FTutor1D

1.0

Generated by Doxygen 1.8.11

Contents

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

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

Chapter 5

Namespace Documentation

5.1 QCP Namespace Reference

Enumerations

- enum MarginSide {
  msLeft = 0x01, msRight = 0x02, msTop = 0x04, msBottom = 0x08,
  msAll = 0xFF, msNone = 0x00 }
- enum AntialiasedElement {
  aeAxes = 0x0001, aeGrid = 0x0002, aeSubGrid = 0x0004, aeLegend = 0x0008,
  aeLegendItems = 0x0010, aePlottables = 0x0020, aeItems = 0x0040, aeScatters = 0x0080,
  aeErrorBars = 0x0100, aeFills = 0x0200, aeZeroLine = 0x0400, aeAll = 0xFFFF,
  aeNone = 0x0000 }
- enum PlottingHint { phNone = 0x000, phFastPolylines = 0x001, phForceRepaint = 0x002, phCacheLabels = 0x004 }
- enum Interaction {
  iRangeDrag = 0x001, iRangeZoom = 0x002, iMultiSelect = 0x004, iSelectPlottables = 0x008,
  iSelectAxes = 0x010, iSelectLegend = 0x020, iSelectItems = 0x040, iSelectOther = 0x080 }

Functions

- bool **isInvalidData** (double value)
- bool **isInvalidData** (double value1, double value2)
- void **setMarginValue** (QMargins &margins, QCP::MarginSide side, int value)
- int **getMarginValue** (const QMargins &margins, QCP::MarginSide side)

5.1.1 Detailed Description

The QCP Namespace contains general enums and QFlags used throughout the QCustomPlot library

5.1.2 Enumeration Type Documentation

5.1.2.1 enum QCP::AntialiasedElement

Defines what objects of a plot can be forcibly drawn antialiased/not antialiased. If an object is neither forcibly drawn antialiased nor forcibly drawn not antialiased, it is up to the respective element how it is drawn. Typically it provides a *setAntialiased* function for this.

`AntialiasedElements` is a flag of or-combined elements of this enum type.

See also

QCustomPlot::setAntialiasedElements, QCustomPlot::setNotAntialiasedElements

Enumerator

aeAxes `0x0001` Axis base line and tick marks

aeGrid `0x0002` Grid lines

aeSubGrid `0x0004` Sub grid lines

aeLegend `0x0008` Legend box

aeLegendItems `0x0010` Legend items

aePlottables `0x0020` Main lines of plottables (excluding error bars, see element aeErrorBars)

aeItems `0x0040` Main lines of items

aeScatters `0x0080` Scatter symbols of plottables (excluding scatter symbols of type ssPixmap)

aeErrorBars `0x0100` Error bars

aeFills `0x0200` Borders of fills (e.g. under or between graphs)

aeZeroLine `0x0400` Zero-lines, see QCPGrid::setZeroLinePen

aeAll `0xFFFF` All elements

aeNone `0x0000` No elements

5.1.2.2 enum QCP::Interaction

Defines the mouse interactions possible with QCustomPlot.

`Interactions` is a flag of or-combined elements of this enum type.

See also

QCustomPlot::setInteractions

Enumerator

iRangeDrag `0x001` Axis ranges are draggable (see QCPAxisRect::setRangeDrag, QCPAxisRect::set↵ RangeDragAxes)

iRangeZoom `0x002` Axis ranges are zoomable with the mouse wheel (see QCPAxisRect::setRangeZoom, QCPAxisRect::setRangeZoomAxes)

iMultiSelect `0x004` The user can select multiple objects by holding the modifier set by QCustomPlot::set↵ MultiSelectModifier while clicking

iSelectPlottables `0x008` Plottables are selectable (e.g. graphs, curves, bars,... see QCPAbstractPlottable)

iSelectAxes `0x010` Axes are selectable (or parts of them, see QCPAxis::setSelectableParts)

iSelectLegend `0x020` Legends are selectable (or their child items, see QCPLegend::setSelectableParts)

iSelectItems `0x040` Items are selectable (Rectangles, Arrows, Textitems, etc. see QCPAbstractItem)

iSelectOther `0x080` All other objects are selectable (e.g. your own derived layerables, the plot title,...)

5.1.2.3 enum QCP::MarginSide

Defines the sides of a rectangular entity to which margins can be applied.

See also

QCPLayoutElement::setAutoMargins, QCPAxisRect::setAutoMargins

Enumerator

msLeft `0x01` left margin

msRight `0x02` right margin

msTop `0x04` top margin

msBottom `0x08` bottom margin

msAll `0xFF` all margins

msNone `0x00` no margin

5.1.2.4 enum QCP::PlottingHint

Defines plotting hints that control various aspects of the quality and speed of plotting.

See also

QCustomPlot::setPlottingHints

Enumerator

phNone `0x000` No hints are set

phFastPolylines `0x001` Graph/Curve lines are drawn with a faster method. This reduces the quality < especially of the line segment joins. (Only relevant for solid line pens.)

phForceRepaint `0x002` causes an immediate repaint() instead of a soft update() when QCustomPlot↩ ::replot() is called with parameter QCustomPlot::rpHint. < This is set by default to prevent the plot from freezing on fast consecutive replots (e.g. user drags ranges with mouse).

phCacheLabels `0x004` axis (tick) labels will be cached as pixmaps, increasing replot performance.

Chapter 6

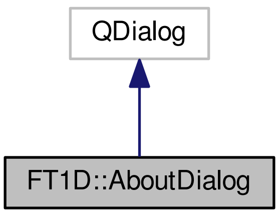
Class Documentation

6.1 FT1D::AboutDialog Class Reference

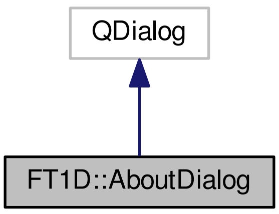
The AboutDialog class is a simple dialog window with information about the application and its creator.

```
#include <aboutdialog.h>
```

Inheritance diagram for FT1D::AboutDialog:



Collaboration diagram for FT1D::AboutDialog:



Public Member Functions

- AboutDialog (QWidget ∗parent, const Translation ∗language, QString icon)

  AboutDialog costructor.
- virtual ∼AboutDialog ()

  Desctructor.

6.1.1 Detailed Description

The AboutDialog class is a simple dialog window with information about the application and its creator.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 FT1D::AboutDialog::AboutDialog (QWidget ∗ *parent*, const **Translation** ∗ *language*, QString *icon*) [explicit]

AboutDialog costructor.

Parameters

parent	the parent object, should be MainWindow
localization	the instance of Localization class, which provides translated labels

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

- src/aboutdialog.h

6.2 QCPAxisPainterPrivate::CachedLabel Struct Reference

Public Attributes

- QPointF **offset**
- QPixmap **pixmap**

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

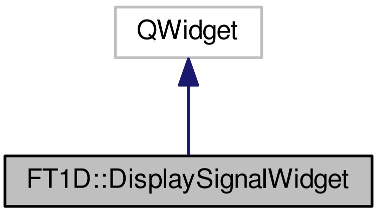
- src/qcustomplot/qcustomplot.h

6.3 FT1D::DisplaySignalWidget Class Reference

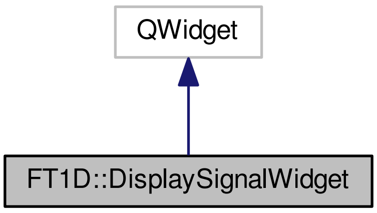
The DisplaySignalWidget class.

```
#include <displaysignalwidget.h>
```

Inheritance diagram for FT1D::DisplaySignalWidget:



Collaboration diagram for FT1D::DisplaySignalWidget:



Public Slots

- void plotDefaultScale ()

  plotDefaultScale rescale the plot so that the original part of the signal just fits the plot
- void displayWithLines (bool value)

  displayWithLines
- void enableCentering (bool enabled)

  enableCentering

Signals

- void needFrequencyUpdate (int idx, double value)

  needFrequencyUpdate mouse was moved so that it points to value at index in the original signal.
- void needUpdateFiltered ()

  needUpdateFiltered request recomputing and redrawing the filtered graph, usually when the selected point was edited
- void editModeNeedUpdate ()

  editModeNeedUpdate request recomputing and redrawing all graphs based on current edit mode state
- void callForSaveState ()

  callForSaveState request recording the current signals
- void callForSaveEditModeState ()

  callForSaveEditModeState request recording the current edit mode state
- void openEditMode ()

  openEditMode notifies the application that an action to open edit mode was triggered.
- void displayValueStatusBar (int x, int index)

  displayValueStatusBar notifies the main window that the value at index index should be displayed in the status bar.
- void mouseLeave ()

  mouseLeave notifies the application that the mouse left the widget.

Public Member Functions

- DisplaySignalWidget (enum DisplaySignalWidgetType type, bool allowEditMode, QWidget ∗parent=0)

  DisplaySignalWidget constructor.
- virtual ∼DisplaySignalWidget ()

  ∼DisplaySignalWidget destructor
- void displaySignal (Signal ∗signal, bool shadowPrevious=false)

  displaySignal displays the signal signal in the plot in the widget
- void plotReplot ()

  plotReplot replots the plot.
- void setAutoScaling (bool val)

  setAutoScaling
- void setDefaultTexts ()

  setDefaultTexts sets defaults values to each text or title or label in the window.
- void setLocalizedTexts (const Translation ∗language)

  setLocalizedTexts sets text, title or label values according to given Translation object'
- void setInteractionsEnabled (bool val)

  setInteractionsEnabled disables or enables interaction in the widget
- void setSibling (DisplaySignalWidget ∗&other)

  setSibling sets a pointer to a sibling widget.
- void forceXAxisUpdate ()

  forceXAxisUpdate Artificially triggers plotXAxisChanged callback

6.3.1 Detailed Description

The DisplaySignalWidget class.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 FT1D::DisplaySignalWidget::DisplaySignalWidget (enum DisplaySignalWidgetType *type*, bool *allowEditMode*, QWidget ∗ *parent* = 0) [explicit]

DisplaySignalWidget constructor.

Parameters

type	type of the widget
allowEditMode	whether edit mode can be triggered from this instance
parent	parent object (MainWindow)

6.3.3 Member Function Documentation

6.3.3.1 void FT1D::DisplaySignalWidget::displaySignal (Signal * *signal*, bool *shadowPrevious* = `false`)

displaySignal displays the signal *signal* in the plot in the widget

Parameters

signal	signal to display
shadowPrevious	if set to true, then previous signal is kept in the graph in gray colour until the next repaint

6.3.3.2 void FT1D::DisplaySignalWidget::displayValueStatusBar (int *x*, int *index*) `[signal]`

displayValueStatusBar notifies the main window that the value at index *index* should be displayed in the status bar.

Parameters

x	x coordinate of the value at index *index*
index	

6.3.3.3 void FT1D::DisplaySignalWidget::displayWithLines (bool *value*) `[slot]`

displayWithLines

Parameters

value	if true, draw graph as a polyline

6.3.3.4 void FT1D::DisplaySignalWidget::enableCentering (bool *enabled*) `[slot]`

enableCentering

Parameters

enabled	if true, draw the graph such that the signal displayed is centered

6.3.3.5 void FT1D::DisplaySignalWidget::needFrequencyUpdate (int *idx*, double *value*) `[signal]`

needFrequencyUpdate mouse was moved so that it points to value at index in the original signal.

Parameters

idx	index of a point
value	value, which is currently at *idx*

6.3.3.6 void FT1D::DisplaySignalWidget::setAutoScaling (bool *val*) `[inline]`

setAutoScaling

Parameters

val	if true, auto rescale after applying a specific operation is enabled.

6.3.3.7 void FT1D::DisplaySignalWidget::setInteractionsEnabled (bool *val*)

setInteractionsEnabled disables or enables interaction in the widget

Parameters

val	true to enable, false to disable.

6.3.3.8 void FT1D::DisplaySignalWidget::setLocalizedTexts (const Translation ∗ *language*)

setLocalizedTexts sets text, title or label values according to given Translation object‘

Parameters

language	Translation object used to set texts

6.3.3.9 void FT1D::DisplaySignalWidget::setSibling (DisplaySignalWidget ∗& *other*)

setSibling sets a pointer to a sibling widget.

This widget is rescaled together with this

Parameters

other	a valid pointer to another widget. Causion: pointer validity is not checked!

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

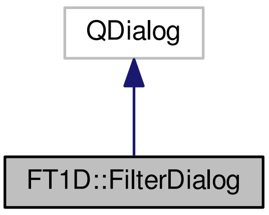
- src/displaysignalwidget.h

6.4 FT1D::FilterDialog Class Reference

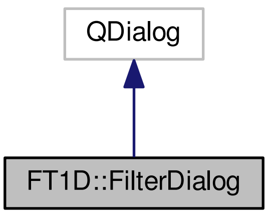
The FilterDialog class is a window to setup a filter.

`#include <filterdialog.h>`

Inheritance diagram for FT1D::FilterDialog:



Collaboration diagram for FT1D::FilterDialog:



Public Member Functions

- FilterDialog (FilterType type, Signal &magnitude, const Translation ∗language, QWidget ∗parent=nullptr)

    FilterDialog constructor creates the filter dialog window.
- virtual ∼FilterDialog ()

    Desctructor.

6.4.1 Detailed Description

The FilterDialog class is a window to setup a filter.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 FT1D::FilterDialog::FilterDialog (**FilterType** *type*, **Signal** & *magnitude*, const **Translation** ∗ *language*, QWidget ∗ *parent* = `nullptr`) `[explicit]`

FilterDialog constructor creates the filter dialog window.

Parameters

type	type of filter, for which the window is created
magnitude	magnitude of the signal, to which we will apply the filter
language	a translation for this window
parent	a parent object, typically MainWindow

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

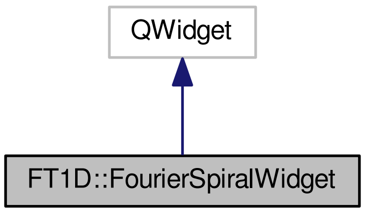
- src/filterdialog.h

6.5 FT1D::FourierSpiralWidget Class Reference

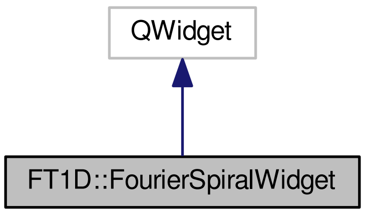
The FourierSpiralWidget class.

```
#include <fourierspiralwidget.h>
```

Inheritance diagram for FT1D::FourierSpiralWidget:



Collaboration diagram for FT1D::FourierSpiralWidget:



Public Member Functions

- FourierSpiralWidget (QWidget ∗parent=0)

  FourierSpiralWidget constructor.
- void displayFrequency (double frequency, double magnitudeVal, double phaseVal, double maxMagnitudeVal, int signalLength)

  displayFrequency the main callback, sets the variables for proper display
- void setNormalized (bool value)

  setNormalized sets, whether or not to scale and shift the basis function
- void clearFrequency ()

  clearFrequency sets displaying enabled to false.
- void setMagnitudeAndPhase (double mag, double pha)

  setMagnitudeAndPhase updates magnitude and phase values.
- void newSignal (int length)

  newSignal changes length of currently loaded signal.

Protected Member Functions

- void paintEvent (QPaintEvent ∗event) Q_DECL_OVERRIDE

  The main painting callback.

6.5.1 Detailed Description

The FourierSpiralWidget class.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 FT1D::FourierSpiralWidget::FourierSpiralWidget (QWidget ∗ *parent* = 0) [explicit]

FourierSpiralWidget constructor.

Parameters

parent	parent object

6.5.3 Member Function Documentation

6.5.3.1 void FT1D::FourierSpiralWidget::clearFrequency ( )

clearFrequency sets displaying enabled to false.

Repaint is called.

6.5.3.2 void FT1D::FourierSpiralWidget::displayFrequency (double *frequency,* double *magnitudeVal,* double *phaseVal,* double *maxMagnitudeVal,* int *signalLength*)

displayFrequency the main callback, sets the variables for proper display

Parameters

frequency	what frequency is to be displayed
magnitudeVal	the magnitude of the selected frequency
phaseVal	the phase of the selected frequency
signalLength	length of the signal for which is the basis function to be displayed

6.5.3.3 void FT1D::FourierSpiralWidget::newSignal (int *length*)

newSignal changes length of currently loaded signal.

Used, when no basis function is displayed, but new signal is loaded in the rest of the application and axes of this widget has to be rescaled. Calls repaint.

Parameters

length	length of the new signal.

6.5.3.4 void FT1D::FourierSpiralWidget::setMagnitudeAndPhase (double *mag,* double *pha*)

setMagnitudeAndPhase updates magnitude and phase values.

If modify is true, repaint is called.

Parameters

mag	current magnitude value
pha	current phase value

6.5.3.5 void FT1D::FourierSpiralWidget::setNormalized (bool *value*)

setNormalized sets, whether or not to scale and shift the basis function

Parameters

value	true means yes, false stands for no

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

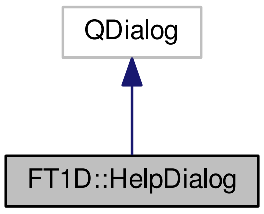
- src/fourierspiralwidget.h

6.6 FT1D::HelpDialog Class Reference

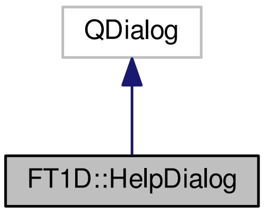
The HelpDialog class is a dialog window with information about usage.

```
#include <helpdialog.h>
```

Inheritance diagram for FT1D::HelpDialog:



Collaboration diagram for FT1D::HelpDialog:



Public Member Functions

- HelpDialog (QWidget ∗parent, const Translation ∗language)

    HelpDialog constructor.
- ∼HelpDialog ()

    Destructor.

6.6.1 Detailed Description

The HelpDialog class is a dialog window with information about usage.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 FT1D::HelpDialog::HelpDialog (QWidget ∗ *parent,* const Translation ∗ *language*) `[explicit]`

HelpDialog constructor.

Parameters

parent	parent widget
language	translation for this object

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

- src/helpdialog.h

6.7 FT1D::Localizations Class Reference

The Localizations class is responsible for managing the language versions (Translations)

`#include <localization.h>`

Public Member Functions

- Localizations ()

    Localizations constructs an empty instance.
- Localizations (const QString &directory)

    Localizations reads the directory and fills the attributes based on files found in that directory.
- QList< QString > getAvailableLanguages () const

    getAvailableLanguages returns list of all available language versions
- bool setLanguage (const QString &language)

    setLanguage sets the language language as the current
- bool initFromDirectory (const QString &directory)

    initFromDirectory implements reading the directory and setting up the attributes based on what's read
- Translation ∗ getCurrentLanguage () const

    getCurrentLanguage returns poitner to the selected language

6.7.1 Detailed Description

The Localizations class is responsible for managing the language versions (Translations)

6.7.2 Constructor & Destructor Documentation

6.7.2.1 FT1D::Localizations::Localizations (const QString & *directory*)

Localizations reads the *directory* and fills the attributes based on files found in that directory.

Parameters

directory	a path to the directory containing the translations

6.7.3 Member Function Documentation

6.7.3.1 QList<QString> FT1D::Localizations::getAvailableLanguages ( ) const

getAvailableLanguages returns list of all available language versions

Returns

list of all available language versions

6.7.3.2 Translation* FT1D::Localizations::getCurrentLanguage ( ) const

getCurrentLanguage returns poitner to the selected language

Returns

pointer to the selected language

6.7.3.3 bool FT1D::Localizations::initFromDirectory (const QString & *directory*)

initFromDirectory implements reading the directory and setting up the attributes based on what's read

Parameters

directory	a path to the directory to process

Returns

bool in case of success, false in case of failure

6.7.3.4 bool FT1D::Localizations::setLanguage (const QString & *language*)

setLanguage sets the language *language* as the current

Parameters

language	name of the language to set

Returns

true in case of success, false if no language with name *language* exists

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

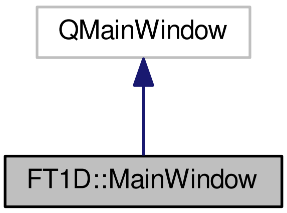
- src/localization.h

6.8 FT1D::MainWindow Class Reference

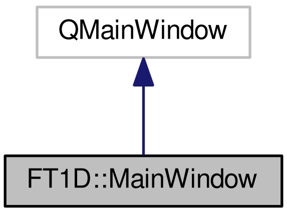
The MainWindow class the main window of the application and the most of the app logic.

```
#include <mainwindow.h>
```

Inheritance diagram for FT1D::MainWindow:



Collaboration diagram for FT1D::MainWindow:



Public Slots

- void showAboutDialog ()

  showAboutDialog action that shows about application dialog
- void showHelpDialog ()

  showHelpDialog action that shows application help dialog
- void openPredefinedSignalsDialog ()

  openPredefinedSignalsDialog
- void recordCurrentState ()

  recordCurrentState

Public Member Functions

- MainWindow (QWidget ∗parent=0)

  MainWindow constructor.
- ∼MainWindow ()

  Destructor.

6.8.1 Detailed Description

The MainWindow class the main window of the application and the most of the app logic.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 FT1D::MainWindow::MainWindow (QWidget ∗ *parent* = 0) [explicit]

MainWindow constructor.

Parameters

parent	parent object, can be NULL

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

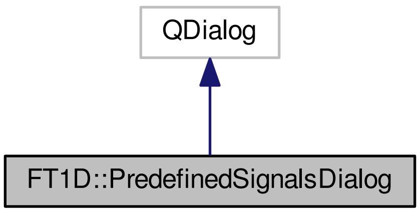
- src/mainwindow.h

6.9 FT1D::PredefinedSignalsDialog Class Reference

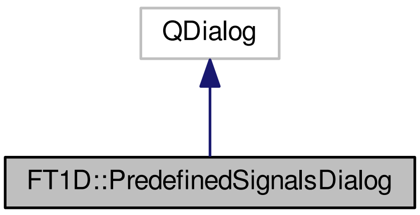
The PredefinedSignalsDialog class is a Dialog in which the user can choose to load one of 8 predefined signals.

```
#include <predefinedsignalsdialog.h>
```

Inheritance diagram for FT1D::PredefinedSignalsDialog:



Collaboration diagram for FT1D::PredefinedSignalsDialog:



Signals

- void signalChosen (QString resourcePath)

  signalChosen a signal to notify that a signal was selected

Public Member Functions

- PredefinedSignalsDialog (QWidget ∗parent, QString signalsFolder, const Translation ∗translation)

  PredefinedSignalsDialog constructor.
- virtual ∼PredefinedSignalsDialog ()

  ∼PredefinedSignalsDialog desctructor

6.9.1 Detailed Description

The PredefinedSignalsDialog class is a Dialog in which the user can choose to load one of 8 predefined signals.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 FT1D::PredefinedSignalsDialog::PredefinedSignalsDialog (QWidget ∗ *parent,* QString *signalsFolder,* const Translation ∗ *translation*)

PredefinedSignalsDialog constructor.

Parameters

parent	parent object
signalsFolder	a resource folder, in which to look for files to load and images to display
translation	provides localized names for this dialog'

6.9.3 Member Function Documentation

6.9.3.1 void FT1D::PredefinedSignalsDialog::signalChosen (QString *resourcePath*) `[signal]`

signalChosen a signal to notify that a signal was selected

Parameters

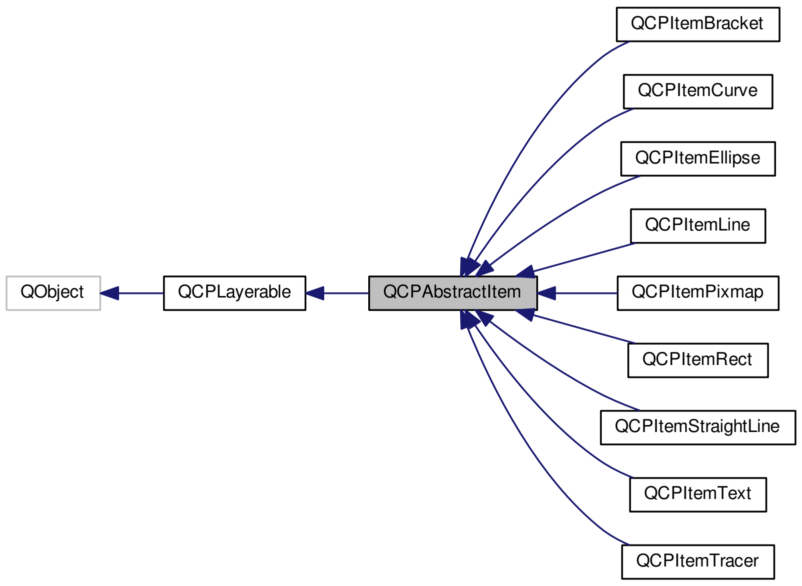
resourcePath	a path to the file which should be loaded

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

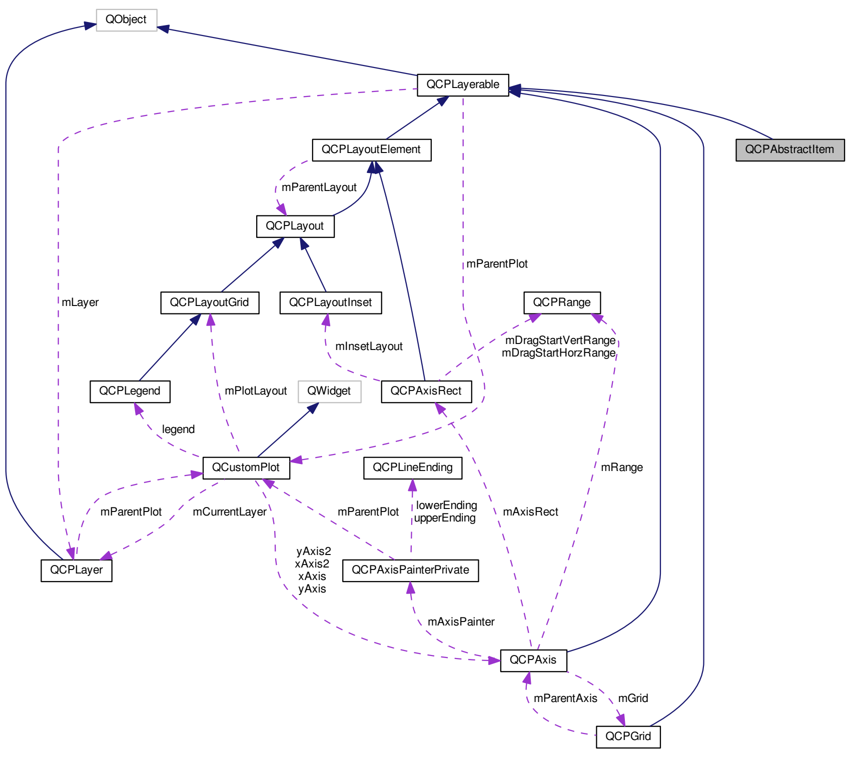
- src/predefinedsignalsdialog.h

6.10 QCPAbstractItem Class Reference

Inheritance diagram for QCPAbstractItem:



Collaboration diagram for QCPAbstractItem:



Signals

- void **selectionChanged** (bool selected)
- void **selectableChanged** (bool selectable)

Public Member Functions

- **QCPAbstractItem** (QCustomPlot ∗parentPlot)
- bool **clipToAxisRect** () const
- QCPAxisRect ∗ **clipAxisRect** () const
- bool **selectable** () const
- bool **selected** () const
- void **setClipToAxisRect** (bool clip)
- void **setClipAxisRect** (QCPAxisRect ∗rect)
- Q_SLOT void **setSelectable** (bool selectable)
- Q_SLOT void **setSelected** (bool selected)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const =0
- QList< QCPItemPosition ∗ > **positions** () const
- QList< QCPItemAnchor ∗ > **anchors** () const
- QCPItemPosition ∗ **position** (const QString &name) const
- QCPItemAnchor ∗ **anchor** (const QString &name) const
- bool **hasAnchor** (const QString &name) const

Protected Member Functions

- virtual QCP::Interaction **selectionCategory** () const
- virtual QRect **clipRect** () const
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)=0
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- double **distSqrToLine** (const QPointF &start, const QPointF &end, const QPointF &point) const
- double **rectSelectTest** (const QRectF &rect, const QPointF &pos, bool filledRect) const
- QCPItemPosition ∗ **createPosition** (const QString &name)
- QCPItemAnchor ∗ **createAnchor** (const QString &name, int anchorId)

Protected Attributes

- bool **mClipToAxisRect**
- QPointer< QCPAxisRect > **mClipAxisRect**
- QList< QCPItemPosition ∗ > **mPositions**
- QList< QCPItemAnchor ∗ > **mAnchors**
- bool **mSelectable**
- bool **mSelected**

Friends

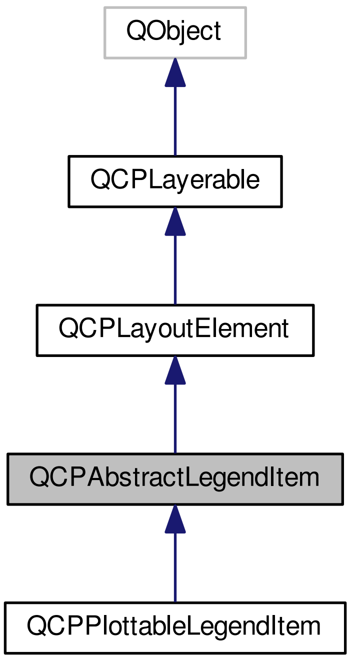
- class **QCustomPlot**
- class **QCPItemAnchor**

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

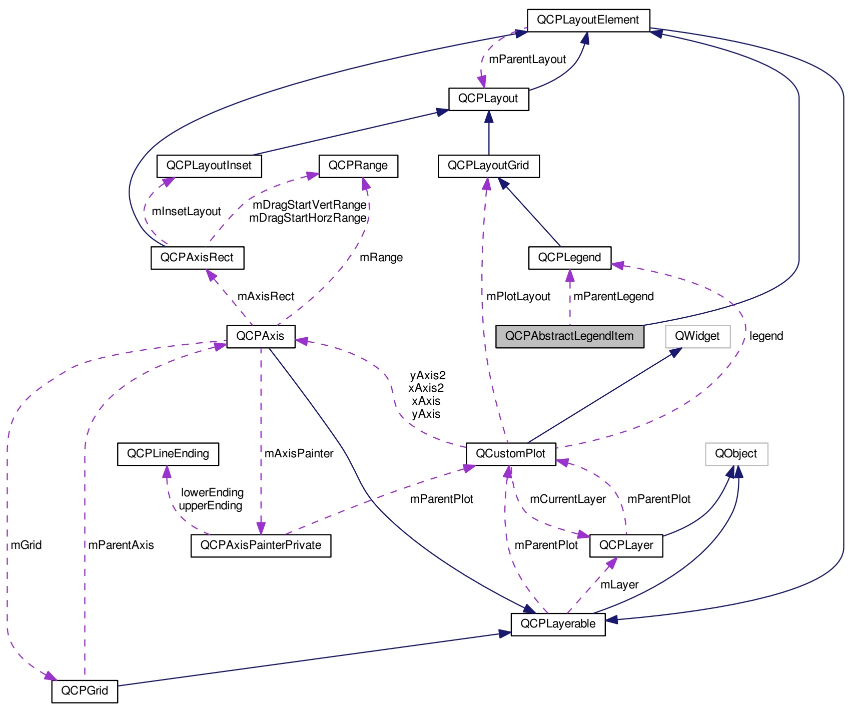
- src/qcustomplot/qcustomplot.h

6.11 QCPAbstractLegendItem Class Reference

Inheritance diagram for QCPAbstractLegendItem:



Collaboration diagram for QCPAbstractLegendItem:



Signals

- void **selectionChanged** (bool selected)
- void **selectableChanged** (bool selectable)

Public Member Functions

- **QCPAbstractLegendItem** (QCPLegend ∗parent)
- QCPLegend ∗ **parentLegend** () const
- QFont **font** () const
- QColor **textColor** () const
- QFont **selectedFont** () const
- QColor **selectedTextColor** () const
- bool **selectable** () const
- bool **selected** () const
- void **setFont** (const QFont &font)
- void **setTextColor** (const QColor &color)
- void **setSelectedFont** (const QFont &font)
- void **setSelectedTextColor** (const QColor &color)
- Q_SLOT void **setSelectable** (bool selectable)
- Q_SLOT void **setSelected** (bool selected)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual QCP::Interaction **selectionCategory** () const
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual QRect **clipRect** () const
- virtual void **draw** (QCPPainter ∗painter)=0
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←↩ Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)

Protected Attributes

- QCPLegend ∗ **mParentLegend**
- QFont **mFont**
- QColor **mTextColor**
- QFont **mSelectedFont**
- QColor **mSelectedTextColor**
- bool **mSelectable**
- bool **mSelected**

