString **readbackLabelVariableSubstitutions2**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- PresentationOptions **presentationOption**
- VariableTypes **variableType1**
- VariableTypes **variableType2**
- double **variableTolerance1**
- double **variableTolerance2**
- QString **userInfo**

12.54.1 Constructor & Destructor Documentation

12.54.1.1 QCaPeriodicPlugin::QCaPeriodicPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.54.2 Property Documentation

12.54.2.1 QString QCaPeriodicPlugin::writeButtonVariable1 `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaPeriodicPlugin.h
- epicsqt/ca_framework/plugins/src/QCaPeriodicPlugin.cpp

12.55 QCaPeriodicPluginManager Class Reference

```
#include <QCaPeriodicPluginManager.h>
```

Public Member Functions

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

12.55.1 Detailed Description

???

12.55.2 Constructor & Destructor Documentation

12.55.2.1 QCaPeriodicPluginManager::QCaPeriodicPluginManager (QObject ∗ *parent* = 0)

???

12.55.3 Member Function Documentation

12.55.3.1 QWidget ∗ QCaPeriodicPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaPeriodicPlugin widget.

12.55.3.2 QString QCaPeriodicPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.55.3.3 QIcon QCaPeriodicPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.55.3.4 QString QCaPeriodicPluginManager::includeFile ( ) const

???

???

12.55.3.5 void QCaPeriodicPluginManager::initialize (QDesignerFormEditorInterface ∗ *formEditor*)

???

12.55.3.6 bool QCaPeriodicPluginManager::isContainer ( ) const

???

12.55.3.7 bool QCaPeriodicPluginManager::isInitialized ( ) const

???

12.55.3.8 QString QCaPeriodicPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.55.3.9 QString QCaPeriodicPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.55.3.10 QString QCaPeriodicPluginManager::whatsThis ( ) const

???

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

- epicsqt/ca_framework/plugins/include/QCaPeriodicPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaPeriodicPluginManager.cpp

12.56 QCaPeriodicPluginTaskMenu Class Reference

Public Member Functions

- **QCaPeriodicPluginTaskMenu** (QCaPeriodicPlugin ∗periodicPlugin, QObject ∗parent)
- QAction ∗ **preferredEditAction** () const
- QList< QAction ∗ > **taskActions** () const

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

- epicsqt/ca_framework/plugins/include/QCaPeriodicPluginTaskMenu.h
- epicsqt/ca_framework/plugins/src/QCaPeriodicPluginTaskMenuExtension.cpp

12.57 QCaPeriodicPluginTaskMenuFactory Class Reference

Public Member Functions

- **QCaPeriodicPluginTaskMenuFactory** (QExtensionManager ∗parent=0)

Protected Member Functions

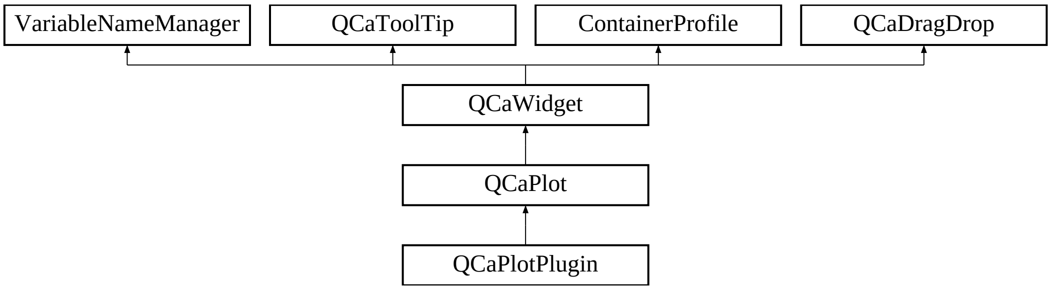
- QObject ∗ **createExtension** (QObject ∗object, const QString &iid, QObject ∗parent) const

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

- epicsqt/ca_framework/plugins/include/QCaPeriodicPluginTaskMenu.h
- epicsqt/ca_framework/plugins/src/QCaPeriodicPluginTaskMenuExtension.cpp

12.58 QCaPlot Class Reference

Inheritance diagram for QCaPlot:



Public Slots

- void requestEnabled (const bool &state)

Signals

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

Public Member Functions

- QCaPlot (QWidget ∗parent=0)
- QCaPlot (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setYMin** (double yMin)
- double **getYMin** ()
- void **setYMax** (double yMax)
- double **getYMax** ()
- void **setAutoScale** (bool autoScale)
- bool **getAutoScale** ()
- void **setAxisEnableX** (bool axisEnableXIn)

- bool **getAxisEnableX** ()
- void **setAxisEnableY** (bool axisEnableYIn)
- bool **getAxisEnableY** ()
- QString **getTitle** ()
- void **setBackgroundColor** (QColor backgroundColor)
- QColor **getBackgroundColor** ()
- void **setTraceStyle** (QwtPlotCurve::CurveStyle traceStyle, const unsigned int variableIndex)
- QwtPlotCurve::CurveStyle **getTraceStyle** (const unsigned int variableIndex)
- void **setTraceColor** (QColor traceColor, const unsigned int variableIndex)
- void **setTraceColor1** (QColor traceColor)
- void **setTraceColor2** (QColor traceColor)
- void **setTraceColor3** (QColor traceColor)
- void **setTraceColor4** (QColor traceColor)
- QColor **getTraceColor** (const unsigned int variableIndex)
- QColor **getTraceColor1** ()
- QColor **getTraceColor2** ()
- QColor **getTraceColor3** ()
- QColor **getTraceColor4** ()
- void **setTraceLegend1** (QString traceLegend)
- void **setTraceLegend2** (QString traceLegend)
- void **setTraceLegend3** (QString traceLegend)
- void **setTraceLegend4** (QString traceLegend)
- QString **getTraceLegend1** ()
- QString **getTraceLegend2** ()
- QString **getTraceLegend3** ()
- QString **getTraceLegend4** ()
- void **setXUnit** (QString xUnit)
- QString **getXUnit** ()
- void **setYUnit** (QString yUnit)
- QString **getYUnit** ()
- void **setGridEnableMajorX** (bool gridEnableMajorXIn)
- void **setGridEnableMajorY** (bool gridEnableMajorYIn)
- void **setGridEnableMinorX** (bool gridEnableMinorXIn)
- void **setGridEnableMinorY** (bool gridEnableMinorYIn)
- bool **getGridEnableMajorX** ()
- bool **getGridEnableMajorY** ()
- bool **getGridEnableMinorX** ()
- bool **getGridEnableMinorY** ()
- void **setGridMajorColor** (QColor gridMajorColorIn)
- void **setGridMinorColor** (QColor gridMinorColorIn)
- QColor **getGridMajorColor** ()
- QColor **getGridMinorColor** ()
- void **setXStart** (double xStart)
- double **getXStart** ()
- void **setXIncrement** (double xIncrement)

- double **getXIncrement** ()
- void **setTimeSpan** (unsigned int timeSpan)
- unsigned int **getTimeSpan** ()
- void **setTickRate** (unsigned int tickRate)
- unsigned int **getTickRate** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void setRunVisible (bool visibleIn)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaFloatingFormatting **floatingFormatting**
- bool **localEnabled**
- bool **allowDrop**
- bool **visible**

12.58.1 Constructor & Destructor Documentation

12.58.1.1 QCaPlot::QCaPlot (QWidget ∗ *parent* = 0)

This class is a CA aware Plot widget based on the Qwt QwtPlot widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.58.1.2 QCaPlot::QCaPlot (const QString & *variableNameIn*, QWidget ∗ *parent* = 0)

Constructor with known variable

12.58.2 Member Function Documentation

12.58.2.1 void QCaPlot::establishConnection (unsigned int *variableIndex*) `[protected, virtual]`

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.58.2.2 bool QCaPlot::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.58.2.3 void QCaPlot::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled, but will use our own setEnabled which will allow alarm states to override current enabled state

12.58.2.4 void QCaPlot::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

12.58.2.5 void QCaPlot::setRunVisible (bool *visibleIn*) [protected]

Manage property to set widget visible or not

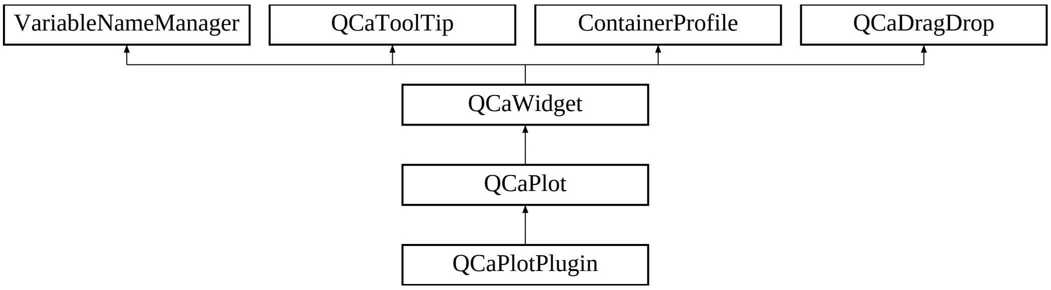
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaPlot.h
- epicsqt/ca_framework/widgets/src/QCaPlot.cpp

12.59 QCaPlotPlugin Class Reference

CA Plot Widget Plugin for designer.

Inheritance diagram for QCaPlotPlugin:



Public Types

- enum **TraceStyles** { **Lines** = QwtPlotCurve::Lines, **Sticks** = QwtPlotCurve::Sticks, **Steps** = QwtPlotCurve::Steps, **Dots** = QwtPlotCurve::Dots }

Public Member Functions

- QCaPlotPlugin (QWidget ∗parent=0)

  Constructors.
- **QCaPlotPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableName1Property** (QString variable1Name)
- QString **getVariableName1Property** ()
- void **setVariableName2Property** (QString variable2Name)
- QString **getVariableName2Property** ()
- void **setVariableName3Property** (QString variable3Name)
- QString **getVariableName3Property** ()
- void **setVariableName4Property** (QString variable4Name)
- QString **getVariableName4Property** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setTraceStyle1** (TraceStyles traceStyle)
- void **setTraceStyle2** (TraceStyles traceStyle)
- void **setTraceStyle3** (TraceStyles traceStyle)
- void **setTraceStyle4** (TraceStyles traceStyle)
- TraceStyles **getTraceStyle1** ()
- TraceStyles **getTraceStyle2** ()
- TraceStyles **getTraceStyle3** ()
- TraceStyles **getTraceStyle4** ()

Properties

- QString **variable1**
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variableSubstitutions**
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **allowDrop**
- QColor **traceColor1**
- QColor **traceColor2**
- QColor **traceColor3**
- QColor **traceColor4**
- TraceStyles **traceStyle1**

- TraceStyles **traceStyle2**
- TraceStyles **traceStyle3**
- TraceStyles **traceStyle4**
- QString **traceLegend1**
- QString **traceLegend2**
- QString **traceLegend3**
- QString **traceLegend4**
- double **yMin**
- double **yMax**
- bool **autoScale**
- bool **axisEnableX**
- bool **axisEnableY**
- bool **gridEnableMajorX**
- bool **gridEnableMajorY**
- bool **gridEnableMinorX**
- bool **gridEnableMinorY**
- QColor **gridMajorColor**
- QColor **gridMinorColor**
- QString **title**
- QColor **backgroundColor**
- QString **xUnit**
- QString **yUnit**
- double **xStart**
- double **xIncrement**
- unsigned **int**

12.59.1 Detailed Description

CA Plot Widget Plugin for designer.

Version

Revision:

#2

Date

DateTime:

2010/09/06 11:58:56

Author

glenn.jackson

12.59.2 Constructor & Destructor Documentation

12.59.2.1 QCaPlotPlugin::QCaPlotPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

for each variable name property manager, set up an index to identify it when it signals and set up a connection to recieve variable name property changes. The variable name property manager class only delivers an updated variable name after the user has stopped typing

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

- epicsqt/ca_framework/plugins/include/QCaPlotPlugin.h
- epicsqt/ca_framework/plugins/src/QCaPlotPlugin.cpp

12.60 QCaPlotPluginManager Class Reference

CA Plot Widget Plugin Manager for designer.

`#include <QCaPlotPluginManager.h>`

Public Member Functions

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

12.60.1 Detailed Description

CA Plot Widget Plugin Manager for designer.

???

Version

Revision:

#1

Date

DateTime:

2010/08/30 16:37:08

Author

glenn.jackson

12.60.2 Constructor & Destructor Documentation

12.60.2.1 QCaPlotPluginManager::QCaPlotPluginManager (QObject $*$ *parent* = 0)

???

12.60.3 Member Function Documentation

12.60.3.1 QWidget $*$ QCaPlotPluginManager::createWidget (QWidget $*$ *parent*)

Widget factory. Creates a QCaPlot widget.

12.60.3.2 QString QCaPlotPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.60.3.3 QIcon QCaPlotPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.60.3.4 QString QCaPlotPluginManager::includeFile ( ) const

???

???

12.60.3.5 void QCaPlotPluginManager::initialize (QDesignerFormEditorInterface $*$ *core*)

???

12.60.3.6 bool QCaPlotPluginManager::isContainer ( ) const

???

12.60.3.7 bool QCaPlotPluginManager::isInitialized ( ) const

???

12.60.3.8 QString QCaPlotPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.60.3.9 QString QCaPlotPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.60.3.10 QString QCaPlotPluginManager::whatsThis ( ) const

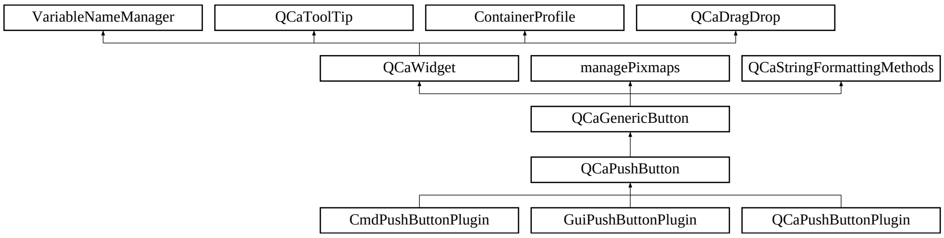
???

