QComboBox Class

QComboBox 小部件是按钮和弹出列表的组合。更多的...

Header:	**#include**
CMake:	find_package(Qt6 REQUIRED COMPONENTS Widgets) target_link_libraries(mytarget PRIVATE Qt6::Widgets)
qmake:	QT += widgets
Inherits:	QWidget
Inherited By:	QFontComboBox

- 所有成员的列表，包括继承的成员

公共类型

enum	**InsertPolicy { NoInsert, InsertAtTop, InsertAtCurrent, InsertAtBottom, InsertAfterCurrent, …, InsertAlphabetically }**
enum	**SizeAdjustPolicy** { AdjustToContents, AdjustToContentsOnFirstShow, AdjustToMinimumContentsLengthWithIcon }

特性

count : const int
currentData : const QVariant
currentIndex : int
currentText : QString
duplicatesEnabled : bool
editable : bool
frame : bool
iconSize : QSize
insertPolicy : InsertPolicy
maxCount : int
maxVisibleItems : int
minimumContentsLength : int
modelColumn : int
placeholderText : QString
sizeAdjustPolicy : SizeAdjustPolicy

公共方法

	QComboBox(QWidget **parent* = nullptr)
virtual	**~QComboBox()**

	QComboBox(QWidget **parent* = nullptr)
void	**addItem**(const QString &*text*, const QVariant &*userData* = QVariant())
void	**addItem**(const QIcon &*icon*, const QString &*text*, const QVariant &*userData* = QVariant())
void	**addItems**(const QStringList &*texts*)
QCompleter *	**completer**() const
int	**count**() const
QVariant	**currentData**(int *role* = Qt::UserRole) const
int	**currentIndex**() const
QString	**currentText**() const
bool	**duplicatesEnabled**() const
int	**findData**(const QVariant &*data*, int *role* = Qt::UserRole, Qt::MatchFlags *flags* = static_castQt::MatchFlags(Qt::MatchExactly|Qt::MatchCaseSensitive)) const
int	**findText**(const QString &*text*, Qt::MatchFlags *flags* = Qt::MatchExactly|Qt::MatchCaseSensitive) const
bool	**hasFrame**() const
virtual void	**hidePopup**()
QSize	**iconSize**() const
void	**insertItem**(int *index*, const QString &*text*, const QVariant &*userData* = QVariant())
void	**insertItem**(int *index*, const QIcon &*icon*, const QString &*text*, const QVariant &*userData* = QVariant())
void	**insertItems**(int *index*, const QStringList &*list*)
QComboBox::InsertPolicy	**insertPolicy**() const
void	**insertSeparator**(int *index*)
bool	**isEditable**() const
QVariant	**itemData**(int *index*, int *role* = Qt::UserRole) const
QAbstractItemDelegate *	**itemDelegate**() const
QIcon	**itemIcon**(int *index*) const
QString	**itemText**(int *index*) const
QLineEdit *	**lineEdit**() const
int	**maxCount**() const
int	**maxVisibleItems**() const

	QComboBox(QWidget **parent* = nullptr)
int	**minimumContentsLength**() const
QAbstractItemModel *	**model**() const
int	**modelColumn**() const
QString	**placeholderText**() const
void	**removeItem**(int *index*)
QModelIndex	**rootModelIndex**() const
void	**setCompleter**(QCompleter **completer*)
void	**setDuplicatesEnabled**(bool *enable*)
void	**setEditable**(bool *editable*)
void	**setFrame**(bool)
void	**setIconSize**(const QSize &*size*)
void	**setInsertPolicy**(QComboBox::InsertPolicy *policy*)
void	**setItemData**(int *index*, const QVariant &*value*, int *role* = Qt::UserRole)
void	**setItemDelegate**(QAbstractItemDelegate **delegate*)
void	**setItemIcon**(int *index*, const QIcon &*icon*)
void	**setItemText**(int *index*, const QString &*text*)
void	**setLineEdit**(QLineEdit **edit*)
void	**setMaxCount**(int *max*)
void	**setMaxVisibleItems**(int *maxItems*)
void	**setMinimumContentsLength**(int *characters*)
virtual void	**setModel**(QAbstractItemModel **model*)
void	**setModelColumn**(int *visibleColumn*)
void	**setPlaceholderText**(const QString &*placeholderText*)
void	**setRootModelIndex**(const QModelIndex &*index*)
void	**setSizeAdjustPolicy**(QComboBox::SizeAdjustPolicy *policy*)
void	**setValidator**(const QValidator **validator*)
void	**setView**(QAbstractItemView **itemView*)
virtual void	**showPopup**()
QComboBox::SizeAdjustPolicy	**sizeAdjustPolicy**() const

