QCheckBox Class

QCheckBox 小部件提供了一个带有文本标签的复选框。更多的...

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

- 所有成员的列表，包括继承的成员

特性

- **tristate** : bool

公共方法

	QCheckBox(QWidget **parent* = nullptr)
	QCheckBox(const QString &*text*, QWidget **parent* = nullptr)
virtual	~QCheckBox()
Qt::CheckState	checkState() const
bool	isTristate() const
void	setCheckState(Qt::CheckState *state*)
void	setTristate(bool *y* = true)

重载的公共方法

virtual QSize	**minimumSizeHint() const override**
virtual QSize	sizeHint() const override

信号

void	**stateChanged**(int *state*)

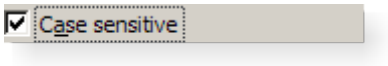
protected function

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

重载的protected function

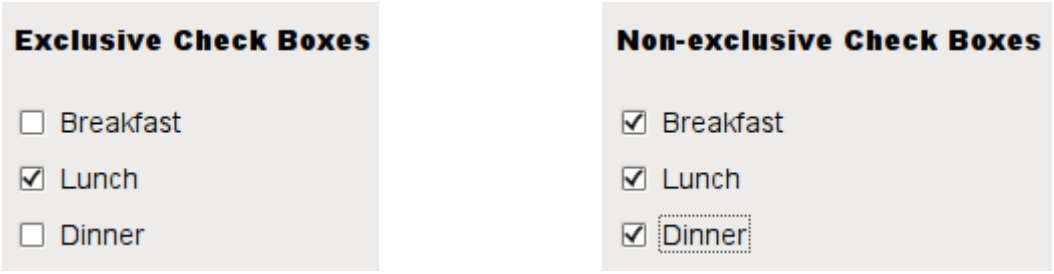
virtual void	**checkStateSet**() **override**
virtual bool	**event**(QEvent **e*) override
virtual bool	**hitButton**(const QPoint &*pos*) const override
virtual void	**mouseMoveEvent**(QMouseEvent **e*) override
virtual void	**nextCheckState**() override
virtual void	**paintEvent**(QPaintEvent *) override

详细说明



QCheckBox 是一个选项按钮，可以打开（选中）或关闭（取消选中）。复选框通常用于表示应用程序中可以启用或禁用而不影响其他功能的功能。可以实施不同类型的行为。例如，一个QButtonGroup可用于对检查按钮进行逻辑分组，从而允许独占复选框。然而，QButtonGroup不提供任何视觉表示。

下图进一步说明了独占和非独占复选框之间的差异。



每当选中或清除复选框时，它都会发出信号stateChanged()。如果您想在每次复选框更改状态时触发操作，请连接到此信号。您可以使用isChecked() 查询复选框是否被选中。

除了通常的选中和未选中状态之外，QCheckBox 还可以选择提供第三种状态来指示“无更改”。每当您需要为用户提供既不选中也不取消选中复选框的选项时，这非常有用。如果您需要第三种状态，请启用它setTristate()，并使用checkState() 查询当前切换状态。

就像QPushButton，复选框显示文本，还可以选择显示小图标。图标设置为setIcon()。文本可以在构造函数中设置或使用setText()。可以通过在首选字符前加上 & 符号来指定快捷键。例如：

```
QCheckBox *checkbox = new QCheckBox("C&ase sensitive", this);
```

在此示例中，快捷键是*Alt+A*。请参阅QShortcut文档了解详细信息。要显示实际的 & 符号，请使用“&&”。

重要的继承函数：text(),setText(),text()、pixmap()、setPixmap()、accel()、setAccel()、isToggleButton()、setDown(),isDown(), isOn(),checkState(),autoRepeat(), isExclusiveToggle(),group(),setAutoRepeat(),toggle(),pressed(),released(),clicked(),toggled(),checkState()，和stateChanged()。

也可以看看QAbstractButton和QRadioButton。

财产文件

tristate : bool

该属性保存复选框是否为三态复选框

默认为 false，即复选框只有两种状态。

访问功能：

bool	**isTristate() const**
void	**setTristate**(bool *y* = true)

成员函数文档

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

使用给定的构造一个复选框*parent*，但没有文字。

*parent*被传递到QAbstractButton构造函数。

*[explicit]QCheckBox::QCheckBox(const QString &text, QWidget *parent = nullptr)*

使用给定的构造一个复选框*parent*和*text*。

*parent*被传递到QAbstractButton构造函数。

[virtual]QCheckBox::~QCheckBox()

析构函数。

Qt::CheckState QCheckBox::checkState() const

返回复选框的选中状态。如果不需要三态支持，也可以使用QAbstractButton::isChecked()，返回一个布尔值。

也可以看看setCheckState（） 和Qt::CheckState。

[override virtual protected]void QCheckBox::checkStateSet()

重新实现：QAbstractButton::checkStateSet()。

*[override virtual protected]bool QCheckBox::event(QEvent *e)*

重新实现：QAbstractButton::event（QEvent *e）。

[override virtual protected]bool QCheckBox::hitButton(const QPoint &pos) const

重新实现：QAbstractButton::hitButton(const QPoint &pos) const。

*[virtual protected]void QCheckBox::initStyleOption(QStyleOptionButton *option) const*

初始化*option*与此值QCheckBox。此方法对于需要的子类很有用QStyleOptionButton，但又不想自己填写所有信息。

也可以看看QStyleOption::initFrom()。

[override virtual]QSize QCheckBox::minimumSizeHint() const

重新实现属性的访问函数：QWidget::minimumSizeHint。

*[override virtual protected]void QCheckBox::mouseMoveEvent(QMouseEvent *e)*

重新实现：QAbstractButton::mouseMoveEvent(QMouseEvent *e)。

[override virtual protected]void QCheckBox::nextCheckState()

重新实现：QAbstractButton::nextCheckState()。

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

重新实现：QAbstractButton::paintEvent（QPaintEvent *e）。

void QCheckBox::setCheckState(Qt::CheckState state)

将复选框的选中状态设置为*state*。如果不需要三态支持，也可以使用QAbstractButton::setChecked()，它接受一个布尔值。

也可以看看checkState（） 和Qt::CheckState。

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

重新实现属性的访问函数：QWidget::sizeHint。

[signal]void QCheckBox::stateChanged(int state)

每当复选框的状态改变时，即每当用户选中或取消选中它时，都会发出此信号。

*state*包含复选框的新内容Qt::CheckState。