* [ОБЗОР КУРСА](https://lyceum.yandex.ru/courses/165/groups/1257)

[Урок QtDesigner](https://lyceum.yandex.ru/courses/165/groups/1257/lessons/1226)

**QtDesigner**

1. [Что такое QtDesigner](https://lyceum.yandex.ru/courses/165/groups/1257/lessons/1226/materials/2200#1)
2. [Как подключить интерфейс к основной программе](https://lyceum.yandex.ru/courses/165/groups/1257/lessons/1226/materials/2200#2)
3. [Размещение виджетов](https://lyceum.yandex.ru/courses/165/groups/1257/lessons/1226/materials/2200#3)

**Аннотация**

*В этом уроке разбирается популярный способ создания графических интерфейсов — с помощью программы QtDesigner.*

**Что такое QtDesigner**

Когда на прошлом занятии мы создавали интерфейсы руками и размещали виджеты на глазок, вы наверняка подумали, что есть какой-то более простой способ. И он действительно есть. Это программа **QtDesigner**, которая включена в сборку PyQT5. Но для ее использования необходимо установить библиотеку pyqt5-tools.

pip install pyqt5-tools

Теперь программа находится на вашем компьютере по адресу:

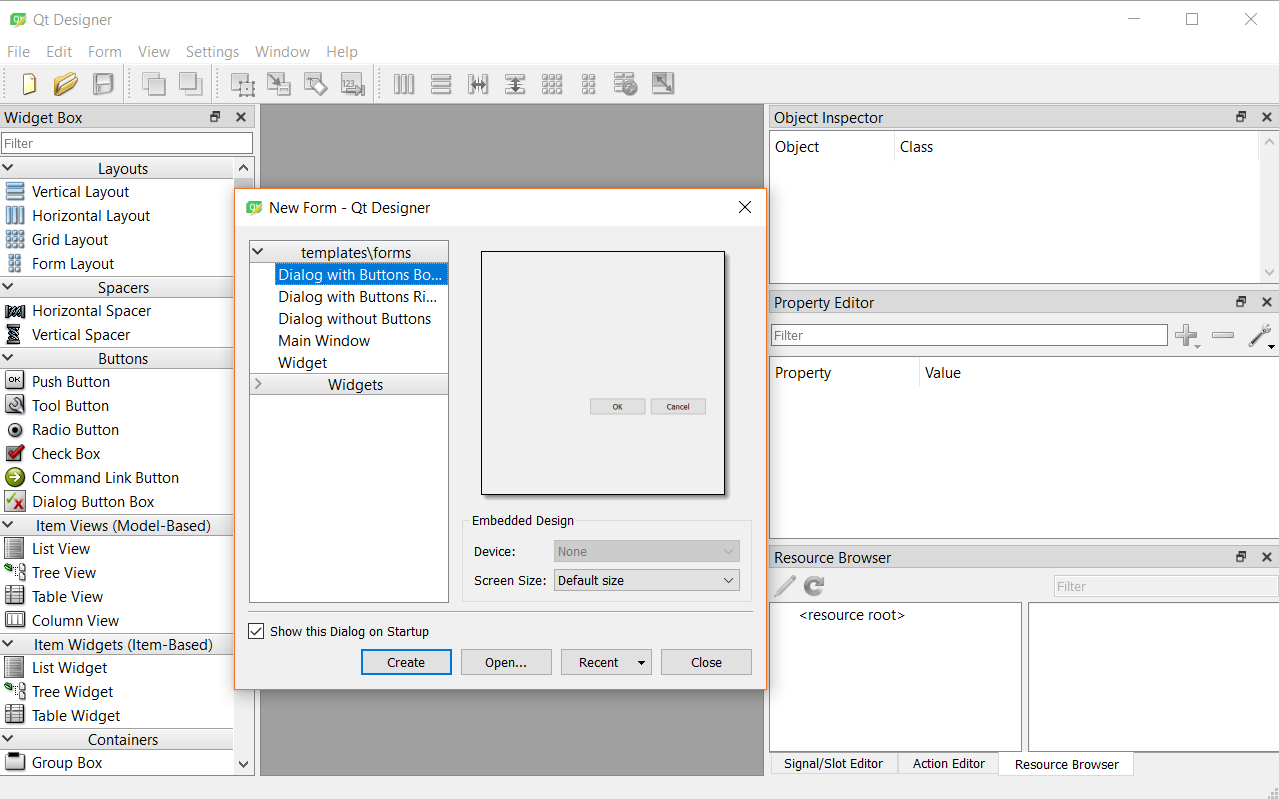
Путь\_к\_папке\_где\_установлен\_Python\Lib\site-packages\pyqt5\_tools\Qt\bin\designer.exe