	QComboBox(QWidget **parent* = nullptr)
const QValidator *	validator() const
QAbstractItemView *	view() const

重载的公共方法

virtual bool	**event(QEvent **event*) override**
virtual QVariant	inputMethodQuery(Qt::InputMethodQuery *query*) const override
virtual QSize	minimumSizeHint() const override
virtual QSize	sizeHint() const override

公共槽

void	**clear()**
void	clearEditText()
void	setCurrentIndex(int *index*)
void	setCurrentText(const QString &*text*)
void	setEditText(const QString &*text*)

信号

void	**activated(int *index*)**
void	currentIndexChanged(int *index*)
void	currentTextChanged(const QString &*text*)
void	editTextChanged(const QString &*text*)
void	highlighted(int *index*)
void	textActivated(const QString &*text*)
void	textHighlighted(const QString &*text*)

protected function

virtual void	**initStyleOption**(QStyleOptionComboBox **option*) const
virtual void	**initStyleOption**(QStyleOptionComboBox **option*) const

重载的protected function

virtual void	**changeEvent**(QEvent **e*) override
virtual void	**contextMenuEvent**(QContextMenuEvent **e*) override
virtual void	**focusInEvent**(QFocusEvent **e*) override
virtual void	**focusOutEvent**(QFocusEvent **e*) override
virtual void	**hideEvent**(QHideEvent **e*) override
virtual void	**inputMethodEvent**(QInputMethodEvent **e*) override
virtual void	**keyPressEvent**(QKeyEvent **e*) override
virtual void	**keyReleaseEvent**(QKeyEvent **e*) override
virtual void	**mousePressEvent**(QMouseEvent **e*) override
virtual void	**mouseReleaseEvent**(QMouseEvent **e*) override
virtual void	**paintEvent**(QPaintEvent **e*) override
virtual void	**resizeEvent**(QResizeEvent **e*) override
virtual void	**showEvent**(QShowEvent **e*) override
virtual void	**wheelEvent**(QWheelEvent **e*) override

详细说明

QComboBox 提供了一种以**占用最小屏幕空间的方式向用户呈现选项列表的方法。**

组合框是一个显示当前项目的选择小部件，并且可以弹出可选项目的列表。组合框可以是可编辑的，允许用户修改列表中的每个项目。

组合框可以包含像素图以及字符串；这insertItem （） 和setItemText() 函数被适当重载。对于可编辑组合框，该函数clearEditText提供 () 来清除显示的字符串而不更改组合框的内容。

如果组合框的当前项发生更改，则会发出三个信号，currentIndexChanged(),currentTextChanged （） 和activated()。currentIndexChanged （） 和currentTextChanged无论更改是通过编程方式还是通过用户交互完成， () 总是会发出，而activated() 仅当更改是由用户交互引起时才会发出。这highlighted当用户突出显示组合框弹出列表中的项目时，会发出 () 信号。所有三个信号都有两种版本，一种带有QString论证和有论证的一个int。如果用户选择或突出显示像素图，则仅int发出信号。每当可编辑组合框的文本发生更改时editTextChanged() 信号被发射。

当用户在可编辑组合框中输入新字符串时，小部件可能会也可能不会插入它，并且可以将其插入到多个位置。默认策略是InsertAtBottom但你可以使用更改它setInsertPolicy()。

可以使用以下命令将输入限制为可编辑组合框QValidator; 看setValidator()。默认情况下，任何输入都会被接受。

可以使用插入函数填充组合框，insertItem （） 和insertItems （） 例如。项目可以更改为setItemText()。可以使用以下命令删除项目removeItem() 并且所有项目都可以通过以下方式删除clear()。当前项目的文本由以下命令返回currentText()，编号项的文本用 text() 返回。当前项目可以设置为setCurrentIndex()。组合框中的项目数通过以下方式返回count(); 可以设置最大项目数setMaxCount()。您可以使用允许编辑setEditable()。对于可编辑组合框，您可以使用设置自动完成setCompleter() 以及用户是否可以添加重复项设置为setDuplicatesEnabled()。

