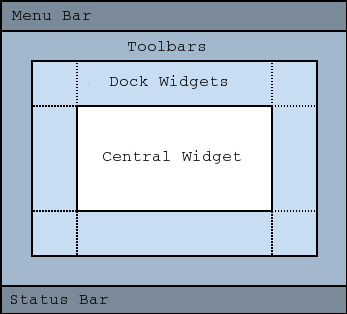
Qt Main Window Framework[**¶**](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QMainWindow.html#qt-main-window-framework)

A main window provides a framework for building an application’s user interface. Qt has [QMainWindow](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QMainWindow.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QMainWindow" \o "PySide2.QtWidgets.PySide2.QtWidgets.QMainWindow) and its [related classes](https://doc.qt.io/qtforpython-5/overviews/widget-classes.html#widgets-classes) for main window management. [QMainWindow](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QMainWindow.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QMainWindow" \o "PySide2.QtWidgets.PySide2.QtWidgets.QMainWindow) has its own layout to which you can add [QToolBar](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QToolBar.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QToolBar" \o "PySide2.QtWidgets.PySide2.QtWidgets.QToolBar) s, [QDockWidget](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QDockWidget.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QDockWidget" \o "PySide2.QtWidgets.PySide2.QtWidgets.QDockWidget) s, a [QMenuBar](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QMenuBar.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QMenuBar" \o "PySide2.QtWidgets.PySide2.QtWidgets.QMenuBar) , and a [QStatusBar](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QStatusBar.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QStatusBar" \o "PySide2.QtWidgets.PySide2.QtWidgets.QStatusBar) . The layout has a center area that can be occupied by any kind of widget. You can see an image of the layout below.



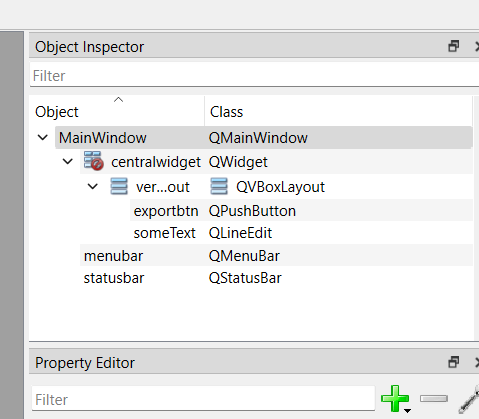
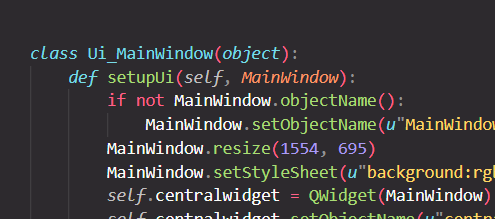
**Note**

Creating a main window without a central widget is not supported. You must have a central widget even if it is just a placeholder.

### Creating Main Window Components

A central widget will typically be a standard Qt widget such as a [QTextEdit](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QTextEdit.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QTextEdit" \o "PySide2.QtWidgets.PySide2.QtWidgets.QTextEdit) or a [QGraphicsView](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QGraphicsView.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QGraphicsView" \o "PySide2.QtWidgets.PySide2.QtWidgets.QGraphicsView) . Custom widgets can also be used for advanced applications. You set the central widget with setCentralWidget() .

Main windows have either a single (SDI) or multiple (MDI) document interface. You create MDI applications in Qt by using a [QMdiArea](https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QMdiArea.html" \l "PySide2.QtWidgets.PySide2.QtWidgets.QMdiArea" \o "PySide2.QtWidgets.PySide2.QtWidgets.QMdiArea) as the central widget.



1. We first need to create mainwindow object in order to build our application.

  MainWindow = QMainWindow()

1. Create object of Ui\_MainWindow()

ui = Ui\_MainWindow()

1. Call setupUi() method of class Ui\_MainWindow() and pass our MainWindow object to it.

ui.setupUi(MainWindow)

setupUi() method setups the MainWindow object according to our design.

1. Show MainWindow obj

MainWindow.show()

<https://www.youtube.com/watch?v=OGVuD0-5INY>

<https://stackoverflow.com/questions/29541524/ui-mainwindow-has-no-attribute-show>

<https://forum.qt.io/topic/138983/attributeerror-ui_mainwindow-object-has-no-attribute-frame_left_menu-solved>

Qt provides four classes for handling image data: [QImage](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QImage.html" \l "PySide2.QtGui.PySide2.QtGui.QImage" \o "PySide2.QtGui.PySide2.QtGui.QImage) , [QPixmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPixmap.html" \l "PySide2.QtGui.PySide2.QtGui.QPixmap" \o "PySide2.QtGui.PySide2.QtGui.QPixmap) , [QBitmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QBitmap.html" \l "PySide2.QtGui.PySide2.QtGui.QBitmap" \o "PySide2.QtGui.PySide2.QtGui.QBitmap) and [QPicture](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPicture.html" \l "PySide2.QtGui.PySide2.QtGui.QPicture" \o "PySide2.QtGui.PySide2.QtGui.QPicture) . [QImage](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QImage.html" \l "PySide2.QtGui.PySide2.QtGui.QImage" \o "PySide2.QtGui.PySide2.QtGui.QImage) is designed and optimized for I/O, and for direct pixel access and manipulation, while [QPixmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPixmap.html" \l "PySide2.QtGui.PySide2.QtGui.QPixmap" \o "PySide2.QtGui.PySide2.QtGui.QPixmap) is designed and optimized for showing images on screen. [QBitmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QBitmap.html" \l "PySide2.QtGui.PySide2.QtGui.QBitmap" \o "PySide2.QtGui.PySide2.QtGui.QBitmap) is only a convenience class that inherits [QPixmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPixmap.html" \l "PySide2.QtGui.PySide2.QtGui.QPixmap" \o "PySide2.QtGui.PySide2.QtGui.QPixmap) , ensuring a depth of 1. The [isQBitmap()](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPixmap.html" \l "PySide2.QtGui.PySide2.QtGui.QPixmap.isQBitmap" \o "PySide2.QtGui.PySide2.QtGui.QPixmap.isQBitmap) function returns true if a [QPixmap](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPixmap.html" \l "PySide2.QtGui.PySide2.QtGui.QPixmap" \o "PySide2.QtGui.PySide2.QtGui.QPixmap) object is really a bitmap, otherwise returns false . Finally, the [QPicture](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPicture.html" \l "PySide2.QtGui.PySide2.QtGui.QPicture" \o "PySide2.QtGui.PySide2.QtGui.QPicture) class is a paint device that records and replays [QPainter](https://doc.qt.io/qtforpython-5/PySide2/QtGui/QPainter.html" \l "PySide2.QtGui.PySide2.QtGui.QPainter" \o "PySide2.QtGui.PySide2.QtGui.QPainter) commands.

<https://www.pythonguis.com/faq/how-to-make-qsplitter-responds-to-double-clicking/>

Event Handlers in pyside

self.QPLabel.mousePressEvent = self.doSomething

and add the event parameter to doSomthing

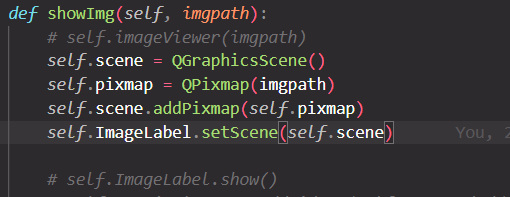
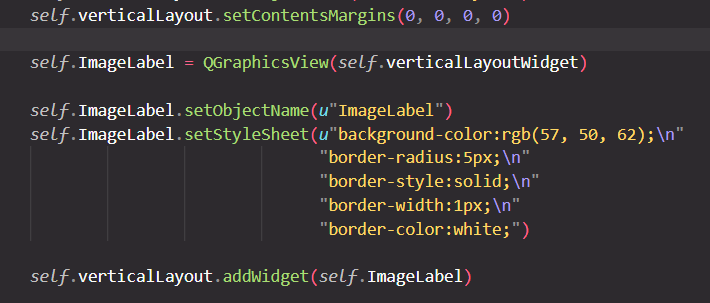
...

def doSomething(self, event):

...

<https://stackoverflow.com/questions/22232705/clickable-event-on-qlabel-in-python-using-pyqt4>

**Generated Qgraphics label logic set**

****

https://stackoverflow.com/questions/30304394/pyqt-graphics-view-next-to-another-widget