Friends

- class **QCPLegend**

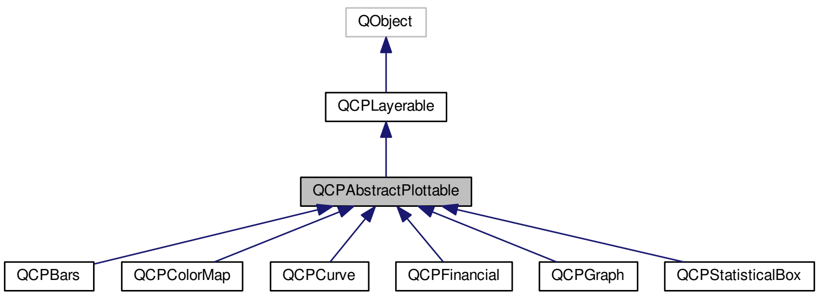
Additional Inherited Members

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

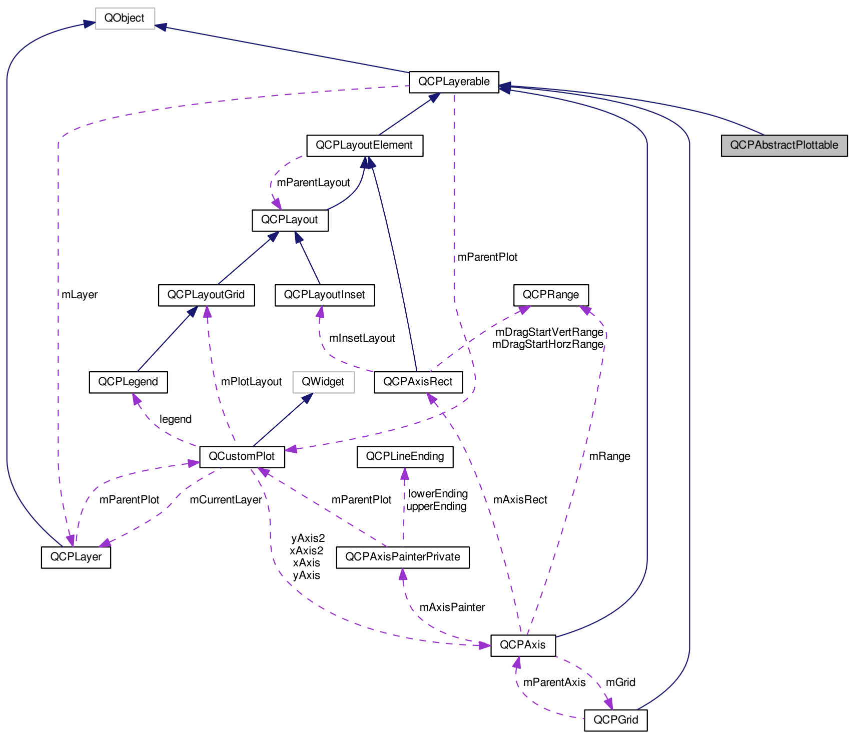
- src/qcustomplot/qcustomplot.h

6.12 QCPAbstractPlottable Class Reference

Inheritance diagram for QCPAbstractPlottable:



Collaboration diagram for QCPAbstractPlottable:



Signals

- void **selectionChanged** (bool selected)
- void **selectableChanged** (bool selectable)

Public Member Functions

- **QCPAbstractPlottable** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- QString **name** () const
- bool **antialiasedFill** () const
- bool **antialiasedScatters** () const
- bool **antialiasedErrorBars** () const
- QPen **pen** () const
- QPen **selectedPen** () const
- QBrush **brush** () const
- QBrush **selectedBrush** () const
- QCPAxis ∗ **keyAxis** () const
- QCPAxis ∗ **valueAxis** () const
- bool **selectable** () const
- bool **selected** () const
- void **setName** (const QString &name)
- void **setAntialiasedFill** (bool enabled)
- void **setAntialiasedScatters** (bool enabled)
- void **setAntialiasedErrorBars** (bool enabled)
- void **setPen** (const QPen &pen)

- void **setSelectedPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setSelectedBrush** (const QBrush &brush)
- void **setKeyAxis** (QCPAxis ∗axis)
- void **setValueAxis** (QCPAxis ∗axis)
- Q_SLOT void **setSelectable** (bool selectable)
- Q_SLOT void **setSelected** (bool selected)
- virtual void **clearData** ()=0
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const =0
- virtual bool **addToLegend** ()
- virtual bool **removeFromLegend** () const
- void **rescaleAxes** (bool onlyEnlarge=false) const
- void **rescaleKeyAxis** (bool onlyEnlarge=false) const
- void **rescaleValueAxis** (bool onlyEnlarge=false) const

Protected Types

- enum SignDomain { sdNegative, sdBoth, sdPositive }

Protected Member Functions

- virtual QRect **clipRect** () const
- virtual void **draw** (QCPPainter ∗painter)=0
- virtual QCP::Interaction **selectionCategory** () const
- void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const =0
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const =0
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const =0
- void **coordsToPixels** (double key, double value, double &x, double &y) const
- const QPointF **coordsToPixels** (double key, double value) const
- void **pixelsToCoords** (double x, double y, double &key, double &value) const
- void **pixelsToCoords** (const QPointF &pixelPos, double &key, double &value) const
- QPen **mainPen** () const
- QBrush **mainBrush** () const
- void **applyFillAntialiasingHint** (QCPPainter ∗painter) const
- void **applyScattersAntialiasingHint** (QCPPainter ∗painter) const
- void **applyErrorBarsAntialiasingHint** (QCPPainter ∗painter) const
- double **distSqrToLine** (const QPointF &start, const QPointF &end, const QPointF &point) const

Protected Attributes

- QString **mName**
- bool **mAntialiasedFill**
- bool **mAntialiasedScatters**
- bool **mAntialiasedErrorBars**
- QPen **mPen**
- QPen **mSelectedPen**
- QBrush **mBrush**
- QBrush **mSelectedBrush**
- QPointer< QCPAxis > **mKeyAxis**
- QPointer< QCPAxis > **mValueAxis**
- bool **mSelectable**
- bool **mSelected**

Friends

- class **QCustomPlot**
- class **QCPAxis**
- class **QCPPlottableLegendItem**

6.12.1 Member Enumeration Documentation

6.12.1.1 enum QCPAbstractPlottable::SignDomain `[protected]`

Represents negative and positive sign domain for passing to getKeyRange and getValueRange.

Enumerator

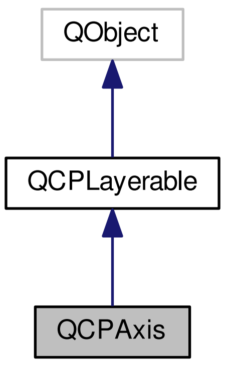
sdNegative	The negative sign domain, i.e. numbers smaller than zero.
sdBoth	Both sign domains, including zero, i.e. all (rational) numbers.
sdPositive	The positive sign domain, i.e. numbers greater than zero.

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

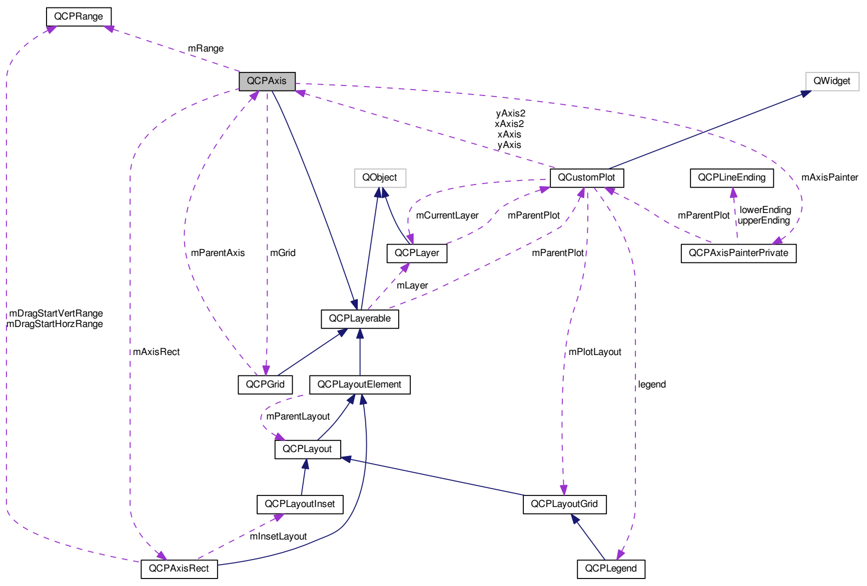
- src/qcustomplot/qcustomplot.h

6.13 QCPAxis Class Reference

Inheritance diagram for QCPAxis:



Collaboration diagram for QCPAxis:



Public Types

- enum AxisType { atLeft = 0x01, atRight = 0x02, atTop = 0x04, atBottom = 0x08 }
- enum LabelType { ltNumber, ltDateTime }
- enum LabelSide { lsInside, lsOutside }
- enum ScaleType { stLinear, stLogarithmic }
- enum SelectablePart { spNone = 0, spAxis = 0x001, spTickLabels = 0x002, spAxisLabel = 0x004 }

Signals

- void **ticksRequest** ()
- void **rangeChanged** (const QCPRange &newRange)
- void **rangeChanged** (const QCPRange &newRange, const QCPRange &oldRange)
- void **scaleTypeChanged** (QCPAxis::ScaleType scaleType)
- void **selectionChanged** (const QCPAxis::SelectableParts &parts)
- void **selectableChanged** (const QCPAxis::SelectableParts &parts)

Public Member Functions

- **QCPAxis** (QCPAxisRect ∗parent, AxisType type)
- AxisType **axisType** () const
- QCPAxisRect ∗ **axisRect** () const
- ScaleType **scaleType** () const
- double **scaleLogBase** () const
- const QCPRange **range** () const
- bool **rangeReversed** () const
- bool **autoTicks** () const
- int **autoTickCount** () const
- bool **autoTickLabels** () const

- bool **autoTickStep** () const
- bool **autoSubTicks** () const
- bool **ticks** () const
- bool **tickLabels** () const
- int **tickLabelPadding** () const
- LabelType **tickLabelType** () const
- QFont **tickLabelFont** () const
- QColor **tickLabelColor** () const
- double **tickLabelRotation** () const
- LabelSide **tickLabelSide** () const
- QString **dateTimeFormat** () const
- Qt::TimeSpec **dateTimeSpec** () const
- QString **numberFormat** () const
- int **numberPrecision** () const
- double **tickStep** () const
- QVector< double > **tickVector** () const
- QVector< QString > **tickVectorLabels** () const
- int **tickLengthIn** () const
- int **tickLengthOut** () const
- int **subTickCount** () const
- int **subTickLengthIn** () const
- int **subTickLengthOut** () const
- QPen **basePen** () const
- QPen **tickPen** () const
- QPen **subTickPen** () const
- QFont **labelFont** () const
- QColor **labelColor** () const
- QString **label** () const
- int **labelPadding** () const
- int **padding** () const
- int **offset** () const
- SelectableParts **selectedParts** () const
- SelectableParts **selectableParts** () const
- QFont **selectedTickLabelFont** () const
- QFont **selectedLabelFont** () const
- QColor **selectedTickLabelColor** () const
- QColor **selectedLabelColor** () const
- QPen **selectedBasePen** () const
- QPen **selectedTickPen** () const
- QPen **selectedSubTickPen** () const
- QCPLineEnding **lowerEnding** () const
- QCPLineEnding **upperEnding** () const
- QCPGrid ∗ **grid** () const
- Q_SLOT void **setScaleType** (QCPAxis::ScaleType type)
- void **setScaleLogBase** (double base)
- Q_SLOT void **setRange** (const QCPRange &range)
- void **setRange** (double lower, double upper)
- void **setRange** (double position, double size, Qt::AlignmentFlag alignment)
- void **setRangeLower** (double lower)
- void **setRangeUpper** (double upper)
- void **setRangeReversed** (bool reversed)
- void **setAutoTicks** (bool on)
- void **setAutoTickCount** (int approximateCount)
- void **setAutoTickLabels** (bool on)
- void **setAutoTickStep** (bool on)

- void **setAutoSubTicks** (bool on)
- void **setTicks** (bool show)
- void **setTickLabels** (bool show)
- void **setTickLabelPadding** (int padding)
- void **setTickLabelType** (LabelType type)
- void **setTickLabelFont** (const QFont &font)
- void **setTickLabelColor** (const QColor &color)
- void **setTickLabelRotation** (double degrees)
- void **setTickLabelSide** (LabelSide side)
- void **setDateTimeFormat** (const QString &format)
- void **setDateTimeSpec** (const Qt::TimeSpec &timeSpec)
- void **setNumberFormat** (const QString &formatCode)
- void **setNumberPrecision** (int precision)
- void **setTickStep** (double step)
- void **setTickVector** (const QVector< double > &vec)
- void **setTickVectorLabels** (const QVector< QString > &vec)
- void **setTickLength** (int inside, int outside=0)
- void **setTickLengthIn** (int inside)
- void **setTickLengthOut** (int outside)
- void **setSubTickCount** (int count)
- void **setSubTickLength** (int inside, int outside=0)
- void **setSubTickLengthIn** (int inside)
- void **setSubTickLengthOut** (int outside)
- void **setBasePen** (const QPen &pen)
- void **setTickPen** (const QPen &pen)
- void **setSubTickPen** (const QPen &pen)
- void **setLabelFont** (const QFont &font)
- void **setLabelColor** (const QColor &color)
- void **setLabel** (const QString &str)
- void **setLabelPadding** (int padding)
- void **setPadding** (int padding)
- void **setOffset** (int offset)
- void **setSelectedTickLabelFont** (const QFont &font)
- void **setSelectedLabelFont** (const QFont &font)
- void **setSelectedTickLabelColor** (const QColor &color)
- void **setSelectedLabelColor** (const QColor &color)
- void **setSelectedBasePen** (const QPen &pen)
- void **setSelectedTickPen** (const QPen &pen)
- void **setSelectedSubTickPen** (const QPen &pen)
- Q_SLOT void **setSelectableParts** (const QCPAxis::SelectableParts &selectableParts)
- Q_SLOT void **setSelectedParts** (const QCPAxis::SelectableParts &selectedParts)
- void **setLowerEnding** (const QCPLineEnding &ending)
- void **setUpperEnding** (const QCPLineEnding &ending)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- Qt::Orientation **orientation** () const
- void **moveRange** (double diff)
- void **scaleRange** (double factor, double center)
- void **setScaleRatio** (const QCPAxis ∗otherAxis, double ratio=1.0)
- void **rescale** (bool onlyVisiblePlottables=false)
- double **pixelToCoord** (double value) const
- double **coordToPixel** (double value) const
- SelectablePart **getPartAt** (const QPointF &pos) const
- QList< QCPAbstractPlottable ∗ > **plottables** () const
- QList< QCPGraph ∗ > **graphs** () const
- QList< QCPAbstractItem ∗ > **items** () const

Static Public Member Functions

- static AxisType **marginSideToAxisType** (QCP::MarginSide side)
- static Qt::Orientation **orientation** (AxisType type)
- static AxisType **opposite** (AxisType type)

Protected Member Functions

- virtual void **setupTickVectors** ()
- virtual void **generateAutoTicks** ()
- virtual int **calculateAutoSubTickCount** (double tickStep) const
- virtual int **calculateMargin** ()
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- virtual QCP::Interaction **selectionCategory** () const
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState↩ Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- void **visibleTickBounds** (int &lowIndex, int &highIndex) const
- double **baseLog** (double value) const
- double **basePow** (double value) const
- QPen **getBasePen** () const
- QPen **getTickPen** () const
- QPen **getSubTickPen** () const
- QFont **getTickLabelFont** () const
- QFont **getLabelFont** () const
- QColor **getTickLabelColor** () const
- QColor **getLabelColor** () const

Protected Attributes

- AxisType **mAxisType**
- QCPAxisRect ∗ **mAxisRect**
- int **mPadding**
- Qt::Orientation **mOrientation**
- SelectableParts **mSelectableParts**
- SelectableParts **mSelectedParts**
- QPen **mBasePen**
- QPen **mSelectedBasePen**
- QString **mLabel**
- QFont **mLabelFont**
- QFont **mSelectedLabelFont**
- QColor **mLabelColor**
- QColor **mSelectedLabelColor**
- bool **mTickLabels**
- bool **mAutoTickLabels**
- LabelType **mTickLabelType**
- QFont **mTickLabelFont**
- QFont **mSelectedTickLabelFont**
- QColor **mTickLabelColor**
- QColor **mSelectedTickLabelColor**
- QString **mDateTimeFormat**
- Qt::TimeSpec **mDateTimeSpec**

- int **mNumberPrecision**
- QLatin1Char **mNumberFormatChar**
- bool **mNumberBeautifulPowers**
- bool **mTicks**
- double **mTickStep**
- int **mSubTickCount**
- int **mAutoTickCount**
- bool **mAutoTicks**
- bool **mAutoTickStep**
- bool **mAutoSubTicks**
- QPen **mTickPen**
- QPen **mSelectedTickPen**
- QPen **mSubTickPen**
- QPen **mSelectedSubTickPen**
- QCPRange **mRange**
- bool **mRangeReversed**
- ScaleType **mScaleType**
- double **mScaleLogBase**
- double **mScaleLogBaseLogInv**
- QCPGrid ∗ **mGrid**
- QCPAxisPainterPrivate ∗ **mAxisPainter**
- int **mLowestVisibleTick**
- int **mHighestVisibleTick**
- QVector< double > **mTickVector**
- QVector< QString > **mTickVectorLabels**
- QVector< double > **mSubTickVector**
- bool **mCachedMarginValid**
- int **mCachedMargin**

Friends

- class **QCustomPlot**
- class **QCPGrid**
- class **QCPAxisRect**

6.13.1 Member Enumeration Documentation

6.13.1.1 enum QCPAxis::AxisType

Defines at which side of the axis rect the axis will appear. This also affects how the tick marks are drawn, on which side the labels are placed etc.

Enumerator

atLeft `0x01` Axis is vertical and on the left side of the axis rect

atRight `0x02` Axis is vertical and on the right side of the axis rect

atTop `0x04` Axis is horizontal and on the top side of the axis rect

atBottom `0x08` Axis is horizontal and on the bottom side of the axis rect

6.13.1.2 enum QCPAxis::LabelSide

Defines on which side of the axis the tick labels (numbers) shall appear.

See also

setTickLabelSide

Enumerator

lsInside Tick labels will be displayed inside the axis rect and clipped to the inner axis rect.

lsOutside Tick labels will be displayed outside the axis rect.

6.13.1.3 enum QCPAxis::LabelType

When automatic tick label generation is enabled (setAutoTickLabels), defines how the coordinate of the tick is interpreted, i.e. translated into a string.

See also

setTickLabelType

Enumerator

ltNumber Tick coordinate is regarded as normal number and will be displayed as such. (see setNumber↩ Format)

ltDateTime Tick coordinate is regarded as a date/time (seconds since 1970-01-01T00:00:00 UTC) and will be displayed and formatted as such. (for details, see setDateTimeFormat)

6.13.1.4 enum QCPAxis::ScaleType

Defines the scale of an axis.

See also

setScaleType

Enumerator

stLinear Linear scaling.

stLogarithmic Logarithmic scaling with correspondingly transformed plots and (major) tick marks at every base power (see setScaleLogBase).

6.13.1.5 enum QCPAxis::SelectablePart

Defines the selectable parts of an axis.

See also

setSelectableParts, setSelectedParts

Enumerator

spNone None of the selectable parts.

spAxis The axis backbone and tick marks.

spTickLabels Tick labels (numbers) of this axis (as a whole, not individually)

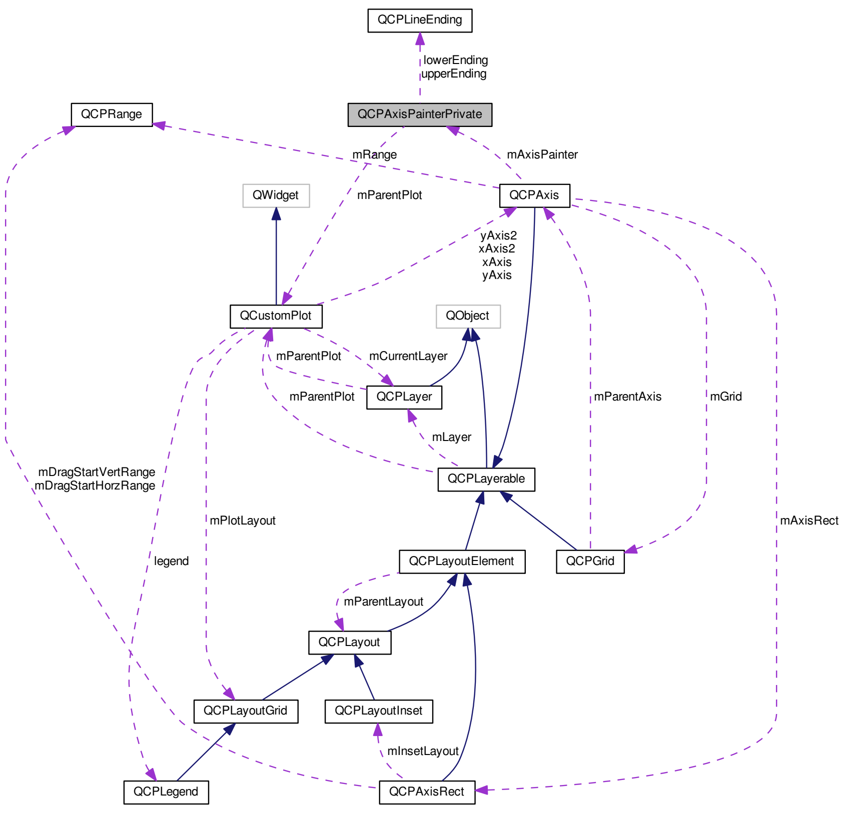
spAxisLabel The axis label.

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

- src/qcustomplot/qcustomplot.h

6.14 QCPAxisPainterPrivate Class Reference

Collaboration diagram for QCPAxisPainterPrivate:



Classes

- struct CachedLabel
- struct TickLabelData

Public Member Functions

- **QCPAxisPainterPrivate** (QCustomPlot ∗parentPlot)
- virtual void **draw** (QCPPainter ∗painter)
- virtual int **size** () const
- void **clearCache** ()
- QRect **axisSelectionBox** () const
- QRect **tickLabelsSelectionBox** () const
- QRect **labelSelectionBox** () const

Public Attributes

- QCPAxis::AxisType **type**
- QPen **basePen**
- QCPLineEnding **lowerEnding**
- QCPLineEnding **upperEnding**
- int **labelPadding**
- QFont **labelFont**
- QColor **labelColor**
- QString **label**
- int **tickLabelPadding**
- double **tickLabelRotation**
- QCPAxis::LabelSide **tickLabelSide**
- bool **substituteExponent**
- bool **numberMultiplyCross**
- int **tickLengthIn**
- int **tickLengthOut**
- int **subTickLengthIn**
- int **subTickLengthOut**
- QPen **tickPen**
- QPen **subTickPen**
- QFont **tickLabelFont**
- QColor **tickLabelColor**
- QRect **axisRect**
- QRect **viewportRect**
- double **offset**
- bool **abbreviateDecimalPowers**
- bool **reversedEndings**
- QVector< double > **subTickPositions**
- QVector< double > **tickPositions**
- QVector< QString > **tickLabels**

Protected Member Functions

- virtual QByteArray **generateLabelParameterHash** () const
- virtual void **placeTickLabel** (QCPPainter ∗painter, double position, int distanceToAxis, const QString &text, QSize ∗tickLabelsSize)
- virtual void **drawTickLabel** (QCPPainter ∗painter, double x, double y, const TickLabelData &labelData) const
- virtual TickLabelData **getTickLabelData** (const QFont &font, const QString &text) const
- virtual QPointF **getTickLabelDrawOffset** (const TickLabelData &labelData) const
- virtual void **getMaxTickLabelSize** (const QFont &font, const QString &text, QSize ∗tickLabelsSize) const

Protected Attributes

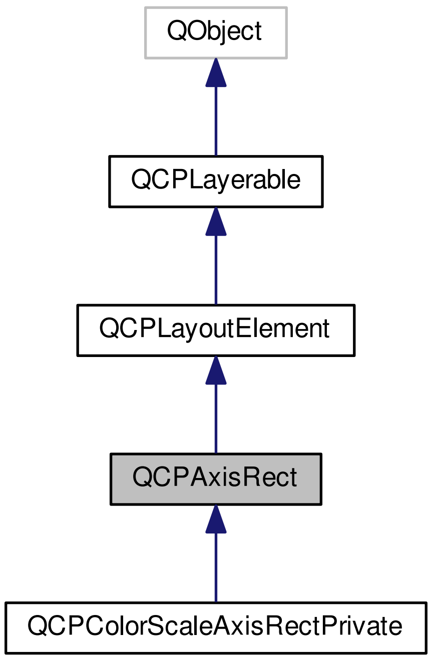
- QCustomPlot ∗ **mParentPlot**
- QByteArray **mLabelParameterHash**
- QCache< QString, CachedLabel > **mLabelCache**
- QRect **mAxisSelectionBox**
- QRect **mTickLabelsSelectionBox**
- QRect **mLabelSelectionBox**

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

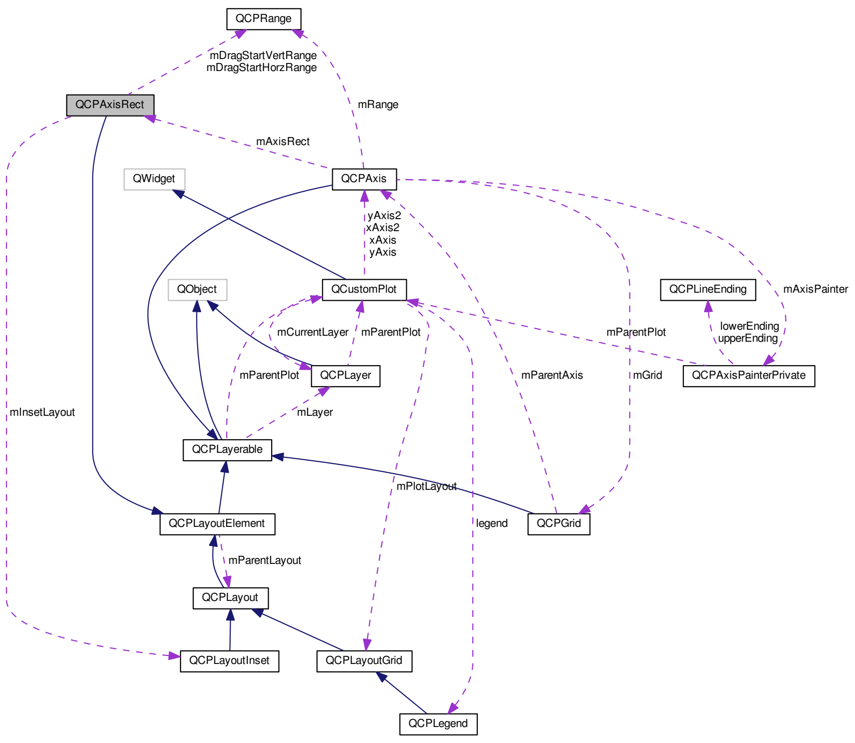
- src/qcustomplot/qcustomplot.h

6.15 QCPAxisRect Class Reference

Inheritance diagram for QCPAxisRect:



Collaboration diagram for QCPAxisRect:



Public Member Functions

- **QCPAxisRect** (QCustomPlot ∗parentPlot, bool setupDefaultAxes=true)
- QPixmap **background** () const
- bool **backgroundScaled** () const
- Qt::AspectRatioMode **backgroundScaledMode** () const
- Qt::Orientations **rangeDrag** () const
- Qt::Orientations **rangeZoom** () const
- QCPAxis ∗ **rangeDragAxis** (Qt::Orientation orientation)
- QCPAxis ∗ **rangeZoomAxis** (Qt::Orientation orientation)
- double **rangeZoomFactor** (Qt::Orientation orientation)
- void **setBackground** (const QPixmap &pm)
- void **setBackground** (const QPixmap &pm, bool scaled, Qt::AspectRatioMode mode=Qt::KeepAspect↩ RatioByExpanding)
- void **setBackground** (const QBrush &brush)
- void **setBackgroundScaled** (bool scaled)
- void **setBackgroundScaledMode** (Qt::AspectRatioMode mode)
- void **setRangeDrag** (Qt::Orientations orientations)
- void **setRangeZoom** (Qt::Orientations orientations)
- void **setRangeDragAxes** (QCPAxis ∗horizontal, QCPAxis ∗vertical)
- void **setRangeZoomAxes** (QCPAxis ∗horizontal, QCPAxis ∗vertical)
- void **setRangeZoomFactor** (double horizontalFactor, double verticalFactor)
- void **setRangeZoomFactor** (double factor)
- int **axisCount** (QCPAxis::AxisType type) const
- QCPAxis ∗ **axis** (QCPAxis::AxisType type, int index=0) const
- QList< QCPAxis ∗ > **axes** (QCPAxis::AxisTypes types) const

- QList< QCPAxis ∗ > **axes** () const
- QCPAxis ∗ **addAxis** (QCPAxis::AxisType type, QCPAxis ∗axis=0)
- QList< QCPAxis ∗ > **addAxes** (QCPAxis::AxisTypes types)
- bool **removeAxis** (QCPAxis ∗axis)
- QCPLayoutInset ∗ **insetLayout** () const
- void **setupFullAxesBox** (bool connectRanges=false)
- QList< QCPAbstractPlottable ∗ > **plottables** () const
- QList< QCPGraph ∗ > **graphs** () const
- QList< QCPAbstractItem ∗ > **items** () const
- int **left** () const
- int **right** () const
- int **top** () const
- int **bottom** () const
- int **width** () const
- int **height** () const
- QSize **size** () const
- QPoint **topLeft** () const
- QPoint **topRight** () const
- QPoint **bottomLeft** () const
- QPoint **bottomRight** () const
- QPoint **center** () const
- virtual void **update** (UpdatePhase phase)
- virtual QList< QCPLayoutElement ∗ > **elements** (bool recursive) const
- virtual void **mousePressEvent** (QMouseEvent ∗event)

Protected Member Functions

- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- virtual int **calculateAutoMargin** (QCP::MarginSide side)
- virtual void **mouseMoveEvent** (QMouseEvent ∗event)
- virtual void **mouseReleaseEvent** (QMouseEvent ∗event)
- virtual void **wheelEvent** (QWheelEvent ∗event)
- void **drawBackground** (QCPPainter ∗painter)
- void **updateAxesOffset** (QCPAxis::AxisType type)

Protected Attributes

- QBrush **mBackgroundBrush**
- QPixmap **mBackgroundPixmap**
- QPixmap **mScaledBackgroundPixmap**
- bool **mBackgroundScaled**
- Qt::AspectRatioMode **mBackgroundScaledMode**
- QCPLayoutInset ∗ **mInsetLayout**
- Qt::Orientations **mRangeDrag**
- Qt::Orientations **mRangeZoom**
- QPointer< QCPAxis > **mRangeDragHorzAxis**
- QPointer< QCPAxis > **mRangeDragVertAxis**
- QPointer< QCPAxis > **mRangeZoomHorzAxis**
- QPointer< QCPAxis > **mRangeZoomVertAxis**
- double **mRangeZoomFactorHorz**
- double **mRangeZoomFactorVert**
- QCPRange **mDragStartHorzRange**

- QCPRange **mDragStartVertRange**
- QCP::AntialiasedElements **mAADragBackup**
- QCP::AntialiasedElements **mNotAADragBackup**
- QPoint **mDragStart**
- bool **mDragging**
- QHash< QCPAxis::AxisType, QList< QCPAxis $*$ > > **mAxes**

Friends

- class **QCustomPlot**
- class **FT1D::DisplaySignalWidget**

Additional Inherited Members

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

- src/qcustomplot/qcustomplot.h

6.16 QCPBarData Class Reference

Public Member Functions

- **QCPBarData** (double key, double value)

Public Attributes

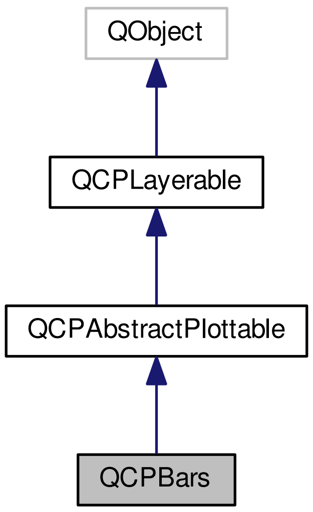
- double **key**
- double **value**

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

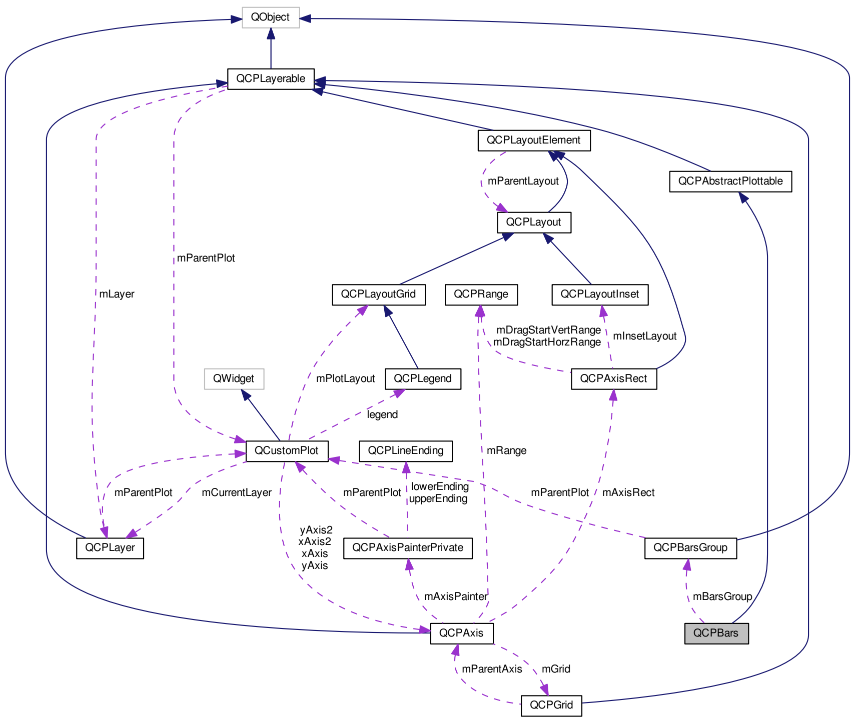
- src/qcustomplot/qcustomplot.h

6.17 QCPBars Class Reference

Inheritance diagram for QCPBars:



Collaboration diagram for QCPBars:



Public Types

- enum WidthType { wtAbsolute, wtAxisRectRatio, wtPlotCoords }

Public Member Functions

- **QCPBars** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- double **width** () const
- WidthType **widthType** () const
- QCPBarsGroup ∗ **barsGroup** () const
- double **baseValue** () const
- QCPBars ∗ **barBelow** () const
- QCPBars ∗ **barAbove** () const
- QCPBarDataMap ∗ **data** () const
- void **setWidth** (double width)
- void **setWidthType** (WidthType widthType)
- void **setBarsGroup** (QCPBarsGroup ∗barsGroup)
- void **setBaseValue** (double baseValue)
- void **setData** (QCPBarDataMap ∗data, bool copy=false)
- void **setData** (const QVector< double > &key, const QVector< double > &value)
- void **moveBelow** (QCPBars ∗bars)
- void **moveAbove** (QCPBars ∗bars)
- void **addData** (const QCPBarDataMap &dataMap)
- void **addData** (const QCPBarData &data)
- void **addData** (double key, double value)
- void **addData** (const QVector< double > &keys, const QVector< double > &values)
- void **removeDataBefore** (double key)
- void **removeDataAfter** (double key)
- void **removeData** (double fromKey, double toKey)
- void **removeData** (double key)
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- void **getVisibleDataBounds** (QCPBarDataMap::const_iterator &lower, QCPBarDataMap::const_iterator &upperEnd) const
- QPolygonF **getBarPolygon** (double key, double value) const
- void **getPixelWidth** (double key, double &lower, double &upper) const
- double **getStackedBaseValue** (double key, bool positive) const

Static Protected Member Functions

- static void **connectBars** (QCPBars ∗lower, QCPBars ∗upper)

Protected Attributes

- QCPBarDataMap ∗ **mData**
- double **mWidth**
- WidthType **mWidthType**
- QCPBarsGroup ∗ **mBarsGroup**
- double **mBaseValue**
- QPointer< QCPBars > **mBarBelow**
- QPointer< QCPBars > **mBarAbove**

Friends

- class **QCustomPlot**
- class **QCPLegend**
- class **QCPBarsGroup**

Additional Inherited Members

6.17.1 Member Enumeration Documentation

6.17.1.1 enum QCPBars::WidthType

Defines the ways the width of the bar can be specified. Thus it defines what the number passed to setWidth actually means.