QComboBox 使用model/view framework用于其弹出列表并存储其项目。默认情况下QStandardItemModel存储物品和QListView子类显示弹出列表。您可以直接访问模型和视图（使用model （） 和view()），但QComboBox还提供了设置和获取项目数据的函数（例如，setItemData （） 和itemText()）。您还可以设置新模型和视图（使用setModel （） 和setView()）。对于组合框标签中的文本和图标，模型中具有以下内容的数据Qt::DisplayRole和Qt::DecorationRole用来。请注意，您无法更改SelectionMode的view()，例如，通过使用setSelectionMode()。

可以参阅QLineEdit,QSpinBox,QRadioButton， 和QButtonGroup。

成员类型文档

enum QComboBox::InsertPolicy

这个枚举指定了什么QComboBox当用户输入新字符串时应该执行的操作。

持续的	价值	描述
QComboBox::NoInsert	0	该字符串不会被插入到组合框中。
QComboBox::InsertAtTop	1	该字符串将作为组合框中的第一项插入。
QComboBox::InsertAtCurrent	2	当前项目将被字符串*替换*。
QComboBox::InsertAtBottom	3	该字符串将插入到组合框中最后一项之后。
QComboBox::InsertAfterCurrent	4	该字符串将插入到组合框中当前项目之后。
QComboBox::InsertBeforeCurrent	5	该字符串将插入到组合框中当前项目之前。
QComboBox::InsertAlphabetically	6	该字符串按字母顺序插入到组合框中。

enum QComboBox::SizeAdjustPolicy

该枚举指定了大小提示如何QComboBox应在添加新内容或内容更改时进行调整。

持续的	价值	描述
QComboBox::AdjustToContents	0	组合框将始终根据内容进行调整

持续的	价值	描述
QComboBox::AdjustToContentsOnFirstShow	1	组合框将在第一次显示时调整其内容。
QComboBox::AdjustToMinimumContentsLengthWithIcon	2	组合框将调整为minimumContentsLength加上图标的空间。出于性能原因，请在大型模型上使用此策略。

财产文件

[read-only]count : const int

该属性保存组合框中的项目数

默认情况下，对于空组合框，此属性的值为 0。

访问功能：

int	**count() const**

[read-only]currentData : const QVariant

该属性保存当前项目的数据

默认情况下，对于空组合框或未设置当前项目的组合框，此属性包含无效的QVariant。

访问功能：

QVariant	**currentData(int *role* = Qt::UserRole) const**

currentIndex : int

该属性保存组合框中当前项目的索引。

插入或删除项目时，当前索引可能会更改。

默认情况下，对于空组合框或未设置当前项目的组合框，此属性的值为 -1。

访问功能：

int	**currentIndex() const**
void	**setCurrentIndex**(int *index*)

通知器信号：

void	**currentIndexChanged**(int *index*)

currentText : QString

该属性保存当前文本

如果组合框可编辑，则当前文本是行编辑显示的值。否则，如果组合框为空或未设置当前项目，则它是当前项目的值或空字符串。

设置器 setCurrentText() 只需调用setEditText() 如果组合框可编辑。否则，如果列表中有匹配的文本，currentIndex被设置为相应的索引。

访问功能：

QString	**currentText() const**
void	**setCurrentText**(const QString &*text*)

通知器信号：

void	**currentTextChanged**(**const QString &*text***)

可以参阅editable和setEditText()。

duplicatesEnabled : bool

该属性保存用户是否可以在组合框中输入重复的项目

请注意，始终可以通过编程方式将重复的项目插入组合框中。

默认情况下，该属性是`false`（不允许重复）。

访问功能：

bool	**duplicatesEnabled() const**
void	**setDuplicatesEnabled**(bool *enable*)

editable : bool

该属性保存组合框是否可以由用户编辑

默认情况下，该属性为`false`。编辑的效果取决于插入策略。

注意：禁用时*editable*状态，验证器和完成器被删除。

访问功能：

bool	**isEditable() const**
void	**setEditable**(bool *editable*)

