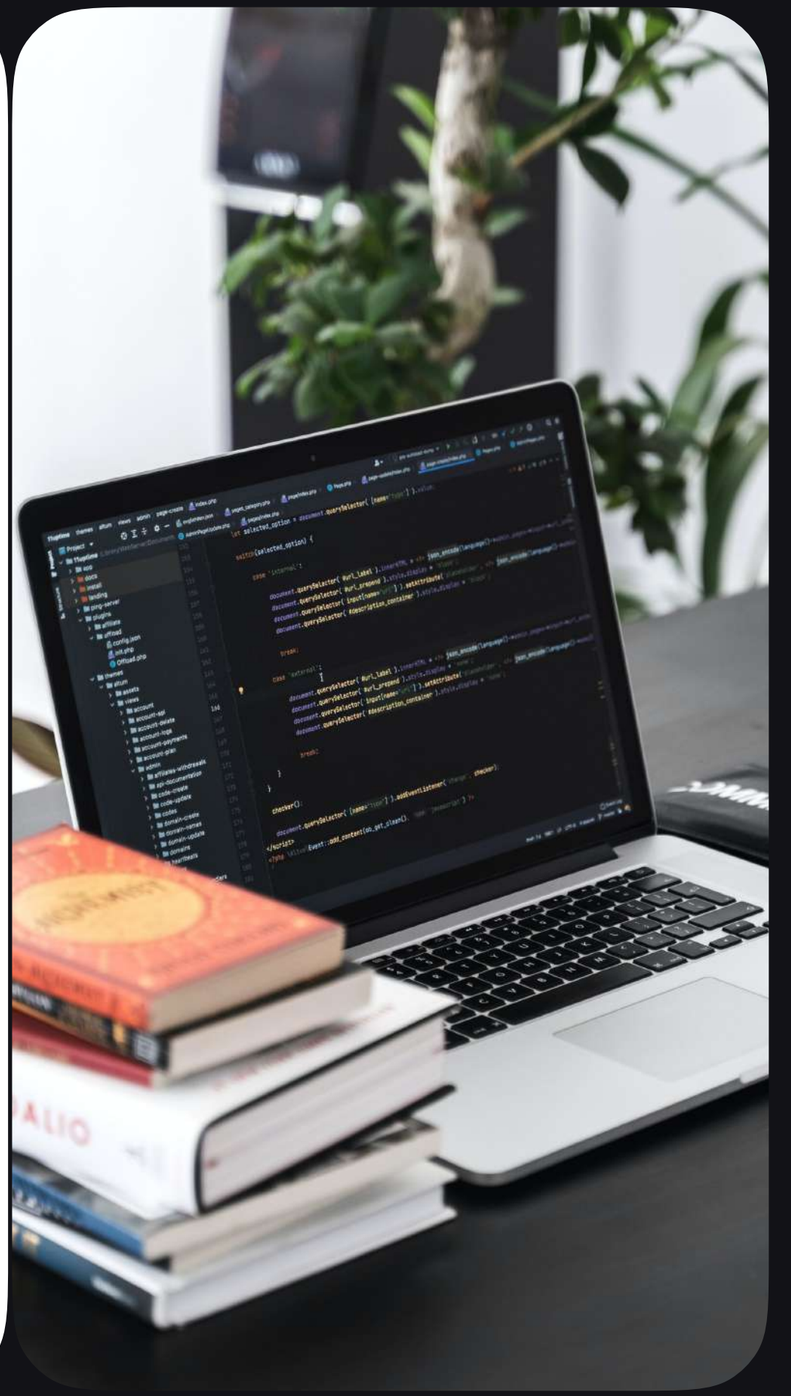
Занятие 11

PyQT: События, типы и обработка событий



PyQT: Сигналы и слоты

События и обработка событий

Обработка событий мыши

Обработка событий клавиатуры

Создание приложения «Калькулятор»