See also

setWidthType, setWidth

Enumerator

wtAbsolute Bar width is in absolute pixels.

wtAxisRectRatio Bar width is given by a fraction of the axis rect size.

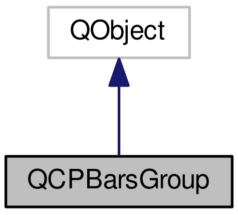
wtPlotCoords Bar width is in key coordinates and thus scales with the key axis range.

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

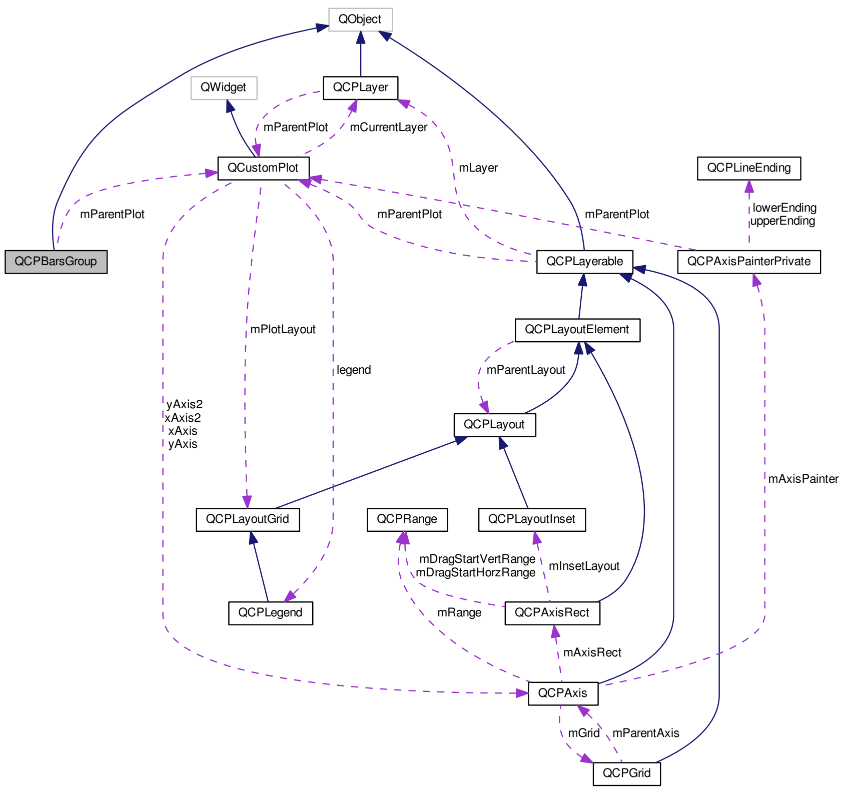
- src/qcustomplot/qcustomplot.h

6.18 QCPBarsGroup Class Reference

Inheritance diagram for QCPBarsGroup:



Collaboration diagram for QCPBarsGroup:



Public Types

- enum SpacingType { stAbsolute, stAxisRectRatio, stPlotCoords }

Public Member Functions

- **QCPBarsGroup** (QCustomPlot ∗parentPlot)
- SpacingType **spacingType** () const
- double **spacing** () const
- void **setSpacingType** (SpacingType spacingType)
- void **setSpacing** (double spacing)
- QList< QCPBars ∗ > **bars** () const
- QCPBars ∗ **bars** (int index) const
- int **size** () const
- bool **isEmpty** () const
- void **clear** ()
- bool **contains** (QCPBars ∗bars) const
- void **append** (QCPBars ∗bars)
- void **insert** (int i, QCPBars ∗bars)
- void **remove** (QCPBars ∗bars)

Protected Member Functions

- void **registerBars** (QCPBars ∗bars)
- void **unregisterBars** (QCPBars ∗bars)
- double **keyPixelOffset** (const QCPBars ∗bars, double keyCoord)
- double **getPixelSpacing** (const QCPBars ∗bars, double keyCoord)

Protected Attributes

- QCustomPlot ∗ **mParentPlot**
- SpacingType **mSpacingType**
- double **mSpacing**
- QList< QCPBars ∗ > **mBars**

Friends

- class **QCPBars**

6.18.1 Member Enumeration Documentation

6.18.1.1 enum QCPBarsGroup::SpacingType

Defines the ways the spacing between bars in the group can be specified. Thus it defines what the number passed to setSpacing actually means.

See also

setSpacingType, setSpacing

Enumerator

stAbsolute Bar spacing is in absolute pixels.

stAxisRectRatio Bar spacing is given by a fraction of the axis rect size.

stPlotCoords Bar spacing is in key coordinates and thus scales with the key axis range.

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

- src/qcustomplot/qcustomplot.h

6.19 QCPColorGradient Class Reference

Public Types

- enum ColorInterpolation { ciRGB, ciHSV }
- enum GradientPreset {
  gpGrayscale, gpHot, gpCold, gpNight,
  gpCandy, gpGeography, gpIon, gpThermal,
  gpPolar, gpSpectrum, gpJet, gpHues }

Public Member Functions

- **QCPColorGradient** (GradientPreset preset=gpCold)
- bool **operator==** (const QCPColorGradient &other) const
- bool **operator!=** (const QCPColorGradient &other) const
- int **levelCount** () const
- QMap< double, QColor > **colorStops** () const
- ColorInterpolation **colorInterpolation** () const
- bool **periodic** () const
- void **setLevelCount** (int n)
- void **setColorStops** (const QMap< double, QColor > &colorStops)
- void **setColorStopAt** (double position, const QColor &color)
- void **setColorInterpolation** (ColorInterpolation interpolation)
- void **setPeriodic** (bool enabled)
- void **colorize** (const double *data, const QCPRange &range, QRgb *scanLine, int n, int dataIndexFactor=1, bool logarithmic=false)
- QRgb **color** (double position, const QCPRange &range, bool logarithmic=false)
- void **loadPreset** (GradientPreset preset)
- void **clearColorStops** ()
- QCPColorGradient **inverted** () const

Protected Member Functions

- void **updateColorBuffer** ()

Protected Attributes

- int **mLevelCount**
- QMap< double, QColor > **mColorStops**
- ColorInterpolation **mColorInterpolation**
- bool **mPeriodic**
- QVector< QRgb > **mColorBuffer**
- bool **mColorBufferInvalidated**

6.19.1 Member Enumeration Documentation

6.19.1.1 enum QCPColorGradient::ColorInterpolation

Defines the color spaces in which color interpolation between gradient stops can be performed.

See also

setColorInterpolation

Enumerator

ciRGB Color channels red, green and blue are linearly interpolated.

ciHSV Color channels hue, saturation and value are linearly interpolated (The hue is interpolated over the shortest angle distance)

6.19.1.2 enum QCPColorGradient::GradientPreset

Defines the available presets that can be loaded with loadPreset. See the documentation there for an image of the presets.

Enumerator

gpGrayscale Continuous lightness from black to white (suited for non-biased data representation)

gpHot Continuous lightness from black over firey colors to white (suited for non-biased data representation)

gpCold Continuous lightness from black over icey colors to white (suited for non-biased data representation)

gpNight Continuous lightness from black over weak blueish colors to white (suited for non-biased data representation)

gpCandy Blue over pink to white.

gpGeography Colors suitable to represent different elevations on geographical maps.

gpIon Half hue spectrum from black over purple to blue and finally green (creates banding illusion but allows more precise magnitude estimates)

gpThermal Colors suitable for thermal imaging, ranging from dark blue over purple to orange, yellow and white.

gpPolar Colors suitable to emphasize polarity around the center, with blue for negative, black in the middle and red for positive values.

gpSpectrum An approximation of the visible light spectrum (creates banding illusion but allows more precise magnitude estimates)

gpJet Hue variation similar to a spectrum, often used in numerical visualization (creates banding illusion but allows more precise magnitude estimates)

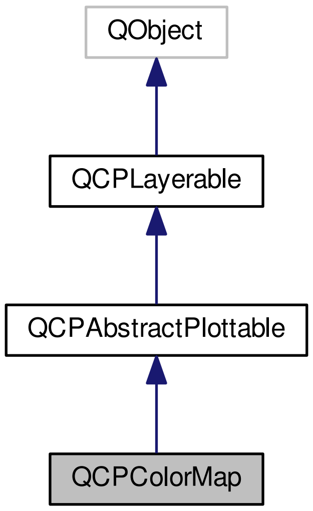
gpHues Full hue cycle, with highest and lowest color red (suitable for periodic data, such as angles and phases, see setPeriodic)

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

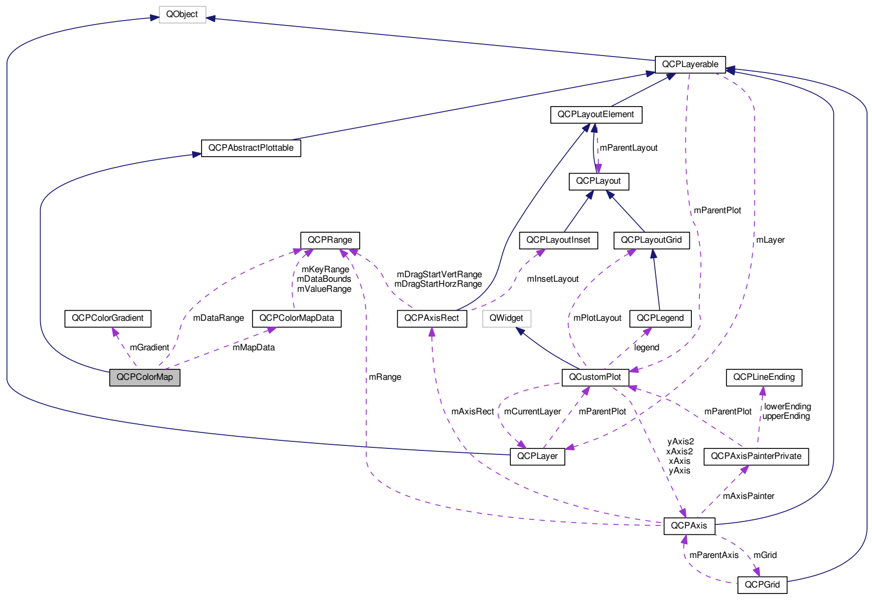
- src/qcustomplot/qcustomplot.h

6.20 QCPColorMap Class Reference

Inheritance diagram for QCPColorMap:



Collaboration diagram for QCPColorMap:



Signals

- void **dataRangeChanged** (QCPRange newRange)
- void **dataScaleTypeChanged** (QCPAxis::ScaleType scaleType)
- void **gradientChanged** (QCPColorGradient newGradient)

Public Member Functions

- **QCPColorMap** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- QCPColorMapData ∗ **data** () const
- QCPRange **dataRange** () const
- QCPAxis::ScaleType **dataScaleType** () const
- bool **interpolate** () const
- bool **tightBoundary** () const
- QCPColorGradient **gradient** () const
- QCPColorScale ∗ **colorScale** () const
- void **setData** (QCPColorMapData ∗data, bool copy=false)
- Q_SLOT void **setDataRange** (const QCPRange &dataRange)
- Q_SLOT void **setDataScaleType** (QCPAxis::ScaleType scaleType)
- Q_SLOT void **setGradient** (const QCPColorGradient &gradient)
- void **setInterpolate** (bool enabled)
- void **setTightBoundary** (bool enabled)
- void **setColorScale** (QCPColorScale ∗colorScale)
- void **rescaleDataRange** (bool recalculateDataBounds=false)
- Q_SLOT void **updateLegendIcon** (Qt::TransformationMode transformMode=Qt::SmoothTransformation, const QSize &thumbSize=QSize(32, 18))
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual void **updateMapImage** ()
- virtual void **draw** (QCPPainter ∗painter)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const

Protected Attributes

- QCPRange **mDataRange**
- QCPAxis::ScaleType **mDataScaleType**
- QCPColorMapData ∗ **mMapData**
- QCPColorGradient **mGradient**
- bool **mInterpolate**
- bool **mTightBoundary**
- QPointer< QCPColorScale > **mColorScale**
- QImage **mMapImage**
- QImage **mUndersampledMapImage**
- QPixmap **mLegendIcon**
- bool **mMapImageInvalidated**

Friends

- class **QCustomPlot**
- class **QCPLegend**

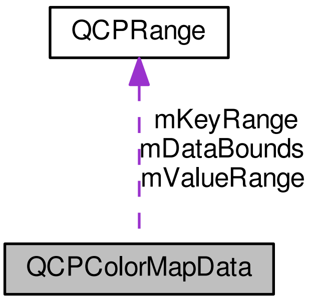
Additional Inherited Members

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

- src/qcustomplot/qcustomplot.h

6.21 QCPColorMapData Class Reference

Collaboration diagram for QCPColorMapData:



Public Member Functions

- **QCPColorMapData** (int keySize, int valueSize, const QCPRange &keyRange, const QCPRange &value←Range)
- **QCPColorMapData** (const QCPColorMapData &other)
- QCPColorMapData & **operator=** (const QCPColorMapData &other)
- int **keySize** () const
- int **valueSize** () const
- QCPRange **keyRange** () const
- QCPRange **valueRange** () const
- QCPRange **dataBounds** () const
- double **data** (double key, double value)
- double **cell** (int keyIndex, int valueIndex)
- void **setSize** (int keySize, int valueSize)
- void **setKeySize** (int keySize)
- void **setValueSize** (int valueSize)
- void **setRange** (const QCPRange &keyRange, const QCPRange &valueRange)
- void **setKeyRange** (const QCPRange &keyRange)
- void **setValueRange** (const QCPRange &valueRange)
- void **setData** (double key, double value, double z)
- void **setCell** (int keyIndex, int valueIndex, double z)
- void **recalculateDataBounds** ()
- void **clear** ()
- void **fill** (double z)
- bool **isEmpty** () const
- void **coordToCell** (double key, double value, int ∗keyIndex, int ∗valueIndex) const
- void **cellToCoord** (int keyIndex, int valueIndex, double ∗key, double ∗value) const

Protected Attributes

- int **mKeySize**
- int **mValueSize**
- QCPRange **mKeyRange**
- QCPRange **mValueRange**
- bool **mIsEmpty**
- double ∗ **mData**
- QCPRange **mDataBounds**
- bool **mDataModified**

Friends

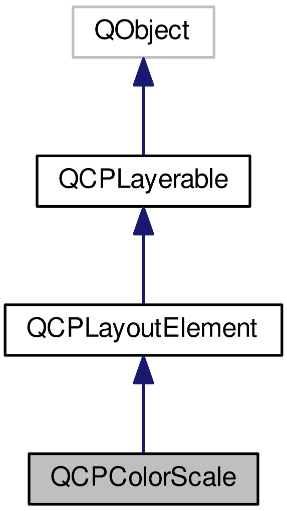
- class **QCPColorMap**

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

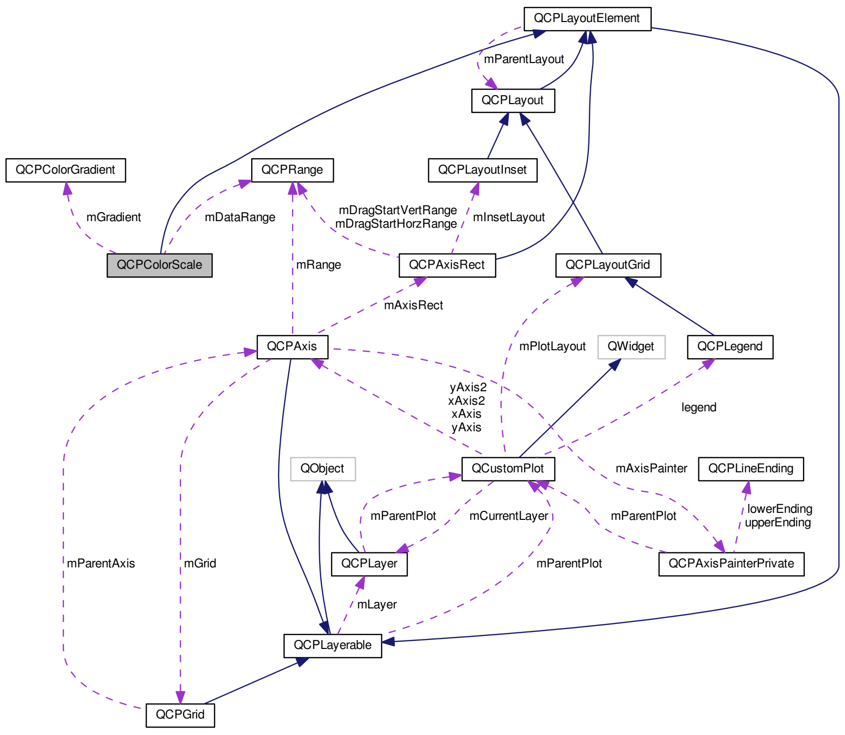
- src/qcustomplot/qcustomplot.h

6.22 QCPColorScale Class Reference

Inheritance diagram for QCPColorScale:



Collaboration diagram for QCPColorScale:



Signals

- void **dataRangeChanged** (QCPRange newRange)
- void **dataScaleTypeChanged** (QCPAxis::ScaleType scaleType)
- void **gradientChanged** (QCPColorGradient newGradient)

Public Member Functions

- **QCPColorScale** (QCustomPlot ∗parentPlot)
- QCPAxis ∗ **axis** () const
- QCPAxis::AxisType **type** () const
- QCPRange **dataRange** () const
- QCPAxis::ScaleType **dataScaleType** () const
- QCPColorGradient **gradient** () const
- QString **label** () const
- int **barWidth** () const
- bool **rangeDrag** () const
- bool **rangeZoom** () const
- void **setType** (QCPAxis::AxisType type)
- Q_SLOT void **setDataRange** (const QCPRange &dataRange)
- Q_SLOT void **setDataScaleType** (QCPAxis::ScaleType scaleType)
- Q_SLOT void **setGradient** (const QCPColorGradient &gradient)
- void **setLabel** (const QString &str)
- void **setBarWidth** (int width)
- void **setRangeDrag** (bool enabled)
- void **setRangeZoom** (bool enabled)
- QList< QCPColorMap ∗ > **colorMaps** () const
- void **rescaleDataRange** (bool onlyVisibleMaps)
- virtual void **update** (UpdatePhase phase)

Protected Member Functions

- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **mousePressEvent** (QMouseEvent ∗event)
- virtual void **mouseMoveEvent** (QMouseEvent ∗event)
- virtual void **mouseReleaseEvent** (QMouseEvent ∗event)
- virtual void **wheelEvent** (QWheelEvent ∗event)

Protected Attributes

- QCPAxis::AxisType **mType**
- QCPRange **mDataRange**
- QCPAxis::ScaleType **mDataScaleType**
- QCPColorGradient **mGradient**
- int **mBarWidth**
- QPointer< QCPColorScaleAxisRectPrivate > **mAxisRect**
- QPointer< QCPAxis > **mColorAxis**

Friends

- class **QCPColorScaleAxisRectPrivate**

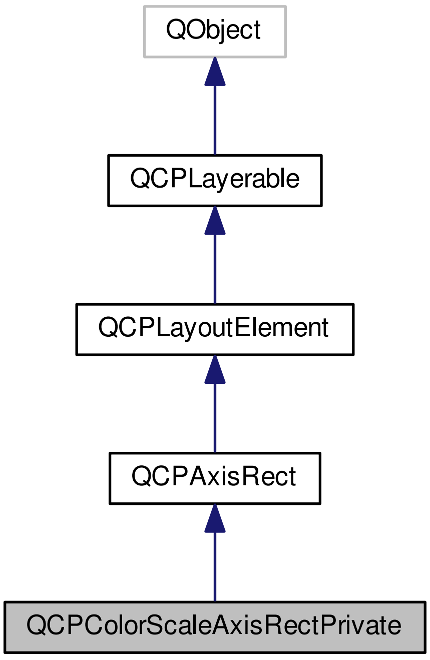
Additional Inherited Members

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

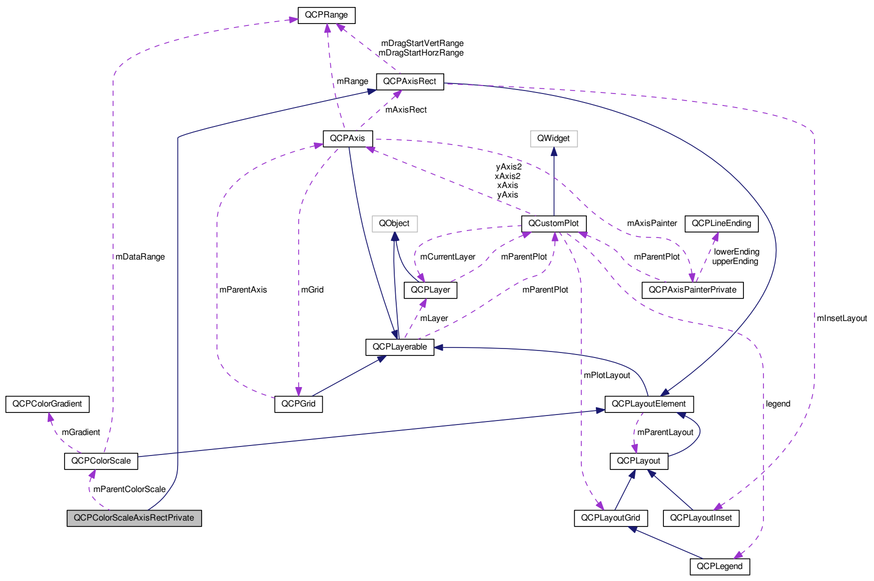
- src/qcustomplot/qcustomplot.h

6.23 QCPColorScaleAxisRectPrivate Class Reference

Inheritance diagram for QCPColorScaleAxisRectPrivate:



Collaboration diagram for QCPColorScaleAxisRectPrivate:



Public Member Functions

- **QCPColorScaleAxisRectPrivate** (QCPColorScale ∗parentColorScale)

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- void **updateGradientImage** ()
- Q_SLOT void **axisSelectionChanged** (QCPAxis::SelectableParts selectedParts)
- Q_SLOT void **axisSelectableChanged** (QCPAxis::SelectableParts selectableParts)

Protected Attributes

- QCPColorScale ∗ **mParentColorScale**
- QImage **mGradientImage**
- bool **mGradientImageInvalidated**

Friends

- class **QCPColorScale**

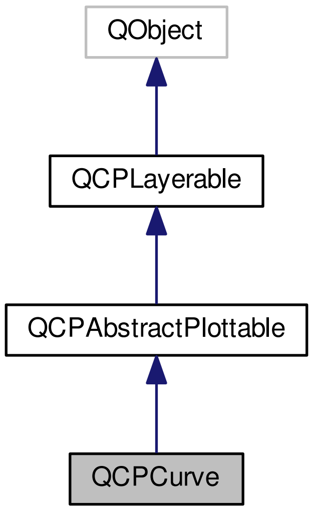
Additional Inherited Members

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

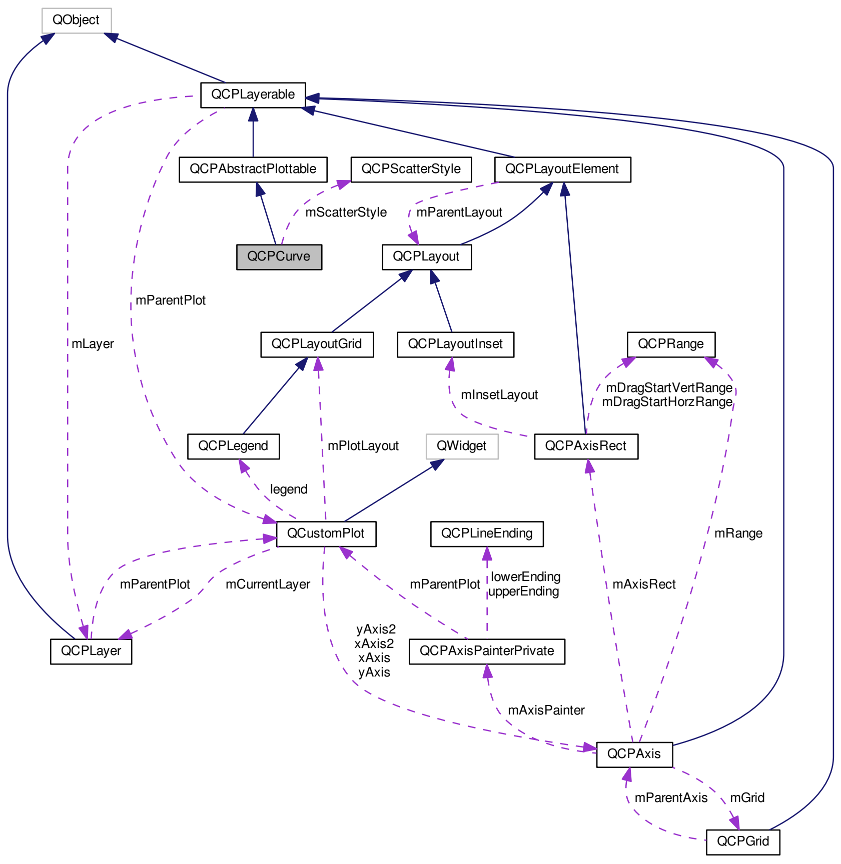
- src/qcustomplot/qcustomplot.h

6.24 QCPCurve Class Reference

Inheritance diagram for QCPCurve:



Collaboration diagram for QCPCurve:



Public Types

- enum LineStyle { lsNone, lsLine }

Public Member Functions

- **QCPCurve** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- QCPCurveDataMap ∗ **data** () const
- QCPScatterStyle **scatterStyle** () const
- LineStyle **lineStyle** () const
- void **setData** (QCPCurveDataMap ∗data, bool copy=false)
- void **setData** (const QVector< double > &t, const QVector< double > &key, const QVector< double > &value)
- void **setData** (const QVector< double > &key, const QVector< double > &value)
- void **setScatterStyle** (const QCPScatterStyle &style)
- void **setLineStyle** (LineStyle style)
- void **addData** (const QCPCurveDataMap &dataMap)
- void **addData** (const QCPCurveData &data)
- void **addData** (double t, double key, double value)
- void **addData** (double key, double value)

- void **addData** (const QVector< double > &ts, const QVector< double > &keys, const QVector< double > &values)
- void **removeDataBefore** (double t)
- void **removeDataAfter** (double t)
- void **removeData** (double fromt, double tot)
- void **removeData** (double t)
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant *details=0) const

Protected Member Functions

- virtual void **draw** (QCPPainter *painter)
- virtual void **drawLegendIcon** (QCPPainter *painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual void **drawScatterPlot** (QCPPainter *painter, const QVector< QPointF > *pointData) const
- void **getCurveData** (QVector< QPointF > *lineData) const
- int **getRegion** (double x, double y, double rectLeft, double rectTop, double rectRight, double rectBottom) const
- QPointF **getOptimizedPoint** (int prevRegion, double prevKey, double prevValue, double key, double value, double rectLeft, double rectTop, double rectRight, double rectBottom) const
- QVector< QPointF > **getOptimizedCornerPoints** (int prevRegion, int currentRegion, double prevKey, double prevValue, double key, double value, double rectLeft, double rectTop, double rectRight, double rectBottom) const
- bool **mayTraverse** (int prevRegion, int currentRegion) const
- bool **getTraverse** (double prevKey, double prevValue, double key, double value, double rectLeft, double rect↩ Top, double rectRight, double rectBottom, QPointF &crossA, QPointF &crossB) const
- void **getTraverseCornerPoints** (int prevRegion, int currentRegion, double rectLeft, double rectTop, double rectRight, double rectBottom, QVector< QPointF > &beforeTraverse, QVector< QPointF > &afterTraverse) const
- double **pointDistance** (const QPointF &pixelPoint) const

Protected Attributes

- QCPCurveDataMap * **mData**
- QCPScatterStyle **mScatterStyle**
- LineStyle **mLineStyle**

Friends

- class **QCustomPlot**
- class **QCPLegend**

Additional Inherited Members

6.24.1 Member Enumeration Documentation

6.24.1.1 enum QCPCurve::LineStyle

Defines how the curve's line is represented visually in the plot. The line is drawn with the current pen of the curve (setPen).

See also

setLineStyle

Enumerator

lsNone No line is drawn between data points (e.g. only scatters)

lsLine Data points are connected with a straight line.

