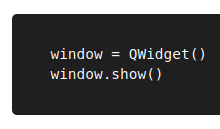
QT Notes

from PyQt5.QtWidgets

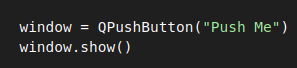
import

QApplication, - Application handler

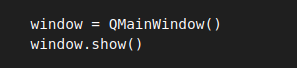
QWidget – creating window



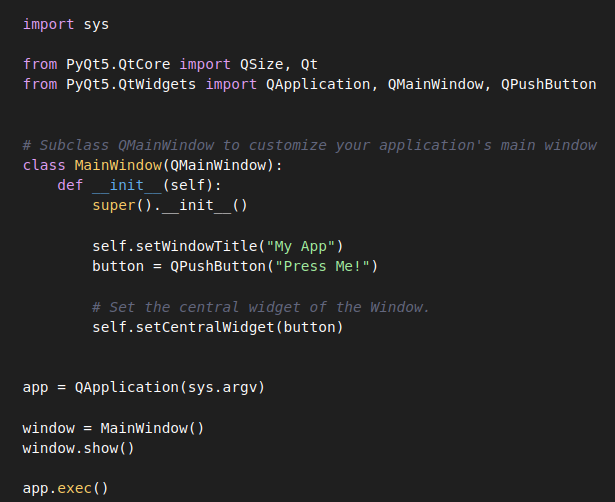
QPushButton – creating button



QMainWindow – creating window (has lot of features)



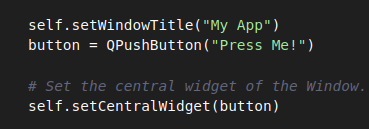
Creating a class called MainWindow(), and passing inside the widget QMainWindow, Inside this class we specify the dialog box using \_\_init\_\_(self)



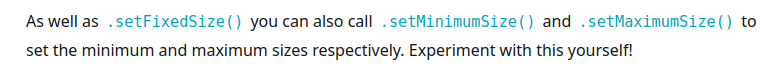
**QMainWindow() :**

**Self.setWindowTitle** - to add title

**Self.setCentralwidget(button)** - to add in the center, by default it covers enter window



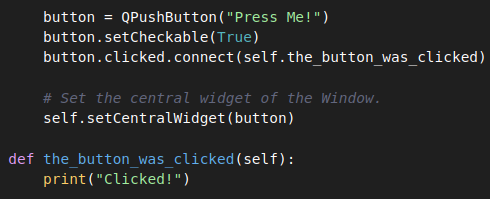
**Self.setFixedSize(QSize(400, 300))** - fixed size window, unable to resize, use QSize to resize





**Signals :**

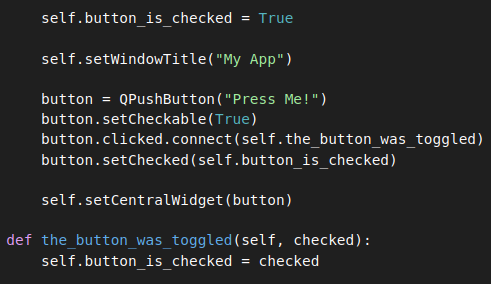
**button.clicked.connect()** and calls to the function below



**Button.setCheckable(True)** - is default false, when button is clicked by providing parameter (checked – anything we can give) ,in the function it shows true or false, it can used as in backed.

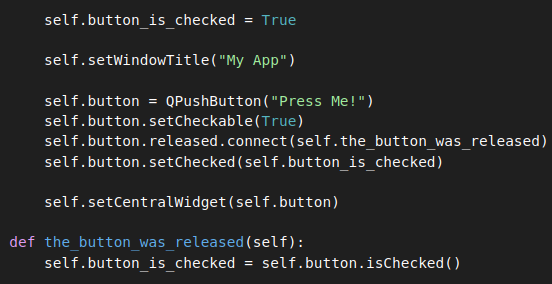


In this below image, after button is clicked, (true or false) , we can store the response, with that we can do what we want



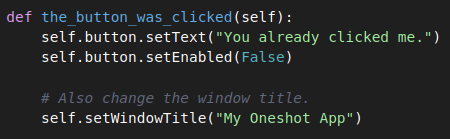
**Self.button.released(),** Below image, its a same thing, but we directly get the respone. Released function doesnt send check results.





**Self.button.SetText(), self.button.setEnabled()**

In the below image, The text of the button is changed by passing a str to .setText(). To disable a button call .setEnabled() with False. Also, we can change the window title when button is pushed.



