Ubuntu 下 VScode Pyqt5 GUI 编程教程

一. 配置 Pyqt5 和 Qt Designer 环境

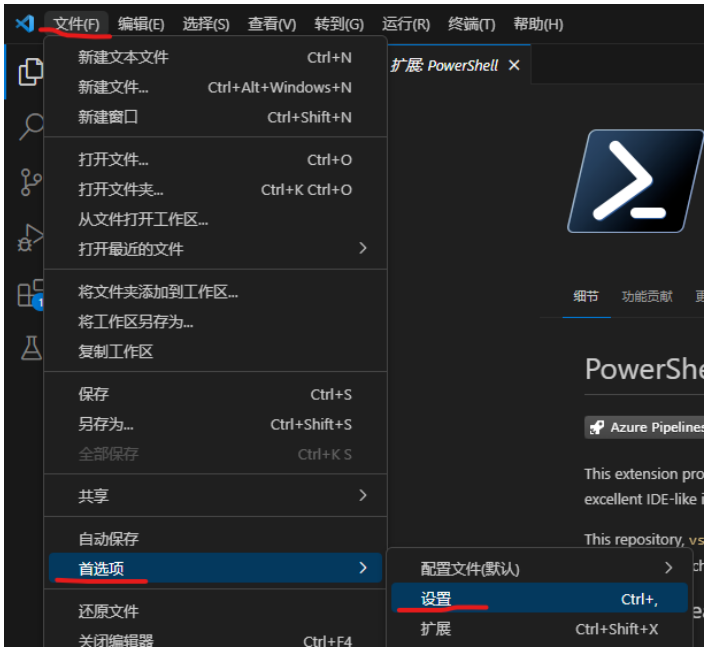
1. 在命令行安装库

```
pip install PyQt5 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

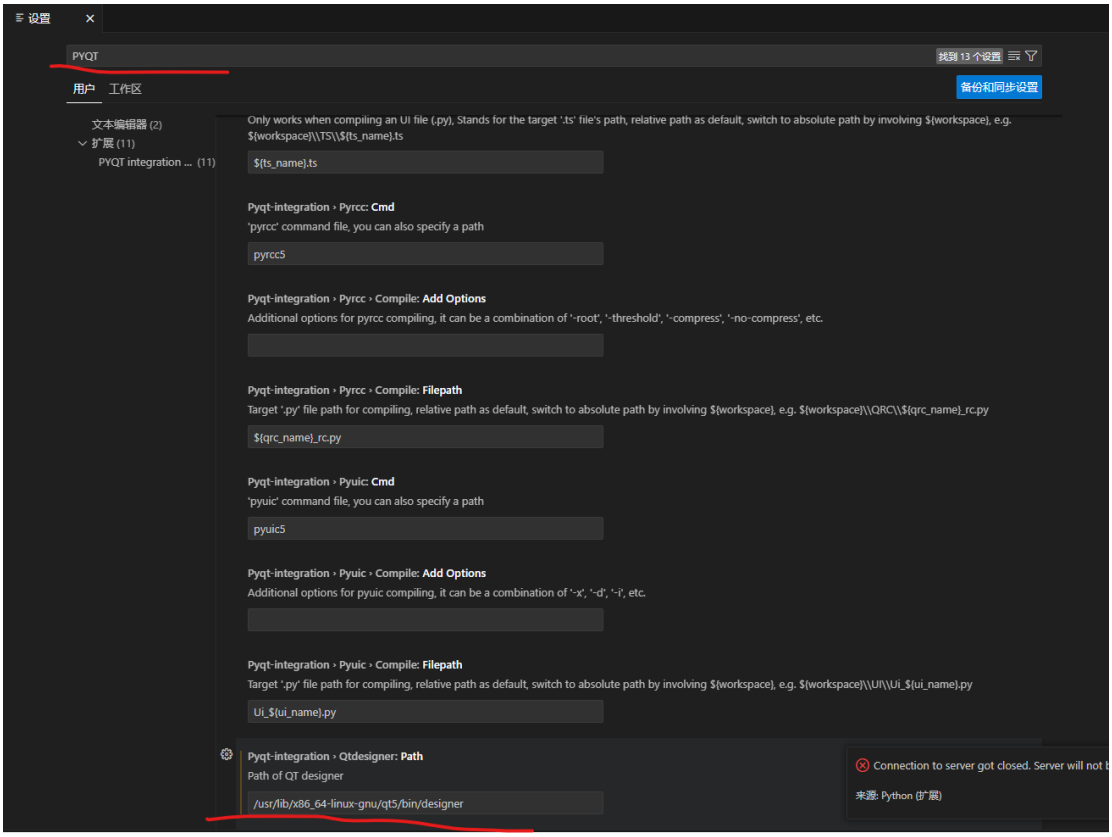
2. 在 VScode 安装“PYQT Integration”插件

3. 配置 Qt Designer 的路径

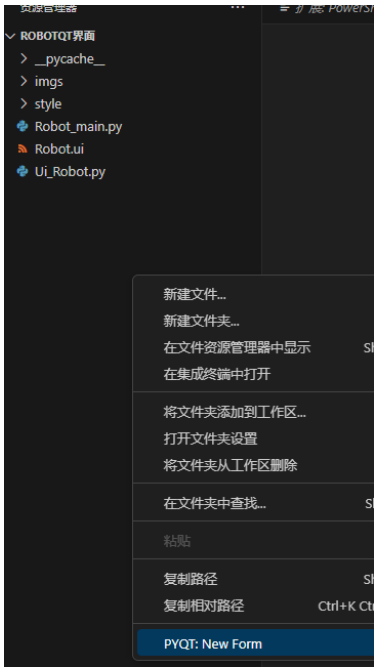
打开设置



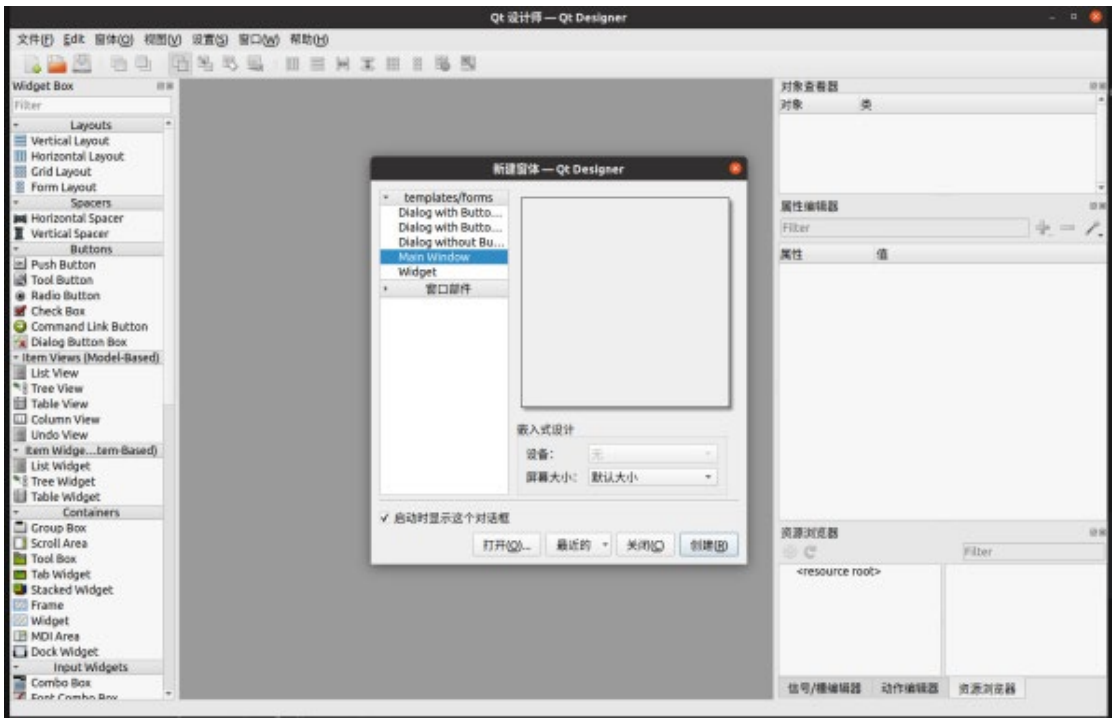
输入 pyqt，在最下一栏“Qtdesigner:path”里，输入路径 /usr/lib/x86_64-linux-gnu/qt5/bin/designer



4. 检查插件是否配置成功 在当前项目的资源管理栏里，右键，会弹出如下面板。