или (для старых версий):

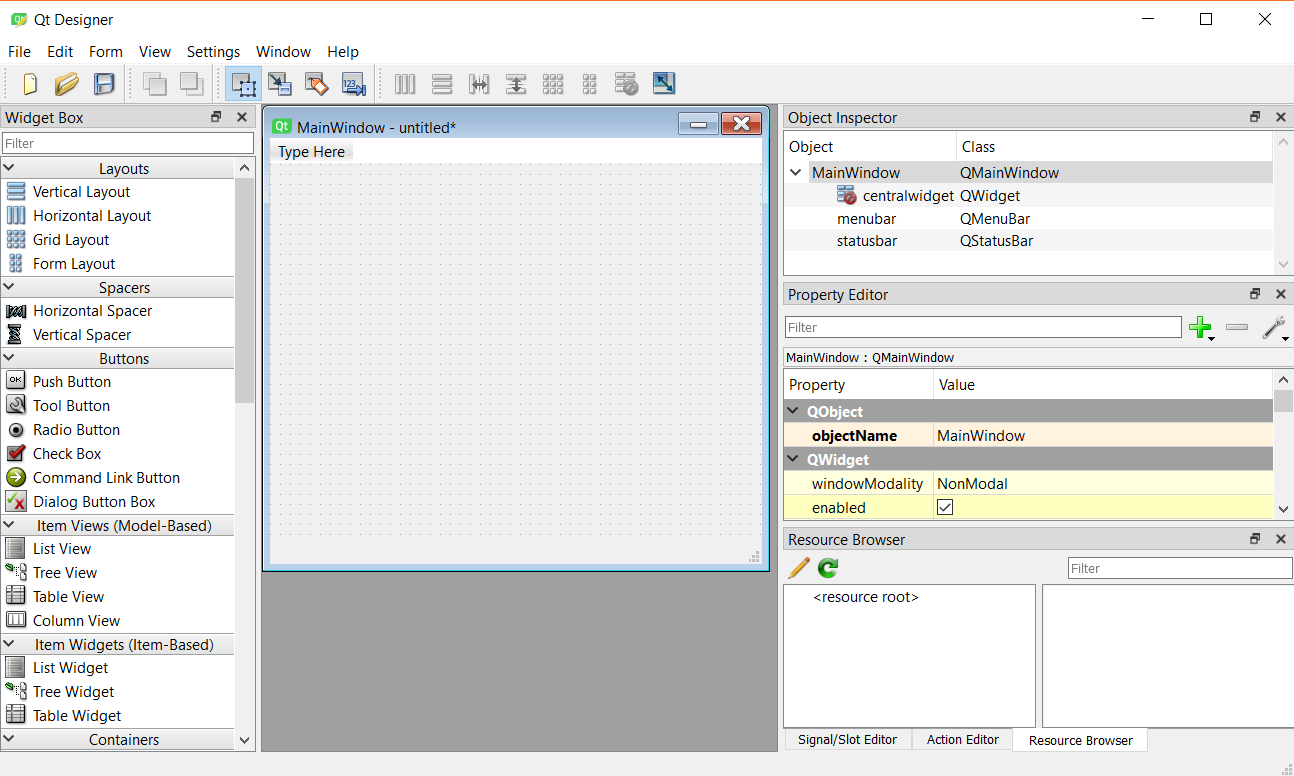
Путь\_к\_папке\_где\_установлен\_Python\Lib\site-packages\pyqt5\_tools\designer.exe

Создайте ярлык к этой программе, поскольку использовать ее придется часто.