In this below program, if title is changed, we cant click the button.

**Self.windowTitleChanged** is only triggered when titl changes





**Qtwidgets - QLabel()**

Self.label = QLabel()

Self.label.**setText(**“Hello”) - set text

**QLineEdit()**

Self.input = QLineEdit()

Self.input.**textChanged.**connect(self.label.setText) - if text changed

**QVBoxLayout()**

layout = QVBoxLayout()

layout.**addwidget(self.input)**

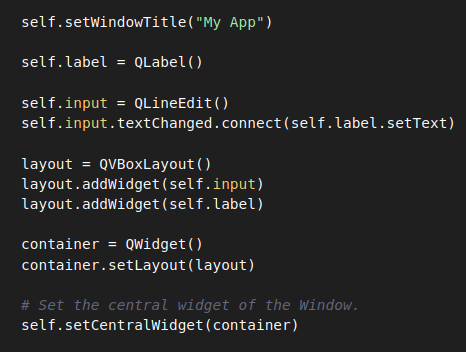
layout.**addwidget(self.label)**

**Qwidget()**

Containter = Qwidget()

Container.**setLayout**(layout)

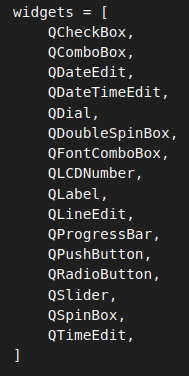
It has more to see, it has QLabel, QLineEdit, QVBoxLayout, Qwidget that connects everything.