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

- src/qcustomplot/qcustomplot.h

6.25 QCPCurveData Class Reference

Public Member Functions

- **QCPCurveData** (double t, double key, double value)

Public Attributes

- double **t**
- double **key**
- double **value**

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

- src/qcustomplot/qcustomplot.h

6.26 QCPData Class Reference

Public Member Functions

- **QCPData** (double key, double value)

Public Attributes

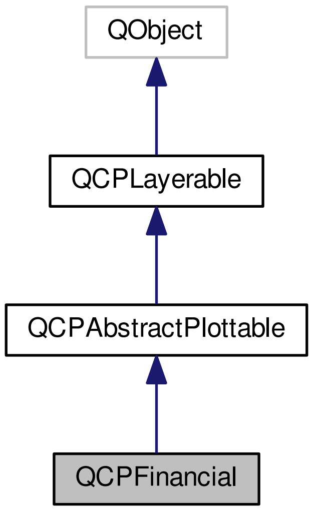
- double **key**
- double **value**
- double **keyErrorPlus**
- double **keyErrorMinus**
- double **valueErrorPlus**
- double **valueErrorMinus**

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

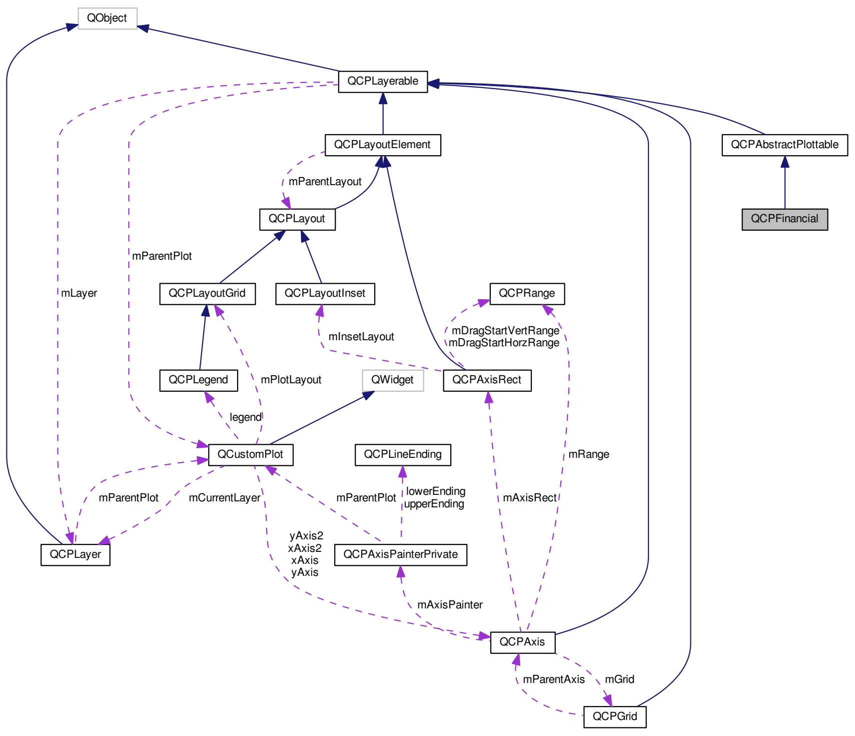
- src/qcustomplot/qcustomplot.h

6.27 QCPFinancial Class Reference

Inheritance diagram for QCPFinancial:



Collaboration diagram for QCPFinancial:



Public Types

- enum ChartStyle { csOhlc, csCandlestick }

Public Member Functions

- **QCPFinancial** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- QCPFinancialDataMap ∗ **data** () const
- ChartStyle **chartStyle** () const
- double **width** () const
- bool **twoColored** () const
- QBrush **brushPositive** () const
- QBrush **brushNegative** () const
- QPen **penPositive** () const
- QPen **penNegative** () const
- void **setData** (QCPFinancialDataMap ∗data, bool copy=false)
- void **setData** (const QVector< double > &key, const QVector< double > &open, const QVector< double > &high, const QVector< double > &low, const QVector< double > &close)
- void **setChartStyle** (ChartStyle style)
- void **setWidth** (double width)
- void **setTwoColored** (bool twoColored)
- void **setBrushPositive** (const QBrush &brush)
- void **setBrushNegative** (const QBrush &brush)
- void **setPenPositive** (const QPen &pen)
- void **setPenNegative** (const QPen &pen)

- void **addData** (const QCPFinancialDataMap &dataMap)
- void **addData** (const QCPFinancialData &data)
- void **addData** (double key, double open, double high, double low, double close)
- void **addData** (const QVector< double > &key, const QVector< double > &open, const QVector< double > &high, const QVector< double > &low, const QVector< double > &close)
- void **removeDataBefore** (double key)
- void **removeDataAfter** (double key)
- void **removeData** (double fromKey, double toKey)
- void **removeData** (double key)
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Static Public Member Functions

- static QCPFinancialDataMap **timeSeriesToOhlc** (const QVector< double > &time, const QVector< double > &value, double timeBinSize, double timeBinOffset=0)

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- void **drawOhlcPlot** (QCPPainter ∗painter, const QCPFinancialDataMap::const_iterator &begin, const QC↩ PFinancialDataMap::const_iterator &end)
- void **drawCandlestickPlot** (QCPPainter ∗painter, const QCPFinancialDataMap::const_iterator &begin, const QCPFinancialDataMap::const_iterator &end)
- double **ohlcSelectTest** (const QPointF &pos, const QCPFinancialDataMap::const_iterator &begin, const Q↩ CPFinancialDataMap::const_iterator &end) const
- double **candlestickSelectTest** (const QPointF &pos, const QCPFinancialDataMap::const_iterator &begin, const QCPFinancialDataMap::const_iterator &end) const
- void **getVisibleDataBounds** (QCPFinancialDataMap::const_iterator &lower, QCPFinancialDataMap↩ ::const_iterator &upper) const

Protected Attributes

- QCPFinancialDataMap ∗ **mData**
- ChartStyle **mChartStyle**
- double **mWidth**
- bool **mTwoColored**
- QBrush **mBrushPositive**
- QBrush **mBrushNegative**
- QPen **mPenPositive**
- QPen **mPenNegative**

Friends

- class **QCustomPlot**
- class **QCPLegend**

Additional Inherited Members

6.27.1 Member Enumeration Documentation

6.27.1.1 enum QCPFinancial::ChartStyle

Defines the possible representations of OHLC data in the plot.

See also

setChartStyle

Enumerator

csOhlc	Open-High-Low-Close bar representation.
csCandlestick	Candlestick representation.

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

- src/qcustomplot/qcustomplot.h

6.28 QCPFinancialData Class Reference

Public Member Functions

- **QCPFinancialData** (double key, double open, double high, double low, double close)

Public Attributes

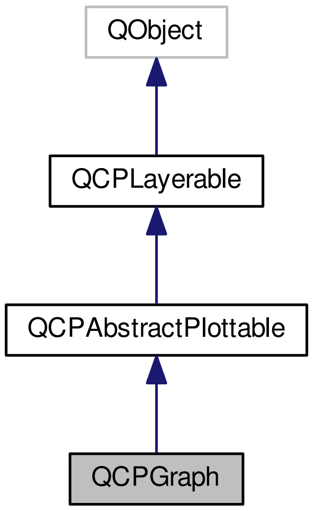
- double **key**
- double **open**
- double **high**
- double **low**
- double **close**

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

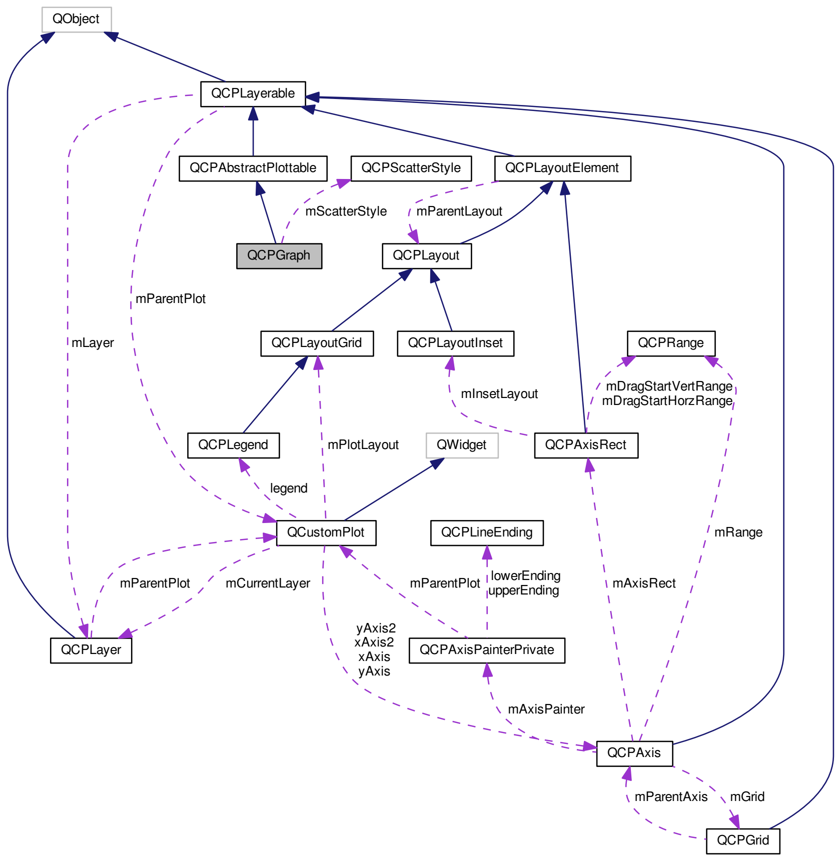
- src/qcustomplot/qcustomplot.h

6.29 QCPGraph Class Reference

Inheritance diagram for QCPGraph:



Collaboration diagram for QCPGraph:



Public Types

- enum LineStyle {
  lsNone, lsLine, lsStepLeft, lsStepRight,
  lsStepCenter, lsImpulse }
- enum ErrorType { etNone, etKey, etValue, etBoth }

Public Member Functions

- **QCPGraph** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- QCPDataMap ∗ **data** () const
- LineStyle **lineStyle** () const
- QCPScatterStyle **scatterStyle** () const
- ErrorType **errorType** () const
- QPen **errorPen** () const
- double **errorBarSize** () const
- bool **errorBarSkipSymbol** () const
- QCPGraph ∗ **channelFillGraph** () const
- bool **adaptiveSampling** () const
- void **setData** (QCPDataMap ∗data, bool copy=false)
- void **setData** (const QVector< double > &key, const QVector< double > &value)

- void **setDataKeyError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &keyError)
- void **setDataKeyError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &keyErrorMinus, const QVector< double > &keyErrorPlus)
- void **setDataValueError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &valueError)
- void **setDataValueError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &valueErrorMinus, const QVector< double > &valueErrorPlus)
- void **setDataBothError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &keyError, const QVector< double > &valueError)
- void **setDataBothError** (const QVector< double > &key, const QVector< double > &value, const QVector< double > &keyErrorMinus, const QVector< double > &keyErrorPlus, const QVector< double > &value←ErrorMinus, const QVector< double > &valueErrorPlus)
- void **setLineStyle** (LineStyle ls)
- void **setScatterStyle** (const QCPScatterStyle &style)
- void **setErrorType** (ErrorType errorType)
- void **setErrorPen** (const QPen &pen)
- void **setErrorBarSize** (double size)
- void **setErrorBarSkipSymbol** (bool enabled)
- void **setChannelFillGraph** (QCPGraph ∗targetGraph)
- void **setAdaptiveSampling** (bool enabled)
- void **addData** (const QCPDataMap &dataMap)
- void **addData** (const QCPData &data)
- void **addData** (double key, double value)
- void **addData** (const QVector< double > &keys, const QVector< double > &values)
- void **removeDataBefore** (double key)
- void **removeDataAfter** (double key)
- void **removeData** (double fromKey, double toKey)
- void **removeData** (double key)
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- void **rescaleAxes** (bool onlyEnlarge, bool includeErrorBars) const
- void **rescaleKeyAxis** (bool onlyEnlarge, bool includeErrorBars) const
- void **rescaleValueAxis** (bool onlyEnlarge, bool includeErrorBars) const

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain, bool includeErrors) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain, bool includeErrors) const
- virtual void **drawFill** (QCPPainter ∗painter, QVector< QPointF > ∗lineData) const
- virtual void **drawScatterPlot** (QCPPainter ∗painter, QVector< QCPData > ∗scatterData) const
- virtual void **drawLinePlot** (QCPPainter ∗painter, QVector< QPointF > ∗lineData) const
- virtual void **drawImpulsePlot** (QCPPainter ∗painter, QVector< QPointF > ∗lineData) const
- void **getPreparedData** (QVector< QCPData > ∗lineData, QVector< QCPData > ∗scatterData) const
- void **getPlotData** (QVector< QPointF > ∗lineData, QVector< QCPData > ∗scatterData) const
- void **getScatterPlotData** (QVector< QCPData > ∗scatterData) const
- void **getLinePlotData** (QVector< QPointF > ∗linePixelData, QVector< QCPData > ∗scatterData) const
- void **getStepLeftPlotData** (QVector< QPointF > ∗linePixelData, QVector< QCPData > ∗scatterData) const

- void **getStepRightPlotData** (QVector< QPointF > ∗linePixelData, QVector< QCPData > ∗scatterData) const
- void **getStepCenterPlotData** (QVector< QPointF > ∗linePixelData, QVector< QCPData > ∗scatterData) const
- void **getImpulsePlotData** (QVector< QPointF > ∗linePixelData, QVector< QCPData > ∗scatterData) const

- void **drawError** (QCPPainter ∗painter, double x, double y, const QCPData &data) const
- void **getVisibleDataBounds** (QCPDataMap::const_iterator &lower, QCPDataMap::const_iterator &upper) const
- int **countDataInBounds** (const QCPDataMap::const_iterator &lower, const QCPDataMap::const_iterator &upper, int maxCount) const
- void **addFillBasePoints** (QVector< QPointF > ∗lineData) const
- void **removeFillBasePoints** (QVector< QPointF > ∗lineData) const
- QPointF **lowerFillBasePoint** (double lowerKey) const
- QPointF **upperFillBasePoint** (double upperKey) const
- const QPolygonF **getChannelFillPolygon** (const QVector< QPointF > ∗lineData) const
- int **findIndexBelowX** (const QVector< QPointF > ∗data, double x) const
- int **findIndexAboveX** (const QVector< QPointF > ∗data, double x) const
- int **findIndexBelowY** (const QVector< QPointF > ∗data, double y) const
- int **findIndexAboveY** (const QVector< QPointF > ∗data, double y) const
- double **pointDistance** (const QPointF &pixelPoint) const

Protected Attributes

- QCPDataMap ∗ **mData**
- QPen **mErrorPen**
- LineStyle **mLineStyle**
- QCPScatterStyle **mScatterStyle**
- ErrorType **mErrorType**
- double **mErrorBarSize**
- bool **mErrorBarSkipSymbol**
- QPointer< QCPGraph > **mChannelFillGraph**
- bool **mAdaptiveSampling**

Friends

- class **QCustomPlot**
- class **QCPLegend**

Additional Inherited Members

6.29.1 Member Enumeration Documentation

6.29.1.1 enum QCPGraph::ErrorType

Defines what kind of error bars are drawn for each data point

Enumerator

etNone	No error bars are shown.
etKey	Error bars for the key dimension of the data point are shown.
etValue	Error bars for the value dimension of the data point are shown.
etBoth	Error bars for both key and value dimensions of the data point are shown.

6.29.1.2 enum QCPGraph::LineStyle

Defines how the graph's line is represented visually in the plot. The line is drawn with the current pen of the graph (setPen).

See also

setLineStyle

Enumerator

lsNone data points are not connected with any lines (e.g. data only represented $<$ with symbols according to the scatter style, see setScatterStyle)

lsLine data points are connected by a straight line

lsStepLeft line is drawn as steps where the step height is the value of the left data point

lsStepRight line is drawn as steps where the step height is the value of the right data point

lsStepCenter line is drawn as steps where the step is in between two data points

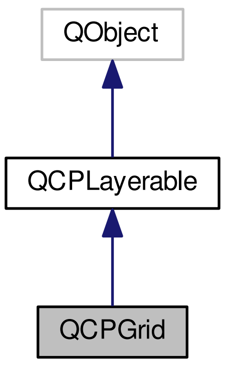
lsImpulse each data point is represented by a line parallel to the value axis, which reaches from the data point to the zero-value-line

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

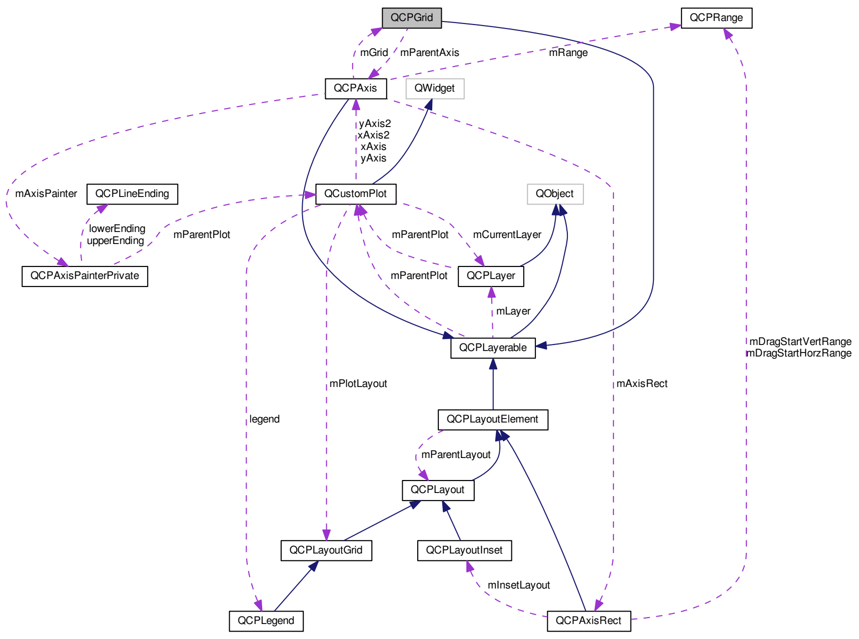
- src/qcustomplot/qcustomplot.h

6.30 QCPGrid Class Reference

Inheritance diagram for QCPGrid:



Collaboration diagram for QCPGrid:



Public Member Functions

- **QCPGrid** (QCPAxis ∗parentAxis)
- bool **subGridVisible** () const
- bool **antialiasedSubGrid** () const
- bool **antialiasedZeroLine** () const
- QPen **pen** () const
- QPen **subGridPen** () const
- QPen **zeroLinePen** () const
- void **setSubGridVisible** (bool visible)
- void **setAntialiasedSubGrid** (bool enabled)
- void **setAntialiasedZeroLine** (bool enabled)
- void **setPen** (const QPen &pen)
- void **setSubGridPen** (const QPen &pen)
- void **setZeroLinePen** (const QPen &pen)

Protected Member Functions

- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- void **drawGridLines** (QCPPainter ∗painter) const
- void **drawSubGridLines** (QCPPainter ∗painter) const

Protected Attributes

- bool **mSubGridVisible**
- bool **mAntialiasedSubGrid**
- bool **mAntialiasedZeroLine**
- QPen **mPen**
- QPen **mSubGridPen**
- QPen **mZeroLinePen**
- QCPAxis ∗ **mParentAxis**

Friends

- class **QCPAxis**

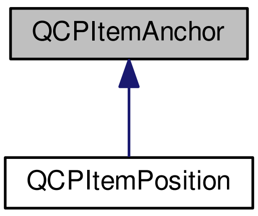
Additional Inherited Members

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

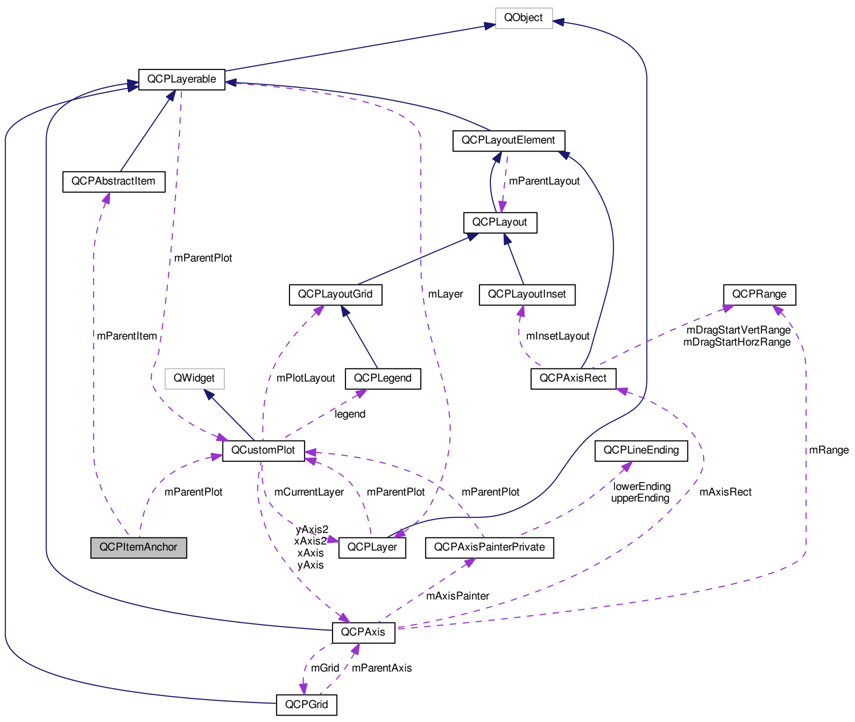
- src/qcustomplot/qcustomplot.h

6.31 QCPItemAnchor Class Reference

Inheritance diagram for QCPItemAnchor:



Collaboration diagram for QCPItemAnchor:



Public Member Functions

- **QCPItemAnchor** (QCustomPlot ∗parentPlot, QCPAbstractItem ∗parentItem, const QString name, int anchorId=-1)
- QString **name** () const
- virtual QPointF **pixelPoint** () const

Protected Member Functions

- virtual QCPItemPosition ∗ **toQCPItemPosition** ()
- void **addChildX** (QCPItemPosition ∗pos)
- void **removeChildX** (QCPItemPosition ∗pos)
- void **addChildY** (QCPItemPosition ∗pos)
- void **removeChildY** (QCPItemPosition ∗pos)

Protected Attributes

- QString **mName**
- QCustomPlot ∗ **mParentPlot**
- QCPAbstractItem ∗ **mParentItem**
- int **mAnchorId**
- QSet< QCPItemPosition ∗ > **mChildrenX**
- QSet< QCPItemPosition ∗ > **mChildrenY**

Friends

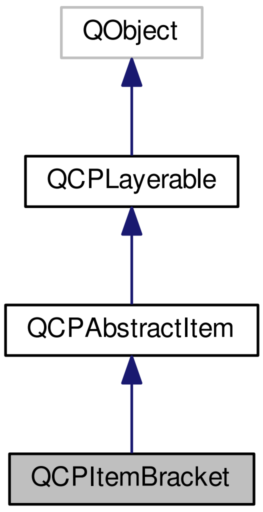
- class **QCPItemPosition**

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

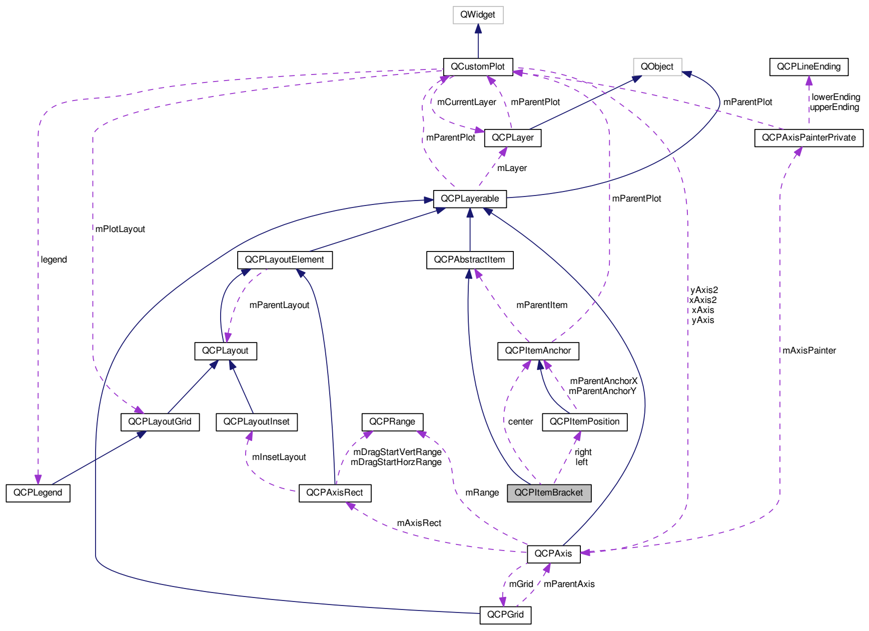
- src/qcustomplot/qcustomplot.h

6.32 QCPItemBracket Class Reference

Inheritance diagram for QCPItemBracket:



Collaboration diagram for QCPItemBracket:



Public Types

- enum BracketStyle { bsSquare, bsRound, bsCurly, bsCalligraphic }

Public Member Functions

- **QCPItemBracket** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- double **length** () const
- BracketStyle **style** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setLength** (double length)
- void **setStyle** (BracketStyle style)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **left**
- QCPItemPosition ∗const **right**
- QCPItemAnchor ∗const **center**

Protected Types

- enum **AnchorIndex** { **aiCenter** }

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- QPen **mainPen** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- double **mLength**
- BracketStyle **mStyle**

Additional Inherited Members

6.32.1 Member Enumeration Documentation

6.32.1.1 enum QCPItemBracket::BracketStyle

Enumerator

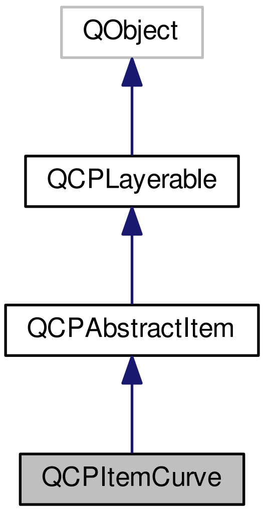
bsSquare	A brace with angled edges.
bsRound	A brace with round edges.
bsCurly	A curly brace.
bsCalligraphic	A curly brace with varying stroke width giving a calligraphic impression.

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

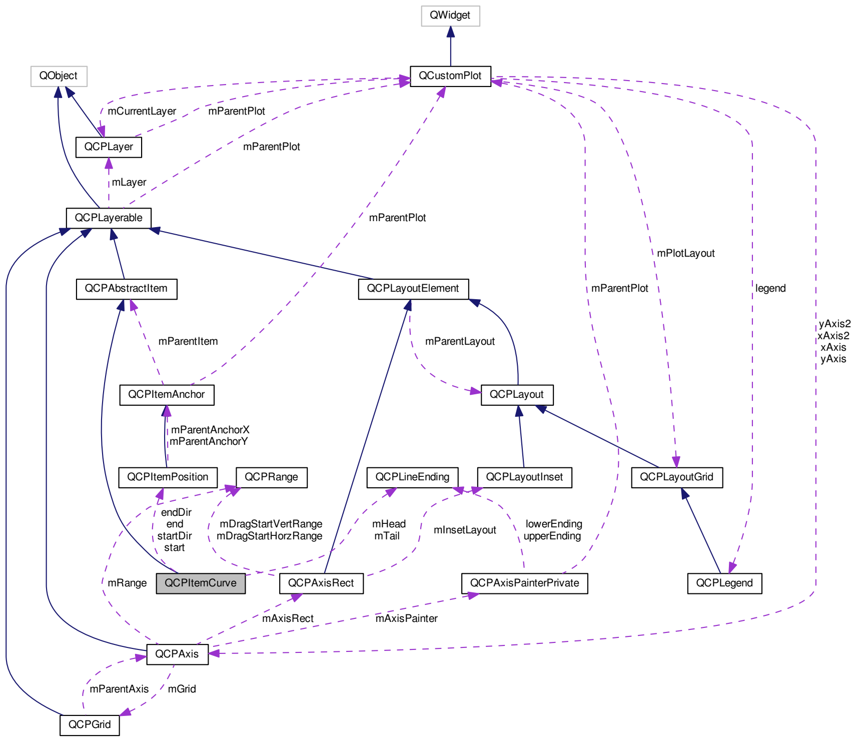
- src/qcustomplot/qcustomplot.h

6.33 QCPItemCurve Class Reference

Inheritance diagram for QCPItemCurve:



Collaboration diagram for QCPItemCurve:



Public Member Functions

- **QCPItemCurve** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- QCPLineEnding **head** () const
- QCPLineEnding **tail** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setHead** (const QCPLineEnding &head)
- void **setTail** (const QCPLineEnding &tail)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **start**
- QCPItemPosition ∗const **startDir**
- QCPItemPosition ∗const **endDir**
- QCPItemPosition ∗const **end**

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- QPen **mainPen** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- QCPLineEnding **mHead**
- QCPLineEnding **mTail**

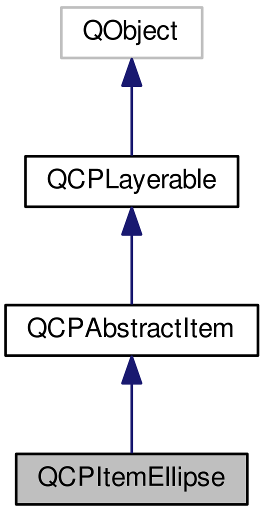
Additional Inherited Members

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

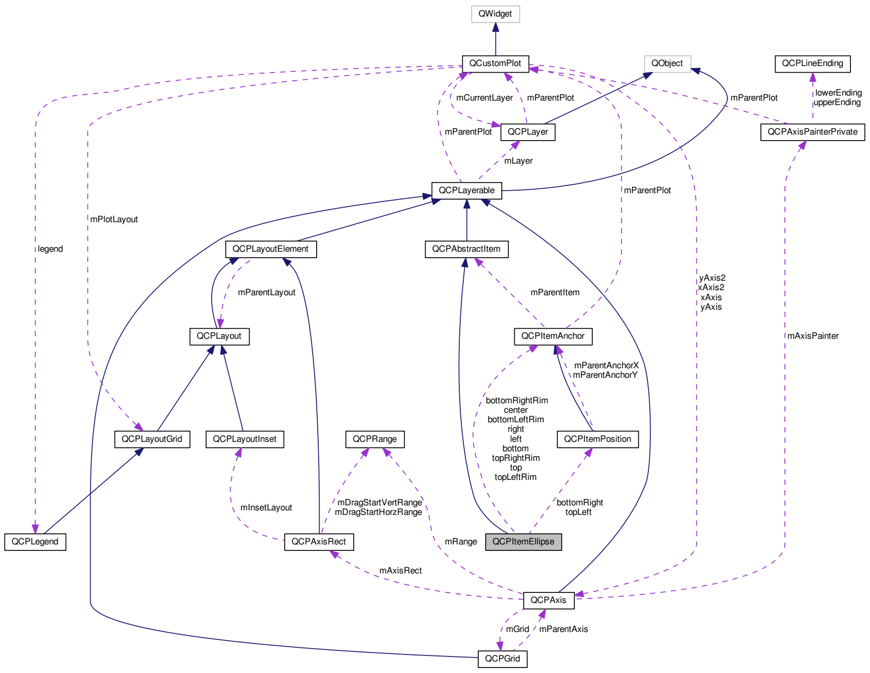
- src/qcustomplot/qcustomplot.h

6.34 QCPItemEllipse Class Reference

Inheritance diagram for QCPItemEllipse:



Collaboration diagram for QCPItemEllipse:



Public Member Functions

- **QCPItemEllipse** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- QBrush **brush** () const
- QBrush **selectedBrush** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setSelectedBrush** (const QBrush &brush)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **topLeft**
- QCPItemPosition ∗const **bottomRight**
- QCPItemAnchor ∗const **topLeftRim**
- QCPItemAnchor ∗const **top**
- QCPItemAnchor ∗const **topRightRim**
- QCPItemAnchor ∗const **right**
- QCPItemAnchor ∗const **bottomRightRim**
- QCPItemAnchor ∗const **bottom**
- QCPItemAnchor ∗const **bottomLeftRim**
- QCPItemAnchor ∗const **left**
- QCPItemAnchor ∗const **center**

Protected Types

- enum **AnchorIndex** {
  aiTopLeftRim, **aiTop**, **aiTopRightRim**, **aiRight**,
  aiBottomRightRim, **aiBottom**, **aiBottomLeftRim**, **aiLeft**,
  aiCenter }

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- QPen **mainPen** () const
- QBrush **mainBrush** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- QBrush **mBrush**
- QBrush **mSelectedBrush**

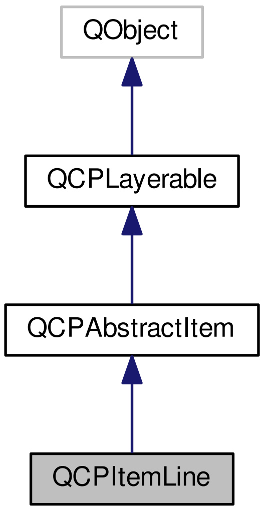
Additional Inherited Members

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

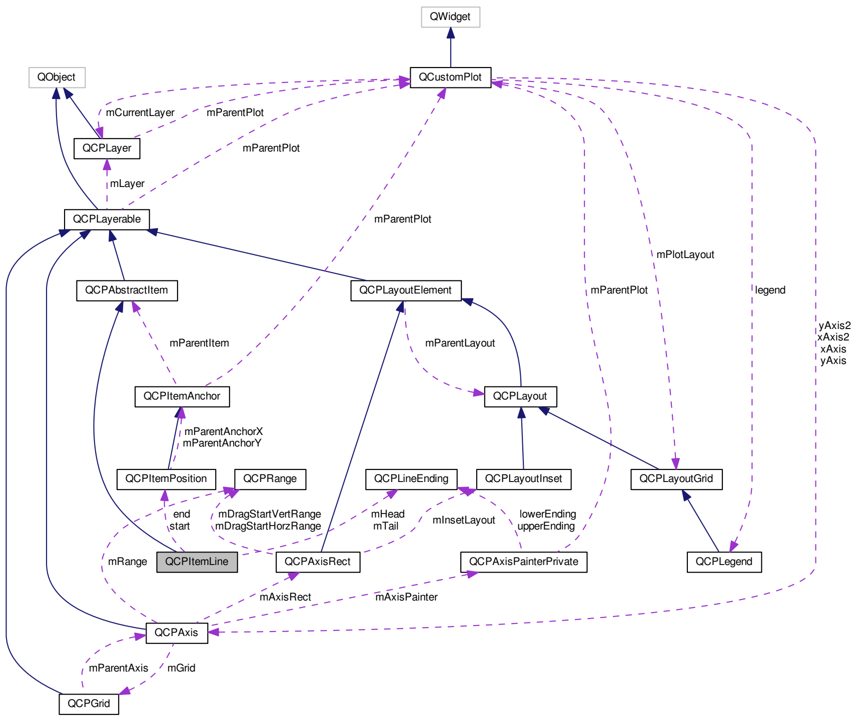
- src/qcustomplot/qcustomplot.h

6.35 QCPItemLine Class Reference

Inheritance diagram for QCPItemLine:



Collaboration diagram for QCPItemLine:



Public Member Functions

- **QCPItemLine** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- QCPLineEnding **head** () const
- QCPLineEnding **tail** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setHead** (const QCPLineEnding &head)
- void **setTail** (const QCPLineEnding &tail)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **start**
- QCPItemPosition ∗const **end**

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- QLineF **getRectClippedLine** (const QVector2D &start, const QVector2D &end, const QRect &rect) const
- QPen **mainPen** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- QCPLineEnding **mHead**
- QCPLineEnding **mTail**

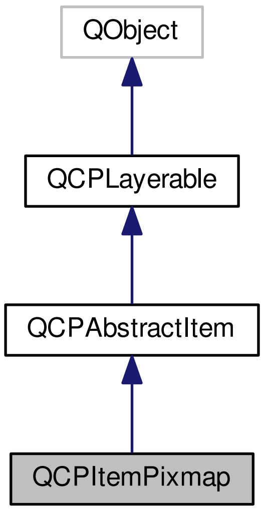
Additional Inherited Members

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

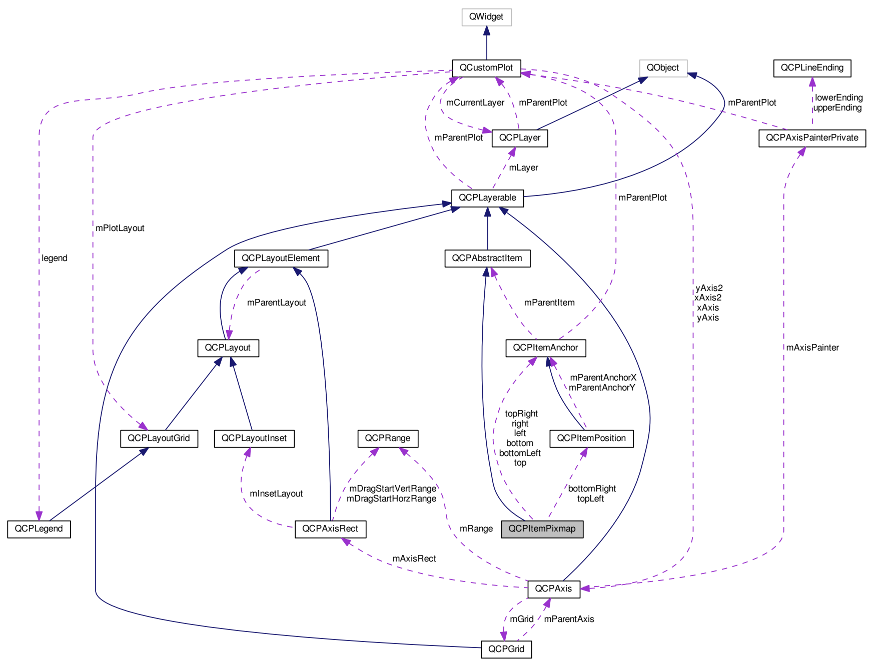
- src/qcustomplot/qcustomplot.h

6.36 QCPItemPixmap Class Reference

Inheritance diagram for QCPItemPixmap:



Collaboration diagram for QCPItemPixmap:



Public Member Functions

- **QCPItemPixmap** (QCustomPlot ∗parentPlot)
- QPixmap **pixmap** () const
- bool **scaled** () const
- Qt::AspectRatioMode **aspectRatioMode** () const
- Qt::TransformationMode **transformationMode** () const
- QPen **pen** () const
- QPen **selectedPen** () const
- void **setPixmap** (const QPixmap &pixmap)
- void **setScaled** (bool scaled, Qt::AspectRatioMode aspectRatioMode=Qt::KeepAspectRatio, Qt::↩ TransformationMode transformationMode=Qt::SmoothTransformation)
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **topLeft**
- QCPItemPosition ∗const **bottomRight**
- QCPItemAnchor ∗const **top**
- QCPItemAnchor ∗const **topRight**
- QCPItemAnchor ∗const **right**
- QCPItemAnchor ∗const **bottom**
- QCPItemAnchor ∗const **bottomLeft**
- QCPItemAnchor ∗const **left**

Protected Types

- enum **AnchorIndex** {
  aiTop, **aiTopRight**, **aiRight**, **aiBottom**,
  aiBottomLeft, **aiLeft** }

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- void **updateScaledPixmap** (QRect finalRect=QRect(), bool flipHorz=false, bool flipVert=false)
- QRect **getFinalRect** (bool ∗flippedHorz=0, bool ∗flippedVert=0) const
- QPen **mainPen** () const

Protected Attributes

- QPixmap **mPixmap**
- QPixmap **mScaledPixmap**
- bool **mScaled**
- bool **mScaledPixmapInvalidated**
- Qt::AspectRatioMode **mAspectRatioMode**
- Qt::TransformationMode **mTransformationMode**
- QPen **mPen**
- QPen **mSelectedPen**

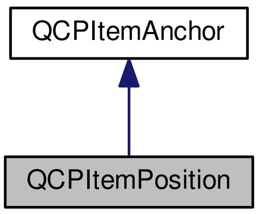
Additional Inherited Members

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

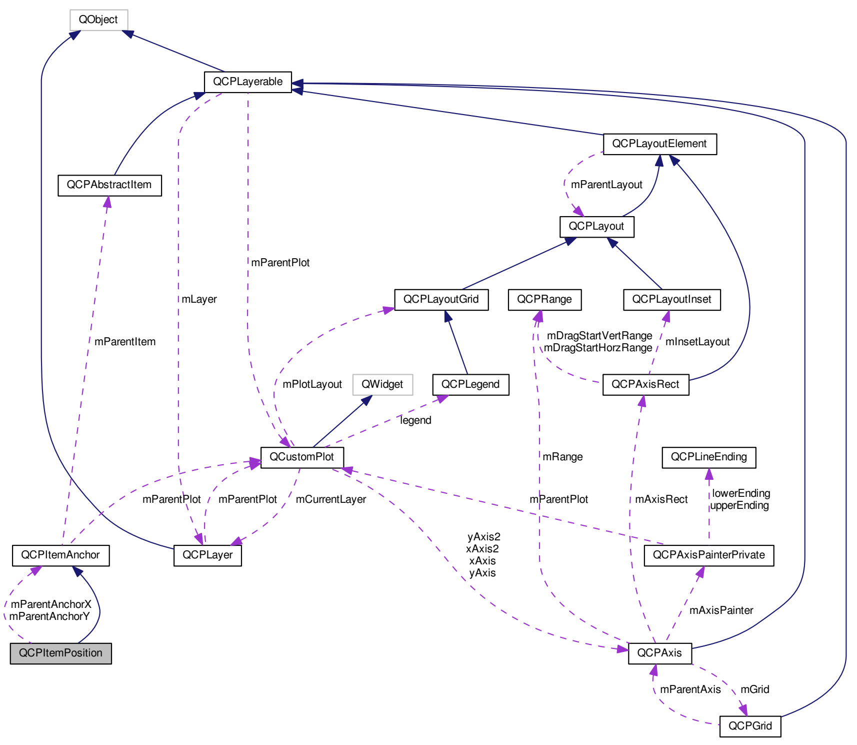
- src/qcustomplot/qcustomplot.h

6.37 QCPItemPosition Class Reference

Inheritance diagram for QCPItemPosition:



Collaboration diagram for QCPItemPosition:



Public Types

- enum PositionType { ptAbsolute, ptViewportRatio, ptAxisRectRatio, ptPlotCoords }

Public Member Functions

- **QCPItemPosition** (QCustomPlot ∗parentPlot, QCPAbstractItem ∗parentItem, const QString name)
- PositionType **type** () const
- PositionType **typeX** () const
- PositionType **typeY** () const
- QCPItemAnchor ∗ **parentAnchor** () const
- QCPItemAnchor ∗ **parentAnchorX** () const
- QCPItemAnchor ∗ **parentAnchorY** () const
- double **key** () const
- double **value** () const
- QPointF **coords** () const
- QCPAxis ∗ **keyAxis** () const
- QCPAxis ∗ **valueAxis** () const
- QCPAxisRect ∗ **axisRect** () const
- virtual QPointF **pixelPoint** () const
- void **setType** (PositionType type)
- void **setTypeX** (PositionType type)
- void **setTypeY** (PositionType type)
- bool **setParentAnchor** (QCPItemAnchor ∗parentAnchor, bool keepPixelPosition=false)
- bool **setParentAnchorX** (QCPItemAnchor ∗parentAnchor, bool keepPixelPosition=false)

- bool **setParentAnchorY** (QCPItemAnchor ∗parentAnchor, bool keepPixelPosition=false)
- void **setCoords** (double key, double value)
- void **setCoords** (const QPointF &coords)
- void **setAxes** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- void **setAxisRect** (QCPAxisRect ∗axisRect)
- void **setPixelPoint** (const QPointF &pixelPoint)

Protected Member Functions

- virtual QCPItemPosition ∗ **toQCPItemPosition** ()

Protected Attributes

- PositionType **mPositionTypeX**
- PositionType **mPositionTypeY**
- QPointer< QCPAxis > **mKeyAxis**
- QPointer< QCPAxis > **mValueAxis**
- QPointer< QCPAxisRect > **mAxisRect**
- double **mKey**
- double **mValue**
- QCPItemAnchor ∗ **mParentAnchorX**
- QCPItemAnchor ∗ **mParentAnchorY**

6.37.1 Member Enumeration Documentation

6.37.1.1 enum QCPItemPosition::PositionType

Defines the ways an item position can be specified. Thus it defines what the numbers passed to setCoords actually mean.

See also

setType

Enumerator

ptAbsolute Static positioning in pixels, starting from the top left corner of the viewport/widget.

ptViewportRatio Static positioning given by a fraction of the viewport size. For example, if you call set↩ Coords(0, 0), the position will be at the top < left corner of the viewport/widget. setCoords(1, 1) will be at the bottom right corner, setCoords(0.5, 0) will be horizontally centered and < vertically at the top of the viewport/widget, etc.

ptAxisRectRatio Static positioning given by a fraction of the axis rect size (see setAxisRect). For example, if you call setCoords(0, 0), the position will be at the top < left corner of the axis rect. setCoords(1, 1) will be at the bottom right corner, setCoords(0.5, 0) will be horizontally centered and < vertically at the top of the axis rect, etc. You can also go beyond the axis rect by providing negative coordinates or coordinates larger than 1.

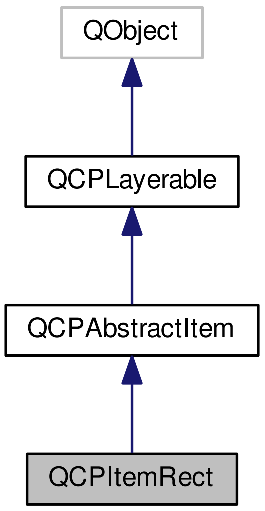
ptPlotCoords Dynamic positioning at a plot coordinate defined by two axes (see setAxes).

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

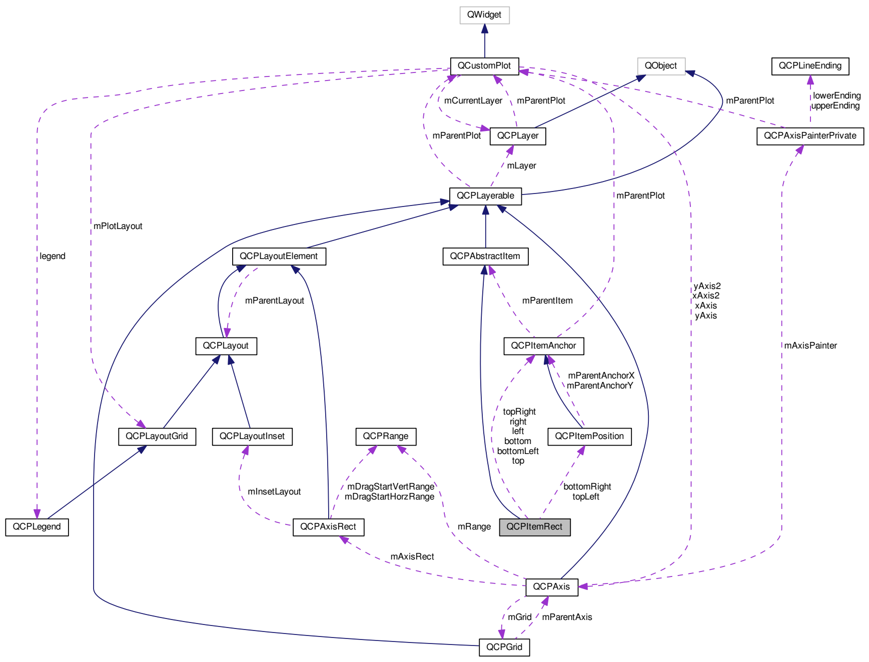
- src/qcustomplot/qcustomplot.h

6.38 QCPItemRect Class Reference

Inheritance diagram for QCPItemRect:



Collaboration diagram for QCPItemRect:



Public Member Functions

- **QCPItemRect** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- QBrush **brush** () const
- QBrush **selectedBrush** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setSelectedBrush** (const QBrush &brush)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **topLeft**
- QCPItemPosition ∗const **bottomRight**
- QCPItemAnchor ∗const **top**
- QCPItemAnchor ∗const **topRight**
- QCPItemAnchor ∗const **right**
- QCPItemAnchor ∗const **bottom**
- QCPItemAnchor ∗const **bottomLeft**
- QCPItemAnchor ∗const **left**

Protected Types

- enum **AnchorIndex** {
  aiTop, **aiTopRight**, **aiRight**, **aiBottom**,
  aiBottomLeft, **aiLeft** }

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- QPen **mainPen** () const
- QBrush **mainBrush** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- QBrush **mBrush**
- QBrush **mSelectedBrush**

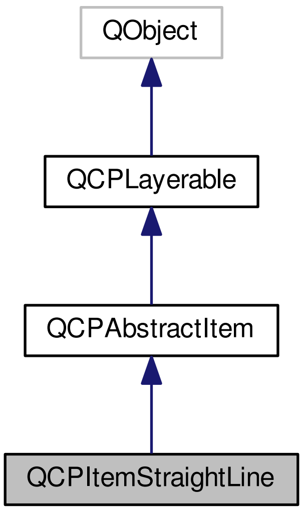
Additional Inherited Members

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

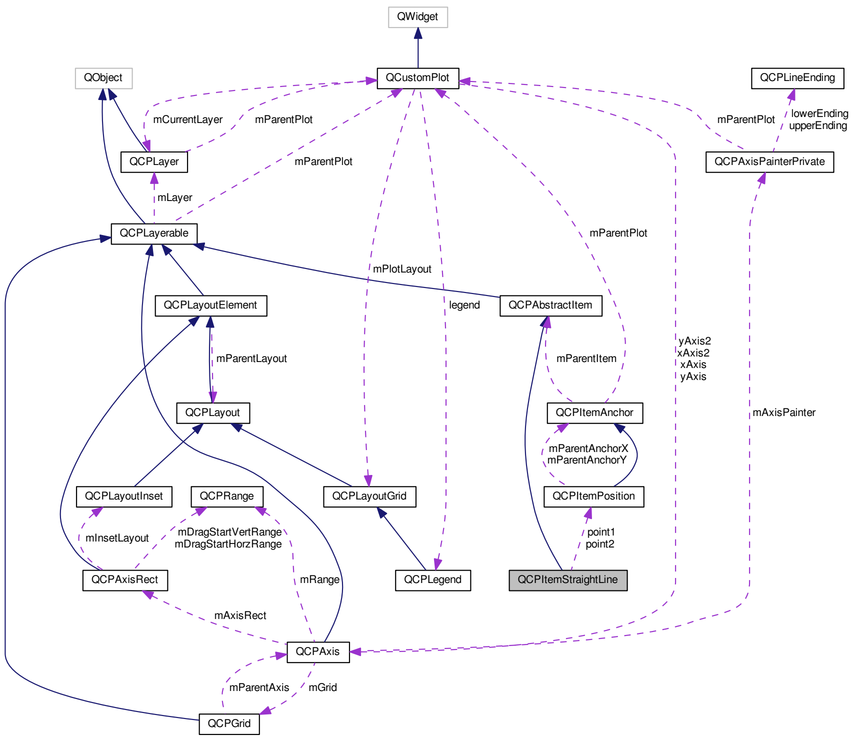
- src/qcustomplot/qcustomplot.h

6.39 QCPItemStraightLine Class Reference

Inheritance diagram for QCPItemStraightLine:



Collaboration diagram for QCPItemStraightLine:



Public Member Functions

- **QCPItemStraightLine** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **point1**
- QCPItemPosition ∗const **point2**

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- double **distToStraightLine** (const QVector2D &point1, const QVector2D &vec, const QVector2D &point) const
- QLineF **getRectClippedStraightLine** (const QVector2D &point1, const QVector2D &vec, const QRect &rect) const
- QPen **mainPen** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**

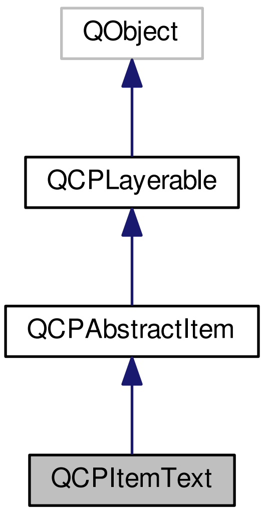
Additional Inherited Members

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

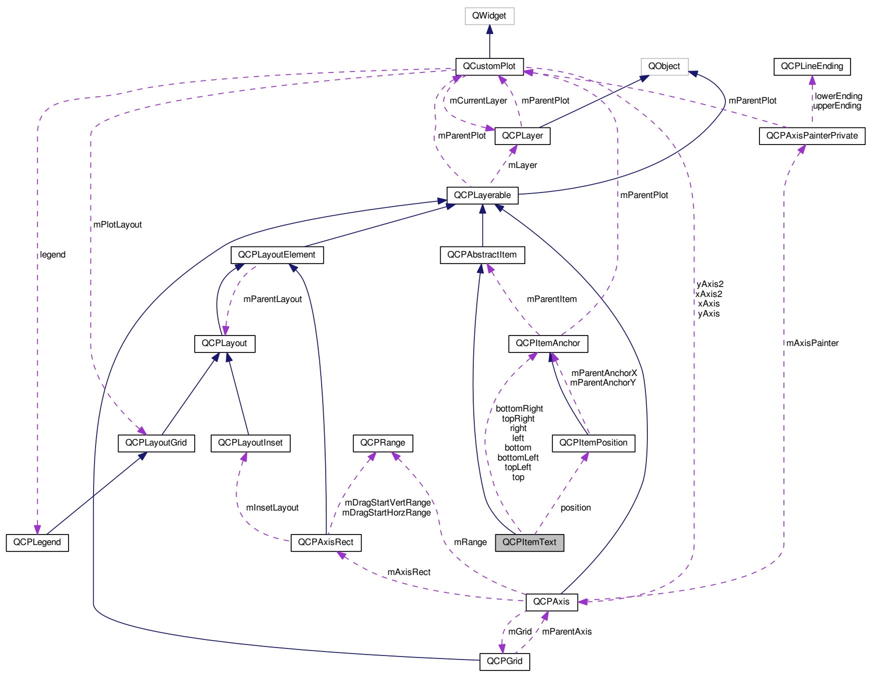
- src/qcustomplot/qcustomplot.h

6.40 QCPItemText Class Reference

Inheritance diagram for QCPItemText:



Collaboration diagram for QCPItemText:



Public Member Functions

- **QCPItemText** (QCustomPlot ∗parentPlot)
- QColor **color** () const
- QColor **selectedColor** () const
- QPen **pen** () const
- QPen **selectedPen** () const
- QBrush **brush** () const
- QBrush **selectedBrush** () const
- QFont **font** () const
- QFont **selectedFont** () const
- QString **text** () const
- Qt::Alignment **positionAlignment** () const
- Qt::Alignment **textAlignment** () const
- double **rotation** () const
- QMargins **padding** () const
- void **setColor** (const QColor &color)
- void **setSelectedColor** (const QColor &color)
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setSelectedBrush** (const QBrush &brush)
- void **setFont** (const QFont &font)
- void **setSelectedFont** (const QFont &font)
- void **setText** (const QString &text)
- void **setPositionAlignment** (Qt::Alignment alignment)
- void **setTextAlignment** (Qt::Alignment alignment)
- void **setRotation** (double degrees)
- void **setPadding** (const QMargins &padding)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Public Attributes

- QCPItemPosition ∗const **position**
- QCPItemAnchor ∗const **topLeft**
- QCPItemAnchor ∗const **top**
- QCPItemAnchor ∗const **topRight**
- QCPItemAnchor ∗const **right**
- QCPItemAnchor ∗const **bottomRight**
- QCPItemAnchor ∗const **bottom**
- QCPItemAnchor ∗const **bottomLeft**
- QCPItemAnchor ∗const **left**

Protected Types

- enum **AnchorIndex** {
  aiTopLeft, **aiTop**, **aiTopRight**, **aiRight**,
  aiBottomRight, **aiBottom**, **aiBottomLeft**, **aiLeft** }

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QPointF **anchorPixelPoint** (int anchorId) const
- QPointF **getTextDrawPoint** (const QPointF &pos, const QRectF &rect, Qt::Alignment positionAlignment) const
- QFont **mainFont** () const
- QColor **mainColor** () const
- QPen **mainPen** () const
- QBrush **mainBrush** () const

Protected Attributes

- QColor **mColor**
- QColor **mSelectedColor**
- QPen **mPen**
- QPen **mSelectedPen**
- QBrush **mBrush**
- QBrush **mSelectedBrush**
- QFont **mFont**
- QFont **mSelectedFont**
- QString **mText**
- Qt::Alignment **mPositionAlignment**
- Qt::Alignment **mTextAlignment**
- double **mRotation**
- QMargins **mPadding**

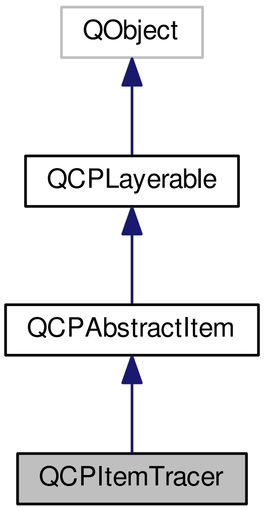
Additional Inherited Members

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

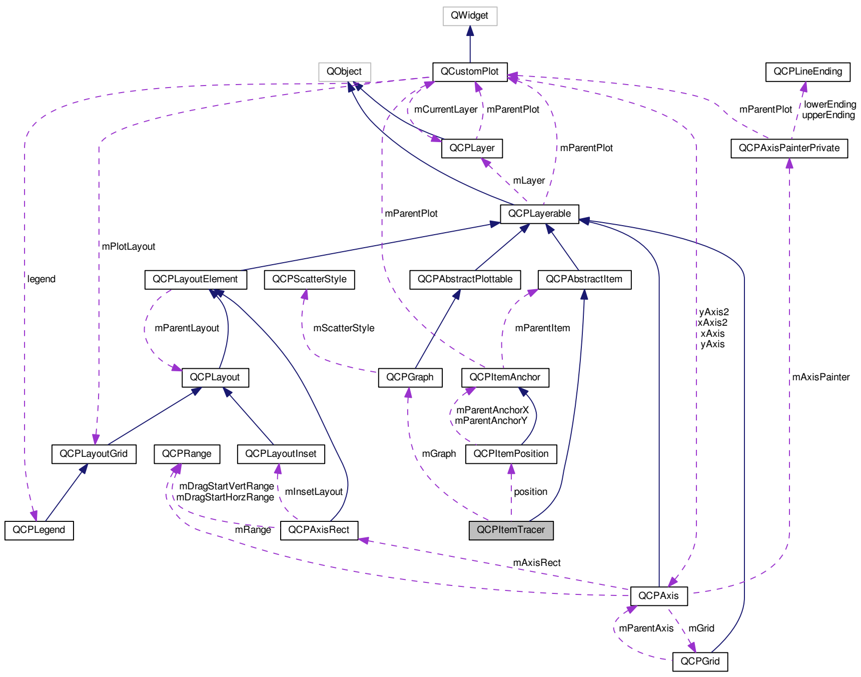
- src/qcustomplot/qcustomplot.h

6.41 QCPItemTracer Class Reference

Inheritance diagram for QCPItemTracer:



Collaboration diagram for QCPItemTracer:



Public Types

- enum TracerStyle {
  tsNone, tsPlus, tsCrosshair, tsCircle,
  tsSquare }

Public Member Functions

- **QCPItemTracer** (QCustomPlot ∗parentPlot)
- QPen **pen** () const
- QPen **selectedPen** () const
- QBrush **brush** () const
- QBrush **selectedBrush** () const
- double **size** () const
- TracerStyle **style** () const
- QCPGraph ∗ **graph** () const
- double **graphKey** () const
- bool **interpolating** () const
- void **setPen** (const QPen &pen)
- void **setSelectedPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setSelectedBrush** (const QBrush &brush)
- void **setSize** (double size)
- void **setStyle** (TracerStyle style)
- void **setGraph** (QCPGraph ∗graph)
- void **setGraphKey** (double key)
- void **setInterpolating** (bool enabled)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- void **updatePosition** ()

Public Attributes

- QCPItemPosition ∗const **position**

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- QPen **mainPen** () const
- QBrush **mainBrush** () const

Protected Attributes

- QPen **mPen**
- QPen **mSelectedPen**
- QBrush **mBrush**
- QBrush **mSelectedBrush**
- double **mSize**
- TracerStyle **mStyle**
- QCPGraph ∗ **mGraph**
- double **mGraphKey**
- bool **mInterpolating**

Additional Inherited Members

6.41.1 Member Enumeration Documentation

6.41.1.1 enum QCPItemTracer::TracerStyle

The different visual appearances a tracer item can have. Some styles size may be controlled with setSize.

See also

setStyle

Enumerator

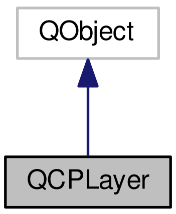
tsNone	The tracer is not visible.
tsPlus	A plus shaped crosshair with limited size.
tsCrosshair	A plus shaped crosshair which spans the complete axis rect.
tsCircle	A circle.
tsSquare	A square.

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

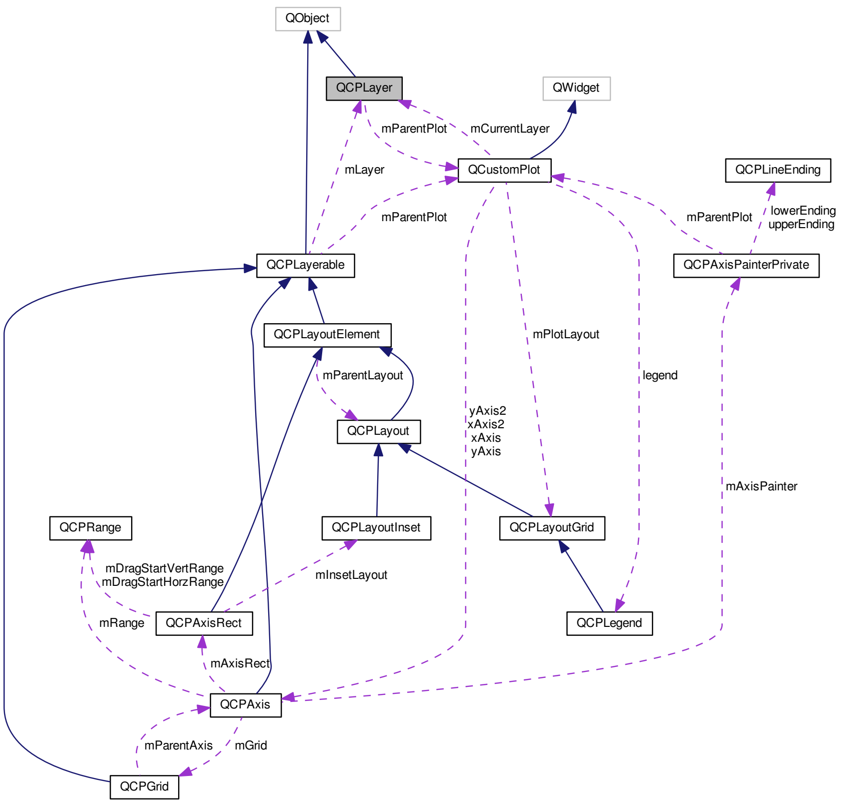
- src/qcustomplot/qcustomplot.h

6.42 QCPLayer Class Reference

Inheritance diagram for QCPLayer:



Collaboration diagram for QCPLayer:



Public Member Functions

- **QCPLayer** (QCustomPlot ∗parentPlot, const QString &layerName)
- QCustomPlot ∗ **parentPlot** () const
- QString **name** () const
- int **index** () const
- QList< QCPLayerable ∗ > **children** () const
- bool **visible** () const
- void **setVisible** (bool visible)

Protected Member Functions

- void **addChild** (QCPLayerable ∗layerable, bool prepend)
- void **removeChild** (QCPLayerable ∗layerable)

Protected Attributes

- QCustomPlot ∗ **mParentPlot**
- QString **mName**
- int **mIndex**
- QList< QCPLayerable ∗ > **mChildren**
- bool **mVisible**

Friends

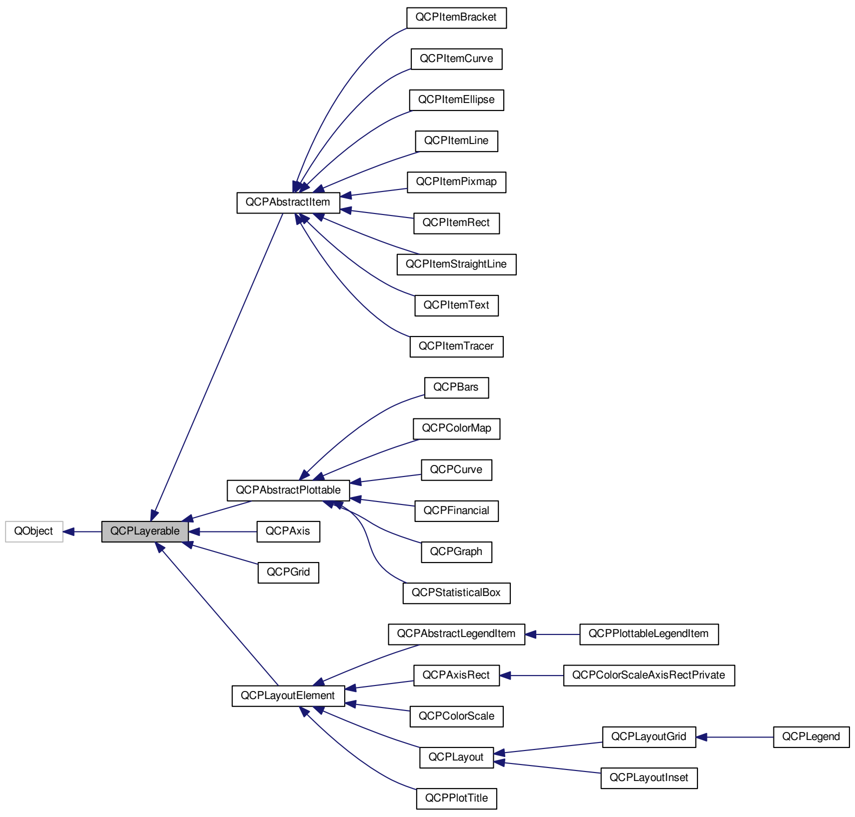
- class **QCustomPlot**
- class **QCPLayerable**