点击 PYQT：New Form ，会弹出“Qt Designer”的设计界面，说明配置没有问题。



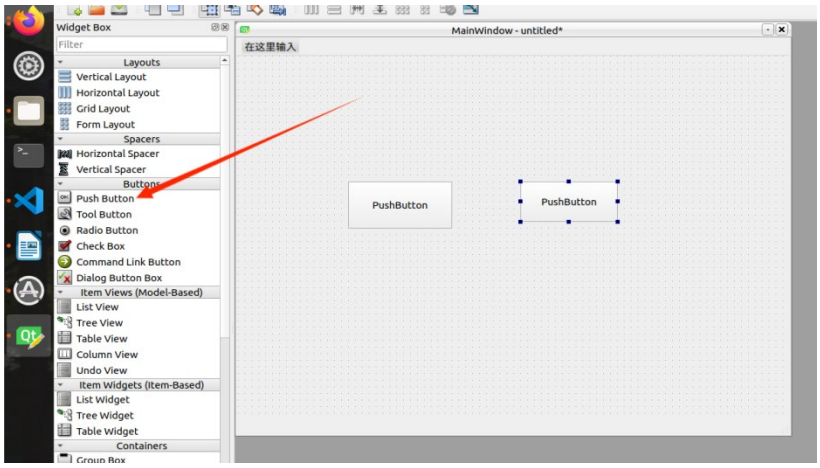
二. 编辑文件

先创建一个简单的界面并保存，再编辑文件使 QT 界面能正常运行，再开始编辑

UI 界面

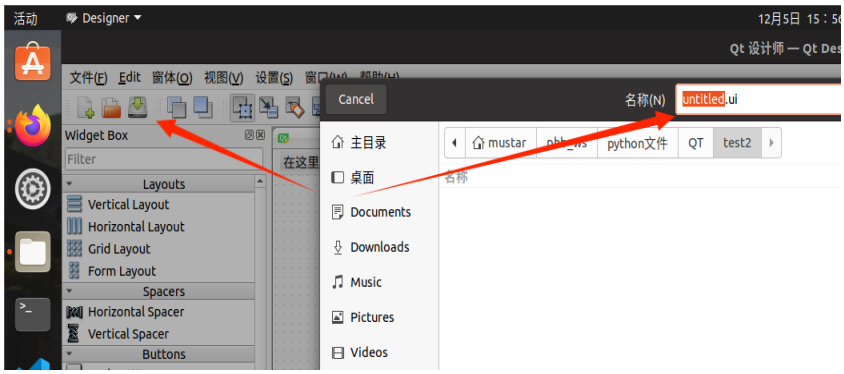
1. 测试界面

选中 Main Window 点击创建，将左边的 PushButton 拖入界面



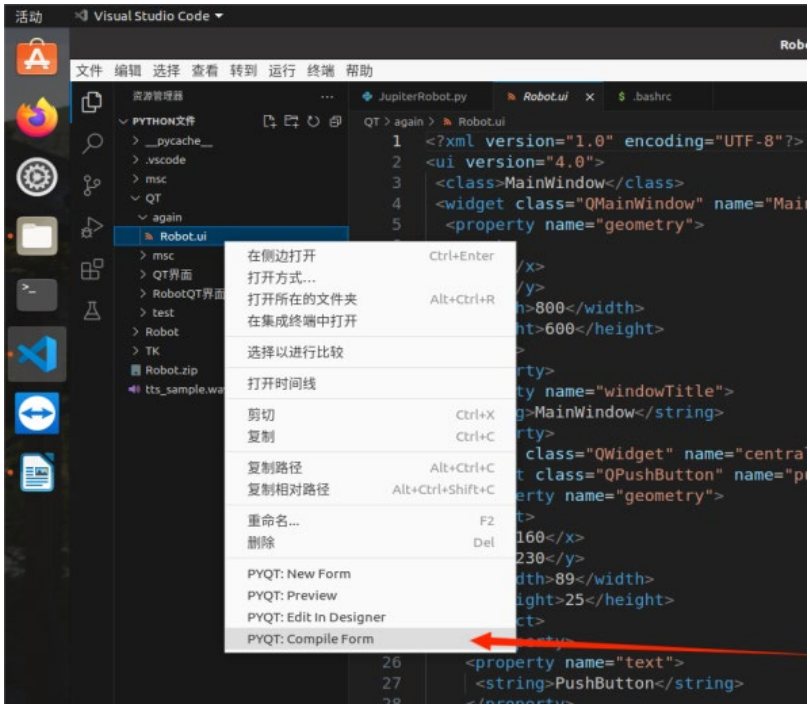
2. 保存文件

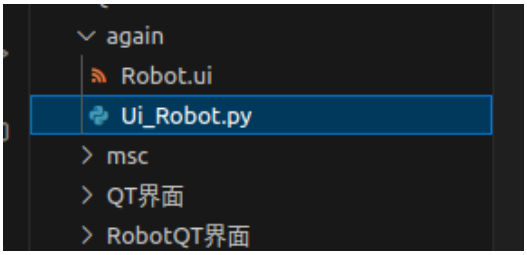
点击左上角的保存或 ctrl+s，首次保存会弹出下面这个界面，选择保存路径（默认为点击 PYQT：New Form 的路径）和文件名



3. ui 文件转 py 文件

右键点击刚刚创建的.ui 文件，点击最下面的 PYQT：Compile Form，会自动生成一个 UI_XXX.py 文件。这个文件为 qt 自动生成，每次编辑完.ui 文件后右键选择最下面的 PYQT：Compile Form 即可自动编辑。这个 py 文件不能直接运行，只是创建了窗体的类，使用需要配合后面的主程序模版。