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

- epicsqt/ca_framework/plugins/include/QCaPlotPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaPlotPluginManager.cpp

12.61 QCaPushButton Class Reference

Inheritance diagram for QCaPushButton:



Public Slots

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

Signals

- void **dbValueChanged** (const QString &out)
- void **newGui** (QString guiName, ASguiForm::creationOptions creationOption)
- void **requestResend** ()

Public Member Functions

- QCaPushButton (QWidget ∗parent=0)
- QCaPushButton (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void **setEnabled** (const bool &state)

Protected Member Functions

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

12.61.1 Constructor & Destructor Documentation

12.61.1.1 QCaPushButton::QCaPushButton (QWidget ∗ *parent* = 0)

This class is a CA aware push button widget based on the Qt push button widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.61.1.2 QCaPushButton::QCaPushButton (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Constructor with known variable

12.61.2 Member Function Documentation

12.61.2.1 bool QCaPushButton::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Reimplemented from QCaGenericButton.

12.61.2.2 void QCaPushButton::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

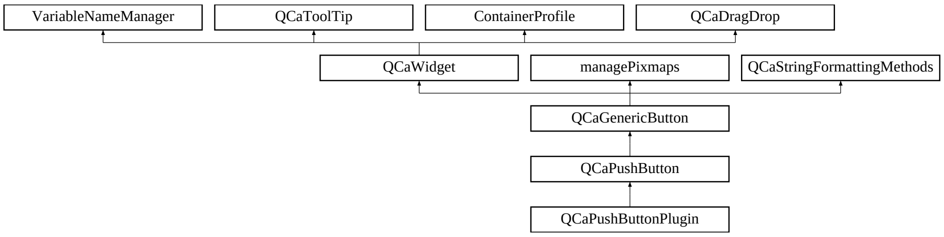
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaPushButton.h
- epicsqt/ca_framework/widgets/src/QCaPushButton.cpp

12.62 QCaPushButtonPlugin Class Reference

CA Push Button Widget Plugin.

Inheritance diagram for QCaPushButtonPlugin:



Public Types

- enum **UpdateOptions** { **Text** = QCaPushButton::UPDATE_TEXT, **Icon** = QCaPushButton::UPDATE_ICON, **TextAndIcon** = QCaPushButton::UPDATE_TEXT_AND_ICON, **State** = QCaPushButton::UPDATE_STATE }
- enum **Formats** {

  Default = QCaStringFormatting::FORMAT_DEFAULT, **Floating** = QCaStringFormatting::FORMAT_FLOATING, **Integer** = QCaStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QCaStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QCaStringFormatting::FORMAT_TIME, **LocalEnumeration** = QCaStringFormatting::FORMAT_LOCAL_ENUMERATE }
- enum **Notations** { **Fixed** = QCaStringFormatting::NOTATION_FIXED, **Scientific** = QCaStringFormatting::NOTATION_SCIENTIFIC, **Automatic** = QCaStringFormatting::NOTATION_AUTOMATIC }
- enum **CreationOptionNames** { **Open** = ASguiForm::CREATION_OPTION_OPEN, **NewTab** = ASguiForm::CREATION_OPTION_NEW_TAB, **NewWindow** = ASguiForm::CREATION_OPTION_NEW_WINDOW }

Public Member Functions

- QCaPushButtonPlugin (QWidget ∗parent=0)

Constructors.

- **QCaPushButtonPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setAltReadbackVariableNameProperty** (QString altReadbackVariableName)
- QString **getAltReadbackVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setUpdateOptionProperty** (UpdateOptions updateOption)
- UpdateOptions **getUpdateOptionProperty** ()
- void **setPixmap0Property** (QPixmap pixmap)
- void **setPixmap1Property** (QPixmap pixmap)
- void **setPixmap2Property** (QPixmap pixmap)
- void **setPixmap3Property** (QPixmap pixmap)
- void **setPixmap4Property** (QPixmap pixmap)
- void **setPixmap5Property** (QPixmap pixmap)
- void **setPixmap6Property** (QPixmap pixmap)
- void **setPixmap7Property** (QPixmap pixmap)
- QPixmap **getPixmap0Property** ()
- QPixmap **getPixmap1Property** ()
- QPixmap **getPixmap2Property** ()
- QPixmap **getPixmap3Property** ()
- QPixmap **getPixmap4Property** ()
- QPixmap **getPixmap5Property** ()
- QPixmap **getPixmap6Property** ()
- QPixmap **getPixmap7Property** ()
- void **setFormatProperty** (Formats format)
- Formats **getFormatProperty** ()
- void **setNotationProperty** (Notations notation)
- Notations **getNotationProperty** ()
- void **setCreationOptionProperty** (CreationOptionNames creationOptionIn)
- CreationOptionNames **getCreationOptionProperty** ()

Properties

- QString variable
- QString **altReadbackVariable**
- QString **variableSubstitutions**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**

- bool **allowDrop**
- UpdateOptions updateOption

  Update options (text, pixmap, or both)
- QPixmap pixmap0

  Pixmaps.
- QPixmap **pixmap1**
- QPixmap **pixmap2**
- QPixmap **pixmap3**
- QPixmap **pixmap4**
- QPixmap **pixmap5**
- QPixmap **pixmap6**
- QPixmap **pixmap7**
- unsigned int

  String formatting properties.
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- QString **localEnumeration**
- Qt::Alignment **alignment**
- Formats **format**
- Notations **notation**
- QString **password**
- bool **writeOnPress**
- bool **writeOnRelease**
- bool **writeOnClick**
- QString pressText

  Text to write on a button release.
- QString **releaseText**
- QString clickText

  Text to write on a button press.
- QString clickCheckedText

  Text to write on a button click.
- QString **labelText**
- QString program

  Text to write on a button click when the button moves to the checked position.
- QStringList arguments

  Program to run.
- QString **guiFile**
- CreationOptionNames creationOption

  GUI file name to launch.

12.62.1 Detailed Description

CA Push Button Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.62.2 Constructor & Destructor Documentation

12.62.2.1 QCaPushButtonPlugin::QCaPushButtonPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

For each variable name property manager, set up an index to identify it when it signals and set up a connection to recieve variable name property changes. The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.62.3 Property Documentation

12.62.3.1 QString QCaPushButtonPlugin::variable `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaPushButtonPlugin.h
- epicsqt/ca_framework/plugins/src/QCaPushButtonPlugin.cpp

12.63 QCaPushButtonPluginManager Class Reference

CA Push Button Widget Plugin Manager for designer.

`#include <QCaPushButtonPluginManager.h>`

Public Member Functions

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

12.63.1 Detailed Description

CA Push Button Widget Plugin Manager for designer.

???

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.63.2 Constructor & Destructor Documentation

12.63.2.1 QCaPushButtonPluginManager::QCaPushButtonPluginManager (QObject ∗ *parent* = 0)

???

12.63.3 Member Function Documentation

12.63.3.1 QWidget ∗ QCaPushButtonPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLabel widget.

12.63.3.2 QString QCaPushButtonPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.63.3.3 QIcon QCaPushButtonPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.63.3.4 QString QCaPushButtonPluginManager::includeFile ( ) const

???

???

12.63.3.5 void QCaPushButtonPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.63.3.6 bool QCaPushButtonPluginManager::isContainer ( ) const

???

12.63.3.7 bool QCaPushButtonPluginManager::isInitialized ( ) const

???

12.63.3.8 QString QCaPushButtonPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.63.3.9 QString QCaPushButtonPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.63.3.10 QString QCaPushButtonPluginManager::whatsThis ( ) const

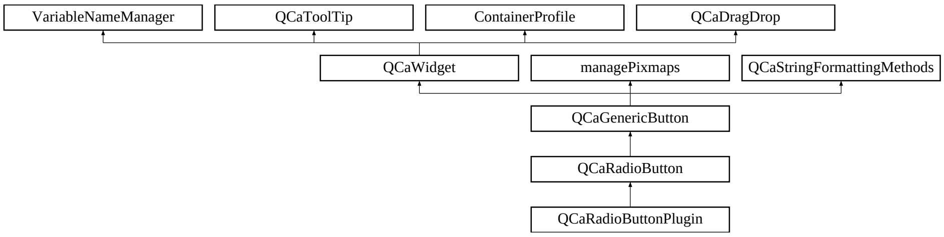
???

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

- epicsqt/ca_framework/plugins/include/QCaPushButtonPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaPushButtonPluginManager.cpp

12.64 QCaRadioButton Class Reference

Inheritance diagram for QCaRadioButton:



Public Slots

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

Signals

- void **dbValueChanged** (const QString &out)
- void **newGui** (QString guiName, ASguiForm::creationOptions creationOption)
- void **requestResend** ()

Public Member Functions

- QCaRadioButton (QWidget ∗parent=0)
- QCaRadioButton (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void **setEnabled** (const bool &state)

Protected Member Functions

- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

12.64.1 Constructor & Destructor Documentation

12.64.1.1 QCaRadioButton::QCaRadioButton (QWidget ∗ *parent* = 0)

This class is a CA aware push button widget based on the Qt push button widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.64.1.2 QCaRadioButton::QCaRadioButton (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Constructor with known variable

12.64.2 Member Function Documentation

12.64.2.1 bool QCaRadioButton::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Reimplemented from QCaGenericButton.

12.64.2.2 void QCaRadioButton::requestEnabled (const bool & *state*) [slot]

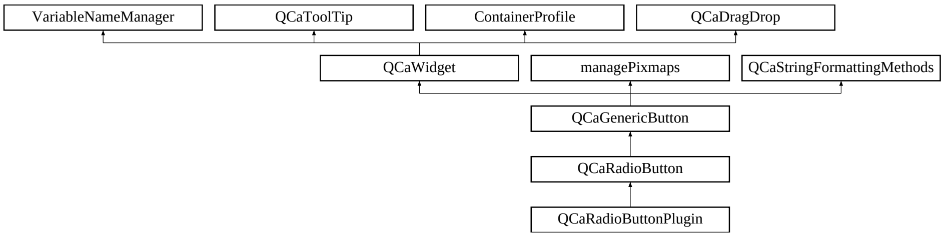
Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

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

- epicsqt/ca_framework/widgets/include/QCaRadioButton.h
- epicsqt/ca_framework/widgets/src/QCaRadioButton.cpp

12.65 QCaRadioButtonPlugin Class Reference

Inheritance diagram for QCaRadioButtonPlugin:



Public Types

- enum **UpdateOptions** { **Text** = QCaRadioButton::UPDATE_TEXT, **Icon** = QCaRadioButton::UPDATE_ICON, **TextAndIcon** = QCaRadioButton::UPDATE_TEXT_AND_ICON, **State** = QCaRadioButton::UPDATE_STATE }
- enum **Formats** {

  Default = QCaStringFormatting::FORMAT_DEFAULT, **Floating** = QCaStringFormatting::FORMAT_FLOATING, **Integer** = QCaStringFormatting::FORMAT_INTEGER, **UnsignedInteger** = QCaStringFormatting::FORMAT_UNSIGNEDINTEGER,

  Time = QCaStringFormatting::FORMAT_TIME }
- enum **Notations** { **Fixed** = QCaStringFormatting::NOTATION_FIXED, **Scientific** = QCaStringFormatting::NOTATION_SCIENTIFIC, **Automatic** = QCaStringFormatting::NOTATION_AUTOMATIC }
- enum **CreationOptionNames** { **Open** = ASguiForm::CREATION_OPTION_OPEN, **NewTab** = ASguiForm::CREATION_OPTION_NEW_TAB, **NewWindow** = ASguiForm::CREATION_OPTION_NEW_WINDOW }

Public Member Functions

- QCaRadioButtonPlugin (QWidget ∗parent=0)

  Constructors.
- **QCaRadioButtonPlugin** (QString variableName, QWidget ∗parent=0)

- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setUpdateOptionProperty** (UpdateOptions updateOption)
- UpdateOptions **getUpdateOptionProperty** ()
- void **setPixmap0Property** (QPixmap pixmap)
- void **setPixmap1Property** (QPixmap pixmap)
- void **setPixmap2Property** (QPixmap pixmap)
- void **setPixmap3Property** (QPixmap pixmap)
- void **setPixmap4Property** (QPixmap pixmap)
- void **setPixmap5Property** (QPixmap pixmap)
- void **setPixmap6Property** (QPixmap pixmap)
- void **setPixmap7Property** (QPixmap pixmap)
- QPixmap **getPixmap0Property** ()
- QPixmap **getPixmap1Property** ()
- QPixmap **getPixmap2Property** ()
- QPixmap **getPixmap3Property** ()
- QPixmap **getPixmap4Property** ()
- QPixmap **getPixmap5Property** ()
- QPixmap **getPixmap6Property** ()
- QPixmap **getPixmap7Property** ()
- void **setFormatProperty** (Formats format)
- Formats **getFormatProperty** ()
- void **setNotationProperty** (Notations notation)
- Notations **getNotationProperty** ()
- void **setCreationOptionProperty** (CreationOptionNames creationOptionIn)
- CreationOptionNames **getCreationOptionProperty** ()

Properties

- QString variable
- QString **variableSubstitutions**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**
- UpdateOptions updateOption

  Update options (text, pixmap, or both)
- QPixmap pixmap0

  Pixmaps.