可以参阅InsertPolicy。

frame : bool

该属性保存组合框是否用框架绘制自身

如果启用（默认），组合框将在框架内绘制自身，否则组合框将在没有任何框架的情况下绘制自身。

访问功能：

bool	**hasFrame() const**
void	**setFrame**(bool)

iconSize : QSize

该属性保存组合框中显示的图标的大小。

除非明确设置，否则返回当前样式的默认值。这个尺寸是图标可以有的最大尺寸；较小尺寸的图标不会放大。

访问功能：

QSize	**iconSize() const**
void	**setIconSize**(const QSize &*size*)

insertPolicy : InsertPolicy

此属性保存用于确定用户插入的项目应出现在组合框中的位置的策略

默认值为InsertAtBottom，表示新项目将出现在项目列表的底部。

访问功能：

QComboBox::InsertPolicy	insertPolicy() const
void	**setInsertPolicy**(QComboBox::InsertPolicy *policy*)

可以参阅InsertPolicy。

maxCount : int

该属性保存组合框中允许的最大项目数

注意：如果您将最大数量设置为小于组合框中当前项目的数量，则多余的项目将被截断。如果您在组合框上设置了外部模型，这也适用。

默认情况下，此属性的值源自可用的最高有符号整数（通常为 2147483647）。

访问功能：

int	maxCount() const
void	**setMaxCount**(int *max*)

maxVisibleItems : int

此属性保存组合框屏幕上允许的最大大小，以项目为单位测量

默认情况下，该属性的值为 10。

注意：对于返回 true 的样式中的不可编辑组合框，此属性将被忽略QStyle::SH_ComboBox_Popup例如 Mac 风格或 Gtk+ 风格。

访问功能：

int	maxVisibleItems() const
void	**setMaxVisibleItems**(int *maxItems*)

minimumContentsLength : int

此属性保存组合框中应容纳的最小字符数。

默认值为 0。

如果该属性设置为正值，minimumSizeHint（） 和sizeHint() 考虑在内。

访问功能：

int	minimumContentsLength() const

int	**minimumContentsLength() const**
void	**setMinimumContentsLength**(int *characters*)

可以参阅sizeAdjustPolicy。

modelColumn : int

此属性保存模型中可见的列。

如果在填充组合框之前设置，弹出视图将不会受到影响，并将显示第一列（使用此属性的默认值）。

默认情况下，该属性的值为 0。

注意：在可编辑组合框中，可见列也将成为completion column。

访问功能：

int	**modelColumn() const**
void	**setModelColumn**(int *visibleColumn*)

placeholderText : QString

设置一个*placeholderText*未设置有效索引时显示的文本

这*placeholderText*当设置了无效索引时将显示。无法在下拉列表中访问该文本。当在添加项目之前调用此函数时，将显示占位符文本，否则您必须调用setCurrentIndex(-1) 如果您想显示占位符文本，则以编程方式。设置空占位符文本以重置设置。

当。。。的时候QComboBox是可编辑的，使用QLineEdit::setPlaceholderText（） 反而。

访问功能：

QString	**placeholderText() const**
void	**setPlaceholderText**(const QString &*placeholderText*)

sizeAdjustPolicy : SizeAdjustPolicy

该属性保存的策略描述当内容更改时组合框的大小如何更改

默认值为AdjustToContentsOnFirstShow。

访问功能：

QComboBox::SizeAdjustPolicy	**sizeAdjustPolicy() const**

QComboBox::SizeAdjustPolicy	sizeAdjustPolicy() const
void	**setSizeAdjustPolicy**(QComboBox::SizeAdjustPolicy *policy*)

可以参阅SizeAdjustPolicy。

成员函数文档

*[explicit]QComboBox::QComboBox(QWidget *parent = nullptr)*

使用给定的构造一个组合框*parent*，使用默认模型QStandardItemModel。

[virtual]QComboBox::~QComboBox()

销毁组合框。

[signal]void QComboBox::activated(int index)

当用户选择组合框中的项目时发送此信号。这几项*index*已通过。请注意，即使选择未更改，也会发送此信号。如果您需要知道选择何时实际改变，请使用信号currentIndexChanged（） 或者currentTextChanged()。

void QComboBox::addItem(const QString &text, const QVariant &userData = QVariant())

