

ТЕМА 1.

Модули Qt для создания приложений с графическим интерфейсом.

Лекция.

Состав QT.

Учебные вопросы

1. Введение
2. Инструментарий PySide2
3. Установка и настройка PySide2

Источники

- Официальная документация: <https://doc.qt.io/qtforpython-5/>
- Прохоренок Н. А., Дронов В. А. Python 3 и PyQt 5. Разработка приложений. 2019 г.

Полезные ссылки

- [13 GUI библиотек на Python](#)
- [PyQt5 vs PySide2](#)
- [Руководство PyQt5](#)
- [Полное руководство PyQt6](#)

Введение

- **Qt** (кью-т, кью-ти, ку-тэ) — фреймворк для разработки кроссплатформенного программного обеспечения на языке программирования C++.
- Для многих языков программирования существуют наборы библиотеки, позволяющие использовать преимущества Qt:
 - Python — PyQt, PySide;
 - Ruby — QtRuby;
 - Java — Qt Jambi;
 - PHP — PHP-Qt и другие.
- Со времени своего появления в 1996 году библиотека легла в основу многих программных проектов. Кроме того, Qt является фундаментом популярной рабочей среды KDE, входящей в состав многих дистрибутивов Linux.

- **PyQt/PySide** — набор расширений (биндингов, привязок) на уровне API графического фреймворка Qt для Python.
- Фреймворки практически полностью реализует возможности Qt. Это более 600 классов, более 6000 функций и методов, включая:
 - набор виджетов графического интерфейса;
 - стили виджетов;
 - доступ к базам данных с помощью SQL (ODBC, MySQL, PostgreSQL, Oracle);
 - поддержку интернационализации (i18n);
 - парсер XML;
 - интеграцию с WebKit, движком рендеринга HTML; поддержку воспроизведения видео и аудио.
- Существует 3 версии:
 - PyQt6 (PySide6)
 - PyQt5 (PySide2)
 - PyQt4 (PySide) поддерживающие соответствующие версии Qt.

Поддерживаемые платформы:

Платформа	Описание
Linux/Unix	
X11	Qt для оконного менеджера X (Linux, FreeBSD, HP-UX, Solaris, AIX, и т. д.)
Wayland	Qt для Wayland.
Встраиваемые Linux-системы	Qt для встраиваемых систем: КПК, смартфонов, и т. д.
Android	Qt для Android, ранее известный как Necessitas.
Платформы Apple	
OS X	Qt для Apple OS X; поддерживает приложения на Cocoa.
iOS	Qt для iOS платформ (iPhone, iPad).
Платформы Microsoft	
Windows	Qt для Microsoft Windows XP, Vista, 7, 8 и 10.
Windows CE	Qt для Windows CE 6 и Windows Embedded Compact 7.
Windows RT	Поддержка для основанных на WinRT приложениях для Windows 8 и Windows Phone 8.

Инструментарий PySide2

Категория/Фреймворк	PyQt	PySide
Лицензия	GPL или коммерческая лицензия	LGPL
Версия Qt	v5.15.6/v6.0.2 (PyQt6)	v5.15.2/v6.2.1+(PySide6)
Платформа	Python 3+	Python 3 и Python 2.7 (только для Linux и MacOS)
Первый стабильный выпуск	Апрель 2016/Январь 2021	Июль 2018/Декабрь 2020
Состав пакета	Основное ядро ~50Mb	Полный пакет со всеми инструментами ~120Mb

Пишите открытое/свободное ПО - можно использовать как PyQt5, так и PySide 2.

Пишите закрытое/коммерческое ПО - бесплатно можно использовать только PySide 2, а для использования PyQt5 потребуется покупать коммерческую лицензию.

Полная поставка включает в себя:

- **Qt Designer** — дизайнер графического интерфейса пользователя.
- **pyuic** — генерирует Python код из файлов, созданных в Qt Designer. Это делает PyQt очень полезным инструментом для быстрого прототипирования. Кроме того, можно добавлять новые графические элементы управления, написанные на Python, в Qt Designer.
- **QTranslator** — служит для локализации интерфейса.

Различия в коде

1. Конвертация файлов .ui в .py

```
# для PySide2
pyside2-uic you_form_name.ui -o you_form_name.py
# для PyQt5
pyuic5 you_form_name.ui -o you_form_name.py
```

1. Использование `exec()` или `exec_()`

```
if __name__ == "__main__":
    app = QtWidgets.QApplication()
    win = MyWin()
    win.show()
    app.exec_() # или app.exec()
```

- **PySide2** – возможен только `exec_()`, т.к. присутствует поддержка Python 2;
- **PyQt5** – возможны оба варианта, т.к. в Python 3 “`exec`” не является ключевым/зарезервированным словом.