- QPixmap **pixmap1**
- QPixmap **pixmap2**
- QPixmap **pixmap3**
- QPixmap **pixmap4**
- QPixmap **pixmap5**
- QPixmap **pixmap6**
- QPixmap **pixmap7**
- unsigned int

  String formatting properties.
- bool **useDbPrecision**
- bool **leadingZero**
- bool **trailingZeros**
- bool **addUnits**
- Qt::Alignment **alignment**
- Formats **format**
- Notations **notation**
- QString **password**
- bool **writeOnPress**
- bool **writeOnRelease**
- bool **writeOnClick**
- QString pressText

  Text to write on a button release.
- QString **releaseText**
- QString clickText

  Text to write on a button press.
- QString clickCheckedText

  Text to write on a button click.
- QString **labelText**
- QString program

  Text to write on a button click when the button moves to the checked position.
- QStringList arguments

  Program to run.
- QString **guiFile**
- CreationOptionNames creationOption

  GUI file name to launch.

12.65.1 Constructor & Destructor Documentation

12.65.1.1 QCaRadioButtonPlugin::QCaRadioButtonPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.65.2 Property Documentation

12.65.2.1 QString QCaRadioButtonPlugin::variable `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaRadioButtonPlugin.h
- epicsqt/ca_framework/plugins/src/QCaRadioButtonPlugin.cpp

12.66 QCaRadioButtonPluginManager Class Reference

```
#include <QCaRadioButtonPluginManager.h>
```

Public Member Functions

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

12.66.1 Detailed Description

???

12.66.2 Constructor & Destructor Documentation

12.66.2.1 QCaRadioButtonPluginManager::QCaRadioButtonPluginManager (QObject ∗ *parent* = 0)

???

12.66.3 Member Function Documentation

12.66.3.1 QWidget ∗ QCaRadioButtonPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaLabel widget.

12.66.3.2 QString QCaRadioButtonPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.66.3.3 QIcon QCaRadioButtonPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.66.3.4 QString QCaRadioButtonPluginManager::includeFile ( ) const

???

???

12.66.3.5 void QCaRadioButtonPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.66.3.6 bool QCaRadioButtonPluginManager::isContainer ( ) const

???

12.66.3.7 bool QCaRadioButtonPluginManager::isInitialized ( ) const

???

12.66.3.8 QString QCaRadioButtonPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.66.3.9 QString QCaRadioButtonPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.66.3.10 QString QCaRadioButtonPluginManager::whatsThis ( ) const

???

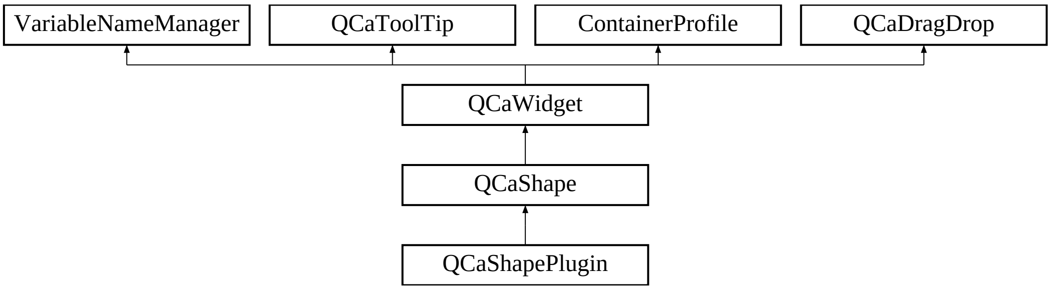
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/plugins/include/QCaRadioButtonPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaRadioButtonPluginManager.cpp

12.67 QCaShape Class Reference

CA Shape Widget.

Inheritance diagram for QCaShape:



Public Types

- enum **Shape** {

  Line, **Points**, **Polyline**, **Polygon**,

  Rect, **RoundedRect**, **Ellipse**, **Arc**,

  Chord, **Pie**, **Path**, **Text**,

  Pixmap }
- enum **Animations** {

  Width, **Height**, **X**, **Y**,

  Transperency, **Rotation**, **ColourHue**, **ColourSaturation**,

  ColourValue, **ColourIndex**, **Penwidth** }

Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged1** (const qlonglong &out)
- void **dbValueChanged2** (const qlonglong &out)
- void **dbValueChanged3** (const qlonglong &out)
- void **dbValueChanged4** (const qlonglong &out)
- void **dbValueChanged5** (const qlonglong &out)
- void **dbValueChanged6** (const qlonglong &out)

Public Member Functions

- QCaShape (QWidget ∗parent=0)
- QCaShape (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setAnimation** (Animations animation, const int index)
- Animations **getAnimation** (const int index)
- void **setScale** (const double scale, const int index)
- double **getScale** (const int index)
- void **setOffset** (const double offset, const int index)
- double **getOffset** (const int index)
- void **setBorder** (const bool border)
- bool **getBorder** ()
- void **setFill** (const bool fill)
- bool **getFill** ()
- void **setSubscribe** (const bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (const bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()
- void **setShape** (Shape shape)
- Shape **getShape** ()
- void **setNumPoints** (const unsigned int numPoints)
- unsigned int **getNumPoints** ()
- void **setOriginTranslation** (const QPoint originTranslation)
- QPoint **getOriginTranslation** ()
- void **setPoint** (const QPoint point, const int index)
- QPoint **getPoint** (const int index)
- void **setColor** (const QColor color, const int index)
- QColor **getColor** (const int index)
- void **setDrawBorder** (const bool drawBorder)
- bool **getDrawBorder** ()
- void **setLineWidth** (const unsigned int lineWidth)

- unsigned int **getLineWidth** ()
- void **setStartAngle** (const double startAngle)
- double **getStartAngle** ()
- void **setRotation** (const double rotation)
- double **getRotation** ()
- void **setArcLength** (const double arcLength)
- double **getArcLength** ()
- void **setText** (const QString text)
- QString **getText** ()

Protected Member Functions

- void colorChange (unsigned int index)
- void establishConnection (unsigned int variableIndex)

  Index into colorsProperty last used when setting brush color.
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **mousePressEvent** (QMouseEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaIntegerFormatting **integerFormatting**
- bool allowDrop

  Integer formatting options.
- double **offsets** [OFFSETS_SIZE]
- double **scales** [SCALES_SIZE]
- Shape **shape**
- QPoint **originTranslation**
- QPoint **points** [POINTS_SIZE]
- unsigned int **numPoints**
- QColor **colors** [COLORS_SIZE]
- Animations **animations** [6]
- bool **fill**
- bool **border**
- double **startAngle**
- double **arcLength**
- QString **text**
- double **rotation**
- unsigned int **lineWidth**
- bool **drawBorder**
- bool **localEnabled**
- unsigned int currentColor

  Act on a color property change. (will update shape if using the color)

12.67.1 Detailed Description

CA Shape Widget.

Version

Revision:

#18

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.67.2 Constructor & Destructor Documentation

12.67.2.1 QCaShape::QCaShape (QWidget ∗ *parent* = 0)

This class is a CA aware shape widget based on QWidget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Create without a known variable. Just manage parental hirarchy.

12.67.2.2 QCaShape::QCaShape (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Create with a known variable. Subscription occurs immedietly.

Call common setup code.

Use the variable name as the first variable

12.67.3 Member Function Documentation

12.67.3.1 void QCaShape::colorChange (unsigned int *index*) [protected]

Reset the brush color if the color the brush is using is changing

12.67.3.2 void QCaShape::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Index into colorsProperty last used when setting brush color.

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.67.3.3 bool QCaShape::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.67.3.4 void QCaShape::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

12.67.3.5 void QCaShape::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

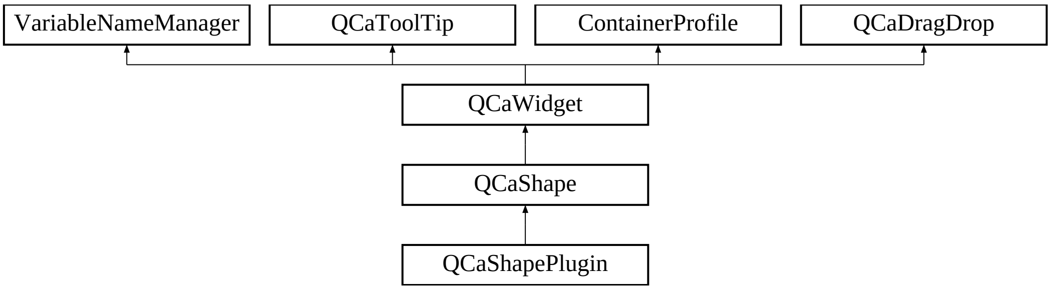
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaShape.h
- epicsqt/ca_framework/widgets/src/QCaShape.cpp

12.68 QCaShapePlugin Class Reference

CA Shape Widget Plugin.

Inheritance diagram for QCaShapePlugin:



Public Types

- enum **animationOptions** {

  Width = QCaShape::Width, **Height** = QCaShape::Height, **X** = QCaShape::X, **Y** = QCaShape::Y,

  Transperency = QCaShape::Transperency, **Rotation** = QCaShape::Rotation, **ColourHue** = QCaShape::ColourHue, **ColourSaturation** = QCaShape::ColourSaturation,

  ColourValue = QCaShape::ColourValue, **ColourIndex** = QCaShape::ColourIndex, **Penwidth** = QCaShape::Penwidth }
- enum **shapeOptions** {

  Line = QCaShape::Line, **Points** = QCaShape::Points, **Polyline** = QCaShape::Polyline, **Polygon** = QCaShape::Polygon,

  Rect = QCaShape::Rect, **RoundedRect** = QCaShape::RoundedRect, **Ellipse** = QCaShape::Ellipse, **Arc** = QCaShape::Arc,

  Chord = QCaShape::Chord, **Pie** = QCaShape::Pie, **Path** = QCaShape::Path, **Text** = QCaShape::Text,

  Pixmap = QCaShape::Pixmap }

Public Member Functions

- QCaShapePlugin (QWidget ∗parent=0)

  Constructors.
- **QCaShapePlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableName1Property** (QString variable1Name)
- QString **getVariableName1Property** ()
- void **setVariableName2Property** (QString variable2Name)
- QString **getVariableName2Property** ()
- void **setVariableName3Property** (QString variable3Name)
- QString **getVariableName3Property** ()
- void **setVariableName4Property** (QString variable4Name)
- QString **getVariableName4Property** ()
- void **setVariableName5Property** (QString variable5Name)
- QString **getVariableName5Property** ()
- void **setVariableName6Property** (QString variable6Name)
- QString **getVariableName6Property** ()
- void **setAnimation1Property** (animationOptions animation)
- animationOptions **getAnimation1Property** ()
- void **setAnimation2Property** (animationOptions animation)
- animationOptions **getAnimation2Property** ()
- void **setAnimation3Property** (animationOptions animation)
- animationOptions **getAnimation3Property** ()
- void **setAnimation4Property** (animationOptions animation)
- animationOptions **getAnimation4Property** ()
- void **setAnimation5Property** (animationOptions animation)
- animationOptions **getAnimation5Property** ()

- void **setAnimation6Property** (animationOptions animation)
- animationOptions **getAnimation6Property** ()
- void **setScale1Property** (double scale)
- double **getScale1Property** ()
- void **setScale2Property** (double scale)
- double **getScale2Property** ()
- void **setScale3Property** (double scale)
- double **getScale3Property** ()
- void **setScale4Property** (double scale)
- double **getScale4Property** ()
- void **setScale5Property** (double scale)
- double **getScale5Property** ()
- void **setScale6Property** (double scale)
- double **getScale6Property** ()
- void **setOffset1Property** (double offset)
- double **getOffset1Property** ()
- void **setOffset2Property** (double offset)
- double **getOffset2Property** ()
- void **setOffset3Property** (double offset)
- double **getOffset3Property** ()
- void **setOffset4Property** (double offset)
- double **getOffset4Property** ()
- void **setOffset5Property** (double offset)
- double **getOffset5Property** ()
- void **setOffset6Property** (double offset)
- double **getOffset6Property** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()
- void **setShapeProperty** (shapeOptions shape)
- shapeOptions **getShapeProperty** ()
- void **setPoint1Property** (QPoint point)
- QPoint **getPoint1Property** ()
- void **setPoint2Property** (QPoint point)
- QPoint **getPoint2Property** ()
- void **setPoint3Property** (QPoint point)
- QPoint **getPoint3Property** ()
- void **setPoint4Property** (QPoint point)
- QPoint **getPoint4Property** ()
- void **setPoint5Property** (QPoint point)
- QPoint **getPoint5Property** ()
- void **setPoint6Property** (QPoint point)
- QPoint **getPoint6Property** ()
- void **setPoint7Property** (QPoint point)
- QPoint **getPoint7Property** ()
- void **setPoint8Property** (QPoint point)

- QPoint **getPoint8Property** ()
- void **setPoint9Property** (QPoint point)
- QPoint **getPoint9Property** ()
- void **setPoint10Property** (QPoint point)
- QPoint **getPoint10Property** ()
- void **setColor1Property** (QColor color)
- QColor **getColor1Property** ()
- void **setColor2Property** (QColor color)
- QColor **getColor2Property** ()
- void **setColor3Property** (QColor color)
- QColor **getColor3Property** ()
- void **setColor4Property** (QColor color)
- QColor **getColor4Property** ()
- void **setColor5Property** (QColor color)
- QColor **getColor5Property** ()
- void **setColor6Property** (QColor color)
- QColor **getColor6Property** ()
- void **setColor7Property** (QColor color)
- QColor **getColor7Property** ()
- void **setColor8Property** (QColor color)
- QColor **getColor8Property** ()
- void **setColor9Property** (QColor color)
- QColor **getColor9Property** ()
- void **setColor10Property** (QColor color)
- QColor **getColor10Property** ()

Properties

- QString variable1
- QString **variable2**
- QString **variable3**
- QString **variable4**
- QString **variable5**
- QString **variable6**
- bool allowDrop

  Integer formatting options.
- animationOptions **animation1**
- animationOptions **animation2**
- animationOptions **animation3**
- animationOptions **animation4**
- animationOptions **animation5**
- animationOptions **animation6**
- double **scale1**
- double **scale2**
- double **scale3**
- double **scale4**

- double **scale5**
- double **scale6**
- double **offset1**
- double **offset2**
- double **offset3**
- double **offset4**
- double **offset5**
- double **offset6**
- bool **border**
- bool **fill**
- QString **variableSubstitutions**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- shapeOptions **shape**
- unsigned **int**
- QPoint **originTranslation**
- 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**
- bool **drawBorder**
- double **startAngle**
- double **rotation**
- double **arcLength**
- QString **text**

12.68.1 Detailed Description

CA Shape Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.68.2 Constructor & Destructor Documentation

12.68.2.1 QCaShapePlugin::QCaShapePlugin (QWidget ∗ *parent* = 0)

Constructors.