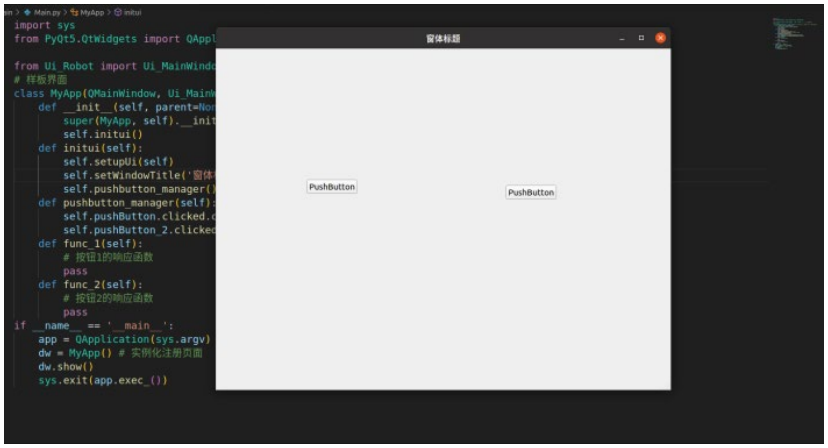
4.创建一个主程序模板文件

在同级目录下创建一个 py 文件，如 Main.py
下面是主程序的模版，需要修改 ui 文件改成你自己的。
Main.py 代码

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

from Ui_Robot import Ui_MainWindow # 这是是改成 ui 转出 py 的文件名
# 样板界面
class MyApp(QMainWindow, Ui_MainWindow): # 此处界面是主窗口，即为 QMainWindow
        def __init__(self, parent=None):
                super(MyApp, self).__init__(parent)
                self.initui()
        def initui(self):
                self.setupUi(self)
                self.setWindowTitle('窗体标题')
                self.pushbutton_manager() # 绑定槽函数
        def pushbutton_manager(self):
                self.pushButton.clicked.connect(self.func_1)
                self.pushButton_2.clicked.connect(self.func_2)
```

```
        def func_1(self):
            # 按钮 1 的响应函数
            pass
        def func_2(self):
            # 按钮 2 的响应函数
            pass
if __name__ == '__main__':
    app = QApplication(sys.argv)
    dw = MyApp() # 实例化注册页面
    dw.show()
    sys.exit(app.exec_())
```