Введение

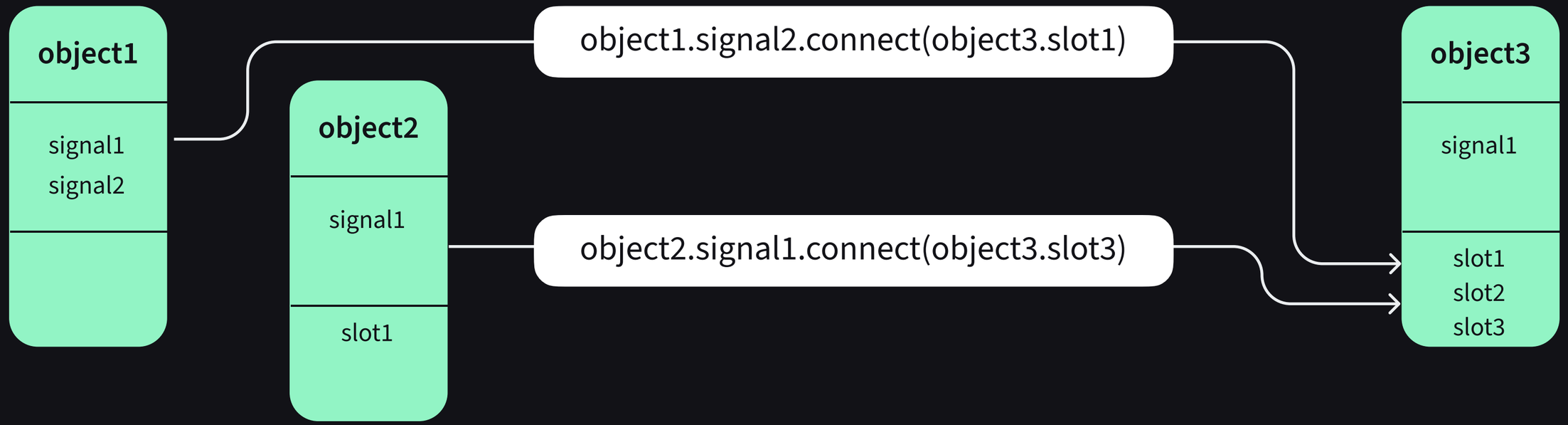
Приложения с графическим интерфейсом являются **событийно-ориентированными**. События могут вызываться пользователем, например, щелчком мыши, или другим способом, например, таймером. Главный цикл приложения обрабатывает события и отправляет их объектам.

В PyQt5 при обработке событий используются определения **сигналы** и **слоты**. Сигналы и слоты используются для связи между объектами, например, виджетами.

Сигналы и слоты

Сигнал — это уведомление, отправляемое виджетами когда что-либо происходит. Это может быть нажатие кнопки, изменение текста в поле ввода или изменение размера окна. Виджеты Qt имеют множество предопределенных сигналов, также есть возможность использовать собственные сигналы.

Слот — это функция или метод, который вызывается на определенный сигнал. Виджеты имеют предопределенные слоты, но на практике часто необходимо добавление собственных слотов, чтобы обрабатывать определенные сигналы.



Слоты и сигналы QPushButton

Slots

- def clear()
- def copy()
- def cut()
- def paste()
- def redo()
- def selectAll()
- def setText(arg__1)
- def undo()

Signals

- def cursorsPositionChanged(arg__1,arg__2)
- def editingFinished()
- def inputRejected()
- def returnPressed()
- def selectionChanged()
- def textChanged(arg__1)
- def textEdited(arg__1)

Посмотрим официальную документацию на виджет QLineEdit.

У виджета уже предопределены следующие слоты и сигналы.

Слоты и сигналы QPushButton

Slots

- def animateClick()
- def click()
- def setChecked(arg__1)
- def setIconSize(size)
- def toggle()

Signals

- def clicked([checked=false])
- def pressed()
- def released()
- def toggled(checked)

Виджет кнопки QPushButton наследует сигналы и слоты виджета QAbstractButton и тоже имеет предопределенные слоты и сигналы.

В официальной документации можно найти информацию о всех слотах и сигналах виджетов.