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

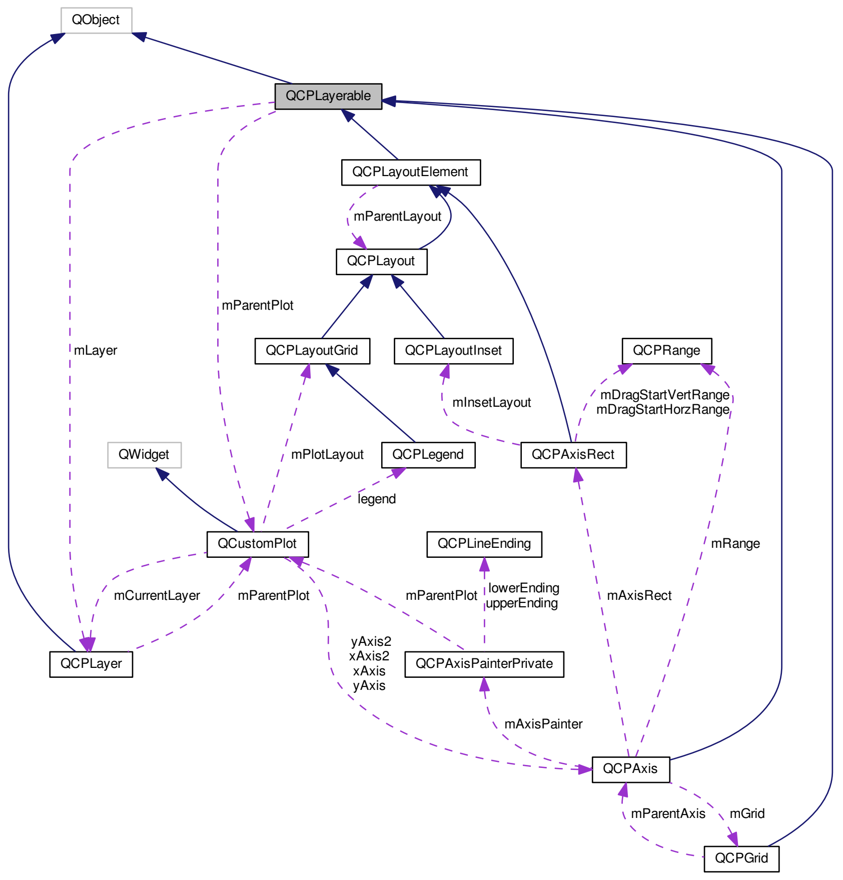
- src/qcustomplot/qcustomplot.h

6.43 QCPLayerable Class Reference

Inheritance diagram for QCPLayerable:



Collaboration diagram for QCPLayerable:



Signals

- void **layerChanged** (QCPLayer ∗newLayer)

Public Member Functions

- **QCPLayerable** (QCustomPlot ∗plot, QString targetLayer=QString(), QCPLayerable ∗parentLayerable=0)
- bool **visible** () const
- QCustomPlot ∗ **parentPlot** () const
- QCPLayerable ∗ **parentLayerable** () const
- QCPLayer ∗ **layer** () const
- bool **antialiased** () const
- void **setVisible** (bool on)
- Q_SLOT bool **setLayer** (QCPLayer ∗layer)
- bool **setLayer** (const QString &layerName)
- void **setAntialiased** (bool enabled)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- bool **realVisibility** () const

Protected Member Functions

- virtual void **parentPlotInitialized** (QCustomPlot ∗parentPlot)
- virtual QCP::Interaction **selectionCategory** () const
- virtual QRect **clipRect** () const
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const =0
- virtual void **draw** (QCPPainter ∗painter)=0
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←↩ Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- void **initializeParentPlot** (QCustomPlot ∗parentPlot)
- void **setParentLayerable** (QCPLayerable ∗parentLayerable)
- bool **moveToLayer** (QCPLayer ∗layer, bool prepend)
- void **applyAntialiasingHint** (QCPPainter ∗painter, bool localAntialiased, QCP::AntialiasedElement overrideElement) const

Protected Attributes

- bool **mVisible**
- QCustomPlot ∗ **mParentPlot**
- QPointer< QCPLayerable > **mParentLayerable**
- QCPLayer ∗ **mLayer**
- bool **mAntialiased**

Friends

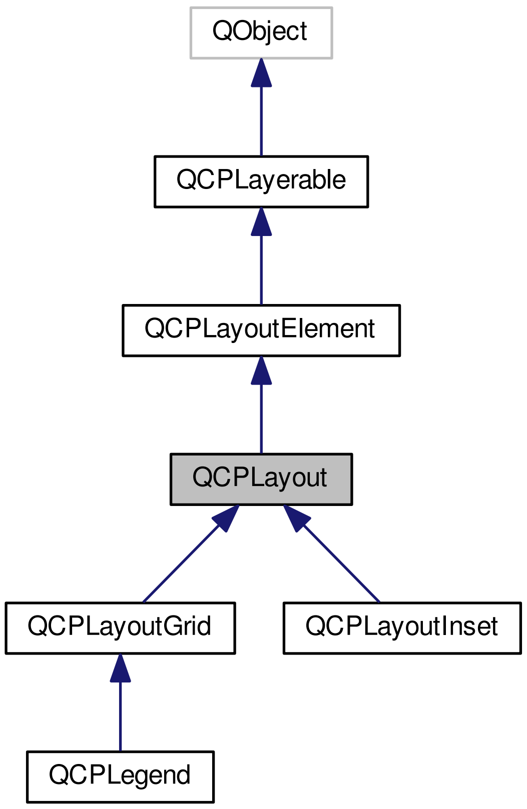
- class **QCustomPlot**
- class **QCPAxisRect**

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

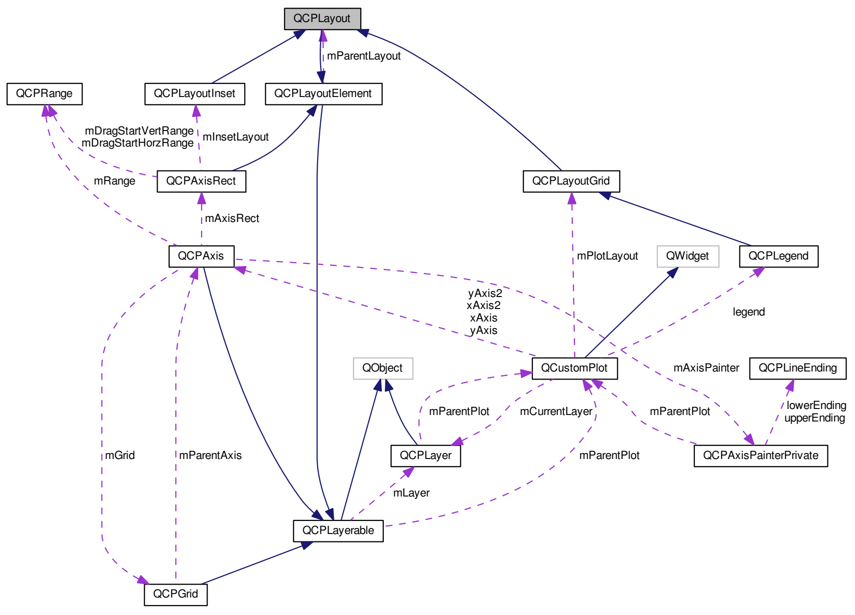
- src/qcustomplot/qcustomplot.h

6.44 QCPLayout Class Reference

Inheritance diagram for QCPLayout:



Collaboration diagram for QCPLayout:



Public Member Functions

- virtual void **update** (UpdatePhase phase)
- virtual QList< QCPLayoutElement ∗ > **elements** (bool recursive) const
- virtual int **elementCount** () const =0
- virtual QCPLayoutElement ∗ **elementAt** (int index) const =0
- virtual QCPLayoutElement ∗ **takeAt** (int index)=0
- virtual bool **take** (QCPLayoutElement ∗element)=0
- virtual void **simplify** ()
- bool **removeAt** (int index)
- bool **remove** (QCPLayoutElement ∗element)
- void **clear** ()

Protected Member Functions

- virtual void **updateLayout** ()
- void **sizeConstraintsChanged** () const
- void **adoptElement** (QCPLayoutElement ∗el)
- void **releaseElement** (QCPLayoutElement ∗el)
- QVector< int > **getSectionSizes** (QVector< int > maxSizes, QVector< int > minSizes, QVector< double > stretchFactors, int totalSize) const

Friends

- class **QCPLayoutElement**

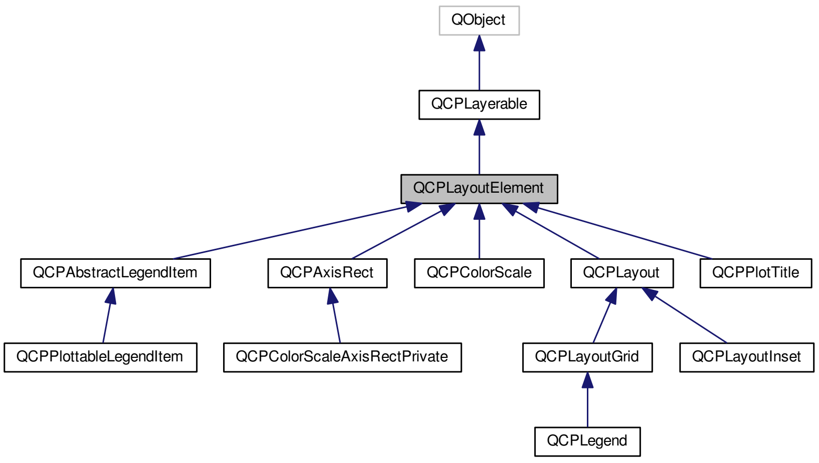
Additional Inherited Members

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

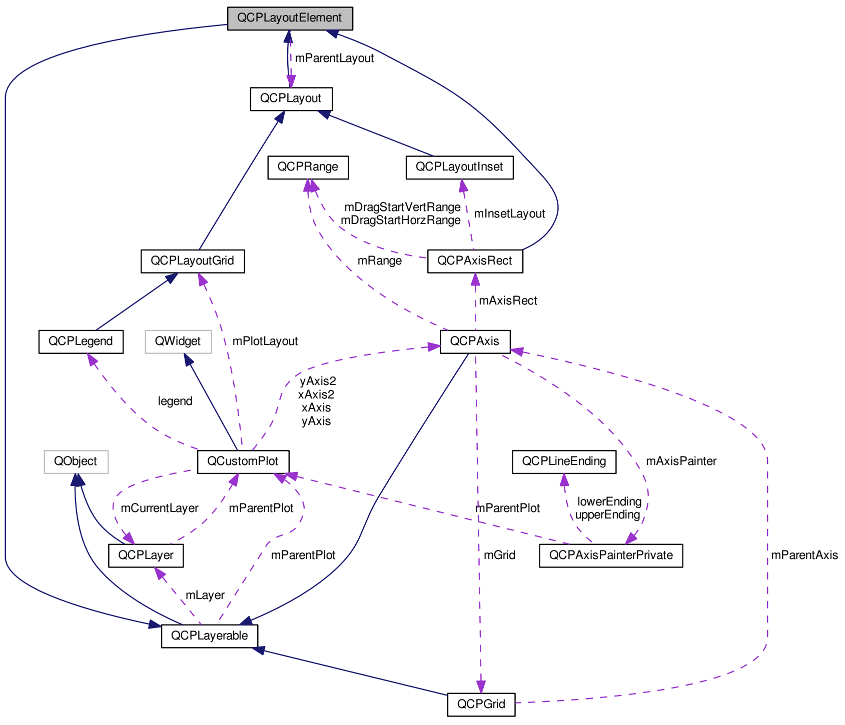
- src/qcustomplot/qcustomplot.h

6.45 QCPLayoutElement Class Reference

Inheritance diagram for QCPLayoutElement:



Collaboration diagram for QCPLayoutElement:



Public Types

- enum UpdatePhase { upPreparation, upMargins, upLayout }

Public Member Functions

- **QCPLayoutElement** (QCustomPlot ∗parentPlot=0)
- QCPLayout ∗ **layout** () const
- QRect **rect** () const
- QRect **outerRect** () const
- QMargins **margins** () const
- QMargins **minimumMargins** () const
- QCP::MarginSides **autoMargins** () const
- QSize **minimumSize** () const
- QSize **maximumSize** () const
- QCPMarginGroup ∗ **marginGroup** (QCP::MarginSide side) const
- QHash< QCP::MarginSide, QCPMarginGroup ∗ > **marginGroups** () const
- void **setOuterRect** (const QRect &rect)
- void **setMargins** (const QMargins &margins)
- void **setMinimumMargins** (const QMargins &margins)
- void **setAutoMargins** (QCP::MarginSides sides)
- void **setMinimumSize** (const QSize &size)
- void **setMinimumSize** (int width, int height)
- void **setMaximumSize** (const QSize &size)
- void **setMaximumSize** (int width, int height)

- void **setMarginGroup** (QCP::MarginSides sides, QCPMarginGroup ∗group)
- virtual void **update** (UpdatePhase phase)
- virtual QSize **minimumSizeHint** () const
- virtual QSize **maximumSizeHint** () const
- virtual QList< QCPLayoutElement ∗ > **elements** (bool recursive) const
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual int **calculateAutoMargin** (QCP::MarginSide side)
- virtual void **mousePressEvent** (QMouseEvent ∗event)
- virtual void **mouseMoveEvent** (QMouseEvent ∗event)
- virtual void **mouseReleaseEvent** (QMouseEvent ∗event)
- virtual void **mouseDoubleClickEvent** (QMouseEvent ∗event)
- virtual void **wheelEvent** (QWheelEvent ∗event)
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- virtual void **parentPlotInitialized** (QCustomPlot ∗parentPlot)

Protected Attributes

- QCPLayout ∗ **mParentLayout**
- QSize **mMinimumSize**
- QSize **mMaximumSize**
- QRect **mRect**
- QRect **mOuterRect**
- QMargins **mMargins**
- QMargins **mMinimumMargins**
- QCP::MarginSides **mAutoMargins**
- QHash< QCP::MarginSide, QCPMarginGroup ∗ > **mMarginGroups**

Friends

- class **QCustomPlot**
- class **QCPLayout**
- class **QCPMarginGroup**

Additional Inherited Members

6.45.1 Member Enumeration Documentation

6.45.1.1 enum QCPLayoutElement::UpdatePhase

Defines the phases of the update process, that happens just before a replot. At each phase, update is called with the according UpdatePhase value.

Enumerator

upPreparation Phase used for any type of preparation that needs to be done before margin calculation and layout.

upMargins Phase in which the margins are calculated and set.

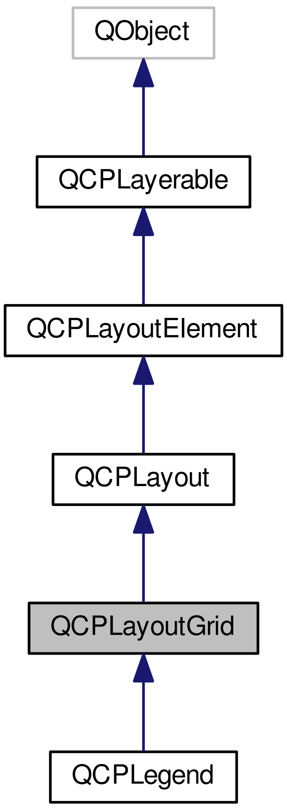
upLayout Final phase in which the layout system places the rects of the elements.

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

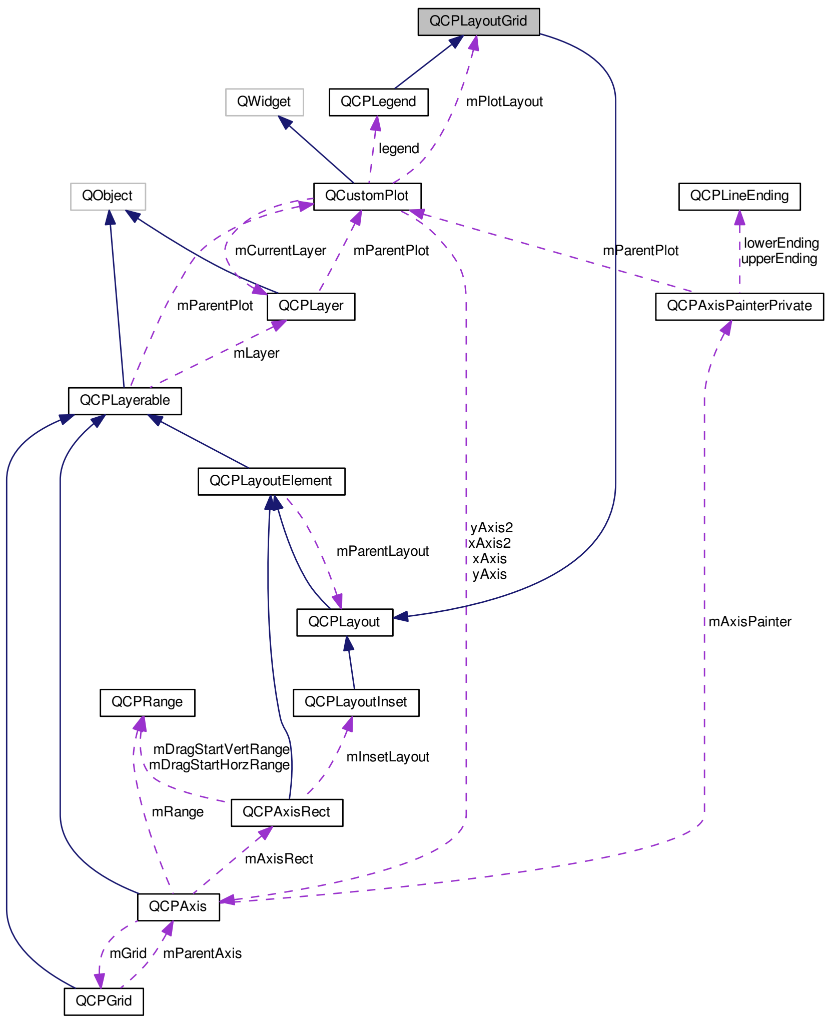
- src/qcustomplot/qcustomplot.h

6.46 QCPLayoutGrid Class Reference

Inheritance diagram for QCPLayoutGrid:



Collaboration diagram for QCPLayoutGrid:



Public Member Functions

- int **rowCount** () const
- int **columnCount** () const
- QList< double > **columnStretchFactors** () const
- QList< double > **rowStretchFactors** () const
- int **columnSpacing** () const
- int **rowSpacing** () const
- void **setColumnStretchFactor** (int column, double factor)
- void **setColumnStretchFactors** (const QList< double > &factors)
- void **setRowStretchFactor** (int row, double factor)
- void **setRowStretchFactors** (const QList< double > &factors)
- void **setColumnSpacing** (int pixels)
- void **setRowSpacing** (int pixels)
- virtual void **updateLayout** ()
- virtual int **elementCount** () const
- virtual QCPLayoutElement ∗ **elementAt** (int index) const

- virtual QCPLayoutElement ∗ **takeAt** (int index)
- virtual bool **take** (QCPLayoutElement ∗element)
- virtual QList< QCPLayoutElement ∗ > **elements** (bool recursive) const
- virtual void **simplify** ()
- virtual QSize **minimumSizeHint** () const
- virtual QSize **maximumSizeHint** () const
- QCPLayoutElement ∗ **element** (int row, int column) const
- bool **addElement** (int row, int column, QCPLayoutElement ∗element)
- bool **hasElement** (int row, int column)
- void **expandTo** (int newRowCount, int newColumnCount)
- void **insertRow** (int newIndex)
- void **insertColumn** (int newIndex)

Protected Member Functions

- void **getMinimumRowColSizes** (QVector< int > ∗minColWidths, QVector< int > ∗minRowHeights) const
- void **getMaximumRowColSizes** (QVector< int > ∗maxColWidths, QVector< int > ∗maxRowHeights) const

Protected Attributes

- QList< QList< QCPLayoutElement ∗ > > **mElements**
- QList< double > **mColumnStretchFactors**
- QList< double > **mRowStretchFactors**
- int **mColumnSpacing**
- int **mRowSpacing**

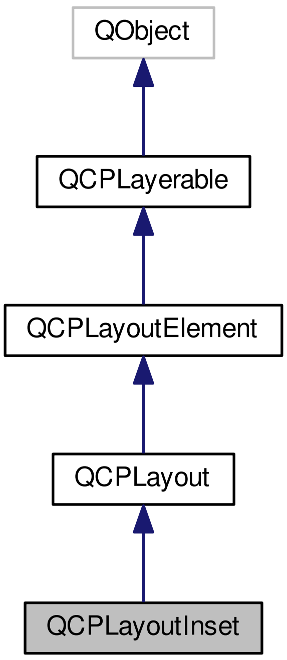
Additional Inherited Members

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

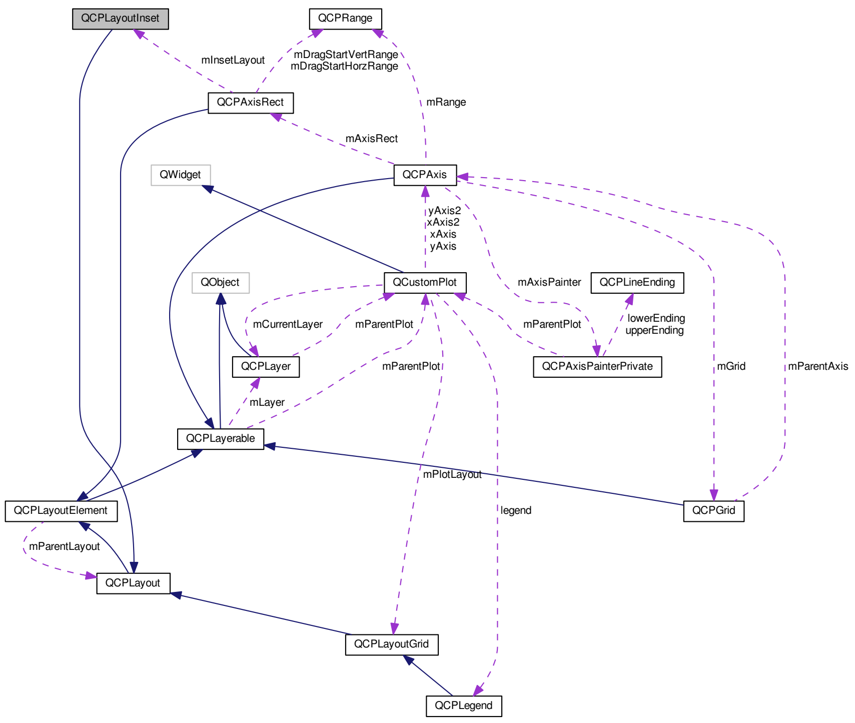
- src/qcustomplot/qcustomplot.h

6.47 QCPLayoutInset Class Reference

Inheritance diagram for QCPLayoutInset:



Collaboration diagram for QCPLayoutInset:



Public Types

- enum InsetPlacement { ipFree, ipBorderAligned }

Public Member Functions

- InsetPlacement **insetPlacement** (int index) const
- Qt::Alignment **insetAlignment** (int index) const
- QRectF **insetRect** (int index) const
- void **setInsetPlacement** (int index, InsetPlacement placement)
- void **setInsetAlignment** (int index, Qt::Alignment alignment)
- void **setInsetRect** (int index, const QRectF &rect)
- virtual void **updateLayout** ()
- virtual int **elementCount** () const
- virtual QCPLayoutElement ∗ **elementAt** (int index) const
- virtual QCPLayoutElement ∗ **takeAt** (int index)
- virtual bool **take** (QCPLayoutElement ∗element)
- virtual void **simplify** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- void **addElement** (QCPLayoutElement ∗element, Qt::Alignment alignment)
- void **addElement** (QCPLayoutElement ∗element, const QRectF &rect)

Protected Attributes

- QList< QCPLayoutElement * > **mElements**
- QList< InsetPlacement > **mInsetPlacement**
- QList< Qt::Alignment > **mInsetAlignment**
- QList< QRectF > **mInsetRect**

Additional Inherited Members

6.47.1 Member Enumeration Documentation

6.47.1.1 enum QCPLayoutInset::InsetPlacement

Defines how the placement and sizing is handled for a certain element in a QCPLayoutInset.

Enumerator

ipFree The element may be positioned/sized arbitrarily, see setInsetRect.

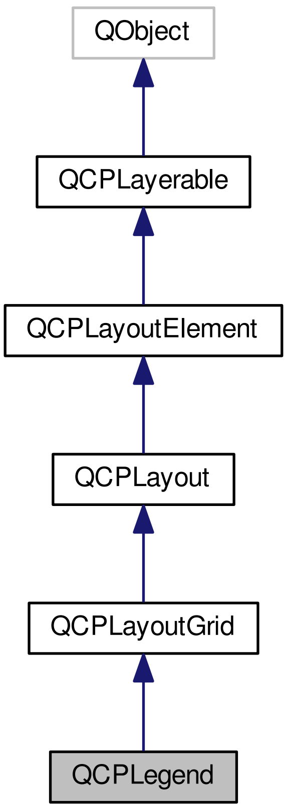
ipBorderAligned The element is aligned to one of the layout sides, see setInsetAlignment.

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

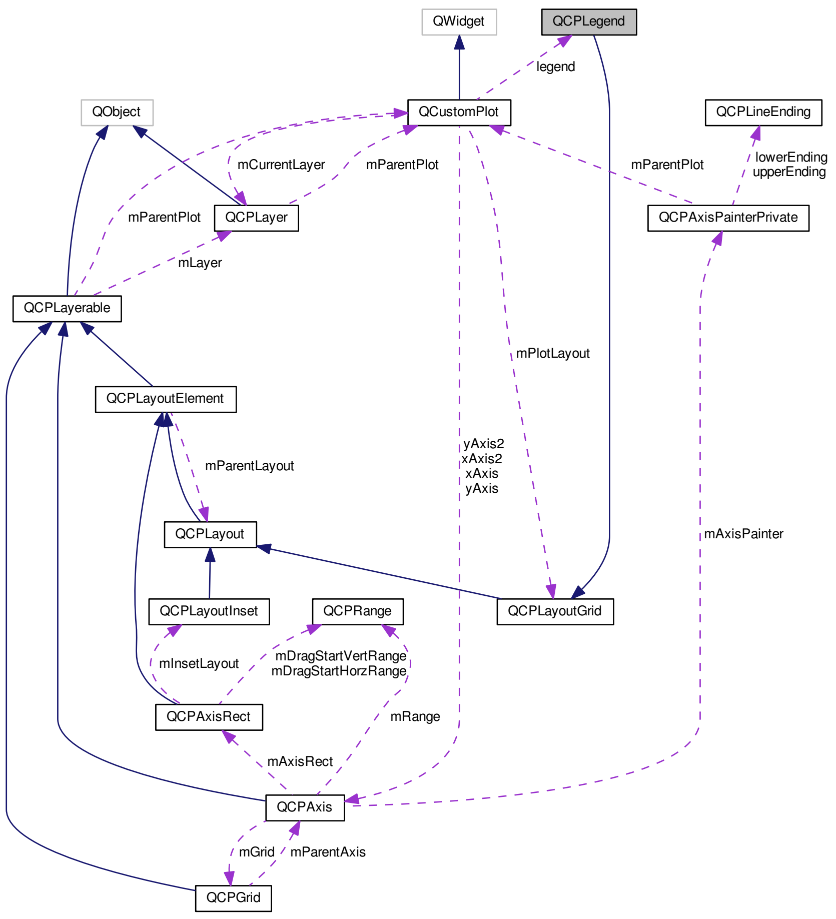
- src/qcustomplot/qcustomplot.h

6.48 QCPLegend Class Reference

Inheritance diagram for QCPLegend:



Collaboration diagram for QCPLegend:



Public Types

- enum SelectablePart { spNone = 0x000, spLegendBox = 0x001, spItems = 0x002 }

Signals

- void **selectionChanged** (QCPLegend::SelectableParts parts)
- void **selectableChanged** (QCPLegend::SelectableParts parts)

Public Member Functions

- QPen **borderPen** () const
- QBrush **brush** () const
- QFont **font** () const
- QColor **textColor** () const
- QSize **iconSize** () const
- int **iconTextPadding** () const
- QPen **iconBorderPen** () const

- SelectableParts **selectableParts** () const
- SelectableParts **selectedParts** () const
- QPen **selectedBorderPen** () const
- QPen **selectedIconBorderPen** () const
- QBrush **selectedBrush** () const
- QFont **selectedFont** () const
- QColor **selectedTextColor** () const
- void **setBorderPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setFont** (const QFont &font)
- void **setTextColor** (const QColor &color)
- void **setIconSize** (const QSize &size)
- void **setIconSize** (int width, int height)
- void **setIconTextPadding** (int padding)
- void **setIconBorderPen** (const QPen &pen)
- Q_SLOT void **setSelectableParts** (const SelectableParts &selectableParts)
- Q_SLOT void **setSelectedParts** (const SelectableParts &selectedParts)
- void **setSelectedBorderPen** (const QPen &pen)
- void **setSelectedIconBorderPen** (const QPen &pen)
- void **setSelectedBrush** (const QBrush &brush)
- void **setSelectedFont** (const QFont &font)
- void **setSelectedTextColor** (const QColor &color)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const
- QCPAbstractLegendItem ∗ **item** (int index) const
- QCPPlottableLegendItem ∗ **itemWithPlottable** (const QCPAbstractPlottable ∗plottable) const
- int **itemCount** () const
- bool **hasItem** (QCPAbstractLegendItem ∗item) const
- bool **hasItemWithPlottable** (const QCPAbstractPlottable ∗plottable) const
- bool **addItem** (QCPAbstractLegendItem ∗item)
- bool **removeItem** (int index)
- bool **removeItem** (QCPAbstractLegendItem ∗item)
- void **clearItems** ()
- QList< QCPAbstractLegendItem ∗ > **selectedItems** () const

Protected Member Functions

- virtual void **parentPlotInitialized** (QCustomPlot ∗parentPlot)
- virtual QCP::Interaction **selectionCategory** () const
- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←
Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- QPen **getBorderPen** () const
- QBrush **getBrush** () const

Protected Attributes

- QPen **mBorderPen**
- QPen **mIconBorderPen**
- QBrush **mBrush**
- QFont **mFont**
- QColor **mTextColor**
- QSize **mIconSize**
- int **mIconTextPadding**
- SelectableParts **mSelectedParts**
- SelectableParts **mSelectableParts**
- QPen **mSelectedBorderPen**
- QPen **mSelectedIconBorderPen**
- QBrush **mSelectedBrush**
- QFont **mSelectedFont**
- QColor **mSelectedTextColor**

Friends

- class **QCustomPlot**
- class **QCPAbstractLegendItem**

6.48.1 Member Enumeration Documentation

6.48.1.1 enum QCPLegend::SelectablePart

Defines the selectable parts of a legend

See also

setSelectedParts, setSelectableParts

Enumerator

spNone `0x000` None

spLegendBox `0x001` The legend box (frame)

spItems `0x002` Legend items individually (see selectedItems)

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

- src/qcustomplot/qcustomplot.h

6.49 QCPLineEnding Class Reference

Public Types

- enum EndingStyle {
  esNone, esFlatArrow, esSpikeArrow, esLineArrow,
  esDisc, esSquare, esDiamond, esBar,
  esHalfBar, esSkewedBar }

Public Member Functions

- **QCPLineEnding** (EndingStyle style, double width=8, double length=10, bool inverted=false)
- EndingStyle **style** () const
- double **width** () const
- double **length** () const
- bool **inverted** () const
- void **setStyle** (EndingStyle style)
- void **setWidth** (double width)
- void **setLength** (double length)
- void **setInverted** (bool inverted)
- double **boundingDistance** () const
- double **realLength** () const
- void **draw** (QCPPainter ∗painter, const QVector2D &pos, const QVector2D &dir) const
- void **draw** (QCPPainter ∗painter, const QVector2D &pos, double angle) const

Protected Attributes

- EndingStyle **mStyle**
- double **mWidth**
- double **mLength**
- bool **mInverted**

6.49.1 Member Enumeration Documentation

6.49.1.1 enum QCPLineEnding::EndingStyle

Defines the type of ending decoration for line-like items, e.g. an arrow.

The width and length of these decorations can be controlled with the functions setWidth and setLength. Some decorations like esDisc, esSquare, esDiamond and esBar only support a width, the length property is ignored.