使用给定的值将项目添加到组合框*text*，并包含指定的*userData*（存储在Qt::UserRole）。该项目将附加到现有项目的列表中。

void QComboBox::addItem(const QIcon &icon, const QString &text, const QVariant &userData = QVariant())

使用给定的值将项目添加到组合框*icon*和*text*，并包含指定的*userData*（存储在Qt::UserRole）。该项目将附加到现有项目的列表中。

void QComboBox::addItems(const QStringList &texts)

添加给定中的每个字符串*texts*到组合框。每个项目依次附加到现有项目列表中。

*[override virtual protected]void QComboBox::changeEvent(QEvent *e)*

重新实现：QWidget::changeEvent（QEvent *事件）。

[slot]void QComboBox::clear()

清除组合框，删除所有项目。

注意：如果您在组合框上设置了外部模型，则调用此函数时该模型仍将被清除。

[slot]void QComboBox::clearEditText()

清除用于在组合框中编辑的行编辑的内容。

*QCompleter *QComboBox::completer() const*

返回用于自动完成组合框文本输入的完成器。

可以参阅setCompleter（） 和editable。

*[override virtual protected]void QComboBox::contextMenuEvent(QContextMenuEvent *e)*

重新实现：QWidget::contextMenuEvent（QContextMenuEvent *事件）。

[signal]void QComboBox::currentIndexChanged(int index)

每当currentIndex组合框中的更改可以通过用户交互或以编程方式进行。这几项*index*如果组合框变空或则传递-1currentIndex被重置。

注意：属性的通知程序信号currentIndex。

[signal]void QComboBox::currentTextChanged(const QString &text)

每当currentText变化。新值传递为*text*。

注意：属性的通知程序信号currentText。

[signal]void QComboBox::editTextChanged(const QString &text)

当组合框的行编辑小部件中的文本更改时，会发出此信号。新文本由以下方式指定*text*。

*[override virtual]bool QComboBox::event(QEvent *event)*

重新实现：QWidget::event（QEvent *事件）。

int QComboBox::findData(const QVariant &data, int role = Qt::UserRole, Qt::MatchFlags flags = static_castQt::MatchFlags(Qt::MatchExactly|Qt::MatchCaseSensitive)) const

返回包含给定项的索引|*data*对于给定的*role*; 否则返回-1。

这*flags*指定如何搜索组合框中的项目。

int QComboBox::findText(const QString &text, Qt::MatchFlags flags = Qt::MatchExactly|Qt::MatchCaseSensitive) const

返回包含给定项的索引|*text*; 否则返回-1。

这*flags*指定如何搜索组合框中的项目。

*[override virtual protected]void QComboBox::focusInEvent(QFocusEvent *e)*

重新实现：QWidget::focusInEvent（QFocusEvent *事件）。

[override virtual protected]void QComboBox::focusOutEvent(QFocusEvent *e)

重新实现：QWidget::focusOutEvent（QFocusEvent *事件）。

[override virtual protected]void QComboBox::hideEvent(QHideEvent *e)

重新实现：QWidget::hideEvent（QHideEvent *事件）。

[virtual]void QComboBox::hidePopup()

隐藏组合框中的项目列表（如果当前可见）并重置内部状态，以便如果自定义弹出窗口显示在重新实现的内部showPopup()，那么您还需要重新实现 hidePopup() 函数来隐藏自定义弹出窗口，并在隐藏自定义弹出窗口小部件时调用基类实现来重置内部状态。

可以参阅showPopup()。

[signal]void QComboBox::highlighted(int index)

当用户突出显示组合框弹出列表中的项目时发送此信号。这几项*index*已通过。

[virtual protected]void QComboBox::initStyleOption(QStyleOptionComboBox *option) const

初始化*option*与此值QComboBox。当子类需要时，此方法非常有用QStyleOptionComboBox，但不想自己填写所有信息。

可以参阅QStyleOption::initFrom()。

[override virtual protected]void QComboBox::inputMethodEvent(QInputMethodEvent *e)

重新实现：QWidget::inputMethodEvent（QInputMethodEvent *事件）。

[override virtual]QVariant QComboBox::inputMethodQuery(Qt::InputMethodQuery query) const