**Widgets :**

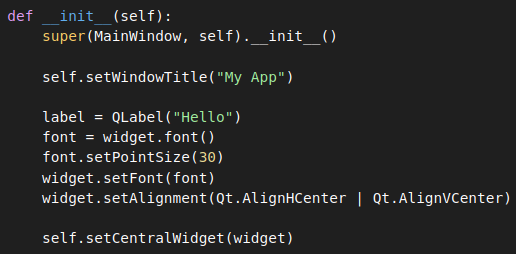
from PyQt5.QtWidgets import

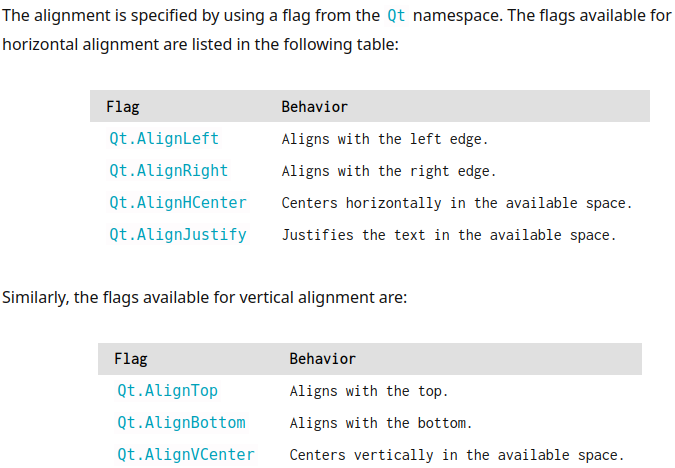


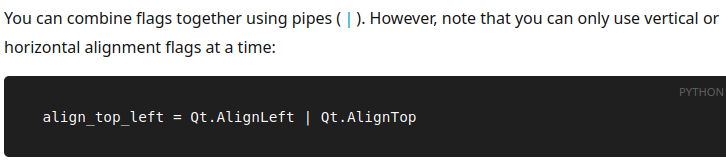
 

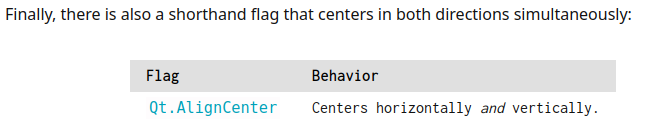
**Qlabel():**

**Label.setAlignment() -** We can align the text, it is useful to be stable

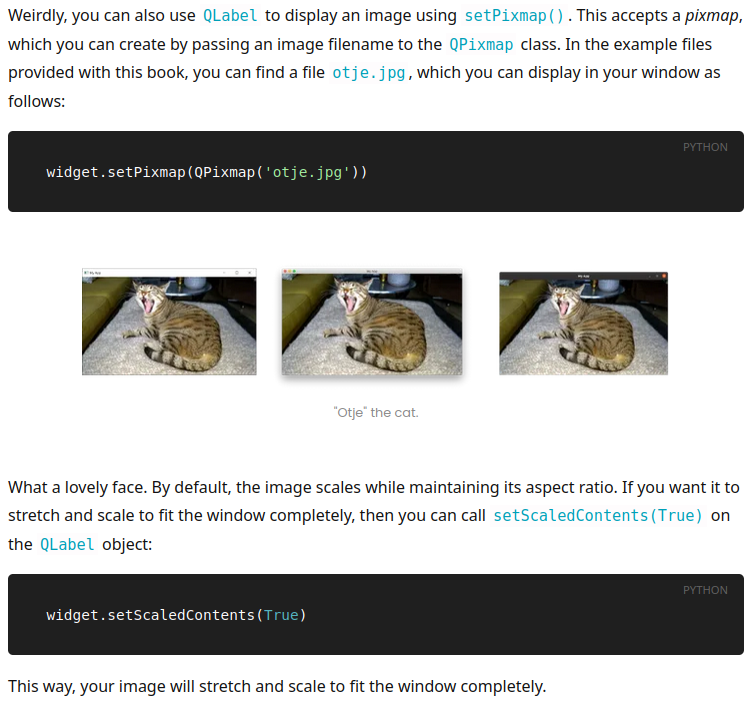








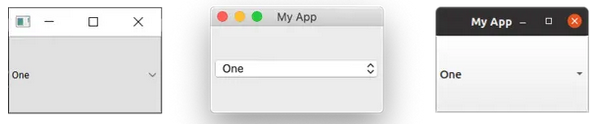
**SetPixmap – to add pictures using Qlabel**



**Qcheckbox – visit in the browser if u want to use, (PyQt5 widgets)**

**Qcombobox – used to show the option, (check website for more)**





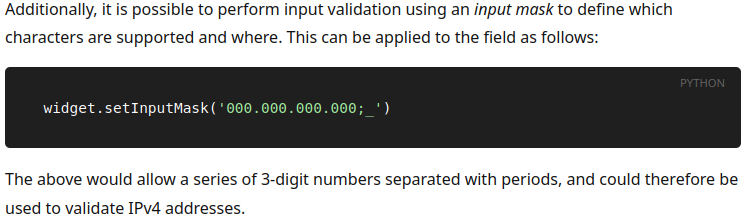
**QListwidget – same as Qcombobox, but we can scroll this**



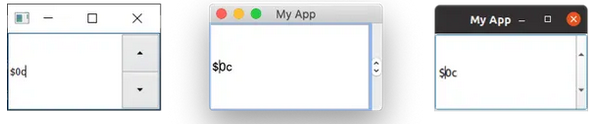
**Qline edit – used to edit the line**







**Qspinbox, Qdoublespinbox** – we can put numbers (see website)



**QSlider** – slide widget (see widget)

**Setting colour** – what this code does is, we can colour the entire screen by using color widget,

the class color can accept any colour, - if we give inside the color class.

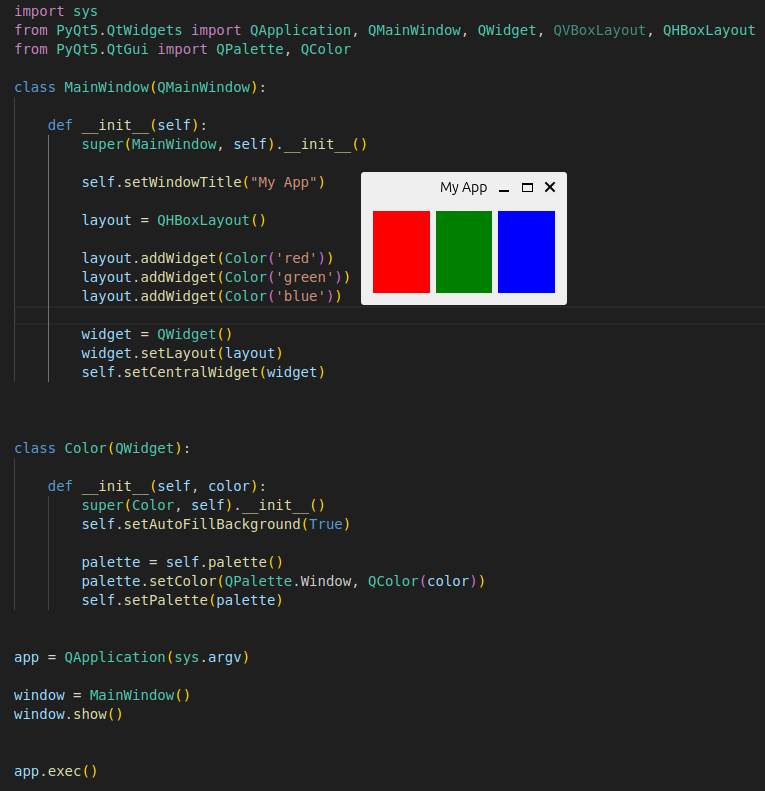
To add it in the main window, we have to use **setCentralwidget()**