See also

QCPItemLine::setHead, QCPItemLine::setTail, QCPItemCurve::setHead, QCPItemCurve::setTail, QCPAxis↩ ::setLowerEnding, QCPAxis::setUpperEnding

Enumerator

esNone No ending decoration.

esFlatArrow A filled arrow head with a straight/flat back (a triangle)

esSpikeArrow A filled arrow head with an indented back.

esLineArrow A non-filled arrow head with open back.

esDisc A filled circle.

esSquare A filled square.

esDiamond A filled diamond (45° rotated square)

esBar A bar perpendicular to the line.

esHalfBar A bar perpendicular to the line, pointing out to only one side (to which side can be changed with setInverted)

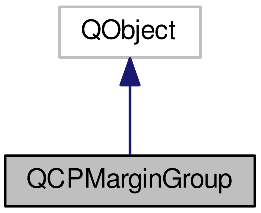
esSkewedBar A bar that is skewed (skew controllable via setLength)

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

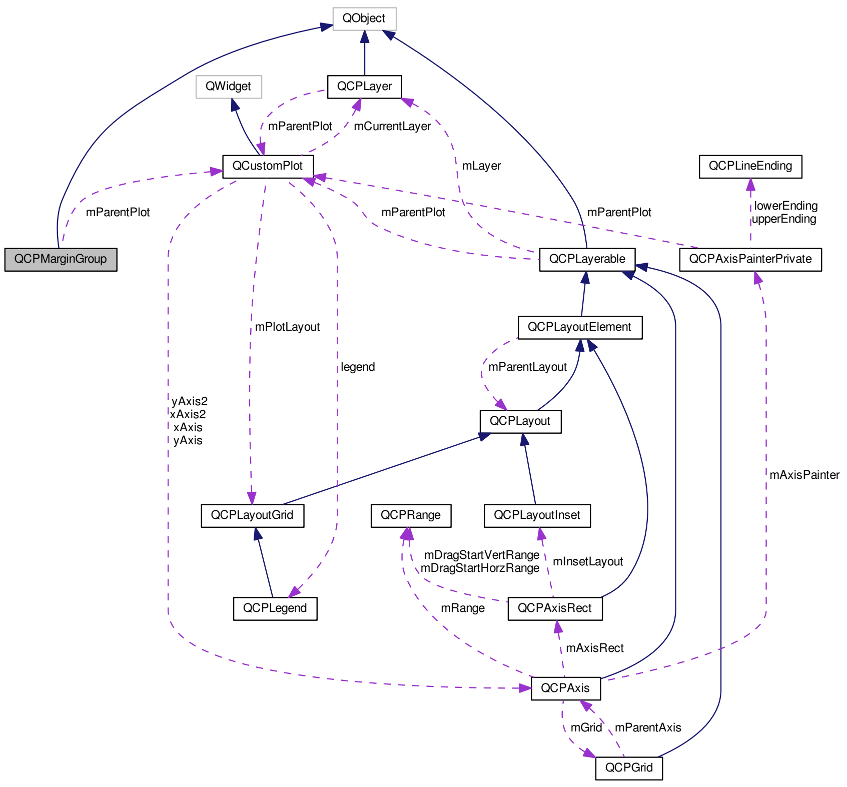
- src/qcustomplot/qcustomplot.h

6.50 QCPMarginGroup Class Reference

Inheritance diagram for QCPMarginGroup:



Collaboration diagram for QCPMarginGroup:



Public Member Functions

- **QCPMarginGroup** (QCustomPlot ∗parentPlot)
- QList< QCPLayoutElement ∗ > **elements** (QCP::MarginSide side) const
- bool **isEmpty** () const
- void **clear** ()

Protected Member Functions

- int **commonMargin** (QCP::MarginSide side) const
- void **addChild** (QCP::MarginSide side, QCPLayoutElement ∗element)
- void **removeChild** (QCP::MarginSide side, QCPLayoutElement ∗element)

Protected Attributes

- QCustomPlot ∗ **mParentPlot**
- QHash< QCP::MarginSide, QList< QCPLayoutElement ∗ > > **mChildren**

Friends

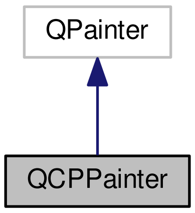
- class **QCPLayoutElement**

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

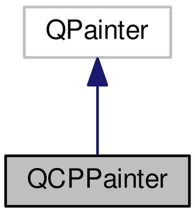
- src/qcustomplot/qcustomplot.h

6.51 QCPPainter Class Reference

Inheritance diagram for QCPPainter:



Collaboration diagram for QCPPainter:



Public Types

- enum PainterMode { pmDefault = 0x00, pmVectorized = 0x01, pmNoCaching = 0x02, pmNonCosmetic = 0x04 }

Public Member Functions

- **QCPPainter** (QPaintDevice ∗device)
- bool **antialiasing** () const
- PainterModes **modes** () const
- void **setAntialiasing** (bool enabled)
- void **setMode** (PainterMode mode, bool enabled=true)
- void **setModes** (PainterModes modes)
- bool **begin** (QPaintDevice ∗device)
- void **setPen** (const QPen &pen)
- void **setPen** (const QColor &color)
- void **setPen** (Qt::PenStyle penStyle)
- void **drawLine** (const QLineF &line)
- void **drawLine** (const QPointF &p1, const QPointF &p2)
- void **save** ()
- void **restore** ()
- void **makeNonCosmetic** ()

Protected Attributes

- PainterModes **mModes**
- bool **mIsAntialiasing**
- QStack< bool > **mAntialiasingStack**

6.51.1 Member Enumeration Documentation

6.51.1.1 enum QCPPainter::PainterMode

Defines special modes the painter can operate in. They disable or enable certain subsets of features/fixes/workarounds, depending on whether they are wanted on the respective output device.

Enumerator

pmDefault `0x00` Default mode for painting on screen devices

pmVectorized `0x01` Mode for vectorized painting (e.g. PDF export). For example, this prevents some antialiasing fixes.

pmNoCaching `0x02` Mode for all sorts of exports (e.g. PNG, PDF,...). For example, this prevents using cached pixmap labels

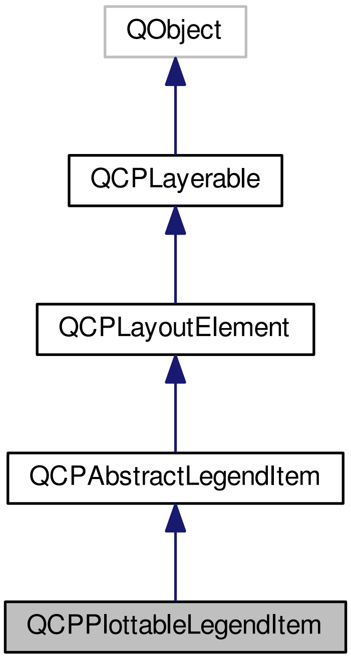
pmNonCosmetic `0x04` Turns pen widths 0 to 1, i.e. disables cosmetic pens. (A cosmetic pen is always drawn with width 1 pixel in the vector image/pdf viewer, independent of zoom.)

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

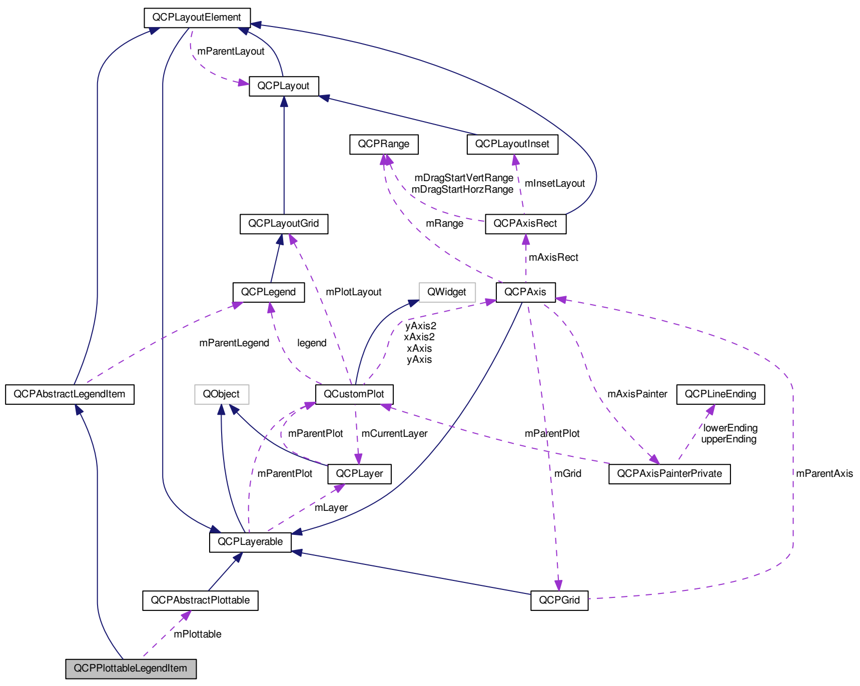
- src/qcustomplot/qcustomplot.h

6.52 QCPPlottableLegendItem Class Reference

Inheritance diagram for QCPPlottableLegendItem:



Collaboration diagram for QCPPlottableLegendItem:



Public Member Functions

- **QCPPlottableLegendItem** (QCPLegend ∗parent, QCPAbstractPlottable ∗plottable)
- QCPAbstractPlottable ∗ **plottable** ()

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual QSize **minimumSizeHint** () const
- QPen **getIconBorderPen** () const
- QColor **getTextColor** () const
- QFont **getFont** () const

Protected Attributes

- QCPAbstractPlottable ∗ **mPlottable**

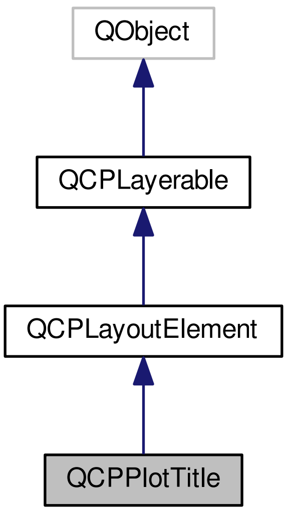
Additional Inherited Members

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

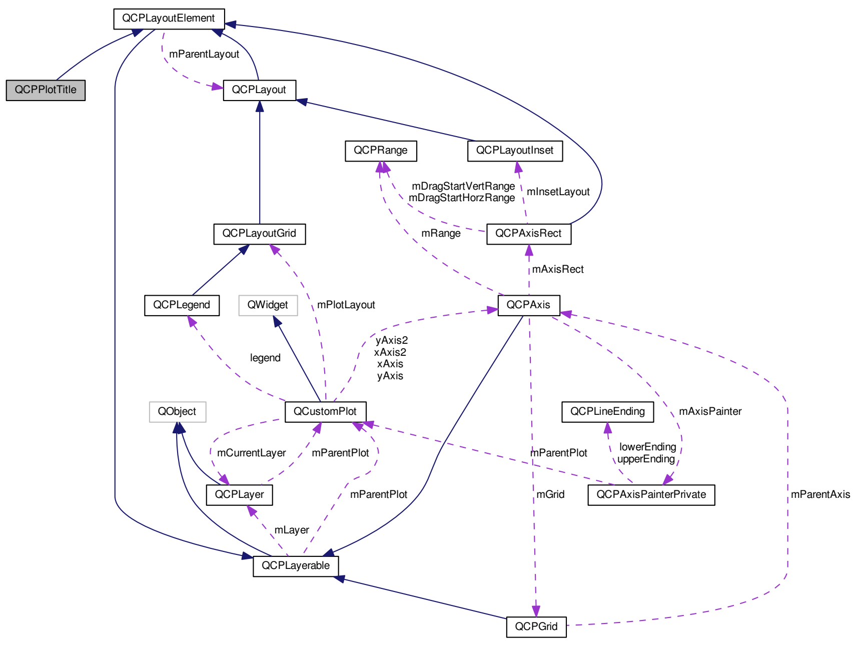
- src/qcustomplot/qcustomplot.h

6.53 QCPPlotTitle Class Reference

Inheritance diagram for QCPPlotTitle:



Collaboration diagram for QCPPlotTitle:



Signals

- void **selectionChanged** (bool selected)
- void **selectableChanged** (bool selectable)

Public Member Functions

- **QCPPlotTitle** (QCustomPlot ∗parentPlot)
- **QCPPlotTitle** (QCustomPlot ∗parentPlot, const QString &text)
- QString **text** () const
- QFont **font** () const
- QColor **textColor** () const
- QFont **selectedFont** () const
- QColor **selectedTextColor** () const
- bool **selectable** () const
- bool **selected** () const
- void **setText** (const QString &text)
- void **setFont** (const QFont &font)
- void **setTextColor** (const QColor &color)
- void **setSelectedFont** (const QFont &font)
- void **setSelectedTextColor** (const QColor &color)
- Q_SLOT void **setSelectable** (bool selectable)
- Q_SLOT void **setSelected** (bool selected)
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual void **applyDefaultAntialiasingHint** (QCPPainter ∗painter) const
- virtual void **draw** (QCPPainter ∗painter)
- virtual QSize **minimumSizeHint** () const
- virtual QSize **maximumSizeHint** () const
- virtual void **selectEvent** (QMouseEvent ∗event, bool additive, const QVariant &details, bool ∗selectionState←↩Changed)
- virtual void **deselectEvent** (bool ∗selectionStateChanged)
- QFont **mainFont** () const
- QColor **mainTextColor** () const

Protected Attributes

- QString **mText**
- QFont **mFont**
- QColor **mTextColor**
- QFont **mSelectedFont**
- QColor **mSelectedTextColor**
- QRect **mTextBoundingRect**
- bool **mSelectable**
- bool **mSelected**

Additional Inherited Members

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

- src/qcustomplot/qcustomplot.h

6.54 QCPRange Class Reference

Public Member Functions

- **QCPRange** (double lower, double upper)
- bool **operator==** (const QCPRange &other) const
- bool **operator!=** (const QCPRange &other) const
- QCPRange & operator+= (const double &value)
- QCPRange & operator-= (const double &value)
- QCPRange & operator∗= (const double &value)
- QCPRange & operator/= (const double &value)
- double **size** () const
- double **center** () const
- void **normalize** ()
- void **expand** (const QCPRange &otherRange)
- QCPRange **expanded** (const QCPRange &otherRange) const
- QCPRange **sanitizedForLogScale** () const
- QCPRange **sanitizedForLinScale** () const
- bool **contains** (double value) const

Static Public Member Functions

- static bool **validRange** (double lower, double upper)
- static bool **validRange** (const QCPRange &range)

Public Attributes

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

Static Public Attributes

- static const double **minRange**
- static const double **maxRange**

Friends

- const QCPRange operator+ (const QCPRange &, double)
- const QCPRange operator+ (double, const QCPRange &)
- const QCPRange operator- (const QCPRange &range, double value)
- const QCPRange operator∗ (const QCPRange &range, double value)
- const QCPRange operator∗ (double value, const QCPRange &range)
- const QCPRange operator/ (const QCPRange &range, double value)

6.54.1 Member Function Documentation

6.54.1.1 QCPRange & QCPRange::operator*= (const double & *value*) `[inline]`

Multiplies both boundaries of the range by *value*.

6.54.1.2 QCPRange & QCPRange::operator+= (const double & *value*) `[inline]`

Adds *value* to both boundaries of the range.

6.54.1.3 QCPRange & QCPRange::operator-= (const double & *value*) `[inline]`

Subtracts *value* from both boundaries of the range.

6.54.1.4 QCPRange & QCPRange::operator/= (const double & *value*) `[inline]`

Divides both boundaries of the range by *value*.

6.54.2 Friends And Related Function Documentation

6.54.2.1 const QCPRange operator* (const QCPRange & *range,* double *value*) `[friend]`

Multiplies both boundaries of the range by *value*.

6.54.2.2 const QCPRange operator* (double *value,* const QCPRange & *range*) `[friend]`

Multiplies both boundaries of the range by *value*.

6.54.2.3 const QCPRange operator+ (const QCPRange & *range,* double *value*) `[friend]`

Adds *value* to both boundaries of the range.

6.54.2.4 const QCPRange operator+ (double *value,* const QCPRange & *range*) `[friend]`

Adds *value* to both boundaries of the range.

6.54.2.5 const QCPRange operator- (const QCPRange & *range,* double *value*) `[friend]`

Subtracts *value* from both boundaries of the range.

6.54.2.6 const QCPRange operator/ (const QCPRange & *range,* double *value*) [friend]

Divides both boundaries of the range by *value*.

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

- src/qcustomplot/qcustomplot.h

6.55 QCPScatterStyle Class Reference

Public Types

- enum ScatterShape {
  ssNone, ssDot, ssCross, ssPlus,
  ssCircle, ssDisc, ssSquare, ssDiamond,
  ssStar, ssTriangle, ssTriangleInverted, ssCrossSquare,
  ssPlusSquare, ssCrossCircle, ssPlusCircle, ssPeace,
  ssPixmap, ssCustom }

Public Member Functions

- **QCPScatterStyle** (ScatterShape shape, double size=6)
- **QCPScatterStyle** (ScatterShape shape, const QColor &color, double size)
- **QCPScatterStyle** (ScatterShape shape, const QColor &color, const QColor &fill, double size)
- **QCPScatterStyle** (ScatterShape shape, const QPen &pen, const QBrush &brush, double size)
- **QCPScatterStyle** (const QPixmap &pixmap)
- **QCPScatterStyle** (const QPainterPath &customPath, const QPen &pen, const QBrush &brush=Qt::NoBrush, double size=6)
- double **size** () const
- ScatterShape **shape** () const
- QPen **pen** () const
- QBrush **brush** () const
- QPixmap **pixmap** () const
- QPainterPath **customPath** () const
- void **setSize** (double size)
- void **setShape** (ScatterShape shape)
- void **setPen** (const QPen &pen)
- void **setBrush** (const QBrush &brush)
- void **setPixmap** (const QPixmap &pixmap)
- void **setCustomPath** (const QPainterPath &customPath)
- bool **isNone** () const
- bool **isPenDefined** () const
- void **applyTo** (QCPPainter ∗painter, const QPen &defaultPen) const
- void **drawShape** (QCPPainter ∗painter, QPointF pos) const
- void **drawShape** (QCPPainter ∗painter, double x, double y) const

Protected Attributes

- double **mSize**
- ScatterShape **mShape**
- QPen **mPen**
- QBrush **mBrush**
- QPixmap **mPixmap**
- QPainterPath **mCustomPath**
- bool **mPenDefined**

6.55.1 Member Enumeration Documentation

6.55.1.1 enum QCPScatterStyle::ScatterShape

Defines the shape used for scatter points.

On plottables/items that draw scatters, the sizes of these visualizations (with exception of ssDot and ssPixmap) can be controlled with the setSize function. Scatters are drawn with the pen and brush specified with setPen and setBrush.

Enumerator

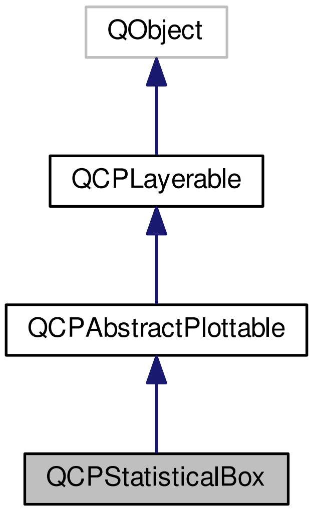
ssNone	no scatter symbols are drawn (e.g. in QCPGraph, data only represented with lines)
ssDot	{ssDot.png} a single pixel (use ssDisc or ssCircle if you want a round shape with a certain radius)
ssCross	{ssCross.png} a cross
ssPlus	{ssPlus.png} a plus
ssCircle	{ssCircle.png} a circle
ssDisc	{ssDisc.png} a circle which is filled with the pen's color (not the brush as with ssCircle)
ssSquare	{ssSquare.png} a square
ssDiamond	{ssDiamond.png} a diamond
ssStar	{ssStar.png} a star with eight arms, i.e. a combination of cross and plus
ssTriangle	{ssTriangle.png} an equilateral triangle, standing on baseline
ssTriangleInverted	{ssTriangleInverted.png} an equilateral triangle, standing on corner
ssCrossSquare	{ssCrossSquare.png} a square with a cross inside
ssPlusSquare	{ssPlusSquare.png} a square with a plus inside
ssCrossCircle	{ssCrossCircle.png} a circle with a cross inside
ssPlusCircle	{ssPlusCircle.png} a circle with a plus inside
ssPeace	{ssPeace.png} a circle, with one vertical and two downward diagonal lines
ssPixmap	a custom pixmap specified by setPixmap, centered on the data point coordinates
ssCustom	custom painter operations are performed per scatter (As QPainterPath, see setCustomPath)

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

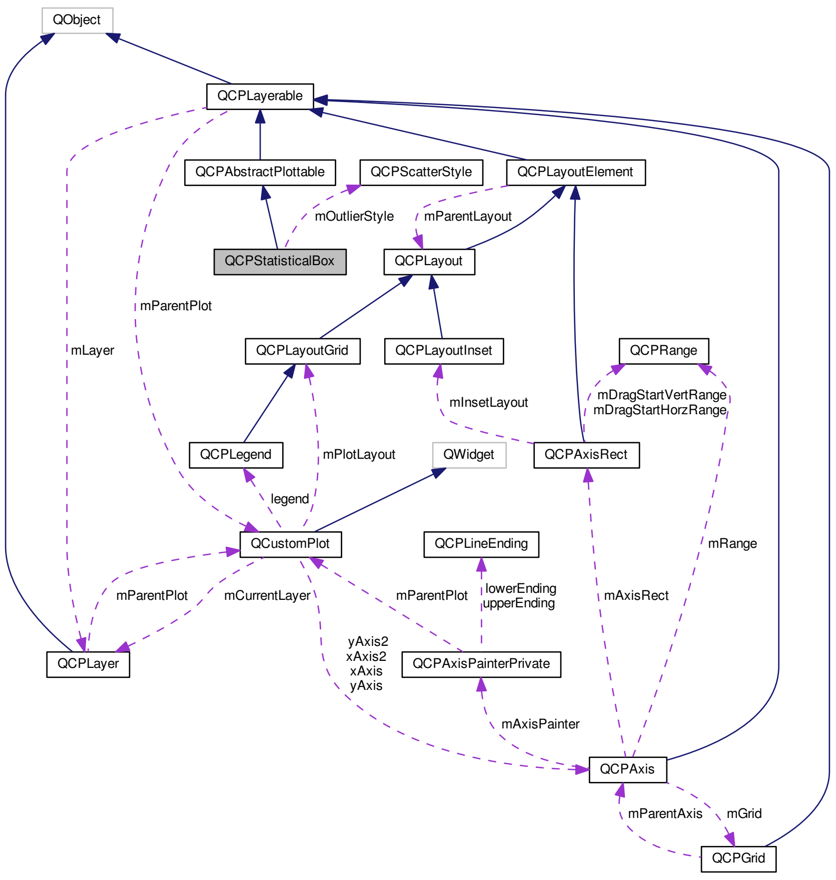
- src/qcustomplot/qcustomplot.h

6.56 QCPStatisticalBox Class Reference

Inheritance diagram for QCPStatisticalBox:



Collaboration diagram for QCPStatisticalBox:



Public Member Functions

- **QCPStatisticalBox** (QCPAxis ∗keyAxis, QCPAxis ∗valueAxis)
- double **key** () const
- double **minimum** () const
- double **lowerQuartile** () const
- double **median** () const
- double **upperQuartile** () const
- double **maximum** () const
- QVector< double > **outliers** () const
- double **width** () const
- double **whiskerWidth** () const
- QPen **whiskerPen** () const
- QPen **whiskerBarPen** () const
- QPen **medianPen** () const
- QCPScatterStyle **outlierStyle** () const
- void **setKey** (double key)
- void **setMinimum** (double value)
- void **setLowerQuartile** (double value)
- void **setMedian** (double value)
- void **setUpperQuartile** (double value)

- void **setMaximum** (double value)
- void **setOutliers** (const QVector< double > &values)
- void **setData** (double key, double minimum, double lowerQuartile, double median, double upperQuartile, double maximum)
- void **setWidth** (double width)
- void **setWhiskerWidth** (double width)
- void **setWhiskerPen** (const QPen &pen)
- void **setWhiskerBarPen** (const QPen &pen)
- void **setMedianPen** (const QPen &pen)
- void **setOutlierStyle** (const QCPScatterStyle &style)
- virtual void **clearData** ()
- virtual double **selectTest** (const QPointF &pos, bool onlySelectable, QVariant ∗details=0) const

Protected Member Functions

- virtual void **draw** (QCPPainter ∗painter)
- virtual void **drawLegendIcon** (QCPPainter ∗painter, const QRectF &rect) const
- virtual QCPRange **getKeyRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual QCPRange **getValueRange** (bool &foundRange, SignDomain inSignDomain=sdBoth) const
- virtual void **drawQuartileBox** (QCPPainter ∗painter, QRectF ∗quartileBox=0) const
- virtual void **drawMedian** (QCPPainter ∗painter) const
- virtual void **drawWhiskers** (QCPPainter ∗painter) const
- virtual void **drawOutliers** (QCPPainter ∗painter) const

Protected Attributes

- QVector< double > **mOutliers**
- double **mKey**
- double **mMinimum**
- double **mLowerQuartile**
- double **mMedian**
- double **mUpperQuartile**
- double **mMaximum**
- double **mWidth**
- double **mWhiskerWidth**
- QPen **mWhiskerPen**
- QPen **mWhiskerBarPen**
- QPen **mMedianPen**
- QCPScatterStyle **mOutlierStyle**

Friends

- class **QCustomPlot**
- class **QCPLegend**

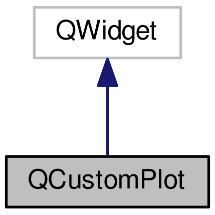
Additional Inherited Members

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

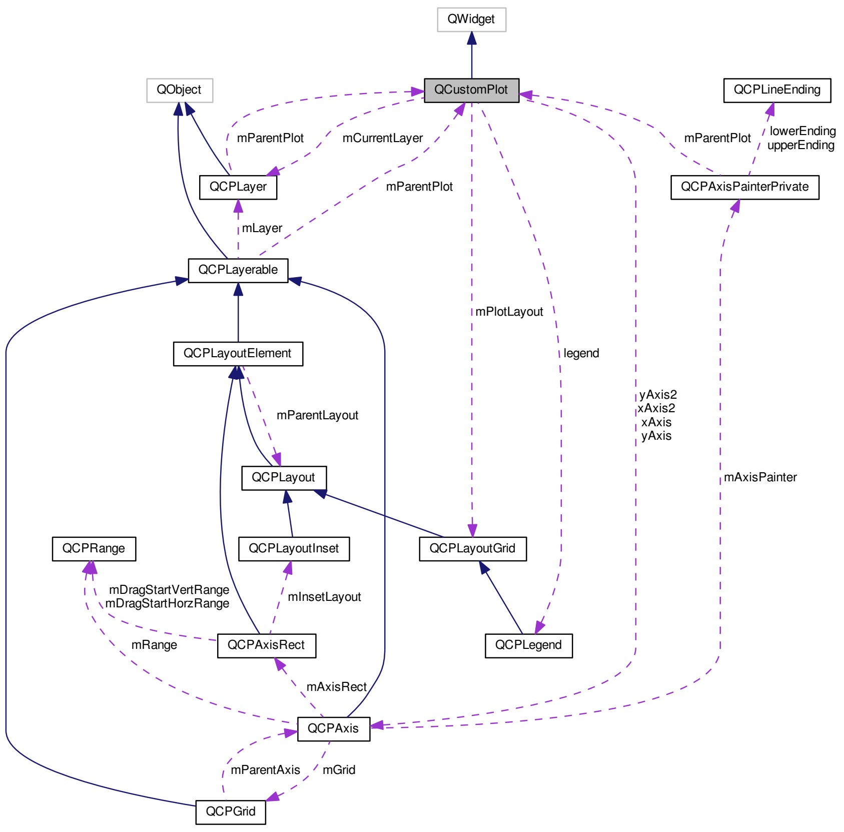
- src/qcustomplot/qcustomplot.h

6.57 QCustomPlot Class Reference

Inheritance diagram for QCustomPlot:



Collaboration diagram for QCustomPlot:



Public Types

- enum LayerInsertMode { limBelow, limAbove }
- enum RefreshPriority { rpImmediate, rpQueued, rpHint }

Signals

- void **mouseDoubleClick** (QMouseEvent ∗event)
- void **mousePress** (QMouseEvent ∗event)
- void **mouseMove** (QMouseEvent ∗event)
- void **mouseRelease** (QMouseEvent ∗event)
- void **mouseWheel** (QWheelEvent ∗event)
- void **plottableClick** (QCPAbstractPlottable ∗plottable, QMouseEvent ∗event)
- void **plottableDoubleClick** (QCPAbstractPlottable ∗plottable, QMouseEvent ∗event)
- void **itemClick** (QCPAbstractItem ∗item, QMouseEvent ∗event)
- void **itemDoubleClick** (QCPAbstractItem ∗item, QMouseEvent ∗event)
- void **axisClick** (QCPAxis ∗axis, QCPAxis::SelectablePart part, QMouseEvent ∗event)
- void **axisDoubleClick** (QCPAxis ∗axis, QCPAxis::SelectablePart part, QMouseEvent ∗event)
- void **legendClick** (QCPLegend ∗legend, QCPAbstractLegendItem ∗item, QMouseEvent ∗event)
- void **legendDoubleClick** (QCPLegend ∗legend, QCPAbstractLegendItem ∗item, QMouseEvent ∗event)
- void **titleClick** (QMouseEvent ∗event, QCPPlotTitle ∗title)
- void **titleDoubleClick** (QMouseEvent ∗event, QCPPlotTitle ∗title)
- void **selectionChangedByUser** ()
- void **beforeReplot** ()
- void **afterReplot** ()

Public Member Functions

- **QCustomPlot** (QWidget ∗parent=0)
- QRect **viewport** () const
- QPixmap **background** () const
- bool **backgroundScaled** () const
- Qt::AspectRatioMode **backgroundScaledMode** () const
- QCPLayoutGrid ∗ **plotLayout** () const
- QCP::AntialiasedElements **antialiasedElements** () const
- QCP::AntialiasedElements **notAntialiasedElements** () const
- bool **autoAddPlottableToLegend** () const
- const QCP::Interactions **interactions** () const
- int **selectionTolerance** () const
- bool **noAntialiasingOnDrag** () const
- QCP::PlottingHints **plottingHints** () const
- Qt::KeyboardModifier **multiSelectModifier** () const
- void **setViewport** (const QRect &rect)
- void **setBackground** (const QPixmap &pm)
- void **setBackground** (const QPixmap &pm, bool scaled, Qt::AspectRatioMode mode=Qt::KeepAspect↩RatioByExpanding)
- void **setBackground** (const QBrush &brush)
- void **setBackgroundScaled** (bool scaled)
- void **setBackgroundScaledMode** (Qt::AspectRatioMode mode)
- void **setAntialiasedElements** (const QCP::AntialiasedElements &antialiasedElements)
- void **setAntialiasedElement** (QCP::AntialiasedElement antialiasedElement, bool enabled=true)
- void **setNotAntialiasedElements** (const QCP::AntialiasedElements &notAntialiasedElements)
- void **setNotAntialiasedElement** (QCP::AntialiasedElement notAntialiasedElement, bool enabled=true)
- void **setAutoAddPlottableToLegend** (bool on)
- void **setInteractions** (const QCP::Interactions &interactions)
- void **setInteraction** (const QCP::Interaction &interaction, bool enabled=true)
- void **setSelectionTolerance** (int pixels)
- void **setNoAntialiasingOnDrag** (bool enabled)
- void **setPlottingHints** (const QCP::PlottingHints &hints)

- void **setPlottingHint** (QCP::PlottingHint hint, bool enabled=true)
- void **setMultiSelectModifier** (Qt::KeyboardModifier modifier)
- QCPAbstractPlottable ∗ **plottable** (int index)
- QCPAbstractPlottable ∗ **plottable** ()
- bool **addPlottable** (QCPAbstractPlottable ∗plottable)
- bool **removePlottable** (QCPAbstractPlottable ∗plottable)
- bool **removePlottable** (int index)
- int **clearPlottables** ()
- int **plottableCount** () const
- QList< QCPAbstractPlottable ∗ > **selectedPlottables** () const
- QCPAbstractPlottable ∗ **plottableAt** (const QPointF &pos, bool onlySelectable=false) const
- bool **hasPlottable** (QCPAbstractPlottable ∗plottable) const
- QCPGraph ∗ **graph** (int index) const
- QCPGraph ∗ **graph** () const
- QCPGraph ∗ **addGraph** (QCPAxis ∗keyAxis=0, QCPAxis ∗valueAxis=0)
- bool **removeGraph** (QCPGraph ∗graph)
- bool **removeGraph** (int index)
- int **clearGraphs** ()
- int **graphCount** () const
- QList< QCPGraph ∗ > **selectedGraphs** () const
- QCPAbstractItem ∗ **item** (int index) const
- QCPAbstractItem ∗ **item** () const
- bool **addItem** (QCPAbstractItem ∗item)
- bool **removeItem** (QCPAbstractItem ∗item)
- bool **removeItem** (int index)
- int **clearItems** ()
- int **itemCount** () const
- QList< QCPAbstractItem ∗ > **selectedItems** () const
- QCPAbstractItem ∗ **itemAt** (const QPointF &pos, bool onlySelectable=false) const
- bool **hasItem** (QCPAbstractItem ∗item) const
- QCPLayer ∗ **layer** (const QString &name) const
- QCPLayer ∗ **layer** (int index) const
- QCPLayer ∗ **currentLayer** () const
- bool **setCurrentLayer** (const QString &name)
- bool **setCurrentLayer** (QCPLayer ∗layer)
- int **layerCount** () const
- bool **addLayer** (const QString &name, QCPLayer ∗otherLayer=0, LayerInsertMode insertMode=limAbove)
- bool **removeLayer** (QCPLayer ∗layer)
- bool **moveLayer** (QCPLayer ∗layer, QCPLayer ∗otherLayer, LayerInsertMode insertMode=limAbove)
- int **axisRectCount** () const
- QCPAxisRect ∗ **axisRect** (int index=0) const
- QList< QCPAxisRect ∗ > **axisRects** () const
- QCPLayoutElement ∗ **layoutElementAt** (const QPointF &pos) const
- Q_SLOT void **rescaleAxes** (bool onlyVisiblePlottables=false)
- QList< QCPAxis ∗ > **selectedAxes** () const
- QList< QCPLegend ∗ > **selectedLegends** () const
- Q_SLOT void **deselectAll** ()
- bool **savePdf** (const QString &fileName, bool noCosmeticPen=false, int width=0, int height=0, const QString &pdfCreator=QString(), const QString &pdfTitle=QString())
- bool **savePng** (const QString &fileName, int width=0, int height=0, double scale=1.0, int quality=-1)
- bool **saveJpg** (const QString &fileName, int width=0, int height=0, double scale=1.0, int quality=-1)
- bool **saveBmp** (const QString &fileName, int width=0, int height=0, double scale=1.0)
- bool **saveRastered** (const QString &fileName, int width, int height, double scale, const char ∗format, int quality=-1)
- QPixmap **toPixmap** (int width=0, int height=0, double scale=1.0)
- void **toPainter** (QCPPainter ∗painter, int width=0, int height=0)
- Q_SLOT void **replot** (QCustomPlot::RefreshPriority refreshPriority=QCustomPlot::rpHint)

Public Attributes

- QCPAxis ∗ **xAxis**
- QCPAxis ∗ **yAxis**
- QCPAxis ∗ **xAxis2**
- QCPAxis ∗ **yAxis2**
- QCPLegend ∗ **legend**

Protected Member Functions

- virtual QSize **minimumSizeHint** () const
- virtual QSize **sizeHint** () const
- virtual void **paintEvent** (QPaintEvent ∗event)
- virtual void **resizeEvent** (QResizeEvent ∗event)
- virtual void **mouseDoubleClickEvent** (QMouseEvent ∗event)
- virtual void **mousePressEvent** (QMouseEvent ∗event)
- virtual void **mouseMoveEvent** (QMouseEvent ∗event)
- virtual void **mouseReleaseEvent** (QMouseEvent ∗event)
- virtual void **wheelEvent** (QWheelEvent ∗event)
- virtual void **draw** (QCPPainter ∗painter)
- virtual void **axisRemoved** (QCPAxis ∗axis)
- virtual void **legendRemoved** (QCPLegend ∗legend)
- void **updateLayerIndices** () const
- QCPLayerable ∗ **layerableAt** (const QPointF &pos, bool onlySelectable, QVariant ∗selectionDetails=0) const
- void **drawBackground** (QCPPainter ∗painter)

Protected Attributes

- QRect **mViewport**
- QCPLayoutGrid ∗ **mPlotLayout**
- bool **mAutoAddPlottableToLegend**
- QList< QCPAbstractPlottable ∗ > **mPlottables**
- QList< QCPGraph ∗ > **mGraphs**
- QList< QCPAbstractItem ∗ > **mItems**
- QList< QCPLayer ∗ > **mLayers**
- QCP::AntialiasedElements **mAntialiasedElements**
- QCP::AntialiasedElements **mNotAntialiasedElements**
- QCP::Interactions **mInteractions**
- int **mSelectionTolerance**
- bool **mNoAntialiasingOnDrag**
- QBrush **mBackgroundBrush**
- QPixmap **mBackgroundPixmap**
- QPixmap **mScaledBackgroundPixmap**
- bool **mBackgroundScaled**
- Qt::AspectRatioMode **mBackgroundScaledMode**
- QCPLayer ∗ **mCurrentLayer**
- QCP::PlottingHints **mPlottingHints**
- Qt::KeyboardModifier **mMultiSelectModifier**
- QPixmap **mPaintBuffer**
- QPoint **mMousePressPos**
- QPointer< QCPLayoutElement > **mMouseEventElement**
- bool **mReplotting**

Friends

- class **QCPLegend**
- class **QCPAxis**
- class **QCPLayer**
- class **QCPAxisRect**
- class **FT1D::DisplaySignalWidget**

6.57.1 Member Enumeration Documentation

6.57.1.1 enum QCustomPlot::LayerInsertMode

Defines how a layer should be inserted relative to an other layer.

See also

addLayer, moveLayer

Enumerator

limBelow Layer is inserted below other layer.

limAbove Layer is inserted above other layer.

6.57.1.2 enum QCustomPlot::RefreshPriority

Defines with what timing the QCustomPlot surface is refreshed after a replot.

See also

replot

Enumerator

rpImmediate The QCustomPlot surface is immediately refreshed, by calling QWidget::repaint() after the replot.

rpQueued Queues the refresh such that it is performed at a slightly delayed point in time after the replot, by calling QWidget::update() after the replot.

rpHint Whether to use immediate repaint or queued update depends on whether the plotting hint QCP::ph←ForceRepaint is set, see setPlottingHints.

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

- src/qcustomplot/qcustomplot.h

6.58 FT1D::Signal Class Reference

The Signal class represents a signal.

```
#include <signal.h>
```

Public Member Functions

- Signal ()

  Constructor, creates an empty signal.
- Signal (const std::string &filename)

  Signal constructor, which loads the signal from the given file.
- Signal (const QVector< double > &x, const QVector< double > &y)

  Signal construcotr, which takes a pair of x and y points coordinates.
- Signal (const Signal &other)

  Signal copy constructor.
- Signal operator= (const Signal &other)

  operator = copies the signal other to this
- bool load_file (const std::string &filename)

  load_file loads signal from the given file.
- bool save_file (const std::string &filename) const

  save_file saves signal to the specified file
- QVector< double > x () const

  x obtain all x-coordinates
- QVector< double > y () const

  y obtain all y-coordinates
- bool empty () const

  empty check, whether the signal is empty or not
- void extend_left ()

  extend_left extend the signal by copying the original part to the left
- void extend_right ()

  extend_right extend the signal by copyting the original part to the right
- void shrink_left ()

  shrink_left shrink the signal by removing one copy of the original from the left
- void shrink_right ()

  shrink_right shrink the signal by removing one copy of the original from the right
- void reset ()

  reset remove all copies
- double min_x () const

  min_x
- double max_x () const

  max_x
- double min_y () const

  min_y
- double max_y () const

  max_y
- double original_min_x () const

  original_min_x
- double original_max_x () const

  original_max_x
- double original_min_y () const

  original_min_y
- double original_max_y () const

  original_max_y
- double range_x () const

  range_x
- double range_y () const

    range_y
- double original_range_x () const

    original_range_x
- double original_range_y () const

    original_range_y
- double allowed_max_x () const

    allowed_max_x
- double allowed_min_x () const

    allowed_min_x
- int original_length () const

    original_length
- void updateAll (int index, double value)

    updateAll change value on index and all its copies to value
- void findYMinMax ()

    findYMinMax updates ymin and ymax attributes
- void clear ()

    clear clears the signal
- Signal applyFilter (Signal &filter) const

    applyFilter applies filter filter to this signal and returns the result

Static Public Member Functions

- static void fourierTransform (Signal &input, Signal &magnitude, Signal &phase)

    fourierTransform computes the fourier transform of signal input
- static void inverseFourierTransform (Signal &magnitude, Signal &phase, Signal &output, QVector< double > x=QVector< double >())

    inverseFourierTransform computes the inverse fourier transform of signal magnitude and phase

Public Attributes

- QMap< double, double > **original**
- double **spacing**

6.58.1 Detailed Description

The Signal class represents a signal.

6.58.2 Constructor & Destructor Documentation

6.58.2.1 FT1D::Signal::Signal (const std::string & *filename*)

Signal constructor, which loads the signal from the given file.

Parameters

filename	path to the file to load

6.58.2.2 FT1D::Signal::Signal (const QVector< double > & *x*, const QVector< double > & *y*)

Signal construcotr, which takes a pair of x and y points coordinates.

Parameters

x	x-axis coordinates
y	y-axis coordinates

6.58.2.3 FT1D::Signal::Signal (const Signal & *other*)

Signal copy constructor.

Parameters

other	signal to copy

6.58.3 Member Function Documentation

6.58.3.1 double FT1D::Signal::allowed_max_x ( ) const [inline]

allowed_max_x

Returns

maximum allowed x coordinate

6.58.3.2 double FT1D::Signal::allowed_min_x ( ) const [inline]

allowed_min_x

Returns

minimum allowed x coordinate

6.58.3.3 Signal FT1D::Signal::applyFilter (Signal & *filter*) const

applyFilter applies filter *filter* to this signal and returns the result

Parameters

filter	filter to apply. This is a signal, that must have the same original length as this

Returns

result of filtering

6.58.3.4 bool FT1D::Signal::empty ( ) const [inline]

empty check, whether the signal is empty or not

Returns

6.58.3.5 static void FT1D::Signal::fourierTransform (Signal & *input,* Signal & *magnitude,* Signal & *phase*) [static]

fourierTransform computes the fourier transform of signal *input*

Parameters

input	signal for which to compute the transform
magnitude	signal of magnitudes of the fourier coefficients
phase	signal of phases of the fourier coefficients

6.58.3.6 static void FT1D::Signal::inverseFourierTransform (Signal & *magnitude,* Signal & *phase,* Signal & *output,* QVector< double > *x* = QVector< double >()) [static]

inverseFourierTransform computes the inverse fourier transform of signal *magnitude* and *phase*

Parameters

magnitude	signal of magnitudes of the fourier coefficients (input)
phase	signal of phases of the fourier coefficients (input)
output	result of the inverse fourier transorm (output)
x	if different x coordinates than 0,1,2,3,... should be used, pass them here

6.58.3.7 bool FT1D::Signal::load_file (const std::string & *filename*)

load_file loads signal from the given file.

Parameters

filename	path to the file to load

Returns

true in case of success, false otherwise (e.g. when the file format is invalid)

6.58.3.8 double FT1D::Signal::max_x ( ) const [inline]

max_x

Returns

maximum x coordinate

6.58.3.9 double FT1D::Signal::max_y ( ) const [inline]

max_y

Returns

maximum y coordinate

6.58.3.10 double FT1D::Signal::min_x ( ) const [inline]

min_x

Returns

minimum x coordinate

6.58.3.11 double FT1D::Signal::min_y ( ) const [inline]

min_y

Returns

minimum y coordinate

6.58.3.12 Signal FT1D::Signal::operator= (const Signal & *other*)

operator = copies the signal *other* to this

Parameters

other	signal to copy

Returns

∗this

6.58.3.13 int FT1D::Signal::original_length ( ) const [inline]

original_length

Returns

length of the original part of the signal

6.58.3.14 double FT1D::Signal::original_max_x ( ) const [inline]

original_max_x

Returns

maximum x coordinate from the original part of the signal

6.58.3.15 double FT1D::Signal::original_max_y ( ) const [inline]

original_max_y

Returns

maximum y coordinate from the original part of the signal

6.58.3.16 double FT1D::Signal::original_min_x ( ) const [inline]

original_min_x

Returns

minimum x coordinate from the original part of the signal

6.58.3.17 double FT1D::Signal::original_min_y ( ) const [inline]

original_min_y

Returns

minimum y coordinate from the original part of the signal

6.58.3.18 double FT1D::Signal::original_range_x ( ) const [inline]

original_range_x

Returns

range of x coordinates of the original part of the signal

6.58.3.19 double FT1D::Signal::original_range_y ( ) const [inline]

original_range_y

Returns

range of the y coordinates of the original part of the signal

6.58.3.20 double FT1D::Signal::range_x ( ) const [inline]

range_x

Returns

range of x coordinates

6.58.3.21 double FT1D::Signal::range_y ( ) const [inline]

range_y

Returns

range of y coordinates

6.58.3.22 bool FT1D::Signal::save_file (const std::string & *filename*) const

save_file saves signal to the specified file

Parameters

filename	where to save the signal

Returns

true in case of success

6.58.3.23 void FT1D::Signal::updateAll (int *index,* double *value*)

updateAll change value on *index* and all its copies to *value*

Parameters

index	index of value in original part of signal to change
value	value to change to

6.58.3.24 QVector<double> FT1D::Signal::x ( ) const [inline]

x obtain all x-coordinates

Returns

6.58.3.25 QVector<double> FT1D::Signal::y ( ) const [inline]

y obtain all y-coordinates

Returns

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

- src/signal.h

6.59 QCPAxisPainterPrivate::TickLabelData Struct Reference

Public Attributes

- QString **basePart**
- QString **expPart**
- QRect **baseBounds**
- QRect **expBounds**
- QRect **totalBounds**
- QRect **rotatedTotalBounds**
- QFont **baseFont**
- QFont **expFont**

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

- src/qcustomplot/qcustomplot.h

6.60 FT1D::Translation Struct Reference

The Translation struct is a language version of all texts in the application.

```
#include <localization.h>
```

Public Member Functions

- Translation ()

  Translation constructs the empty object.
- Translation (const QString &languageName, const QString &countryCode, const QDomElement &data)

  Translation constructs the object given all important properties.
- Translation (const Translation &other)

  Translation copy constructor.
- Translation ∗ getTranslationForWindow (const QString &windowName) const

  getTranslationForWindow given windowName, obtains the subtree of the data starting with window element having the name equal to the windowName
- Translation ∗ getTranslationForElement (const QString &elementName) const

  getTranslationForElement obtains the subtree of the data starting with UIElement having the name equal to the elementName
- Translation ∗ getTranslationForUseCase (const QString &name) const

  getTranslationForUseCase obtains the subtree of the data starting with UseCase element having the name equal to the name
- Translation ∗ getTranslationForElement (const int id) const

  getTranslationForElement obtains the subtree of the data starting with UIElement having the index equal to the id
- QString getTitle () const

  getTitle obtains the text contained between child <title> and <title> tag if it has no title tag, returns the content of <text> and </text> tag, or an empty string
- QString getText () const

  getText obtains the text contained between child <text> and <text> tag
- QString getChildElementText (const QString &elementName) const

  getChildElementText obtains text of the child element
- QString getChildElementText (const int elementIndex) const

  getChildElementText obtains text of the child element

Public Attributes

- QString **languageName**
- QString **countryCode**
- QDomElement **data**

6.60.1 Detailed Description

The Translation struct is a language version of all texts in the application.

6.60.2 Constructor & Destructor Documentation

6.60.2.1 FT1D::Translation::Translation (const QString & *languageName*, const QString & *countryCode*, const QDomElement & *data*)

Translation constructs the object given all important properties.

Parameters

languageName	name of the language version
countryCode	country code corresponding to this language version
data	translation data in form of a DOM tree

6.60.2.2 FT1D::Translation::Translation (const **Translation** & *other*)

Translation copy constructor.

Parameters

other	an object to copy

6.60.3 Member Function Documentation

6.60.3.1 QString FT1D::Translation::getChildElementText (const QString & *elementName*) const

getChildElementText obtains text of the child element

Parameters

elementName	element name, for which to get the text

Returns

text of the child element, of an empty string in case of failure

6.60.3.2 QString FT1D::Translation::getChildElementText (const int *elementIndex*) const

getChildElementText obtains text of the child element

Parameters

elementIndex	element index, for which to get the text

Returns

text of the child element, of an empty string in case of failure

6.60.3.3 QString FT1D::Translation::getText ( ) const

getText obtains the text contained between child <text> and <text> tag

Returns

demanded string, or empty string in case of failure

6.60.3.4 QString FT1D::Translation::getTitle ( ) const

getTitle obtains the text contained between child <title> and <title> tag if it has no title tag, returns the content of <text> and </text> tag, or an empty string

Returns

6.60.3.5 Translation∗ FT1D::Translation::getTranslationForElement (const QString & *elementName*) const

getTranslationForElement obtains the subtree of the data starting with UIElement having the name equal to the *elementName*

Parameters

elementName	name of the element for which to acquire the translation

Returns

a pointer to newly allocated instance of translation, or nullptr if such translation was not found

6.60.3.6 Translation∗ FT1D::Translation::getTranslationForElement (const int *id*) const

getTranslationForElement obtains the subtree of the data starting with UIElement having the index equal to the *id*

Parameters

id	id of the element for which to acquire the translation

Returns

a pointer to newly allocated instance of translation, or nullptr if such translation was not found

6.60.3.7 Translation∗ FT1D::Translation::getTranslationForUseCase (const QString & *name*) const

getTranslationForUseCase obtains the subtree of the data starting with UseCase element having the name equal to the *name*

Parameters

name	name of the use case for which to acquire the translation

Returns

a pointer to newly allocated instance of translation, or nullptr if such translation was not found

6.60.3.8 Translation∗ FT1D::Translation::getTranslationForWindow (const QString & *windowName*) const

getTranslationForWindow given *windowName*, obtains the subtree of the data starting with window element having the name equal to the *windowName*

Parameters

windowName	name of the window for which to acquire the translation

Returns

a pointer to newly allocated instance of translation, or nullptr if such translation was not found

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

- src/localization.h

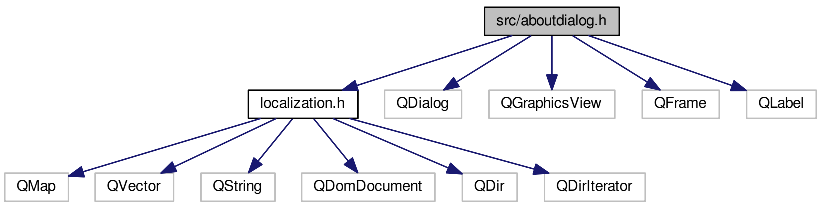
Chapter 7

File Documentation

7.1 src/aboutdialog.h File Reference

```
#include "localization.h"
#include <QDialog>
#include <QGraphicsView>
#include <QFrame>
#include <QLabel>
```

Include dependency graph for aboutdialog.h:



Classes

- class FT1D::AboutDialog

  The AboutDialog class is a simple dialog window with information about the application and its creator.

7.1.1 Detailed Description

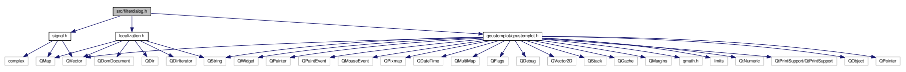
Author

Ján Bella xbella1@fi.muni.cz

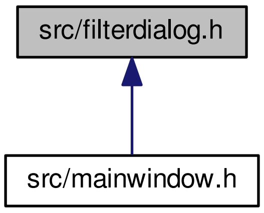
7.2 src/filterdialog.h File Reference

```
#include "localization.h"
#include "qcustomplot/qcustomplot.h"
#include "signal.h"
```

Include dependency graph for filterdialog.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FT1D::FilterDialog

  The FilterDialog class is a window to setup a filter.

Enumerations

- enum FT1D::FilterType {
  ILPF, **IHPF**, **LPGAUSS**, **HPGAUSS**,
  LPBUTTERWORTH, **HPBUTTERWORTH**, **BANDPASS** }

  The FilterType enum denotes type of the filter.

7.2.1 Detailed Description

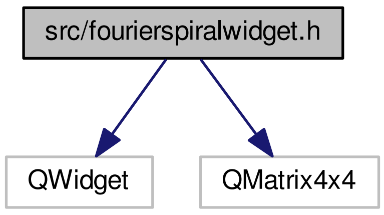
Author

Ján Bella `xbella1@fi.muni.cz`

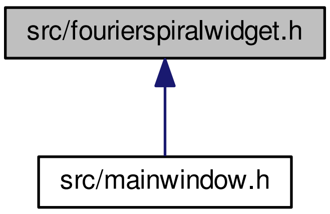
7.3 src/fourierspiralwidget.h File Reference

```
#include <QWidget>
#include <QMatrix4x4>
```

Include dependency graph for fourierspiralwidget.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FT1D::FourierSpiralWidget

  The FourierSpiralWidget class.

7.3.1 Detailed Description

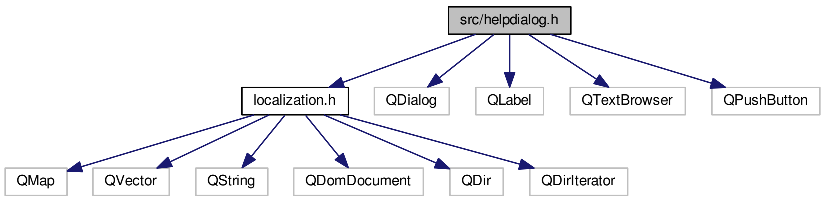
Author

Ján Bella `xbella1@fi.muni.cz`

7.4 src/helpdialog.h File Reference

```
#include "localization.h"
#include <QDialog>
#include <QLabel>
#include <QTextBrowser>
#include <QPushButton>
```

Include dependency graph for helpdialog.h:



Classes

- class FT1D::HelpDialog

  The HelpDialog class is a dialog window with information about usage.

7.4.1 Detailed Description

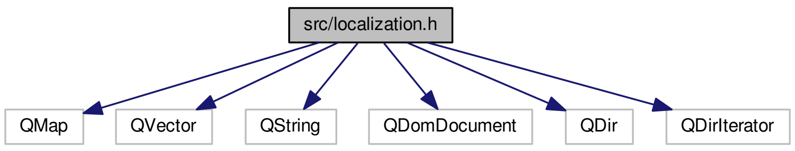
Author

Ján Bella `xbella1@fi.muni.cz`

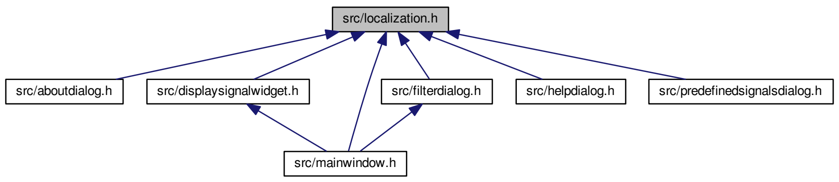
7.5 src/localization.h File Reference

```
#include <QMap>
#include <QVector>
#include <QString>
#include <QDomDocument>
#include <QDir>
#include <QDirIterator>
```

Include dependency graph for localization.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct FT1D::Translation

  The Translation struct is a language version of all texts in the application.
- class FT1D::Localizations

  The Localizations class is responsible for managing the language versions (Translations)

7.5.1 Detailed Description

Author

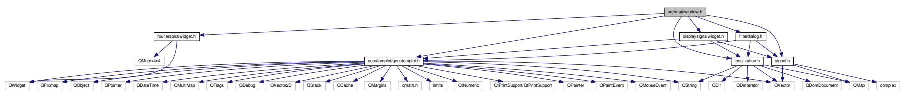
Ján Bella xbella1@fi.muni.cz

The file provides definitions of classes Localization and Translation.

7.6 src/mainwindow.h File Reference

```
#include "displaysignalwidget.h"
#include "qcustomplot/qcustomplot.h"
#include "signal.h"
#include "localization.h"
#include "filterdialog.h"
#include "fourierspiralwidget.h"
```

Include dependency graph for mainwindow.h:



Classes

- class FT1D::MainWindow

  The MainWindow class the main window of the application and the most of the app logic.

7.6.1 Detailed Description

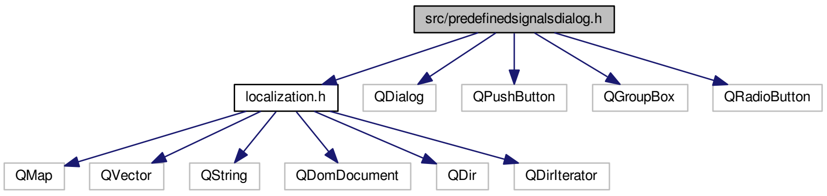
Author

Ján Bella xbella1@fi.muni.cz

7.7 src/predefinedsignalsdialog.h File Reference

```
#include "localization.h"
#include <QDialog>
#include <QPushButton>
#include <QGroupBox>
#include <QRadioButton>
```

Include dependency graph for predefinedsignalsdialog.h:



Classes

- class FT1D::PredefinedSignalsDialog

  The PredefinedSignalsDialog class is a Dialog in which the user can choose to load one of 8 predefined signals.

7.7.1 Detailed Description

Author

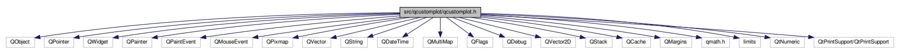
Ján Bella xbella1@fi.muni.cz

7.8 src/qcustomplot/qcustomplot.h File Reference

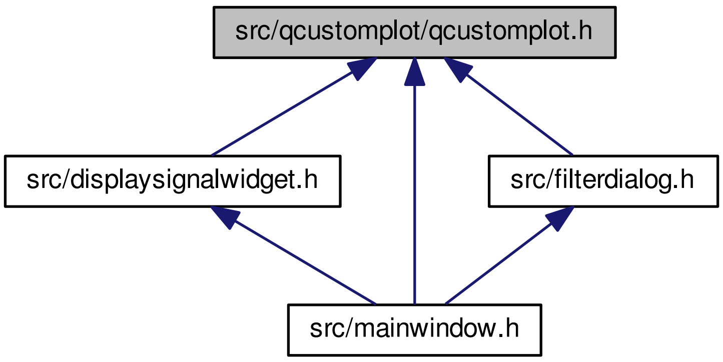
```
#include <QObject>
```

```
#include <QPointer>
#include <QWidget>
#include <QPainter>
#include <QPaintEvent>
#include <QMouseEvent>
#include <QPixmap>
#include <QVector>
#include <QString>
#include <QDateTime>
#include <QMultiMap>
#include <QFlags>
#include <QDebug>
#include <QVector2D>
#include <QStack>
#include <QCache>
#include <QMargins>
#include <qmath.h>
#include <limits>
#include <QtNumeric>
#include <QtPrintSupport/QtPrintSupport>
```

Include dependency graph for qcustomplot.h:



This graph shows which files directly or indirectly include this file:



Classes

- class QCPScatterStyle
- class QCPPainter
- class QCPLayer
- class QCPLayerable
- class QCPRange

- class QCPMarginGroup
- class QCPLayoutElement
- class QCPLayout
- class QCPLayoutGrid
- class QCPLayoutInset
- class QCPLineEnding
- class QCPGrid
- class QCPAxis
- class QCPAxisPainterPrivate
- struct QCPAxisPainterPrivate::CachedLabel
- struct QCPAxisPainterPrivate::TickLabelData
- class QCPAbstractPlottable
- class QCPItemAnchor
- class QCPItemPosition
- class QCPAbstractItem
- class QCustomPlot
- class QCPColorGradient
- class QCPAxisRect
- class QCPAbstractLegendItem
- class QCPPlottableLegendItem
- class QCPLegend
- class QCPPlotTitle
- class QCPColorScaleAxisRectPrivate
- class QCPColorScale
- class QCPData
- class QCPGraph
- class QCPCurveData
- class QCPCurve
- class QCPBarsGroup
- class QCPBarData
- class QCPBars
- class QCPStatisticalBox
- class QCPColorMapData
- class QCPColorMap
- class QCPFinancialData
- class QCPFinancial
- class QCPItemStraightLine
- class QCPItemLine
- class QCPItemCurve
- class QCPItemRect
- class QCPItemText
- class QCPItemEllipse
- class QCPItemPixmap
- class QCPItemTracer
- class QCPItemBracket

Namespaces

- QCP

Typedefs

- typedef QMap< double, QCPData > QCPDataMap
- typedef QMapIterator< double, QCPData > **QCPDataMapIterator**
- typedef QMutableMapIterator< double, QCPData > **QCPDataMutableMapIterator**
- typedef QMap< double, QCPCurveData > QCPCurveDataMap
- typedef QMapIterator< double, QCPCurveData > **QCPCurveDataMapIterator**
- typedef QMutableMapIterator< double, QCPCurveData > **QCPCurveDataMutableMapIterator**
- typedef QMap< double, QCPBarData > QCPBarDataMap
- typedef QMapIterator< double, QCPBarData > **QCPBarDataMapIterator**
- typedef QMutableMapIterator< double, QCPBarData > **QCPBarDataMutableMapIterator**
- typedef QMap< double, QCPFinancialData > QCPFinancialDataMap
- typedef QMapIterator< double, QCPFinancialData > **QCPFinancialDataMapIterator**
- typedef QMutableMapIterator< double, QCPFinancialData > **QCPFinancialDataMutableMapIterator**

Enumerations

- enum QCP::MarginSide {
  QCP::msLeft = 0x01, QCP::msRight = 0x02, QCP::msTop = 0x04, QCP::msBottom = 0x08,
  QCP::msAll = 0xFF, QCP::msNone = 0x00 }
- enum QCP::AntialiasedElement {
  QCP::aeAxes = 0x0001, QCP::aeGrid = 0x0002, QCP::aeSubGrid = 0x0004, QCP::aeLegend = 0x0008,
  QCP::aeLegendItems = 0x0010, QCP::aePlottables = 0x0020, QCP::aeItems = 0x0040, QCP::aeScatters = 0x0080,
  QCP::aeErrorBars = 0x0100, QCP::aeFills = 0x0200, QCP::aeZeroLine = 0x0400, QCP::aeAll = 0xFFFF,
  QCP::aeNone = 0x0000 }
- enum QCP::PlottingHint { QCP::phNone = 0x000, QCP::phFastPolylines = 0x001, QCP::phForceRepaint = 0x002, QCP::phCacheLabels = 0x004 }
- enum QCP::Interaction {
  QCP::iRangeDrag = 0x001, QCP::iRangeZoom = 0x002, QCP::iMultiSelect = 0x004, QCP::iSelectPlottables = 0x008,
  QCP::iSelectAxes = 0x010, QCP::iSelectLegend = 0x020, QCP::iSelectItems = 0x040, QCP::iSelectOther = 0x080 }

Functions

- bool **QCP::isInvalidData** (double value)
- bool **QCP::isInvalidData** (double value1, double value2)
- void **QCP::setMarginValue** (QMargins &margins, QCP::MarginSide side, int value)
- int **QCP::getMarginValue** (const QMargins &margins, QCP::MarginSide side)
- **Q_DECLARE_TYPEINFO** (QCPScatterStyle, Q_MOVABLE_TYPE)
- **Q_DECLARE_TYPEINFO** (QCPRange, Q_MOVABLE_TYPE)
- const QCPRange operator+ (const QCPRange &range, double value)
- const QCPRange operator+ (double value, const QCPRange &range)
- const QCPRange operator- (const QCPRange &range, double value)
- const QCPRange operator∗ (const QCPRange &range, double value)
- const QCPRange operator∗ (double value, const QCPRange &range)
- const QCPRange operator/ (const QCPRange &range, double value)
- **Q_DECLARE_TYPEINFO** (QCPLineEnding, Q_MOVABLE_TYPE)
- **Q_DECLARE_TYPEINFO** (QCPData, Q_MOVABLE_TYPE)
- **Q_DECLARE_TYPEINFO** (QCPCurveData, Q_MOVABLE_TYPE)
- **Q_DECLARE_TYPEINFO** (QCPBarData, Q_MOVABLE_TYPE)
- **Q_DECLARE_TYPEINFO** (QCPFinancialData, Q_MOVABLE_TYPE)

7.8.1 Typedef Documentation

7.8.1.1 QCPBarDataMap

Container for storing QCPBarData items in a sorted fashion. The key of the map is the key member of the QCP←BarData instance.

This is the container in which QCPBars holds its data.

See also

QCPBarData, QCPBars::setData

7.8.1.2 QCPCurveDataMap

Container for storing QCPCurveData items in a sorted fashion. The key of the map is the t member of the QCP←CurveData instance.

This is the container in which QCPCurve holds its data.

See also

QCPCurveData, QCPCurve::setData

7.8.1.3 QCPDataMap

Container for storing QCPData items in a sorted fashion. The key of the map is the key member of the QCPData instance.

This is the container in which QCPGraph holds its data.

See also

QCPData, QCPGraph::setData

7.8.1.4 QCPFinancialDataMap

Container for storing QCPFinancialData items in a sorted fashion. The key of the map is the key member of the QCPFinancialData instance.

This is the container in which QCPFinancial holds its data.

See also

QCPFinancial, QCPFinancial::setData

7.8.2 Function Documentation

7.8.2.1 const **QCPRange** operator∗ (const **QCPRange** & *range,* double *value*) `[inline]`

Multiplies both boundaries of the range by *value*.

7.8.2.2 const **QCPRange** operator∗ (double *value,* const **QCPRange** & *range*) `[inline]`

Multiplies both boundaries of the range by *value*.

7.8.2.3 const **QCPRange** operator+ (const **QCPRange** & *range,* double *value*) `[inline]`

Adds *value* to both boundaries of the range.

7.8.2.4 const **QCPRange** operator+ (double *value,* const **QCPRange** & *range*) `[inline]`

Adds *value* to both boundaries of the range.

7.8.2.5 const **QCPRange** operator- (const **QCPRange** & *range,* double *value*) `[inline]`

Subtracts *value* from both boundaries of the range.

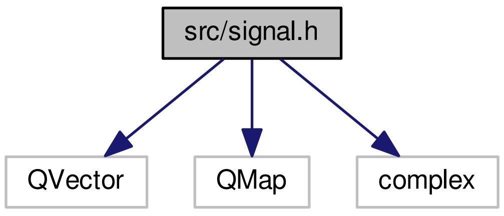
7.8.2.6 const **QCPRange** operator/ (const **QCPRange** & *range,* double *value*) `[inline]`

Divides both boundaries of the range by *value*.

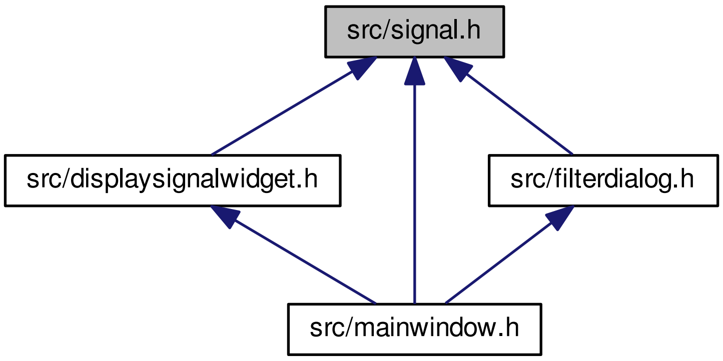
7.9 src/signal.h File Reference

```
#include <QVector>
#include <QMap>
#include <complex>
```

Include dependency graph for signal.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FT1D::Signal

  The Signal class represents a signal.

Macros

- #define **NUM_COPIES_ALLOWED** 3

7.9.1 Detailed Description

Author

Ján Bella `xbella1@fi.muni.cz`