重新实现：QWidget::inputMethodQuery(Qt::InputMethodQuery query) const。

void QComboBox::insertItem(int index, const QString &text, const QVariant &userData = QVariant())

插入*text*和*userData*（存储在Qt::UserRole）进入给定的组合框*index*。

如果索引等于或高于项目总数，则新项目将追加到现有项目列表中。如果索引为零或负数，则新项目将添加到现有项目列表的前面。

可以参阅insertItems()。

void QComboBox::insertItem(int index, const QIcon &icon, const QString &text, const QVariant &userData = QVariant())

插入*icon*,*text*和*userData*（存储在Qt::UserRole）进入给定的组合框*index*。

如果索引等于或高于项目总数，则新项目将追加到现有项目列表中。如果索引为零或负数，则新项目将添加到现有项目列表的前面。

可以参阅insertItems()。

void QComboBox::insertItems(int index, const QStringList &list)

插入来自的字符串*list*作为单独的项目进入组合框，从*index*指定的。

如果索引等于或高于项目总数，则新项目将追加到现有项目列表中。如果索引为零或负数，则新项目将添加到现有项目列表的前面。

可以参阅insertItem()。

void QComboBox::insertSeparator(int index)

将分隔符项插入组合框中给定的位置*index*。

如果索引等于或高于项目总数，则新项目将追加到现有项目列表中。如果索引为零或负数，则新项目将添加到现有项目列表的前面。

可以参阅insertItem()。

QVariant QComboBox::itemData(int index, int role = Qt::UserRole) const

返回给定的数据*role*在给定的*index*在组合框中，或无效QVariant如果没有该角色的数据。

可以参阅setItemData()。

*QAbstractItemDelegate *QComboBox::itemDelegate() const*

返回弹出列表视图使用的项目委托。

可以参阅setItemDelegate()。

QIcon QComboBox::itemIcon(int index) const

返回给定的图标*index*在组合框中。

可以参阅setItemIcon()。

QString QComboBox::itemText(int index) const

返回给定的文本*index*在组合框中。

可以参阅setItemText()。

*[override virtual protected]void QComboBox::keyPressEvent(QKeyEvent *e)*

重新实现：QWidget::keyPressEvent（QKeyEvent *事件）。

*[override virtual protected]void QComboBox::keyReleaseEvent(QKeyEvent *e)*

重新实现：QWidget::keyReleaseEvent（QKeyEvent *事件）。

QLineEdit *QComboBox::lineEdit() const

返回用于编辑组合框中的项目的行编辑，或者nullptr如果没有行编辑。

只有可编辑的组合框才可以进行行编辑。

可以参阅setLineEdit()。

[override virtual]*QSize* QComboBox::minimumSizeHint() const

重新实现属性的访问函数：QWidget::minimumSizeHint。

QAbstractItemModel *QComboBox::model() const

返回组合框使用的模型。

可以参阅setModel()。

[override virtual protected]void QComboBox::mousePressEvent(*QMouseEvent* *e)

重新实现：QWidget::mousePressEvent（QMouseEvent *事件）。

[override virtual protected]void QComboBox::mouseReleaseEvent(*QMouseEvent* *e)

重新实现：QWidget::mouseReleaseEvent（QMouseEvent *事件）。

[override virtual protected]void QComboBox::paintEvent(*QPaintEvent* *e)

重新实现：QWidget::paintEvent（QPaintEvent *事件）。

void QComboBox::removeItem(int index)

删除给定的项目*index*从组合框中。如果删除索引，这将更新当前索引。

如果出现以下情况，该函数不执行任何操作*index*超出范围。

*[override virtual protected]void QComboBox::resizeEvent(QResizeEvent *e)*

重新实现：QWidget::resizeEvent（QResizeEvent *事件）。

QModelIndex QComboBox::rootModelIndex() const

返回组合框中项目的根模型项目索引。

可以参阅setRootModelIndex()。

*void QComboBox::setCompleter(QCompleter *completer)*

设置*completer*来代替当前的完成器。如果*completer*是`nullptr`，自动完成功能被禁用。

默认情况下，对于可编辑组合框，QCompleter执行不区分大小写的内联完成是自动创建的。

注意：当editable属性变为`false`，或者当行编辑被调用替换时setLineEdit()。在 a 上设置完成器QComboBox不可编辑的将被忽略。

可以参阅completer()。

[slot]void QComboBox::setEditText(const QString &text)

设置*text*在组合框的文本编辑中。

void QComboBox::setItemData(int index, const QVariant &value, int role = Qt::UserRole)

设置数据*role*对于给定的项目*index*在组合框中指定*value*。

可以参阅itemData()。

*void QComboBox::setItemDelegate(QAbstractItemDelegate *delegate)*

设置项目*delegate*对于弹出列表视图。组合框取得委托的所有权。

任何现有委托都将被删除，但不会被删除。QComboBox不拥有所有权*delegate*。

警告：您不应在组合框、小部件映射器或视图之间共享委托的同一实例。这样做可能会导致不正确或不直观的编辑行为，因为连接到给定委托的每个视图都可能会收到closeEditor() 信号，并尝试访问、修改或关闭已关闭的编辑器。

可以参阅itemDelegate()。

void QComboBox::setItemIcon(int index, const QIcon &icon)

设置*icon*对于给定的项目*index*在组合框中。

可以参阅itemIcon()。

void QComboBox::setItemText(int index, const QString &text)

设置*text*对于给定的项目*index*在组合框中。

可以参阅itemText()。

*void QComboBox::setLineEdit(QLineEdit *edit)*

设置线路*edit*使用而不是当前行编辑小部件。

组合框获得行编辑的所有权。

注意：由于组合框的行编辑拥有QCompleter，任何先前的调用setCompleter() 将不再有任何作用。

可以参阅lineEdit()。

*[virtual]void QComboBox::setModel(QAbstractItemModel *model)*

将模型设置为*model*。*model*一定不能`nullptr`。如果你想清除模型的内容，请调用clear()。

注意：如果组合框是可编辑的，则*model*也将在行编辑的完成器上设置。

可以参阅model(),clear（），和setCompleter()。

void QComboBox::setRootModelIndex(const QModelIndex &index)

设置根模型项*index*对于组合框中的项目。

可以参阅rootModelIndex()。

*void QComboBox::setValidator(const QValidator *validator)*

设置*validator*来代替当前的验证器使用。

注意：当验证器被删除时editable属性变为`false`.

可以参阅validator()。

*void QComboBox::setView(QAbstractItemView *itemView)*

将组合框弹出窗口中使用的视图设置为给定的*itemView*。组合框获得视图的所有权。

注意：如果您想使用便捷视图（例如QListWidget,QTableWidget或者QTreeWidget），请务必致电setModel在调用此函数之前，在带有便利小部件模型的组合框中使用 () 。

可以参阅view()。

*[override virtual protected]void QComboBox::showEvent(QShowEvent *e)*

重新实现：QWidget::showEvent（QShowEvent *事件）。

[virtual]void QComboBox::showPopup()

显示组合框中的项目列表。如果列表为空，则不会显示任何项目。

如果您重新实现此函数以显示自定义弹出窗口，请确保调用hidePopup() 重置内部状态。

可以参阅hidePopup()。

[override virtual]QSize QComboBox::sizeHint() const

重新实现属性的访问函数：QWidget::sizeHint。

此实现会缓存大小提示，以避免在内容动态更改时调整大小。要使缓存值无效，请更改sizeAdjustPolicy。

[signal]void QComboBox::textActivated(const QString &text)

当用户选择组合框中的项目时发送此信号。这几项*text*已通过。请注意，即使选择未更改，也会发送此信号。如果您需要知道选择何时实际改变，请使用信号currentIndexChanged（） 或者currentTextChanged()。

[signal]void QComboBox::textHighlighted(const QString &text)

当用户突出显示组合框弹出列表中的项目时发送此信号。这几项*text*已通过。

*const QValidator *QComboBox::validator() const*

返回用于约束组合框文本输入的验证器。

可以参阅setValidator（） 和editable。

*QAbstractItemView *QComboBox::view() const*

返回用于组合框弹出窗口的列表视图。

可以参阅setView()。

*[override virtual protected]void QComboBox::wheelEvent(QWheelEvent *e)*

重新实现：QWidget::wheelEvent（QWheelEvent *事件）。