Давайте запустим программу и посмотрим, что в ней можно сделать.



При запуске открывается окно с предложением выбрать шаблон для формы или виджет, на основе которого мы будем делать свой интерфейс. Выберем **Main Window**. Откроется пустое окно.



Теперь рассмотрим, что у нас есть, кроме пустой формы, которую мы будем заполнять.

Слева — меню виджетов, **Widget Box**. В нем они сгруппированы в зависимости от их функциональности. Отдельно кнопки, отдельно виджеты для ввода данных и так далее.

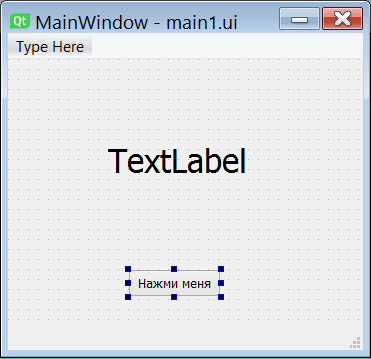
Справа — Инспектор объектов (**Object Inspector**), Редактор свойств (**Property Editor**) и Браузер ресурсов (**Resourse Browser**). Остановимся на первых двух. В **Инспекторе объектов** отображается информация об используемых виджетах. Для каждого виджета указывается его имя и класс. Кроме того, можно увидеть иерархическую структуру всего интерфейса.

Чтобы разместить виджет на форме, его надо просто перетащить из меню виджетов. При этом информация об этом виджете автоматически появится в Инспекторе объектов.

**Редактор свойств** помогает изменять значения тех или иных атрибутов виджета (например, текст или размер). Расположение свойств меняется кнопкой с изображением гаечного ключа.

Попробуйте создать простейший интерфейс из кнопки (PushButton) и текстового поля (TextLabel), поиграйте с **Редактором свойств**, разберитесь, где поменять название и размеры кнопки, как изменить шрифт в TextLabel. После того как у вас вышло что-то похожее на картинку, ниже сохраните полученный дизайн. Для сохранения в меню **Файл** выберите вкладку **Сохранить как**, найдите папку проекта и впишите имя.

Помните: название объекта (атрибут objectName) и текст, который может быть на нем показан (атрибут text), — это разные вещи.



Давайте посмотрим, как выглядит наш дизайн с точки зрения компьютера. Для этого мы должны открыть созданный нами файл с помощью любого текстового редактора. Лучше использовать не просто «Блокнот», а, например, SublimeText. Когда мы откроем наш файл, увидим, что это просто XML-документ, описывающий всю структуру GUI.

**Как подключить интерфейс к основной программе**

Теперь у нас есть дизайн, но нам надо подключить его к программе. Для этого есть два способа.

**Способ первый: загрузка ui-файла**

import sys

from PyQt5 import uic

from PyQt5.QtWidgets import QApplication, QWidget, QMainWindow

class MyWidget(QMainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

uic.loadUi('main1.ui',self)

self.pushButton.clicked.connect(self.run)

def run(self):

self.label.setText("OK")

app = QApplication(sys.argv)

ex = MyWidget()

ex.show()

sys.exit(app.exec\_())

Для загрузки ui-файла импортируйте класс uic, а затем в конструкторе вызовите метод loadUi, где одним из параметров указывается файл с интерфейсом. В нашем примере он называется main1.ui и лежит в той же папке, что и запускаемый нами скрипт.

После выполнения метода loadUi все виджеты становятся полями класса, имена для которых мы задали в редакторе свойств. Затем можно работать с ними точно так же, как в предыдущем уроке. Что мы и делаем, подключая обработчик нажатия кнопки.

**Способ второй: использование pyuic**

Независимо от способа подключения, ui-файл конвертируется в класс на Python. В первом случае это происходит при запуске (метод loadUi), но замедляет программу. Поэтому часто файл конвертируют заранее вручную.

Для этого нужна консольная утилита **pyuic5**.

Чтобы ею воспользоваться, нужно открыть командную строку (терминал), перейти в ту папку, где лежит ваш ui-файл, и выполнить следующую команду:

pyuic5 ui\_file.ui -o ui\_file.py

В той же папке появится «питоновский» файл с классом, аналогичным тому, что мы писали в первом уроке. Но в нем нет никаких функций, и, если мы попробуем его запустить, ничего не произойдет.

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QMainWindow

from ui\_file import Ui\_MainWindow

class MyWidget(QMainWindow,Ui\_MainWindow):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setupUi(self)

self.pushButton.clicked.connect(self.run)

def run(self):

self.label.setText("OK")

app = QApplication(sys.argv)

ex = MyWidget()

ex.show()

sys.exit(app.exec\_())

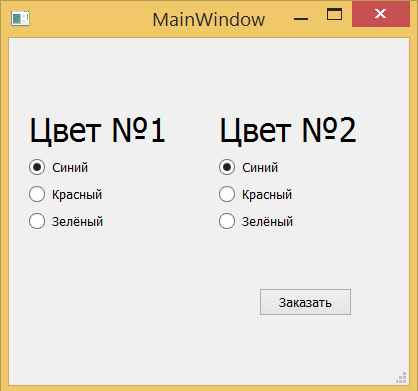
Здесь вы столкнулись с множественным наследованием. Предками нашего класса являются и QMainWindow, и Ui\_MainWindow. От первого унаследованы методы, а от второго — дизайн. В остальном работа схожа: мы вызываем метод setupUi из Ui\_MainWindow, а затем просто работаем с полями.

**Размещение виджетов**

В QtDesigner можно размещать виджеты на экране не хаотично, а упорядоченно. Этот процесс называется разметкой.

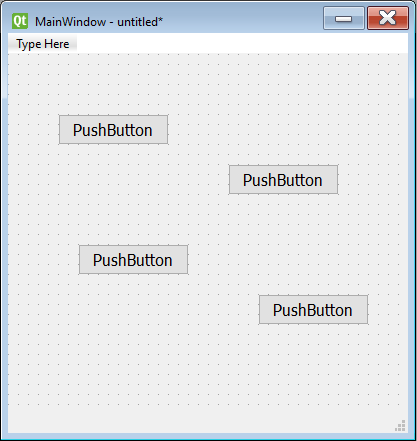
Для этого есть виджеты, которые называются **Layout** (Разметка): Vertical, Horizontal, Grid и Form Layout.

Но они нужны не только для красивого размещения элементов интерфейса, но и для работы с Radio Button. Когда мы работаем с ними, можем выбрать только одну из кнопок. А что делать в том случае, когда у нас несколько факторов, которые нужно выбирать? Если мы просто разместим все Radio Button на нашем виджете, то никак не сможем выбрать два. А вот если часть из них поместить в какой-нибудь Layout — легко.

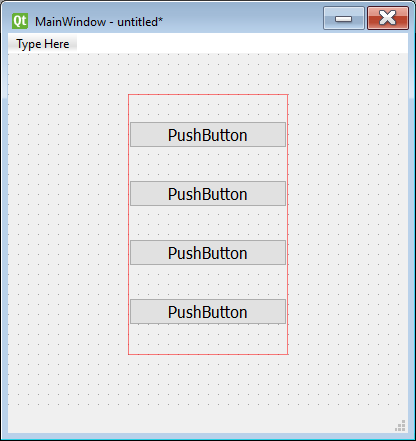


Давайте посмотрим на то, как будут выглядеть привычные нам PushButton при применении к ним различных layout-ов.

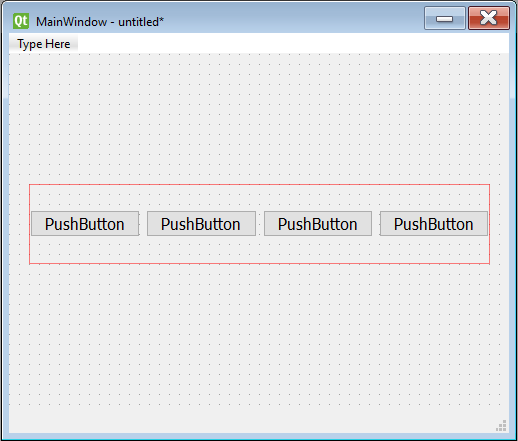
Разместим изначально наши кнопки на форме в случайном порядке.



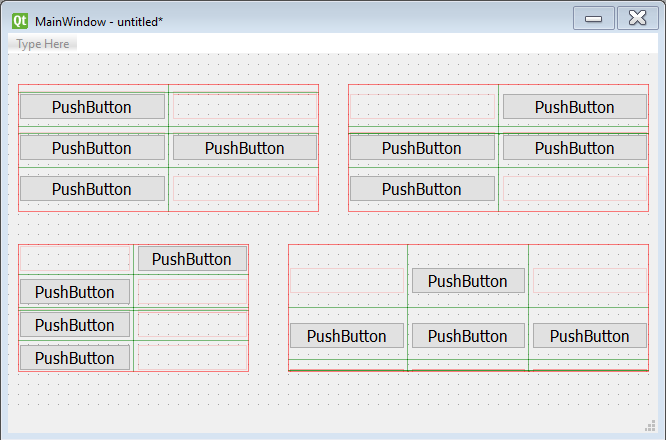
Затем поместим их внутрь **Vertical Layout**.



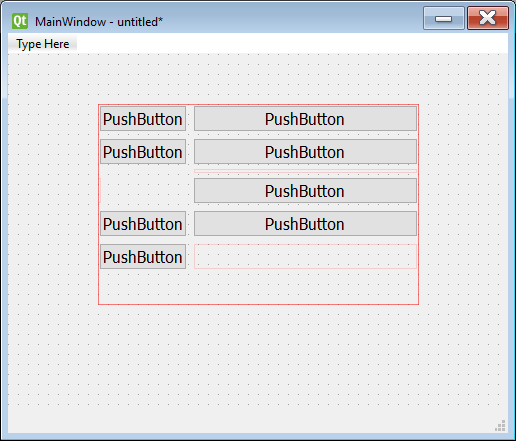
Теперь — в **Horizontal Layout**.



В случае использования **Grid Layout** появляются различные варианты:



А **Form Layout** является подвидом **Grid Layout**. У него фиксированное число столбцов, равное двум.



Чтобы добавить виджет в **Grid Layout**, можно воспользоваться функцией grid.addWidget(elem, x, y). В качестве параметров функции выступает виджет и координаты, по которым его нужно расположить.

Кроме Layout, существуют и другие группы. Чаще всего они нужны, когда у нас есть много виджетов с одинаковыми функциями. Например, у нас много Radio Button, но они все отвечают за один и тот же параметр — цвет. Чтобы объединить Radio Button в группу, необходимо выделить их, нажать на правую кнопку мыши, кликнуть пункт меню **Assign to button group** и выбрать либо пункт создания новой группы (**New button group**), либо уже созданную. Чтобы «повесить» на всю группу обработчик событий, нужно вызвать уже знакомый нам метод connect. Но теперь он применяется не к одной кнопке, а сразу к группе.

self.buttonGroup.buttonClicked.connect(self.run)

А чтобы получить список всех кнопок группы, примените метод .buttons().

[Справка](https://yandex.ru/support/lyceum-students)

Исключительное право на учебную программу и все сопутствующие ей учебные материалы, доступные в рамках проекта «Яндекс.Лицей», принадлежат АНО ДПО «ШАД». Воспроизведение, копирование, распространение и иное использование программы и материалов допустимо только с предварительного письменного согласия АНО ДПО «ШАД».

© 2018 – 2020  ООО «[Яндекс](https://yandex.ru/)»

Чаты