Пример

```python
import sys

from PyQt5.QtWidgets import QApplication, QPushButton, QWidget

class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(300, 200)
        self.setWindowTitle("Кликер")
        button = QPushButton("Нажми меня!", self)
        button.setGeometry(100, 100, 100, 23)
        button.clicked.connect(self.button_was_clicked)

    def button_was_clicked(self):
        print('click')

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

Мы воспользовались сигналом clicked и создали собственную слот функцию button_was_clicked. Функция принимает сигнал clicked от QPushButton.

В Python любая функция (или метод) может использоваться как слот для этого нужно подключить к ней сигнал (connect).

Подключение виджетов напрямую

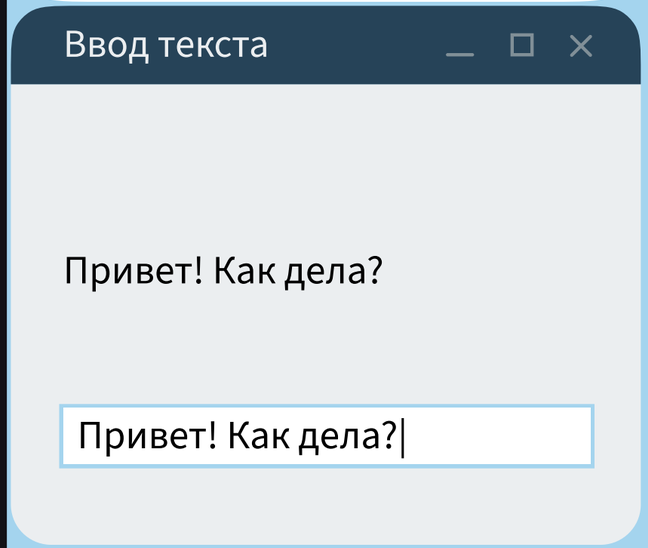
```
import sys

from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QLabel, QLineEdit

class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(250, 150)
        self.setWindowTitle("Ввод текста")
        label = QLabel()
        line_edit = QLineEdit()
        vbox=QVBoxLayout()
        vbox.addWidget(label)
        vbox.addWidget(line_edit)
        self.setLayout(vbox)
        line_edit.textChanged.connect(label.setText)

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

Сигнал textChanged виджета QLineEdit соединен со слотом setText QLabel.
При изменении текста в QLineEdit он меняется в QLabel.



Отправитель сигнала

```
import sys

from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QPushButton

class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(250, 150)
        self.setWindowTitle("Кликер")
        button_1 = QPushButton("Нажми меня 1", self)
        button_2 = QPushButton("Нажми меня 2", self)
        vbox = QVBoxLayout()
        vbox.addWidget(button_1)
        vbox.addWidget(button_2)
        self.setLayout(vbox)
        button_1.clicked.connect(self.button_was_clicked)
        button_2.clicked.connect(self.button_was_clicked)

    def button_was_clicked(self):
        sender = self.sender()
        print(sender.text())

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

Одна и та же функция может быть подсоединена к нескольким сигналам. В этом случае важно знать, какой именно виджет отправил сигнал. Для этого PyQt имеет метод sender().

События

Любое взаимодействие пользователя с приложением — это событие. События в PyQt — это объекты (наследуются от класса QEvent), в которых содержится информация о событии. События передаются обработчикам в виджете, где произошло это событие.

События

События, связанные с мышью, это событие QMouseEvent, которое обрабатывается, например, обработчиками:

- mousePressEvent — кнопка мыши нажата
- mouseDoubleClickEvent — двойной клик мыши

События, связанные с клавиатурой, создают события QKeyEvent и могут обрабатываться например:

- keyPressEvent — клавиша на клавиатуре нажата

В официальной документации можно найти информацию обо всех событиях и их обработчиках.

Пример

```
import sys

from PyQt5.QtCore import Qt
from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QLabel

class MainWindow(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(250, 150)
        self.setWindowTitle("Кликер")
        self.label = QLabel(self)
        self.label.setAlignment(Qt.AlignHCenter | Qt.AlignVCenter)
        vbox = QVBoxLayout()
        vbox.addWidget(self.label)
        self.setLayout(vbox)

    def mousePressEvent(self, event):
        self.label.setText(f'{event.x()}, {event.y()}')

    def keyPressEvent(self, event):
        self.label.setText(event.text())

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

События можно перехватывать, просто переопределяя метод обработчика в этом классе. Например, переопределим обработчики **mousePressEvent** и **keyPressEvent** в нашем классе.