CME 466
Design of an Advanced Digital System
Shahim Vedaei and Khan Wahid
Winter 2024

QT Designer and PyQt5

1. Overview:

Design the UI (.ui)	**Qt Designer** How to install: https://build-system.fman.io/qt-designer-download
Convert (.ui) to (.py)	**pyuic5** How to install: *sudo apt-get install pyqt5-dev pyqt5-dev-tools* *pip3 install pyqt5-tools* How to use: *pyuic5 <input_file_.ui> -o <output_file.py>*
Import the UI (.py) in python	Import the generated file from previous step into your main code.
Use the widgets and write the backend	Import the pyqt5 package. Use the imported widgets in your code. How to install PyQt5: *sudo apt-get install python3-pyqt5* *pip3 install pyqt5* Resource: https://www.tutorialspoint.com/pyqt/index.htm

2. Qt Designer:

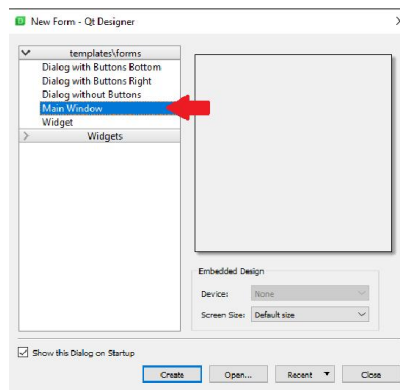Open Qt Designer. Select "Main Window", then press the "create" button.
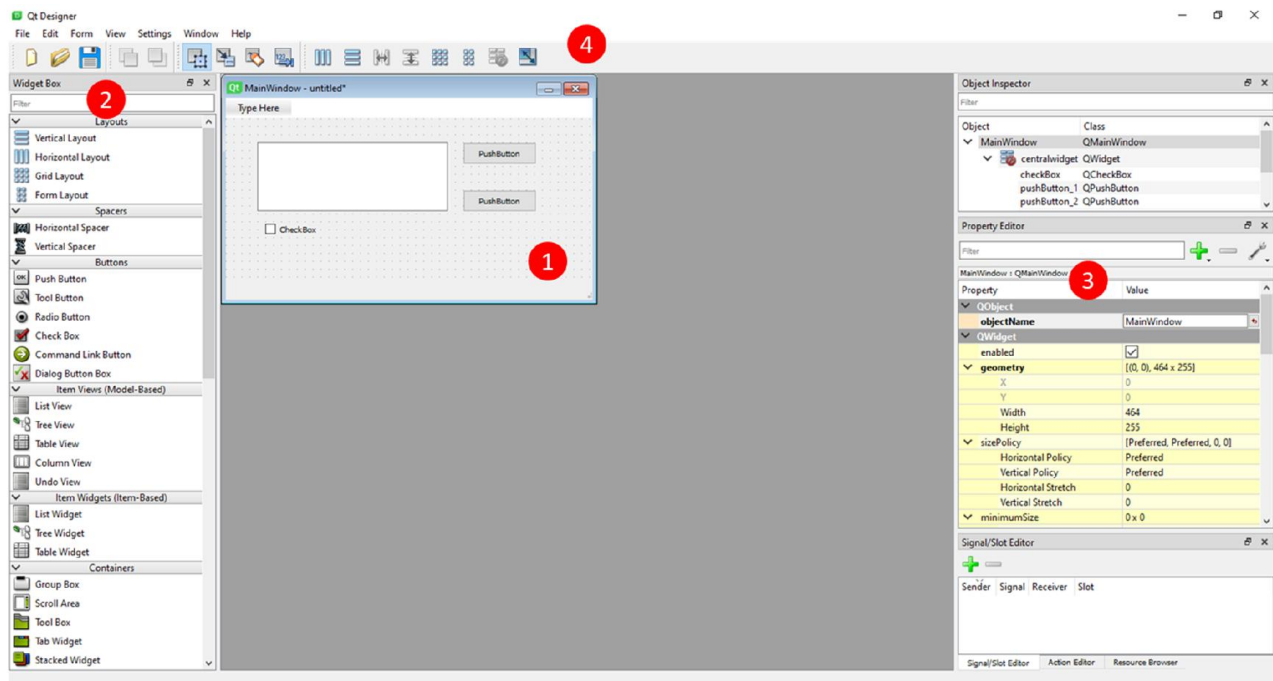

Fig. 1. Create Main Window form

Fig. 1. Qt Designer software

Fig. 1. Shows the Qt Designer software.

1- is the main window form, you are able to resize the form by clicking on the edge of this form and manually adjust the form size.

2- Section 2, is the widget box. Select the desired widgets, then drag and drop it in side the form. Place the widgets in a desired position.

3- Using Property Editor window, you can access to the selected widget's properties. The objectName is a unique name assigned to a widget that can later be used in programming.

4- The toolbar section will be used to save/ open your file. Quick access to some tools such as layouts are also provided.

2.1 Useful widgets for this Lab:
pushButton
textEdit
checkBox

Info on more widgets can be found here: https://doc.qt.io/qt-6/qwidget.html
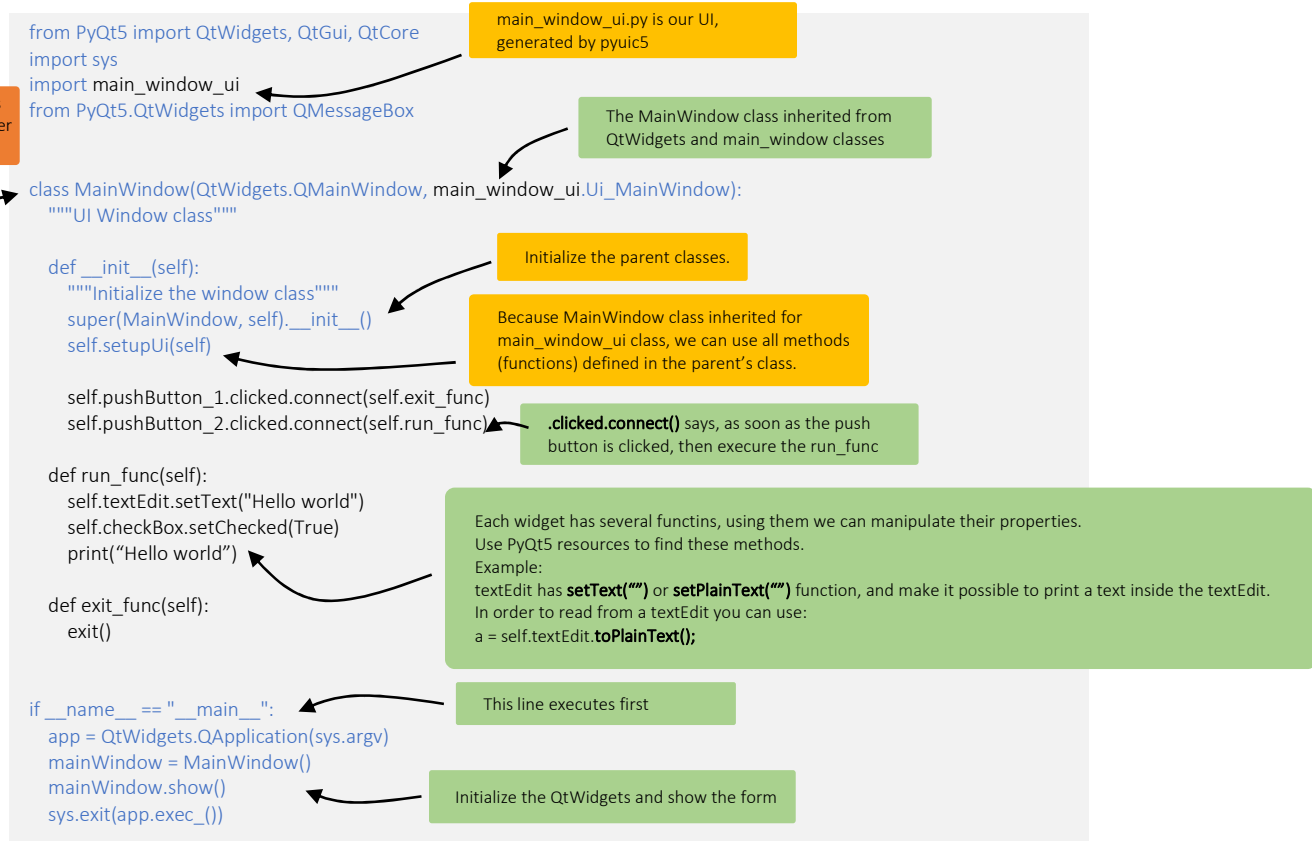
2.2 Convert .ui file to .py:
Save the design with .ui format.
Then, using the pyuic5 tool, convert .ui file to .py.

```
pyuic5 main_window.ui –o main_window_ui.py
```

In order to have a same naming style, please convert the ui and add **_ui.py** to its end. Hence, we can easily distinguish other python files from the generated ui files.

3. PyQt5: content inside the *main.py* is shown below:

```python
from PyQt5 import QtWidgets, QtGui, QtCore
import sys
import main_window_ui
from PyQt5.QtWidgets import QMessageBox


class MainWindow(QtWidgets.QMainWindow, main_window_ui.Ui_MainWindow):
    """UI Window class"""

    def __init__(self):
        """Initialize the window class"""
        super(MainWindow, self).__init__()
        self.setupUi(self)

        self.pushButton_1.clicked.connect(self.exit_func)
        self.pushButton_2.clicked.connect(self.run_func)

    def run_func(self):
        self.textEdit.setText("Hello world")
        self.checkBox.setChecked(True)
        print("Hello world")

    def exit_func(self):
        exit()


if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    mainWindow = MainWindow()
    mainWindow.show()
    sys.exit(app.exec_())
```

Annotations:
- main_window_ui.py is our UI, generated by pyuic5
- All of our codes will be run under this class
- The MainWindow class inherited from QtWidgets and main_window classes
- Initialize the parent classes.
- Because MainWindow class inherited for main_window_ui class, we can use all methods (functions) defined in the parent's class.
- **.clicked.connect()** says, as soon as the push button is clicked, then execure the run_func
- Each widget has several functins, using them we can manipulate their properties. Use PyQt5 resources to find these methods. Example: textEdit has **setText("")** or **setPlainText("")** function, and make it possible to print a text inside the textEdit. In order to read from a textEdit you can use: a = self.textEdit.**toPlainText();**
- This line executes first
- Initialize the QtWidgets and show the form

You can use the blue lines as a template to show the form window, then add your own code inside the MainWindow class.

4. Sample code: some sample codes have been posted in canvas. You can download and play with them to get yourself familiar with.

You can change the background colour of a frame:

self.frame1.setStyleSheet("background-color: rgb(170, 0, 0)")

You can also display an image inside a frame:

self.frame1.setStyleSheet("background-image: url(image.jpg); background-position: center;")

You can display a string inside a text box:

self.textEdit1.setText("hello world")

You can check a checkbox:

self.checkBox1.setChecked(True)