QT6 CheckBox, SpinBox, ComboBox

QCheckbox

Это кнопка выбора, которую можно включить (установить флажок) или выключить (снять флажок). Флажки обычно используются для обозначения функций в приложении, которые можно включать или отключать, не затрагивая другие. При изменении состояния флажка выдается сигнал StateChanged(). Метод IsChecked() используется для запроса, установлен ли флажок.

Методы:

Метод	Описание
isChecked()	
setIcon()	
setText()	
setChecked()	

Сигналы:

stateChanged

QSpinbox

QSpinBox предназначен для обработки целых чисел и дискретных наборов значений, позволяет выбирать значение, нажимая кнопки вверх / вниз или нажимая клавиши вверх / вниз на клавиатуре, чтобы увеличить / уменьшить отображаемое в данный момент значение. Также возможно ввести значение вручную.

Методы:

Метод	Описание
value()	текущее выбранное целое значение
text()	отображения текста в окне прокрутки
setMinimum()	
setMaximum()	

Метод	Описание
setPrefix()	текстовый префикс, добавляемый перед значением, возвращаемым полем прокрутки.
setSuffix()	текст суффикса, добавляемый к значению, возвращаемому блоком spin.

Сигналы:

valueChanged()

editingFinished() выдается при потере фокуса на spinbox. Предполагаю, актуально для приложений с web backend при передаче финальных данных.

```
from PyQt6.QtWidgets import QApplication, QWidget, QLabel, QHBoxLayout, QLineEdit
from PyQt6.QtWidgets import QSpinBox
import sys

class Window(QWidget):
    def __init__(self):
        super().__init__()
        self.setGeometry(200,200, 700, 400)
        self.setWindowTitle("Python GUI Development")
        self.create_button()

    def create_button(self):
        hbox = QHBoxLayout()
        label = QLabel("Laptop price: ")
        self.lineedit = QLineEdit()
        self.spinbox = QSpinBox()
        self.spinbox.valueChanged.connect(self.spin_selected)

        self.total_result = QLineEdit()

        hbox.addWidget(label)
        hbox.addWidget(self.lineedit)
        hbox.addWidget(self.spinbox)
        hbox.addWidget(self.total_result)
        self.setLayout(hbox)

    def spin_selected(self):
        if self.lineedit.text() != 0:
```

```
            price = int(self.lineedit.text())
            totalPrice = self.spinbox.value() * price
            self.total_result.setText(str(totalPrice))

app = QApplication(sys.argv)
window = Window()
window.show()
sys.exit(app.exec())
```

QComboBox

Виджет выбора, отображающий текущий элемент. Также отображает список выбираемых элементов. Может быть редактируемым.

Также есть специализированный ComboBox: для выбора шрифтов (fontComboBox).

Методы:

Метод	**Описание**
setItemText()	Устанавливает или изменяет текст элемента в поле со списком.
removeItem()	Удаляет определенный элемент из поля со списком.
clear()	Удаляет все элементы из поля со списком.
currentText()	Возвращает текст текущего элемента, то есть элемента, который выбран в данный момент.
setCurrentIndex()	Устанавливает текущий индекс поля со списком, то есть задает желаемый элемент в поле со списком в качестве выбранного в данный момент элемента.
count()	Возвращает количество элементов в поле со списком.
setEditable()	Сделайте поле со списком доступным для редактирования, то есть пользователь можно редактировать элементы в поле со списком.
addItem()	Добавляет указанное содержимое в поле со списком.
itemText()	Возвращает текст в указанное расположение индекса в поле со списком.
currentIndex()	Возвращает индексное местоположение текущего выбранного элемента в поле со списком. Если поле со списком пусто или в поле со списком в данный момент не выбран ни один элемент, метод вернет значение -1 в качестве индекса.

Сигналы

currentIndexChanged() выбор нового элемента

editTextChanged() изменение текста в редактируемом комбобоксе

```
from PyQt6.QtWidgets import QApplication, QWidget, QComboBox, QLabel, QHBoxLayout, QVBoxLayout
import sys

class Window(QWidget):
    def __init__(self):
        super().__init__()
        self.setGeometry(200,200, 700, 400)
        self.setWindowTitle("Python GUI Development")
        self.create_button()

    def create_button(self):
        hbox = QHBoxLayout()
        label = QLabel('Select type: ')
        self.combo = QComboBox()
        self.combo.addItem('')
        self.combo.addItem('Current account')
        self.combo.addItem('Deposite account')
        self.combo.addItem('Saving account')
        self.combo.currentTextChanged.connect(self.updresult)
        hbox.addWidget(label)
        hbox.addWidget(self.combo)

        vbox = QVBoxLayout()
        vbox.addLayout(hbox)

        self.label_result = QLabel('')
        vbox.addWidget(self.label_result)

        self.setLayout(vbox)

    def updresult(self):
        self.label_result.setText('Your type: ' + self.combo.currentText())

app = QApplication(sys.argv)
window = Window()
```

```
window.show()
sys.exit(app.exec())
```

Revision #7
Created 31 January 2026 16:06:53 by Admin
Updated 8 February 2026 16:27:29 by Admin