**QVboxlayout** – it will add it to the qmainwindow vertically,

QHboxlayout – it will add it to the qmainwindow horizontally,

To add it in the main window, we have to use **setCentralwidget()**

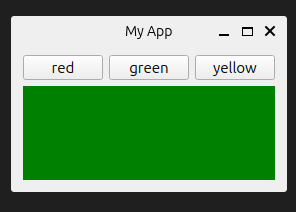


The layout has **layout.addwidget(), widget.setlayout(),**

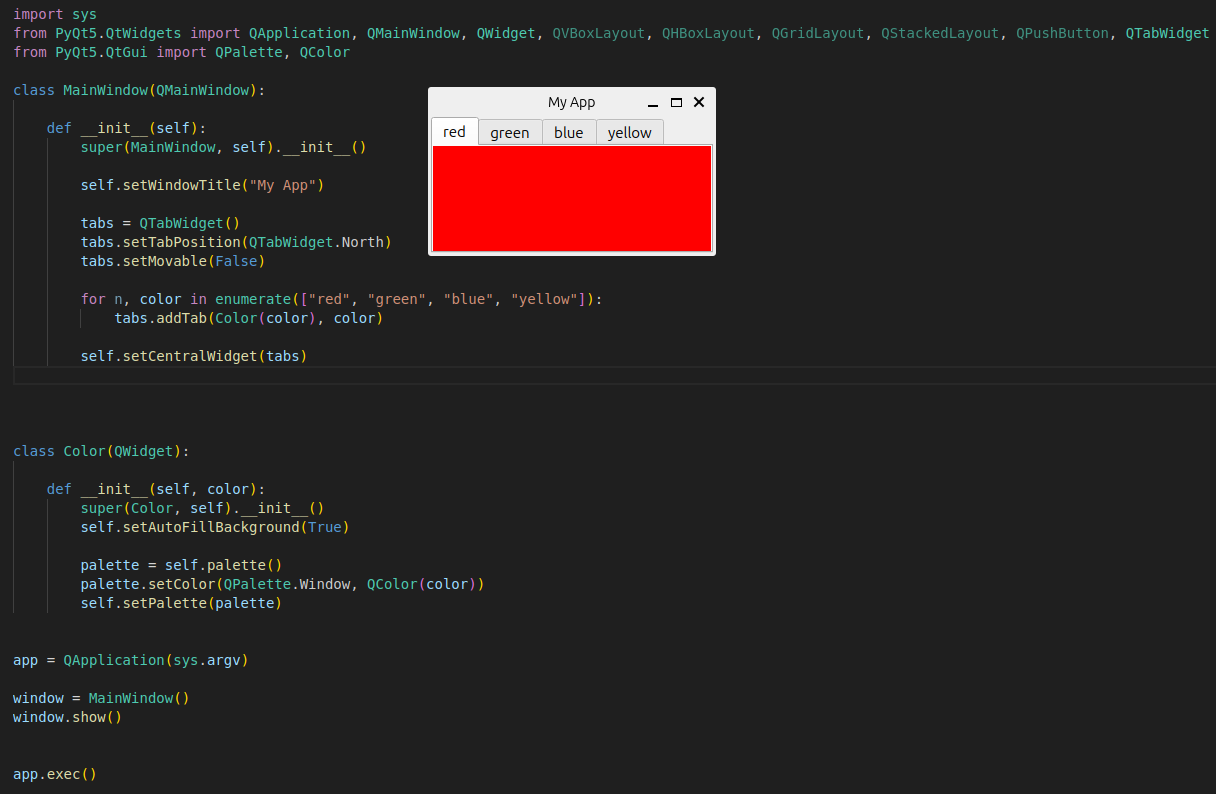
**Nestedlayout** – add multiple widgets in one dialog box(Qwidget) - see in the website

**Qgridlayout** – used to place the widgets in certain place. Like grid, (maybe useful – see in the website)

**Qstackedlayout** – add one layout on top of other layout, (may be useful, see in the website)



**Qtabwidget** – it has tab feature (see website)



To create a small dialog box:

