Introduction to PyQt

Building Graphical User Interfaces in Python with PyQt

What is PyQt5?

- ➔ *Qt5 is an Application Framework from a Finnish company called Qt Group PLC, once part of Digia. Qt was first released in 1995 by Trolltech.*
- ➔ *An Application Framework is a Software Framework driven by the structure imposed on projects by the GUI system used.*
- ➔ *PyQt5 is a set of Python bindings for Qt5 and can be used with Python 2.x or 3.x.*
- ➔ *PyQt5 is developed by a British company called Riverbank computing.*
- ➔ *Qt is more than a GUI toolkit. It includes abstractions of network sockets, threads, Unicode, regular expressions, SQL databases, SVG, OpenGL, XML, a fully functional web browser, a help system, a multimedia framework, as well as a rich collection of GUI widgets.*
- ➔ *Qt classes employ a signal/slot mechanism for communicating between objects that is type safe but loosely coupled making it easy to create reusable software components.*

Installation and Modules

- ➔ *Comes with Anaconda, for miniconda, “conda install PyQt5”*
- ➔ *Without anaconda/miniconda, “sudo pip install pyqt5”*
- ➔ *PyQt5 classes divided into modules, QtCore, QtGui, QtWidgets, QtMultimedia, QtBluetooth, QtNetwork, QtPositioning, Enginio, QtWebSockets, QtWebKit, QtWebKitWidgets, QtXml, QtSvg, QtSql, QtTest.*
- ➔ *QtCore - non-GUI functionality - time, files, directories, data types, streams, URLs, mime types, threads/processes.*
- ➔ *QtGui - windowing system integration, event handling, 2D graphics, basic imaging, fonts and text.*
- ➔ *QtWidgets - User Interface elements for building desktop UIs.*
- ➔ *QtWebKit - classes for a web browser implementation based on WebKit2.*
- ➔ *QtWebKitWidgets - WebKit1 based implementation of web browser.*

A Few More Modules

- *QtXml - Classes for working with XML - SAX and DOM APIs.*
- *QtSvg - Scalable Vector Graphics - for 2D graphics.*
- *QtSql - Classes for working with databases.*

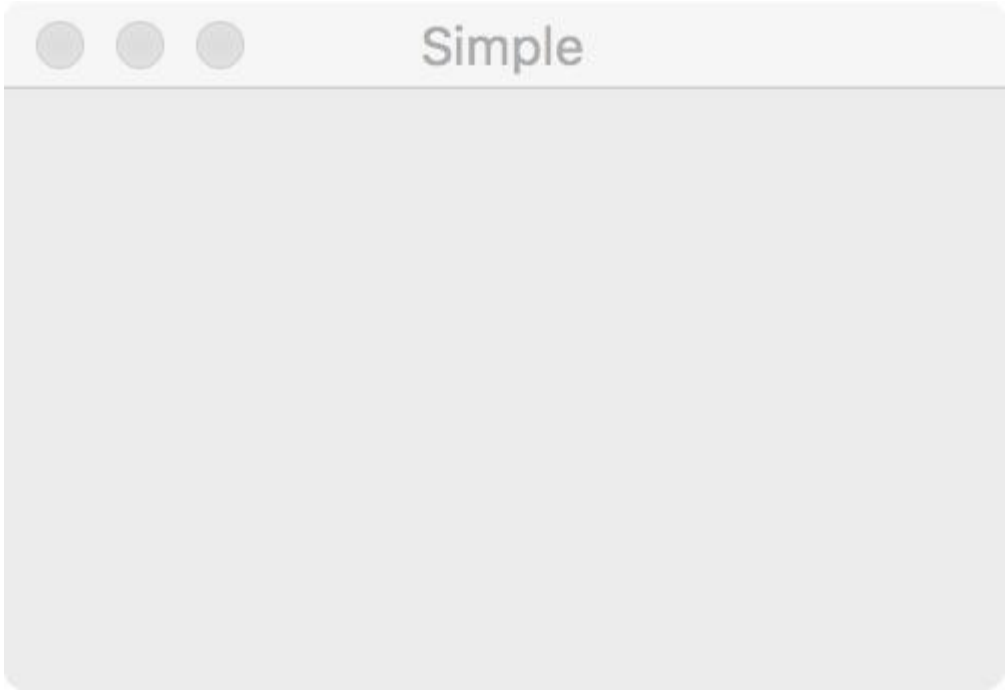
first.py

```
import sys
from PyQt5.QtWidgets import QApplication, QWidget

app = QApplication(sys.argv)   # Every PyQt5 App must create application object

w = QWidget()                  # QWidget is the base class for all user interface objects
                               # A widget with no parent is called a 'window'
w.resize(250, 150)             # Set the size of the window
w.move(300, 300)               # Set the position of the window on the screen
w.setWindowTitle('Simple')     # Set the title displayed at the top of the widget
w.show()                       # Display the window on the screen.
sys.exit(app.exec_())          # Start the mainloop of the application, ends if widget
                               # destroyed or exit() called.  Sys.exit() gives clean exit
```

first.py running



second.py

```
import sys
from PyQt5.QtWidgets import (QWidget,
                             QPushButton, QApplication)

class Example(QWidget):

    def __init__(self):
        super().__init__()

        self.initUI()

    def initUI(self):

        qbtn = QPushButton('Quit', self)  # self=parent
        qbtn.clicked.connect(
                             QApplication.instance().quit)
        qbtn.resize(qbtn.sizeHint())
        qbtn.move(50, 50)

        self.setGeometry(300, 300, 250, 150)
        self.setWindowTitle('Quit button')
        self.show()


if __name__ == '__main__':

    app = QApplication(sys.argv)
    ex = Example()
    sys.exit(app.exec_())
```

Second.py running

third.py

```
import sys
from PyQt5.QtWidgets import QWidget, QMessageBox, QApplication


class Example(QWidget):

    def __init__(self):
        super().__init__()

        self.initUI()


    def initUI(self):

        self.setGeometry(300, 300, 250, 150)
        self.setWindowTitle('Message box')
        self.show()
```

third.py continued

```
    def closeEvent(self, event):          # override the default closeEvent event handler

        reply = QMessageBox.question(self, 'Message',
            "Are you sure to quit?", QMessageBox.Yes |
            QMessageBox.No, QMessageBox.No)

        if reply == QMessageBox.Yes:
            event.accept()
        else:
            event.ignore()

if __name__ == '__main__':

    app = QApplication(sys.argv)
    ex = Example()
    sys.exit(app.exec_())
```

third.py running

Inherit from QMainWindow instead of QWidget

We can use QMainWindow in place of QWidget.

A QMainWindow has various pieces, invisible until set up.

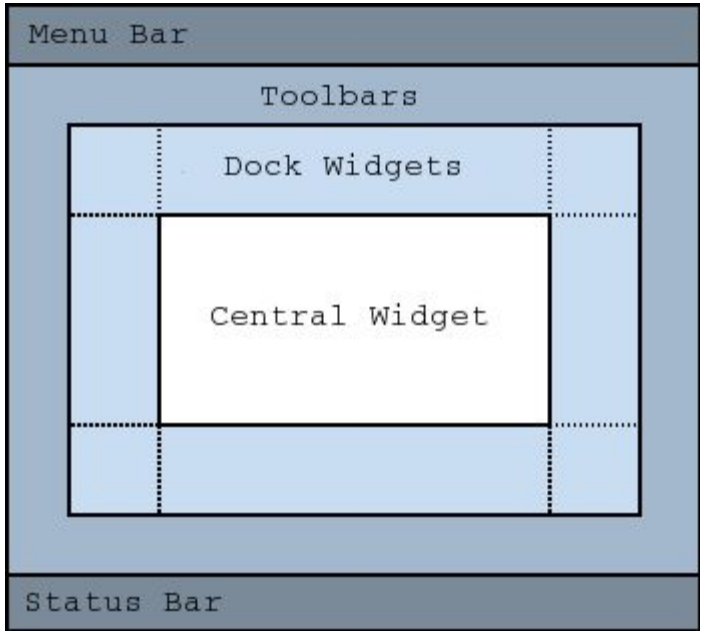
StatusBar : self.statusBar()
MenuBar : self.menuBar()
ToolBar: self.addToolBar()

Dock Widgets

Central Widget is like QWidget.

```
class MyProgram(QMainWindow):

self.main_widget = QWidget(self)
self.setCentralWidget(self.main_widget)
```



fourth.py

```
class Example(QMainWindow):

    def __init__(self):
        super().__init__()

        self.initUI()


    def initUI(self):

        self.statusBar().showMessage('Ready')

        self.setGeometry(300, 300, 250, 150)
        self.setWindowTitle('Statusbar')
        self.show()
```

forth.py running

Next Steps

- Layouts
  - Absolute Positioning
  - Box Layout
  - QGridLayout
- Events
  - Signals and Slots
  - Reimplementing event handler
  - Event Objects
  - Event Senders
  - Emitting Signals
- Dialogs
- Widgets
  - QCheckBox
  - Toggle Button
  - QSlider
  - Etc.
- Pixelmaps, QSplitter, QComboBox, Drag and Drop, etc.

Next Steps, continued

- Painting, QPen, QBrush, Bézier curve, etc.
- Embedding MatPlotLib plots in PyQt
- Custom Widgets
- More detailed examples

PyQt Browser

```
from PyQt5.QtCore import QUrl
from PyQt5.QtWidgets import *
from PyQt5.QtWebEngineWidgets import (QWebEngineSettings,
                                      QWebEngineView, QWebEnginePage)

class Web(QWebEngineView):

    def load(self, url):
        self.setUrl(QUrl(url))

    def adjustTitle(self):
        self.setWindowTitle(self.title())

    def disableJS(self):
        settings = QWebEngineSettings.globalSettings()
        settings.setAttribute(QWebEngineSettings.JavascriptEnabled, False)
```

PyQt Browser, Continued

```
class Main(QWidget):

    def __init__(self):
        super().__init__()
        self.initUI()

    def initUI(self):
        self.setWindowTitle('Name')
        self.setWindowIcon(QIcon('icon.png'))
        self.setGeometry(100, 100, 1400, 1000)

        web = Web()
        web.load("http://lowell.edu")

        lay = QVBoxLayout(self)
        lay.addWidget(web)
```

PyQt Browser, running