???

for each variable name property manager, set up an index to identify it when it signals and set up a connection to recieve variable name property changes. The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.68.3 Property Documentation

12.68.3.1 QString QCaShapePlugin::variable1 [read, write]

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaShapePlugin.h
- epicsqt/ca_framework/plugins/src/QCaShapePlugin.cpp

12.69 QCaShapePluginManager Class Reference

CA Shape Widget Plugin Manager for designer.

```
#include <QCaShapePluginManager.h>
```

Public Member Functions

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

12.69.1 Detailed Description

CA Shape Widget Plugin Manager for designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.69.2 Constructor & Destructor Documentation

12.69.2.1 QCaShapePluginManager::QCaShapePluginManager (QObject ∗ *parent* = 0)

???

12.69.3 Member Function Documentation

12.69.3.1 QWidget ∗ QCaShapePluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaShapePlugin widget.

12.69.3.2 QString QCaShapePluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.69.3.3 QIcon QCaShapePluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.69.3.4 QString QCaShapePluginManager::includeFile ( ) const

???

???

12.69.3.5 void QCaShapePluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.69.3.6 bool QCaShapePluginManager::isContainer ( ) const

???

12.69.3.7 bool QCaShapePluginManager::isInitialized ( ) const

???

12.69.3.8 QString QCaShapePluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.69.3.9 QString QCaShapePluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.69.3.10 QString QCaShapePluginManager::whatsThis ( ) const

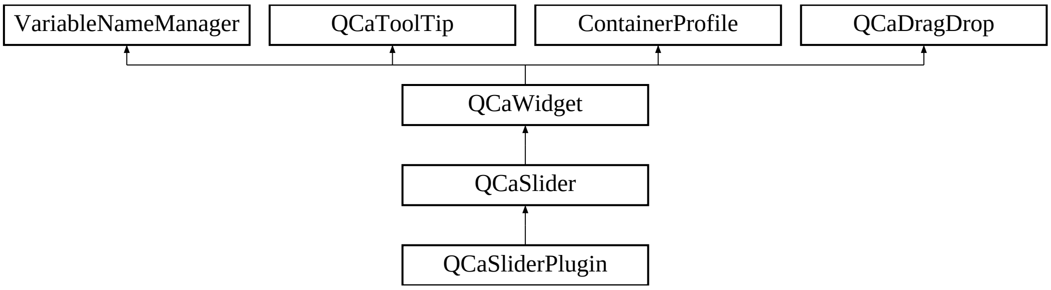
???

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

- epicsqt/ca_framework/plugins/include/QCaShapePluginManager.h
- epicsqt/ca_framework/plugins/src/QCaShapePluginManager.cpp

12.70 QCaSlider Class Reference

Inheritance diagram for QCaSlider:



Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged** (const qlonglong &out)

Public Member Functions

- QCaSlider (QWidget ∗parent=0)
- QCaSlider (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setWriteOnChange** (bool writeOnChange)
- bool **getWriteOnChange** ()
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaIntegerFormatting **integerFormatting**
- bool writeOnChange

  Integer formatting options.
- bool localEnabled

  Write changed value to database when ever the position changes.
- bool **allowDrop**

12.70.1 Constructor & Destructor Documentation

12.70.1.1 QCaSlider::QCaSlider (QWidget ∗ *parent* = 0)

This class is a CA aware slider widget based on the Qt slider widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Constructor with no initialisation

12.70.1.2 QCaSlider::QCaSlider (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Constructor with known variable

12.70.2 Member Function Documentation

12.70.2.1 void QCaSlider::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.70.2.2 bool QCaSlider::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.70.2.3 void QCaSlider::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

12.70.2.4 void QCaSlider::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

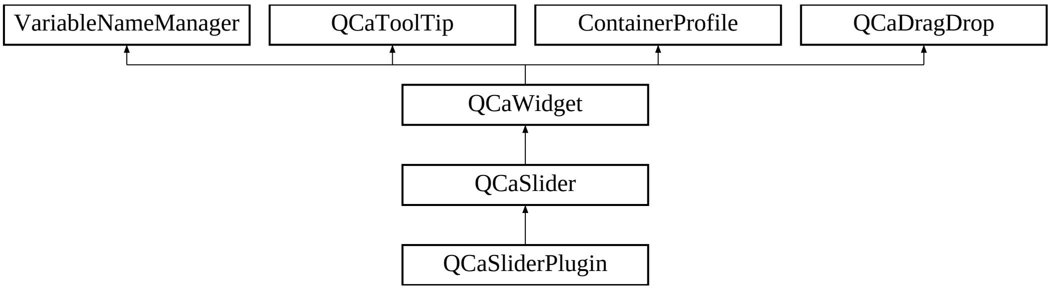
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaSlider.h
- epicsqt/ca_framework/widgets/src/QCaSlider.cpp

12.71 QCaSliderPlugin Class Reference

CA Slider Widget Plugin.

Inheritance diagram for QCaSliderPlugin:



Public Member Functions

- QCaSliderPlugin (QWidget ∗parent=0)

  Constructors.
- **QCaSliderPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()

Properties

- QString variable
- QString **variableSubstitutions**
- bool **enabled**
- bool **allowDrop**
- bool writeOnChange

  Integer formatting options.
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.

12.71.1 Detailed Description

CA Slider Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.71.2 Constructor & Destructor Documentation

12.71.2.1 QCaSliderPlugin::QCaSliderPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.71.3 Property Documentation

12.71.3.1 QString QCaSliderPlugin::variable `[read, write]`

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaSliderPlugin.h
- epicsqt/ca_framework/plugins/src/QCaSliderPlugin.cpp

12.72 QCaSliderPluginManager Class Reference

CA Slider Widget Plugin Manager for designer.

```
#include <QCaSliderPluginManager.h>
```

Public Member Functions

- QCaSliderPluginManager (QObject ∗parent=0)
- bool isContainer () const
- bool isInitialized () const

- QIcon icon () const
- QString group () const
- QString includeFile () const
- QString name () const
- QString toolTip () const
- QString whatsThis () const
- QWidget ∗ createWidget (QWidget ∗parent)
- void initialize (QDesignerFormEditorInterface ∗core)

12.72.1 Detailed Description

CA Slider Widget Plugin Manager for designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.72.2 Constructor & Destructor Documentation

12.72.2.1 QCaSliderPluginManager::QCaSliderPluginManager (QObject ∗ *parent* = 0)

???

12.72.3 Member Function Documentation

12.72.3.1 QWidget ∗ QCaSliderPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QCaSliderPlugin widget.

12.72.3.2 QString QCaSliderPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.72.3.3 QIcon QCaSliderPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.72.3.4 QString QCaSliderPluginManager::includeFile ( ) const

???

???

12.72.3.5 void QCaSliderPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.72.3.6 bool QCaSliderPluginManager::isContainer ( ) const

???

12.72.3.7 bool QCaSliderPluginManager::isInitialized ( ) const

???

12.72.3.8 QString QCaSliderPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.72.3.9 QString QCaSliderPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.72.3.10 QString QCaSliderPluginManager::whatsThis ( ) const

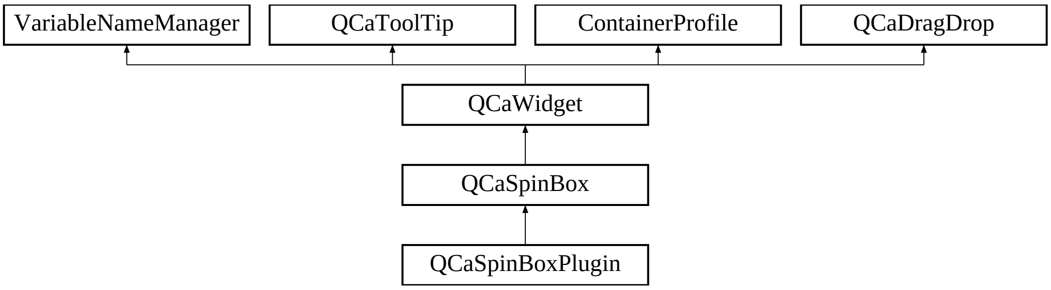
???

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

- epicsqt/ca_framework/plugins/include/QCaSliderPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaSliderPluginManager.cpp

12.73 QCaSpinBox Class Reference

Inheritance diagram for QCaSpinBox:



Public Slots

- void requestEnabled (const bool &state)

Signals

- void **dbValueChanged** (const qlonglong &out)

Public Member Functions

- QCaSpinBox (QWidget ∗parent=0)
- QCaSpinBox (const QString &variableName, QWidget ∗parent=0)
- bool isEnabled () const
- void setEnabled (bool state)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setSubscribe** (bool subscribe)
- bool **getSubscribe** ()
- void **setVariableAsToolTip** (bool variableAsToolTip)
- bool **getVariableAsToolTip** ()
- void **setAllowDrop** (bool allowDropIn)
- bool **getAllowDrop** ()

Protected Member Functions

- void establishConnection (unsigned int variableIndex)
- void **dragEnterEvent** (QDragEnterEvent ∗event)
- void **dropEvent** (QDropEvent ∗event)
- void **setDropText** (QString text)
- QString **getDropText** ()

Protected Attributes

- QCaIntegerFormatting **integerFormatting**
- bool **localEnabled**
- bool **allowDrop**

12.73.1 Constructor & Destructor Documentation

12.73.1.1 QCaSpinBox::QCaSpinBox (QWidget ∗ *parent* = 0)

This class is a CA aware spin box widget based on the Qt spin box widget. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details

Create a CA aware spin box with no variable name yet

12.73.1.2 QCaSpinBox::QCaSpinBox (const QString & *variableNameIn,* QWidget ∗ *parent* = 0)

Create a CA aware spin box with a variable name already known

12.73.2 Member Function Documentation

12.73.2.1 void QCaSpinBox::establishConnection (unsigned int *variableIndex*) [protected, virtual]

Start updating. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. This function may also be used to initiate updates when loaded as a plugin.

Implements VariableNameManager.

12.73.2.2 bool QCaSpinBox::isEnabled ( ) const

Override the default widget isEnabled to allow alarm states to override current enabled state

Return what the state of widget would be if connected.

12.73.2.3 void QCaSpinBox::requestEnabled (const bool & *state*) [slot]

Slot similar to default widget setEnabled slot, but will use our own setEnabled which will allow alarm states to override current enabled state

12.73.2.4 void QCaSpinBox::setEnabled (bool *state*)

Override the default widget setEnabled to allow alarm states to override current enabled state

Note the new 'enabled' state

Set the enabled state of the widget only if connected

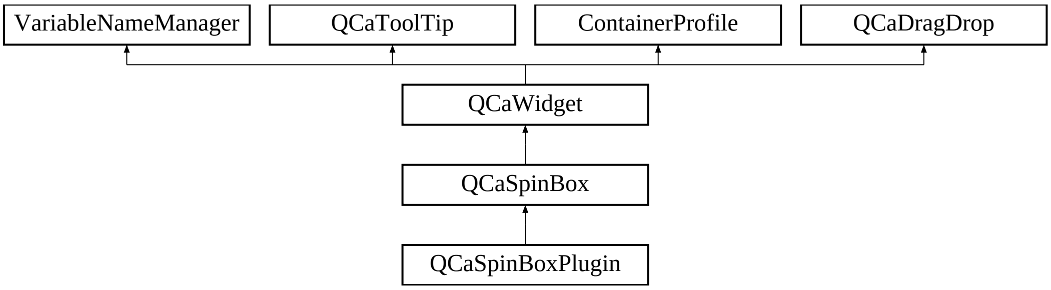
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/widgets/include/QCaSpinBox.h
- epicsqt/ca_framework/widgets/src/QCaSpinBox.cpp

12.74 QCaSpinBoxPlugin Class Reference

CA Spinbox Widget Plugin.

Inheritance diagram for QCaSpinBoxPlugin:



Public Member Functions

- QCaSpinBoxPlugin (QWidget ∗parent=0)
- **QCaSpinBoxPlugin** (QString variableName, QWidget ∗parent=0)
- void **setVariableNameProperty** (QString variableName)
- QString **getVariableNameProperty** ()
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)
- QString **getVariableNameSubstitutionsProperty** ()

Properties

- QString variable
- QString **variableSubstitutions**
- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.
- bool **enabled**
- bool **allowDrop**

12.74.1 Detailed Description

CA Spinbox Widget Plugin.

Version

Revision:

#4

Date

DateTime:

2010/09/06 11:58:56

Author

andrew.rhyder

12.74.2 Constructor & Destructor Documentation

12.74.2.1 QCaSpinBoxPlugin::QCaSpinBoxPlugin (QWidget * *parent* = 0)

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.74.3 Property Documentation

12.74.3.1 QString QCaSpinBoxPlugin::variable [read, write]

Note, a property macro in the form 'Q_PROPERTY(QString variableName READ ...' doesn't work. A property name ending with 'Name' results in some sort of string a variable being displayed, but will only accept alphanumeric and won't generate callbacks on change.

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

- epicsqt/ca_framework/plugins/include/QCaSpinBoxPlugin.h
- epicsqt/ca_framework/plugins/src/QCaSpinBoxPlugin.cpp

12.75 QCaSpinBoxPluginManager Class Reference

CA Spinbox Widget Plugin Manager for designer.

```
#include <QCaSpinBoxPluginManager.h>
```

Public Member Functions

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

12.75.1 Detailed Description

CA Spinbox Widget Plugin Manager for designer.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.75.2 Constructor & Destructor Documentation

12.75.2.1 QCaSpinBoxPluginManager::QCaSpinBoxPluginManager (QObject ∗ *parent* = 0)

???

12.75.3 Member Function Documentation

12.75.3.1 QWidget $*$ QCaSpinBoxPluginManager::createWidget (QWidget $*$ *parent*)

Widget factory. Creates a QCaSpinBoxPlugin widget.

12.75.3.2 QString QCaSpinBoxPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.75.3.3 QIcon QCaSpinBoxPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.75.3.4 QString QCaSpinBoxPluginManager::includeFile ( ) const

???

???

12.75.3.5 void QCaSpinBoxPluginManager::initialize (QDesignerFormEditorInterface $*$ *core*)

???

12.75.3.6 bool QCaSpinBoxPluginManager::isContainer ( ) const

???

12.75.3.7 bool QCaSpinBoxPluginManager::isInitialized ( ) const

???

12.75.3.8 QString QCaSpinBoxPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.75.3.9 QString QCaSpinBoxPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.75.3.10 QString QCaSpinBoxPluginManager::whatsThis ( ) const

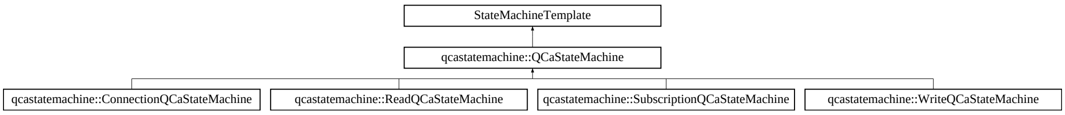
???

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

- epicsqt/ca_framework/plugins/include/QCaSpinBoxPluginManager.h
- epicsqt/ca_framework/plugins/src/QCaSpinBoxPluginManager.cpp

12.76 qcastatemachine::QCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::QCaStateMachine:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::ConnectionQCaStateMachine
qcastatemachine::ReadQCaStateMachine
qcastatemachine::SubscriptionQCaStateMachine
qcastatemachine::WriteQCaStateMachine

Public Member Functions

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

Public Attributes

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

12.76.1 Constructor & Destructor Documentation

12.76.1.1 QCaStateMachine::QCaStateMachine (void ∗ *parent*)

Startup default attributes.

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

12.77 QCaStateMachine Class Reference

Statemachine architecture.

12.77.1 Detailed Description

Statemachine architecture.

Version

Revision:

#8

Date

DateTime:

2010/06/23 07:49:40

Author

anthony.owen

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

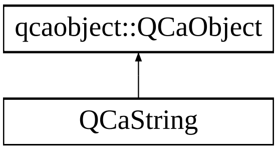
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

12.78 QCaString Class Reference

String wrapper for QCaObject variant data.

`#include <QCaString.h>`

Inheritance diagram for QCaString:



Public Slots

- void writeString (const QString &data)

Signals

- void **stringChanged** (const QString &value, QCaAlarmInfo &alarmInfo, QCaDateTime &timeStamp, const unsigned int &variableIndex)

Public Member Functions

- **QCaString** (QString recordName, QObject ∗eventObject, QCaStringFormatting ∗stringFormattingIn, unsigned int variableIndexIn)
- **QCaString** (QString recordName, QObject ∗eventObject, QCaStringFormatting ∗stringFormattingIn, unsigned int variableIndexIn, UserMessage ∗userMessageIn)

12.78.1 Detailed Description

String wrapper for QCaObject variant data.

Version

Revision:

#6

Date

DateTime:

2010/09/06 13:16:04

Author

andrew.rhyder

Version

Revision:

#7

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.78.2 Member Function Documentation

12.78.2.1 void QCaString::writeString (const QString & *data*) [slot]

Take a new text value and write it to the database. The type of data formatted (text, floating, integer, etc) will be determined by the record data type, How the text is parsed will be determined by the string formatting. For example, assume the record is a floating point record and the text is formatted as an integer. The string is parsed as in integer (123 is OK, 123.456 would fail), then converted to a floating point number. The above example is pedantic if the string formatting

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

- epicsqt/ca_framework/data/include/QCaString.h
- epicsqt/ca_framework/data/src/QCaString.cpp

12.79 QCaStringFormatting Class Reference

Formats the string for QCaString data.

```
#include <QCaStringFormatting.h>
```

Public Types

- enum **formats** {

  FORMAT_DEFAULT, **FORMAT_FLOATING**, **FORMAT_INTEGER**, **FORMAT_UNSIGNEDINTEGER**,

  FORMAT_TIME, **FORMAT_LOCAL_ENUMERATE** }
- enum **notations** { **NOTATION_FIXED** = QTextStream::FixedNotation, **NOTATION_SCIENTIFIC** = QTextStream::ScientificNotation, **NOTATION_AUTOMATIC** = QTextStream::SmartNotation }
- enum **arrayActions** { **APPEND**, **ASCII**, **INDEX** }

Public Member Functions

- QCaStringFormatting ()
- QString formatString (const QVariant &value)
- QVariant formatValue (const QString &text)
- void setDbEgu (QString egu)
- void setDbEnumerations (QStringList enumerations)

  Units to be added (or removed) from the formatted string if 'addUnits' flag is set.
- void setDbPrecision (unsigned int dbPrecisionIn)
- void setDbVariableIsStatField (bool isStatField)
- void setPrecision (unsigned int precision)
- void setUseDbPrecision (bool useDbPrecision)
- void setLeadingZero (bool leadingZero)

- void setTrailingZeros (bool trailingZeros)
- void setFormat (formats format)
- void setRadix (unsigned int radix)
- void setNotation (notations notation)
- void setArrayAction (arrayActions arrayActionIn)
- void setArrayIndex (unsigned int arrayIndexIn)
- void setAddUnits (bool addUnits)
- void setLocalEnumeration (QString localEnumerationIn)
- unsigned int getPrecision ()
- bool getUseDbPrecision ()
- bool getLeadingZero ()
- bool getTrailingZeros ()
- formats getFormat ()
- unsigned int getRadix ()
- notations getNotation ()
- arrayActions getArrayAction ()
- unsigned int getArrayIndex ()
- bool getAddUnits ()
- QString getLocalEnumeration ()

12.79.1 Detailed Description

Formats the string for QCaString data.

Version

Revision:

#9

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.79.2 Constructor & Destructor Documentation

12.79.2.1 QCaStringFormatting::QCaStringFormatting ( )

Construction

12.79.3 Member Function Documentation

12.79.3.1 QString QCaStringFormatting::formatString (const QVariant & *value*)

Generate a string given a value, using formatting defined within this class.

12.79.3.2 QVariant QCaStringFormatting::formatValue (const QString & *text*)

Generate a value given a string, using formatting defined within this class.

??? Temporarily use the variants text convertion.

12.79.3.3 bool QCaStringFormatting::getAddUnits ( )

Get the 'include engineering units' flag.

12.79.3.4 QCaStringFormatting::arrayActions QCaStringFormatting::getArrayAction ( )

Return the action to take when formatting an array (treat as ascii characters, a series of numbers, etc)

12.79.3.5 unsigned int QCaStringFormatting::getArrayIndex ( )

Return the index to select a value from array of values (not relevent when the array is treated as ascii)

12.79.3.6 QCaStringFormatting::formats QCaStringFormatting::getFormat ( )

Get the type of information being formatted. See setFormat() for the use of the format type.

12.79.3.7 bool QCaStringFormatting::getLeadingZero ( )

Get the 'leading zero' flag. See setLeadingZero() for the use of the 'leading zero' flag.

12.79.3.8 QString QCaStringFormatting::getLocalEnumeration ( )

Get the local enumeration strings. See setLocalEnumeration() for the use of 'localEnumeration'.

12.79.3.9 QCaStringFormatting::notations QCaStringFormatting::getNotation ( )

Return the floating point notation

12.79.3.10 unsigned int QCaStringFormatting::getPrecision ( )

Get the precision. See setPrecision() for the use of 'precision'.

12.79.3.11 unsigned int QCaStringFormatting::getRadix ( )

Get the numerical base. See setRadix() for the use of 'radix'.

12.79.3.12 bool QCaStringFormatting::getTrailingZeros ( )

Get the 'trailing zeros' flag. See setTrailingZeros() for the use of the 'trailing zeros' flag.

12.79.3.13 bool QCaStringFormatting::getUseDbPrecision ( )

Get the 'use DB precision' flag. See setUseDbPrecision() for the use of the 'use DB precision' flag.

12.79.3.14 void QCaStringFormatting::setAddUnits (bool *AddUnitsIn*)

Set or clear a flag to include the engineering units in a string

12.79.3.15 void QCaStringFormatting::setArrayAction (arrayActions *arrayActionIn*)

Set how arrays are converted to text (Treates as an array of ascii characters, an array of values, etc)

12.79.3.16 void QCaStringFormatting::setArrayIndex (unsigned int *arrayIndexIn*)

Set which value from an array is formatted (not relevent when the array is processed as ascii)

12.79.3.17 void QCaStringFormatting::setDbEgu (QString *egu*)

Set up the engineering units that will be added to or removed from strings if 'addUnits' flag is set

12.79.3.18 void QCaStringFormatting::setDbEnumerations (QStringList *enumerations*)

Units to be added (or removed) from the formatted string if 'addUnits' flag is set.

Set up the enumeration values. Thses are used if avaiable if the formatting is FORMAT_DEFAULT

12.79.3.19 void QCaStringFormatting::setDbPrecision (unsigned int *dbPrecisionIn*)

Set up the precision specified by the database. This precision is used when formatting floating point numbers if 'useDbPrecision' is set

12.79.3.20 void QCaStringFormatting::setDbVariableIsStatField (bool *isStatField*)

Set up a flag indicating the variable represents a STAT field. This is required as formatting enumerated strings for the STAT field is a special case.

12.79.3.21 void QCaStringFormatting::setFormat (formats *formatIn*)

Set the type of information being displayed (floating point number, date/time, etc).

12.79.3.22 void QCaStringFormatting::setLeadingZero (bool *leadingZeroIn*)

Set or clear a flag to display a leading zero before a decimal point for floating point numbers between 1 and -1 Relevent when formatting the string as a floating point number.

12.79.3.23 void QCaStringFormatting::setLocalEnumeration (QString *localEnumerationIn*)

Set the string used to specify local enumeration. This is used when a value is to be enumerated and the value is either not the VAL field, or the database does not provide any enumeration, or the database enumeration is not appropriate

12.79.3.24 void QCaStringFormatting::setNotation (notations *notation*)

Set the notation (floating, scientific, or automatic) Note, the notation enumerations match the QTextStream RealNumberNotation values

12.79.3.25 void QCaStringFormatting::setPrecision (unsigned int *precisionIn*)

Set the precision - the number of significant digits displayed when formatting a floating point number. Relevent when formatting the string as a floating point number. Note, this will only be used if 'useDbPrecision' is false.

12.79.3.26 void QCaStringFormatting::setRadix (unsigned int *radix*)

Set the numer system base. Relevent when formatting the string as an interger of floating point number. Any radix of 2 or more is accepted. Check the conversion code that uses this number to see what values are expected. At the time of writing (16/2/9) it is anticipated that floating point numbers will always be base 10 and integer numbers will be base 2, 8, 10, or 16.

12.79.3.27 void QCaStringFormatting::setTrailingZeros (bool *trailingZerosIn*)

Set or clear a flag to display trailing zeros at the end up the precision for floating point Relevent when formatting the string as a floating point number.

12.79.3.28 void QCaStringFormatting::setUseDbPrecision (bool *useDbPrecisionIn*)

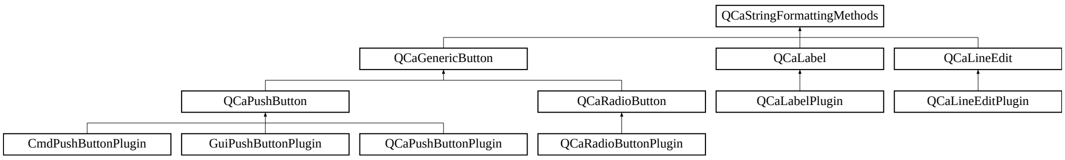
Set or clear a flag to ignore the precision held by this class and use the precision read from the database record. The precision is the number of significant digits displayed when formatting a floating point number. Relevent when formatting the string as a floating point number.

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

- epicsqt/ca_framework/data/include/QCaStringFormatting.h
- epicsqt/ca_framework/data/src/QCaStringFormatting.cpp

12.80 QCaStringFormattingMethods Class Reference

Inheritance diagram for QCaStringFormattingMethods:



Public Member Functions

- virtual void **stringFormattingChange** ()=0
- void setPrecision (unsigned int precision)
- unsigned int **getPrecision** ()
- void **setUseDbPrecision** (bool useDbPrecision)
- bool **getUseDbPrecision** ()
- void **setLeadingZero** (bool leadingZero)
- bool **getLeadingZero** ()
- void **setTrailingZeros** (bool trailingZeros)
- bool **getTrailingZeros** ()
- void **setAddUnits** (bool addUnits)
- bool **getAddUnits** ()
- void **setLocalEnumeration** (QString localEnumeration)
- QString **getLocalEnumeration** ()
- void **setFormat** (QCaStringFormatting::formats format)
- QCaStringFormatting::formats **getFormat** ()
- void **setRadix** (unsigned int radix)

- unsigned int **getRadix** ()
- void **setNotation** (QCaStringFormatting::notations notation)
- QCaStringFormatting::notations **getNotation** ()
- void **setArrayAction** (QCaStringFormatting::arrayActions arrayAction)
- QCaStringFormatting::arrayActions **getArrayAction** ()
- void **setArrayIndex** (unsigned int arrayIndex)
- unsigned int **getArrayIndex** ()

Protected Attributes

- QCaStringFormatting **stringFormatting**

12.80.1 Member Function Documentation

12.80.1.1 void QCaStringFormattingMethods::setPrecision (unsigned int *precision*)

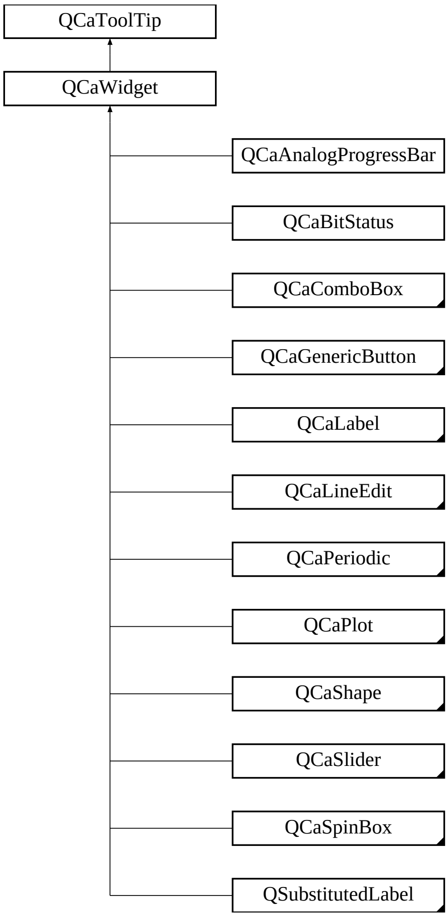
This class is used by QCaWidgets when setting properties managed by the QCaStringformatting class. This class calls matching set and get methods from the QCaStringformatting class but also calls back the QCawidget when setting properties so the widget can update itself based on the changes.

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

- epicsqt/ca_framework/widgets/include/QCaStringFormattingMethods.h
- epicsqt/ca_framework/widgets/src/QCaStringFormattingMethods.cpp

12.81 QCaToolTip Class Reference

Inheritance diagram for QCaToolTip:



Public Member Functions

- void updateToolTipVariable (const QString &variable)
- void **updateToolTipAlarm** (const QString &alarm)
- void **updateToolTipConnection** (bool connection)
- virtual void **updateToolTip** (const QString &tip)=0

12.81.1 Member Function Documentation

12.81.1.1 void QCaToolTip::updateToolTipVariable (const QString & *variable*)

Update the tool tip as requested by QCaWidget.

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

- epicsqt/ca_framework/widgets/include/QCaToolTip.h
- epicsqt/ca_framework/widgets/src/QCaToolTip.cpp

12.82 QCaVariableNamePropertyManager Class Reference

Handles variable name sets.

`#include <QCaVariableNamePropertyManager.h>`

Signals

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

Public Member Functions

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

12.82.1 Detailed Description

Handles variable name sets.

Handles user manipulation of variable name sets and associated macro substitutions.

Version

Revision:

#3

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.82.2 Constructor & Destructor Documentation

12.82.2.1 QCaVariableNamePropertyManager::QCaVariableNamePropertyManager ( )

Construction

Setup a timer so rapid changes to the variable name property are ignored. Only after the user has stopped typing for a while will the entry be used. The timer will be set on the first keystroke and reset with each subsequent keystroke untill the keystrokes stop for longer than the timeout period. Note, timers are not required if there is no user entering variable names of macro substitutions.

Default to first (and only) variable

12.82.3 Member Function Documentation

12.82.3.1 QString QCaVariableNamePropertyManager::getSubstitutionsProperty ( )

Return the Variable Name Substitutions property.

12.82.3.2 QString QCaVariableNamePropertyManager::getVariableNameProperty ( )

Return the Variable Name property.

12.82.3.3 void QCaVariableNamePropertyManager::setSubstitutionsProperty (QString *substitutionsPropertyIn*)

Set the Variable Name Substitutions property. This changes with every keystroke. Store the new value but don't subscribe yet. Instead, set a timer that will expire only when changes stop occuring for a while.

If the substitutions have changed as a result of a user typing, save them and set (or reset) a timer to complete when changes stop occuring. The timer will signal subscribeDelayExpired() If the change was not interactive, use the change immedietly.

12.82.3.4 void QCaVariableNamePropertyManager::setVariableIndex (unsigned int *variableIndexIn*)

Set the variable index. Used when multiple variables can affect an object.

12.82.3.5 void QCaVariableNamePropertyManager::setVariableNameProperty (QString *variableNamePropertyIn*)

Set the Variable Name property This changes with every keystroke. Store the new value but don't subscribe yet. Instead, set a timer that will expire only when changes stop occuring for a while.

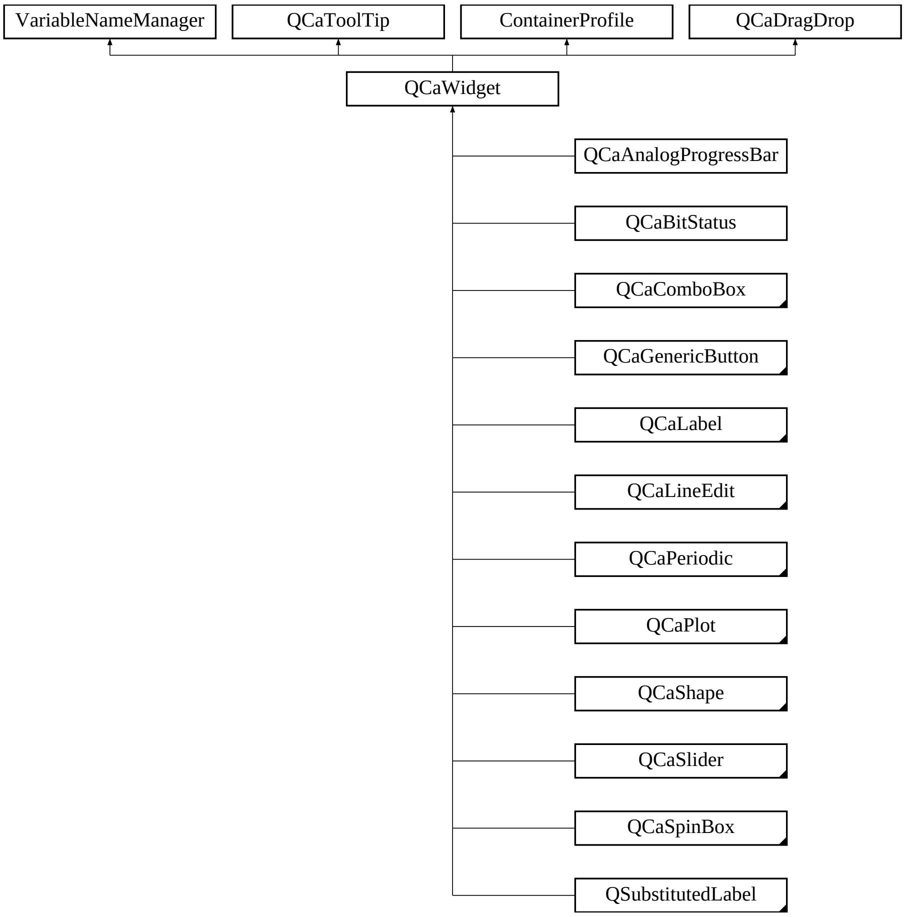
If the variable name has changed as a result of a user typing, save it and set (or reset) a timer to complete when changes stop occuring. The timer will signal subscribeDelayExpired() If the change was not interactive, use the change immedietly.

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

- epicsqt/ca_framework/data/include/QCaVariableNamePropertyManager.h
- epicsqt/ca_framework/data/src/QCaVariableNamePropertyManager.cpp

12.83 QCaWidget Class Reference

Inheritance diagram for QCaWidget:



Public Member Functions

- QCaWidget (QWidget ∗ownerIn)
- virtual ∼QCaWidget ()

Constructor.

- void activate ()

  Manager of messages to the user.

Public Attributes

- UserMessage userMessage

  Destructor.

Protected Member Functions

- void setNumVariables (unsigned int numVariablesIn)

  Initiate updates.
- qcaobject::QCaObject ∗ createConnection (unsigned int variableIndex)

  Flag the tool tip should be set to the variable name.
- virtual void setup ()=0

  Create a CA connection. Return a QCaObject if successfull.
- virtual qcaobject::QCaObject ∗ createQcaItem (unsigned int variableIndex)=0

  ???
- qcaobject::QCaObject ∗ getQcaItem (unsigned int variableIndex)

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

Protected Attributes

- bool subscribe

  Set the number of variables that will stream data updates to the widget. Default of 1 if not called.
- bool variableAsToolTip

  Flag if data updates should be requested.

12.83.1 Constructor & Destructor Documentation

12.83.1.1 QCaWidget::QCaWidget (QWidget ∗ *owner*)

This class is used as a base for all CA aware wigets, such as QCaLabel, QCaSpinBox, etc. It manages common issues including creating a source of CA data updates, handling error, warning and status messages, and setting tool tips based on variable names.

Note, there is tight integration between the CA aware widget classes, this class, and its base classes, especially VariableNameManager and QCaToolTip.

In particular, this class manages QCaObject classes that stream updates to the CA aware widget class. But this class, however, doesn't know how to format the data, or how the updates will be used. To resolve this, this class asks its parent class (such as

QCaLabel) to create the QCaObject class in what ever flavour it wants, by calling the virtual function createQcaItem. A QCaLabel, for example, wants string updates so it creates a QCaString which is based on a QCaObject class and formats all updates as strings.

The CA aware parent class (such as QCaLabel) defines a variable by calling VariableNameManager::setVariableName(). The VariableNameManager class calls the establishConnection function of the CA aware parent class, such as QCaLabel when it has a new variable name.

This class uses its base QCaToolTip class to format tool tips. that class in turn calls the CA aware parent class (such as QCaLabel) directly to make use of a new tool tip.

After construction, a CA aware widget is activated (starts updating) by calling it's establishConnection() function in one of two ways:

1) The variable name or variable name substitutions is changed by calling setVariableName or setVariableNameSubstitutions respectively. These functions are in the VariableNameManager class. The VariableNameManager 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 ASguiForm 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 ASguiForm class calls the activate() function in this class (QCaWiget). the activate() function calls establishConnection() in the CA aware widget for each variable. This simulates what the VariableNameManager 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 QCaWidget 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 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 QCaLabel creates a QCaString object. The QCaString 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 QCaLabel connects the 'stringChanged' signal fromthe QCaString object to its setLabelText slot.

Constructor

Initially flag no variables array is defined. This will be corrected when the first variable is declared

Default properties

12.83.1.2 QCaWidget::∼QCaWidget ( ) [virtual]

Constructor.

Destruction: Delete all variable sources for the widgeet

12.83.2 Member Function Documentation

12.83.2.1 void QCaWidget::activate ( )

Manager of messages to the user.

Initiate updates. This is only required when QCa widgets are loaded within a form and not directly by 'designer'. When loaded directly by 'designer' they are activated (a CA connection is established) as soon as either the variable name or variable name substitution properties are set

12.83.2.2 qcaobject::QCaObject ∗ QCaWidget::createConnection (unsigned int *variableIndex*) [protected]

Flag the tool tip should be set to the variable name.

Create a CA connection and initiates updates if required. This is called by the establishConnection function of CA aware widgets based on this class, such as a QCaLabel. If successfull it will return the QCaObject based object supplying data update signals

If the index is invalid do nothing This same test is also valid if qcaItem has never been set up yet as numVariables will be zero

If a variable name is present, add it to the tip

Add a space between variable names

Add the variable name to the tip

Remove any existing QCa connection

Connect to new variable. If a new variable name is present, ask the CA aware widget based on this class to create an appropriate object based on a QCaObject (by calling its createQcaItem() function). If that is successfull, supply it with a mechanism for handling errors and subscribe to the new variable if required.

12.83.2.3 qcaobject::QCaObject ∗ QCaWidget::getQcaItem (unsigned int *variableIndex*) [protected]

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

Return a reference to one of the qCaObjects used to stream CA data updates to the widget This is called by CA aware widgets based on this class, such as a QCaLabel, mainly when they want to connect to its signals to recieve data updates.

If the index is invalid return NULL. This same test is also valid if qcaItem has never been set up yet as numVariables will be zero

12.83.2.4 void QCaWidget::setNumVariables (unsigned int *numVariablesIn*) [protected]

Initiate updates.

Set the number of variables that will be used for this widget. Create an array of QCaObject based objects to suit. This is called by the CA aware widgets based on this class, such as a QCaLabel.

Get the number of variables that will be used by this widget Don't accept zero or the qca array will be invalid

Set up the number of variables managed by the variable name manager

Allocate the array of QCa objects

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

- epicsqt/ca_framework/data/include/QCaWidget.h
- epicsqt/ca_framework/data/src/QCaWidget.cpp

12.84 QCaWidgets Class Reference

Public Member Functions

- QCaWidgets (QObject ∗parent=0)
- virtual QList< QDesignerCustomWidgetInterface ∗ > customWidgets () const

12.84.1 Constructor & Destructor Documentation

12.84.1.1 QCaWidgets::QCaWidgets (QObject ∗ *parent* = 0)

Add all plugins that will be provided in QT Designer.

12.84.2 Member Function Documentation

12.84.2.1 QList< QDesignerCustomWidgetInterface ∗ > QCaWidgets::customWidgets ( ) const [virtual]

Multiplugin interface.

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

- epicsqt/ca_framework/plugins/include/QCaDesignerPlugin.h
- epicsqt/ca_framework/plugins/src/QCaDesignerPlugin.cpp

12.85 QEpicsPV Class Reference

Public Slots

- const QVariant & set (QVariant value, int delay=-1)

  Sets new value for the field.
- void setPV (const QString &_pvName="")

  Sets new PV.