1. Слоты и сигналы:

```
from PySide2.QtCore import Signal, Slot
from PyQt5.QtCore import pyqtSignal, pyqtSlot
```

```
my_signal_ps2 = Signal()
my_signal_pq2 = pyqtSignal()
```

```
@Slot
def my_slot_ps2():
    pass
```

```
@pyqtSlot
def my_slot_pq2():
    pass
```

С помощью определённых конструкций, можно обеспечить некоторую "совместимость":

```
from pip._internal.operations.freeze import freeze
```

```
for package in freeze(local_only=True):
    print(package)
```

```
    if 'PyQt5' in package:
        print('Work with PyQt5')
        from PyQt5 import QtGui, QtWidgets, QtCore
        from PyQt5.QtCore import pyqtSignal as Signal, pyqtSlot as
Slot
        break
```

```
    elif 'PySide2' in package:
        print('Work with PySide2')
        from PySide2 import QtGui, QtWidgets, QtCore
        from PySide2.QtCore import Signal, Slot
        break
```

Основные компоненты Qt

Модули Qt

QtCore

QtConcurrent

QtOpenGL

QtPrintSupport

QtQuickControls2

QtSvg

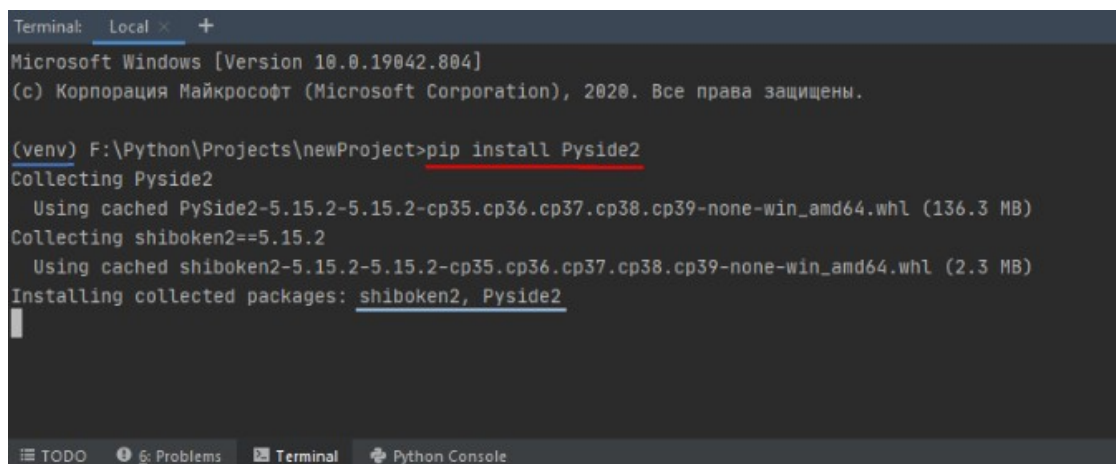
QtUiTools

- **QtCore** – основные функции, не связанные с графическим интерфейсом.
- **QtGui** – расширяет функциональность графического интерфейса.
- **QtWidgets** – работа с виджетами Qt.
- **QtConcurrent** – высокоуровневое API для работы с потоками.
- **QtHelp** – интеграция онлайн-документации в приложения.
- **QtNetwork** – позволяет писать клиент-серверные (TCP/IP) приложения.
- **QtOpenGL/QtOpenGLFunctions/QtOpenGLWidgets** – работа с 2D/3D графикой.
- **QtPrintSupport** – класс обеспечивающий поддержку печати.
- **QtQml/QtQuick/QtQuickControls2/QtQuickWidgets** – API для использования Qt QML (Qt Meta/Modeling Language) и создания настраиваемых высокодинамичных графических пользовательских интерфейсов с плавными переходами и эффектами.
- **QtSql** – содержит драйвера для обеспечения интеграции БД приложением.
- **QtSvg/QtSvgWidgets** – для работы с файлами SVG.
- **QtTest** – классы для модульного тестирования.
- **QtUiTools** – служит для обработки форм, созданных с помощью Qt Designer.
- **QtXml** – обеспечивает работу с потоками чтения и записи XML документов и реализацию их в форме SAX (Simple API for XML) и DOM (Document Object Model)

Установка и настройка

1. Устанавливаем PySide2

`pip install PySide2`



```

Terminal: Local x +
Microsoft Windows [Version 10.0.19042.804]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

(venv) F:\Python\Projects\newProject>pip install Pyside2
Collecting Pyside2
  Using cached PySide2-5.15.2-5.15.2-cp35.cp36.cp37.cp38.cp39-none-win_amd64.whl (136.3 MB)
Collecting shiboken2==5.15.2
  Using cached shiboken2-5.15.2-5.15.2-cp35.cp36.cp37.cp38.cp39-none-win_amd64.whl (2.3 MB)
Installing collected packages: shiboken2, Pyside2

```

Примечание:

Пакет shiboken2 – генератор привязок (binding generator), используемый для обеспечения связи между классами Qt (C++) и кодом PySide2.

1. Создаём .py файл и импортируем библиотеку

```
from PySide2 import QtWidgets
```

1. Создаём класс окна
- ```
class MyFirstWindow(QtWidgets.QWidget):
```

```
 def __init__(self, parent=None):
 super().__init__(parent)
```

```
class MyFirstWindow(QtWidgets.QWidget):

 def __init__(self, parent=None):
 super().__init__(parent)
```

1. Создаём объект приложения и объект окна
- ```
if __name__ == "__main__":
    app = QtWidgets.QApplication() # Создаем объект приложения

    myWindow = MyFirstWindow() # Создаём объект окна
    myWindow.show() # Показываем окно

    app.exec_() # Запускает бесконечный цикл выполнения приложения
```

```
if __name__ == "__main__":
    app = QtWidgets.QApplication() # Создаем объект приложения

    myWindow = MyFirstWindow() # Создаём объект окна
    myWindow.show() # Показываем окно

    sys.exit(app.exec_()) # Запускает бесконечный цикл выполнения приложения
```