Signals

- void connectionChanged (bool connected)
- void connected ()

  Emitted on successful connection.
- void disconnected ()

  Emitted on disconnection, destruction and in ::setPV() method.
- void valueChanged (const QVariant &value)
- void valueUpdated (const QVariant &value)
- void valueInited (const QVariant &value)

  Emitted on first value update after the connection was established.

Public Member Functions

- QEpicsPV (const QString &_pvName, QObject ∗parent=0)
- QEpicsPV (QObject ∗parent=0)
- ∼QEpicsPV ()

  Destructor.
- const QVariant & get () const
- void needUpdated () const

  Starts monitoring for the updates.
- const QVariant & getUpdated (int delay=defaultDelay) const

  Returns an updated value of the field.
- bool isConnected () const
- const QStringList & getEnum () const
- const QString & pv () const
- const QVariant & getReady (int delay=defaultDelay) const

  Ensures that the object is ready to operate.

Static Public Member Functions

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

  Static version of the ::getUpdated() method.
- static QVariant set (QString &_pvName, const QVariant &value, int delay=-1)

  Static version of the ::set() method.

12.85.1 Constructor & Destructor Documentation

12.85.1.1 QEpicsPV::QEpicsPV (const QString & *_pvName,* QObject ∗ *parent* = 0)

Constructor.

Parameters

_pvName	new PV name.
parent	Object's parent.

12.85.1.2 QEpicsPV::QEpicsPV (QObject ∗ *parent* = 0)

Constructor.

Parameters

parent	Object's parent.

12.85.2 Member Function Documentation

12.85.2.1 void QEpicsPV::connectionChanged (bool *connected*) [signal]

Emitted whenever the connection status has changed.

Parameters

connected	new connection status.

12.85.2.2 QVariant QEpicsPV::get (const QString & *_pvName,* int *delay* = defaultDelay) [static]

Static version of the ::getUpdated() method.

Constructs new object of the QEpicsPV type, gets it ready and returns the value. Useful for the single shot readouts.

Parameters

_pvName	PV name of the field.
delay	Maximum waiting time in milliseconds. If zero, then unlimited waiting.

Returns

Value of the field. If the value was not updated during the specified delay then returns an invalid object.

12.85.2.3 const QVariant & QEpicsPV::get ( ) const

Returns current value of the field.

Returns

Latest value of the PV field or invalid object if the field is not connected or before first update.

12.85.2.4 const QStringList & QEpicsPV::getEnum ( ) const

PV's enumeration.

Returns

The enumeration of the PV field, or the empty list if the PV is not an enumeration.

12.85.2.5 const QVariant & QEpicsPV::getReady (int *delay* = `defaultDelay`) const

Ensures that the object is ready to operate.

Connection to the EPICS PV and first update takes certain time after the ::setPV() method or the constructor with name are called. This method waits for the PV to get connected and the value to be initialized before returning.

Parameters

delay	Maximum waiting time in milliseconds. If zero, then unlimited waiting. Note that the actual waiting time can be twice as large as the delay: one delay to wait for the connection and one more to wait for the first update.

Returns

Field's updated value. If the connection could not be established or the value was not updated then returns an invalid data.

12.85.2.6 const QVariant & QEpicsPV::getUpdated (int *delay* = `defaultDelay`) const

Returns an updated value of the field.

Same as ::get(), but ensures that the PV's value was updated since the last call to ::needUpdated() method or establishing the connection.

Parameters

delay	Maximum waiting time in milliseconds. If zero, then unlimited waiting.

Returns

Updated value of the field. If the value was not updated during the specified delay then returns an invalid object.

12.85.2.7 bool QEpicsPV::isConnected ( ) const

Connection status.

Returns

true if connected, false otherwise.

12.85.2.8 void QEpicsPV::needUpdated ( ) const

Starts monitoring for the updates.

Makes sence only in combination with further call to ::getUpdated(). Ensures that the next call to ::getUpdated() method will return only after a new value of the PV is received.

12.85.2.9 const QString & QEpicsPV::pv ( ) const

PV name.

Returns

PV name.

12.85.2.10 const QVariant & QEpicsPV::set (QVariant *value*, int *delay* = `-1`) `[slot]`

Sets new value for the field.

Parameters

value	Value to be set.
delay	Negative delay tells that the confirmation of the update is nor required (just puts the new value and returns). If non-negative, then sets the maximum time in milliseconds to wait for the update event before return. Zero delay sets unlimited waiting time. Note that the method does not confirm the updated value to be equal to the one set.

Returns

Last updated value or invalid object if could not confirm the update within the specified delay time.

12.85.2.11 QVariant QEpicsPV::set (QString & *_pvName*, const QVariant & *value*, int *delay* = -1) [static]

Static version of the ::set() method.

Constructs new object of the QEpicsPV type, gets it ready, sets the new value and returns the result of the ::set() method. Useful for the single shot write.

Parameters

_pvName	PV name of the field.
value	Value to be set.
delay	Negative delay tells that the confirmation of the update is nor required (just puts the new value and returns). If non-negative, then sets the maximum time in milliseconds to wait for the update event before return. Zero delay sets unlimited waiting time. Note that the method does not confirm the updated value to be equal to the one set.

Returns

Updated value or invalid object if could not confirm the update within the specified delay time.

12.85.2.12 void QEpicsPV::setPV (const QString & *_pvName* = "") [slot]

Sets new PV.

First disconnects any previous PV (if any was connected) and connects to the new one (if non-empty).

Parameters

_pvName	New PV name.

12.85.2.13 void QEpicsPV::valueChanged (const QVariant & *value*) [signal]

Emitted whenever the field is updated with a value different from the old one.

Parameters

value	new value.

12.85.2.14 void QEpicsPV::valueUpdated (const QVariant & *value*) [signal]

Emitted whenever the field is updated (even if the new value is equal to the old one).

Parameters

value	new value.

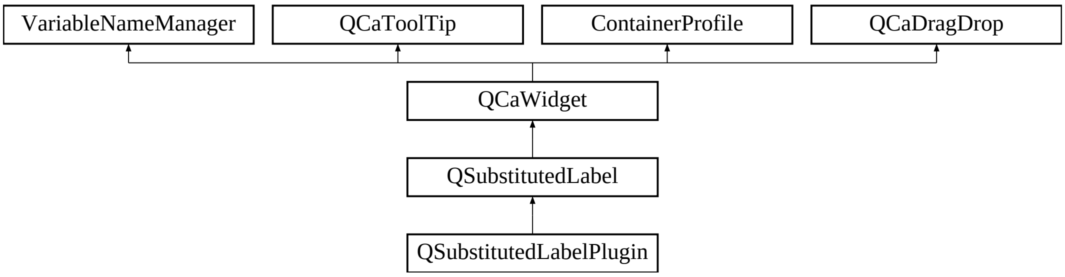
The documentation for this class was generated from the following files:

- epicsqt/ca_framework/data/include/qepicspv.h
- epicsqt/ca_framework/data/src/qepicspv.cpp

12.86 QSubstitutedLabel Class Reference

```
#include <QSubstitutedLabel.h>
```

Inheritance diagram for QSubstitutedLabel:



Public Member Functions

- QSubstitutedLabel (QWidget ∗parent=0)
- void establishConnection (unsigned int variableIndex)
- void **setVariableNameAndSubstitutions** (QString variableNameIn, QString variableNameSubstitutionsIn, unsigned int variableIndex)
- void **setLabelTextProperty** (QString labelTextIn)
- QString **getLabelTextProperty** ()
- QString **getLabelTextPropertyFormat** ()
- void **setLabelTextPropertyFormat** (QString labelTextIn)

Protected Attributes

- QString **labelText**

12.86.1 Detailed Description

This class is based on the Qt label widget. It is NOT a CA aware widget. It does, however, use the QCaWidget and its VariableNameManager base class to manage substituting the label's text in the same way most other widgets manage substitutions in variable names. It is tighly integrated with the base class QCaWidget. Refer to QCaWidget.cpp for details, but watch out for references to varable names when it is actually just talking about the text that will be displayed in the QLabel widget. This widget is required to implement some virtual functions required by QCaWidget for managing data connections. These functions are just stubs.

12.86.2 Constructor & Destructor Documentation

12.86.2.1 QSubstitutedLabel::QSubstitutedLabel (QWidget ∗ *parent* = 0)

This class is a label that uses the same macro substitution mechanisms as other CA aware label widgets. It is intended to enable unique titles and text in sub forms.

Constructor with no initialisation

12.86.3 Member Function Documentation

12.86.3.1 void QSubstitutedLabel::establishConnection (unsigned int *variableIndex*) [virtual]

Update the label text with the required substitutions. Implementation of VariableNameManager's virtual funtion to establish a connection to a PV as the variable name has changed. NOTE, In this usage, the variable name manager is used to manage substitutions in the text displayed in the label, so the 'Variable Name' is not used by this widget (as for most other widgets), rather the substitutions are applied to fixed text.

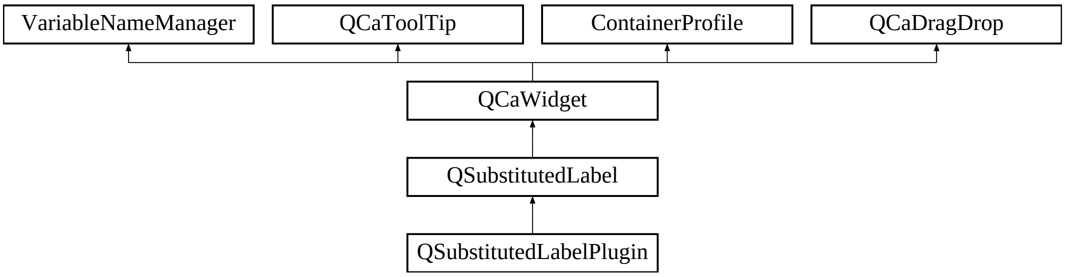
Implements VariableNameManager.

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

- epicsqt/ca_framework/widgets/include/QSubstitutedLabel.h
- epicsqt/ca_framework/widgets/src/QSubstitutedLabel.cpp

12.87 QSubstitutedLabelPlugin Class Reference

Inheritance diagram for QSubstitutedLabelPlugin:



Public Member Functions

- QSubstitutedLabelPlugin (QWidget ∗parent=0)

  Constructors.
- void **setVariableNameSubstitutionsProperty** (QString variableNameSubstitutions)

- QString **getVariableNameSubstitutionsProperty** ()

Properties

- QString labelText

  Label text to be substituted.
- QString textSubstitutions

12.87.1 Constructor & Destructor Documentation

12.87.1.1 QSubstitutedLabelPlugin::QSubstitutedLabelPlugin (QWidget ∗ *parent* = 0)

Constructors.

???

Set up a connection to recieve variable name property changes The variable name property manager class only delivers an updated variable name after the user has stopped typing

12.87.2 Property Documentation

12.87.2.1 QString QSubstitutedLabelPlugin::textSubstitutions [read, write]

Text substitutions. NOTE, this plugin uses the variable-name-and-substitutions mechanism used to manage variable names to manage text displayed in the label.

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

- epicsqt/ca_framework/plugins/include/QSubstitutedLabelPlugin.h
- epicsqt/ca_framework/plugins/src/QSubstitutedLabelPlugin.cpp

12.88 QSubstitutedLabelPluginManager Class Reference

```
#include <QSubstitutedLabelPluginManager.h>
```

Public Member Functions

- QSubstitutedLabelPluginManager (QObject ∗parent=0)
- bool isContainer () const
- bool isInitialized () const
- QIcon icon () const
- QString group () const
- QString includeFile () const
- QString name () const

- QString toolTip () const
- QString whatsThis () const
- QWidget ∗ createWidget (QWidget ∗parent)
- void initialize (QDesignerFormEditorInterface ∗core)

12.88.1 Detailed Description

???

12.88.2 Constructor & Destructor Documentation

12.88.2.1 QSubstitutedLabelPluginManager::QSubstitutedLabelPluginManager (QObject ∗ *parent* = 0)

???

12.88.3 Member Function Documentation

12.88.3.1 QWidget ∗ QSubstitutedLabelPluginManager::createWidget (QWidget ∗ *parent*)

Widget factory. Creates a QSubstitutedLabel widget.

12.88.3.2 QString QSubstitutedLabelPluginManager::group ( ) const

Name of group Qt Designer will add widget to.

12.88.3.3 QIcon QSubstitutedLabelPluginManager::icon ( ) const

Icon for widget. Used by Qt Designer in widget list.

12.88.3.4 QString QSubstitutedLabelPluginManager::includeFile ( ) const

???

???

12.88.3.5 void QSubstitutedLabelPluginManager::initialize (QDesignerFormEditorInterface ∗ *core*)

???

12.88.3.6 bool QSubstitutedLabelPluginManager::isContainer ( ) const

???

12.88.3.7 bool QSubstitutedLabelPluginManager::isInitialized ( ) const

???

12.88.3.8 QString QSubstitutedLabelPluginManager::name ( ) const

Name for widget. Used by Qt Designer in widget list.

12.88.3.9 QString QSubstitutedLabelPluginManager::toolTip ( ) const

Tool tip for widget. Used by Qt Designer in widget list.

12.88.3.10 QString QSubstitutedLabelPluginManager::whatsThis ( ) const

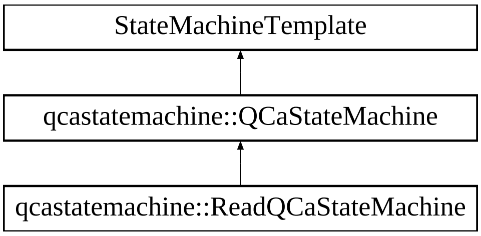
???

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

- epicsqt/ca_framework/plugins/include/QSubstitutedLabelPluginManager.h
- epicsqt/ca_framework/plugins/src/QSubstitutedLabelPluginManager.cpp

12.89 qcastatemachine::ReadQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::ReadQCaStateMachine:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::ReadQCaStateMachine

Public Member Functions

- ReadQCaStateMachine (void ∗parent)
- bool process (int requestedState)

12.89.1 Constructor & Destructor Documentation

12.89.1.1 ReadQCaStateMachine::ReadQCaStateMachine (void ∗ *parent*)

Startup for the Reading statemachine.

12.89.2 Member Function Documentation

12.89.2.1 bool ReadQCaStateMachine::process (int *requestedState*) [virtual]

Process the Reading statemachine.

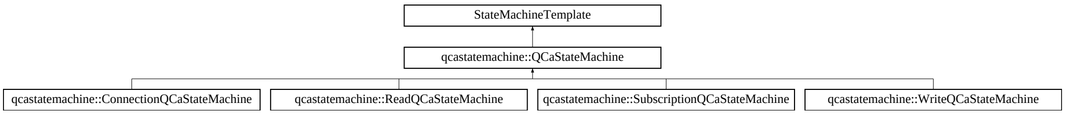
Implements qcastatemachine::QCaStateMachine.

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

12.90 StateMachineTemplate Class Reference

Inheritance diagram for StateMachineTemplate:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::ConnectionQCaStateMachine
qcastatemachine::ReadQCaStateMachine
qcastatemachine::SubscriptionQCaStateMachine
qcastatemachine::WriteQCaStateMachine

Public Member Functions

- virtual bool **process** (int requestedState)=0

Public Attributes

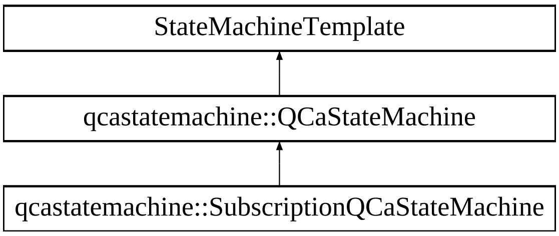
- int **currentState**
- int **requestState**

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h

12.91 qcastatemachine::SubscriptionQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::SubscriptionQCaStateMachine:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::SubscriptionQCaStateMachine

Public Member Functions

- SubscriptionQCaStateMachine (void ∗parent)
- bool process (int requestedState)

12.91.1 Constructor & Destructor Documentation

12.91.1.1 SubscriptionQCaStateMachine::SubscriptionQCaStateMachine (void ∗ *parent*)

Startup for the Subscription statemachine.

12.91.2 Member Function Documentation

12.91.2.1 bool SubscriptionQCaStateMachine::process (int *requestedState*) [virtual]

Process the Subscription statemachine.

Implements qcastatemachine::QCaStateMachine.

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

12.92 trace Class Reference

Public Attributes

- QVector< QCaDateTime > **timeStamps**
- QVector< double > **xdata**
- QVector< double > **ydata**
- QwtPlotCurve ∗ **curve**
- QColor **color**
- QString **legend**
- bool **waveform**
- QwtPlotCurve::CurveStyle **style**

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

- epicsqt/ca_framework/widgets/include/QCaPlot.h

12.93 userInfoStruct Class Reference

Public Attributes

- bool **enable**
- double **value1**
- double **value2**
- QString **elementText**

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

- epicsqt/ca_framework/widgets/include/QCaPeriodic.h

12.94 QCaPeriodic::userInfoStructArray Struct Reference

Public Attributes

- userInfoStruct **array** [NUM_ELEMENTS]

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

- epicsqt/ca_framework/widgets/include/QCaPeriodic.h

12.95 UserMessage Class Reference

User message manager.

```
#include <UserMessage.h>
```

Signals

- void **statusMessage** (QString message)
- void **warningMessage** (QString message)
- void **errorMessage** (QString message)
- void **generalMessage** (QString message)

Public Member Functions

- UserMessage ()

  Construction.
- ∼UserMessage ()

  Destruction.
- void setup (QObject ∗statusMessageConsumer, QObject ∗warningMessageConsumer, QObject ∗errorMessageConsumer)
- void setup (QObject ∗generalMessageConsumer)
- void sendStatusMessage (QString message)
- void sendStatusMessage (QString message, QString source)
- void sendWarningMessage (QString message)
- void sendWarningMessage (QString message, QString source)
- void sendErrorMessage (QString message)
- void sendErrorMessage (QString message, QString source)

12.95.1 Detailed Description

User message manager.

Version

Revision:

#4

Date

DateTime:

2010/06/23 07:49:40

Author

andrew.rhyder

12.95.2 Constructor & Destructor Documentation

12.95.2.1 UserMessage::UserMessage ( )

Construction.

Description:

A class to manage user messages.

Prior to use the class instance is supplied with slot references to receive status, warning, and error messages.

Once setup, this class can be used to present messages to the user.

If the slots used change only the call to UserMessage::setup() needs to change. The methods used to present messages to the user are overloaded so the caller can supply either a message string, or a message string and a source string to help identify where teh message originated.

A reference to an instance of this class can be passed to an object that needs to present messages to the user, but has no user context. That is, it does not know what slots to connect to.

FUTURE USES Perhaps add additioinal setup methods to provide log file locations, connections to slots that take more than a single QString, etc.

12.95.3 Member Function Documentation

12.95.3.1 void UserMessage::sendErrorMessage (QString *message*)

Send an error message to the user. A string containing the message is required

12.95.3.2 void UserMessage::sendErrorMessage (QString *message,* QString *source*)

Send an error message to the user. A string containing the message and a string containing information as to the source of the message is required

12.95.3.3 void UserMessage::sendStatusMessage (QString *message,* QString *source*)

Send a status message to the user. A string containing the message and a string containing information as to the source of the message is required

12.95.3.4 void UserMessage::sendStatusMessage (QString *message*)

Send a status message to the user. A string containing the message is required

12.95.3.5 void UserMessage::sendWarningMessage (QString *message*)

Send a warning message to the user. A string containing the message is required

12.95.3.6 void UserMessage::sendWarningMessage (QString *message,* QString *source*)

Send a warning message to the user. A string containing the message and a string containing information as to the source of the message is required

12.95.3.7 void UserMessage::setup (QObject ∗ *generalMessageConsumer*)

Set up the signal / slot connections required for user messages.

The caller can supply a slot that accepts a QString for general messages. Once set up, this class can be used to send messages to the user without knowledge of what signal needs to be emited.

12.95.3.8 void UserMessage::setup (QObject ∗ *statusMessageConsumer,* QObject ∗ *warningMessageConsumer,* QObject ∗ *errorMessageConsumer*)

Set up the signal / slot connections required for various types of messages.

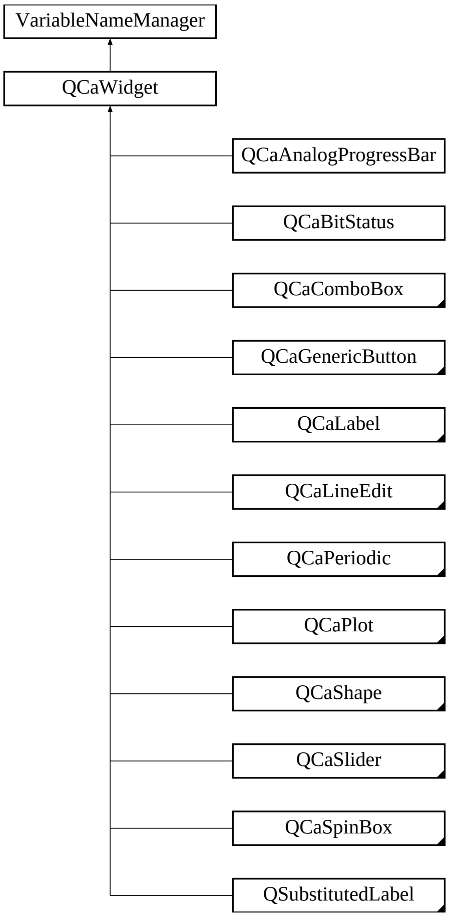
The caller can supply a slot that accepts a QString for each of status, warning, and error messages. Once set up, this class can be used to send messages to the user without knowledge of what signal needs to be emited.

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

- epicsqt/ca_framework/data/include/UserMessage.h
- epicsqt/ca_framework/data/src/UserMessage.cpp

12.96 VariableNameManager Class Reference

Inheritance diagram for VariableNameManager:



Public Member Functions

- VariableNameManager ()
- void variableNameManagerInitialise (unsigned int numVariables)
- QString getOriginalVariableName (unsigned int variableIndex)

  Used to set up the number of variables required (defaults to 1 if this is not called)
- QString getVariableNameSubstitutions ()

  Return variable name prior to any macro substitutions. (eg, SR$SECTOR$V)
- QString getSubstitutedVariableName (unsigned int variableIndex)

  Return macro substitutions used for variable names (eg, SECTOR=01,V=PRESURE)
- void initialiseVariableNameSubstitutions (const QString &substitutions)

  Return variable name after macro substitutions.
- virtual void establishConnection (unsigned int variableIndex)=0

  Initialise macro substitutions. May be used on creation.
- void setVariableNameSubstitutionsOverride (const QString &substitutions)

  Create a CA connection and initiates updates if required.

- void setVariableName (const QString &variableName, unsigned int variableIndex)
- void setVariableNameSubstitutions (const QString &substitutions)

  Accept a new variable name which may include substitution keys preceeded by $.
- QString substituteThis (const QString string)

  Accept a new set of macro substitutions in the form KEY1=VALUE1,KEY2=VALUE2.

12.96.1 Constructor & Destructor Documentation

12.96.1.1 VariableNameManager::VariableNameManager ( )

Assume one variable name.

12.96.2 Member Function Documentation

12.96.2.1 QString VariableNameManager::getOriginalVariableName (unsigned int *variableIndex*)

Used to set up the number of variables required (defaults to 1 if this is not called)

Get the current variable name. Supply a variable index if this class is managing more than one variable name.

12.96.2.2 QString VariableNameManager::getSubstitutedVariableName (unsigned int *variableIndex*)

Return macro substitutions used for variable names (eg, SECTOR=01,V=PRESURE)

Get the current variable name with substitutions applied.

12.96.2.3 QString VariableNameManager::getVariableNameSubstitutions ( )

Return variable name prior to any macro substitutions. (eg, SR$SECTOR$V)

Get the current variable name substitutions. Note the substitutions for the first variable are always returned as the same substitutions are used for every entry in the variableNames list.

12.96.2.4 void VariableNameManager::setVariableName (const QString & *variableNameIn,* unsigned int *variableIndex*)

Set the variable name. Macro substitution will be performed. A new connection is established.

Save the variable name and request the variableName data if updates are required

12.96.2.5 void VariableNameManager::setVariableNameSubstitutions (const QString & *macroSubstitutionsIn*)

Accept a new variable name which may include substitution keys preceeded by $.

Set the variable name substitutions. Note, if there is more than one variable name in the list, the same substitutions are used for every entry in the variableNames list. Macro substitution will be performed. A new connection is established.

12.96.2.6 void VariableNameManager::setVariableNameSubstitutionsOverride (const QString & *macroSubstitutionsOverrideIn*)

Create a CA connection and initiates updates if required.

Override variable name substitutions. This is called when any macro substitutions set by default are overridden by the creator.

12.96.2.7 QString VariableNameManager::substituteThis (const QString *string*)

Accept a new set of macro substitutions in the form KEY1=VALUE1,KEY2=VALUE2.

Perform a set of substitutions throughout a string. Replace $MACRO1 with VALUE1, $MACRO2 with VALUE2, etc wherever they appear in the string.

Start with the initial string

Generate a list where each item in the list is a single substitution in the form MACRO1=VALUE1

Perform all substitutions

Get the key to search for and the value to replace it with. If both a key and value are present (and nothing more) perform the substitution. Note, the key will only be replaced if it is prefixed with a $.

12.96.2.8 void VariableNameManager::variableNameManagerInitialise (unsigned int *numVariables*)

Define the required number of variables to manage. The list of variable names is initially conatins a single variable name so this need only be called if more than one variable name is required.

Clear out any existing variables

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

- epicsqt/ca_framework/data/include/VariableNameManager.h
- epicsqt/ca_framework/data/src/VariableNameManager.cpp

12.97 WidgetRef Class Reference

Public Member Functions

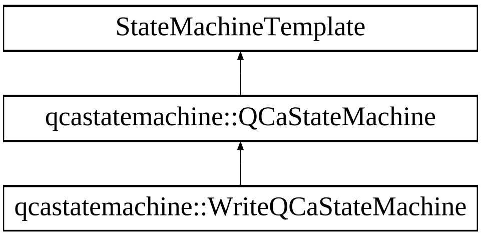
- **WidgetRef** (QCaWidget ∗refIn)
- QCaWidget ∗ **getRef** ()

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

- epicsqt/ca_framework/plugins/include/ContainerProfile.h

12.98 qcastatemachine::WriteQCaStateMachine Class Reference

Inheritance diagram for qcastatemachine::WriteQCaStateMachine:


StateMachineTemplate
qcastatemachine::QCaStateMachine
qcastatemachine::WriteQCaStateMachine

Public Member Functions

- WriteQCaStateMachine (void ∗parent)
- bool process (int requestedState)

12.98.1 Constructor & Destructor Documentation

12.98.1.1 WriteQCaStateMachine::WriteQCaStateMachine (void ∗ *parent*)

Startup for the Writing statemachine.

12.98.2 Member Function Documentation

12.98.2.1 bool WriteQCaStateMachine::process (int *requestedState*) [virtual]

Process the Writing statemachine.

Implements qcastatemachine::QCaStateMachine.

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

- epicsqt/ca_framework/data/include/QCaStateMachine.h
- epicsqt/ca_framework/data/src/QCaStateMachine.cpp

Index