可以直接运行 Main.py，显示效果如下



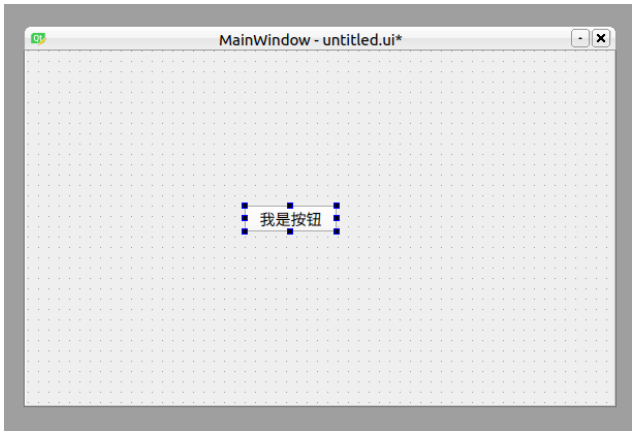
三. 编辑 UI 界面

文件能正常运行后，我们就可以开始编辑 UI 界面了，右键点击.ui 文件，选择 PYQT:Edit In Designer 就可以回到编辑界面，第一步先将界面调整至合适的大小，随后就可以加入各种控件

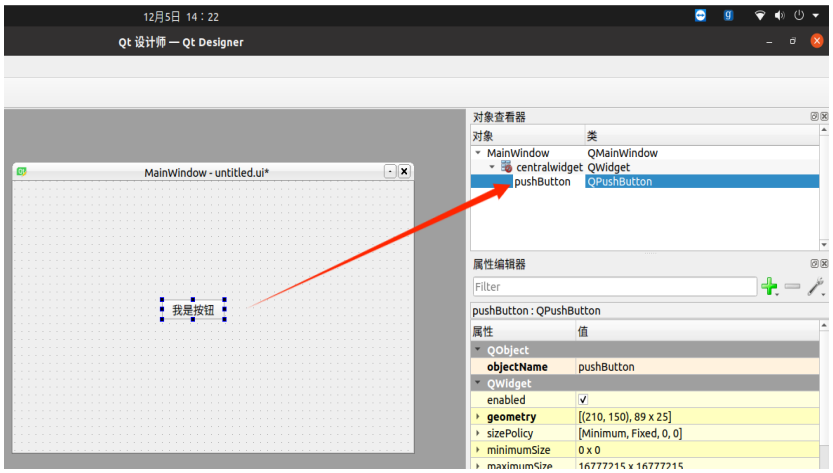
1.常用功能介绍

（1）PushButton

按钮是制作 UI 界面最常见的控件，QT 也提供了按钮。拖入后双击可以改变按钮上显示的文字



也可以双击右侧的对象查看器中的对象，修改对象名，在按钮较多时，可以按照自己的习惯给每个按钮命名，方便后续绑定函数和管理。



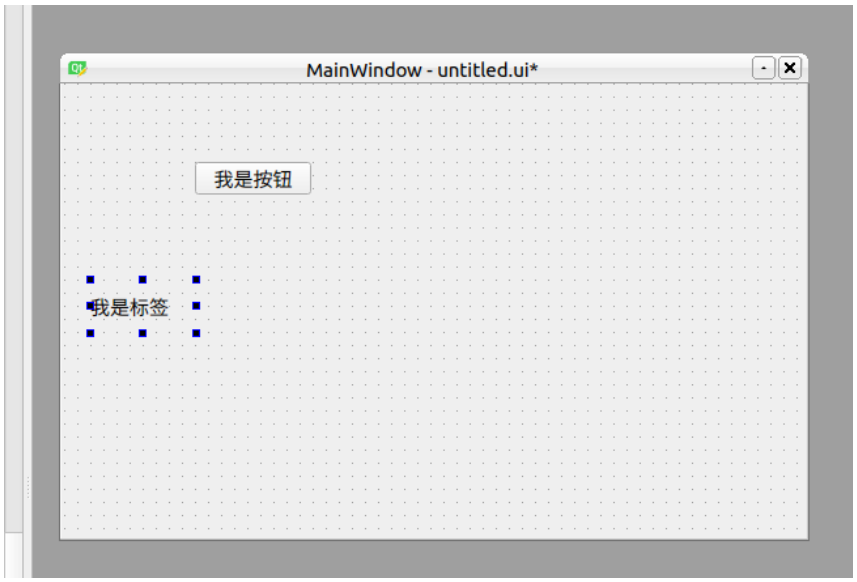
给按钮绑定函数：

```
self.pushButton.clicked.connect(self.cmd)
```

以上示例是将对象名为 pushButton 的按钮绑定点击时触发 cmd 的方法。

（2）Label

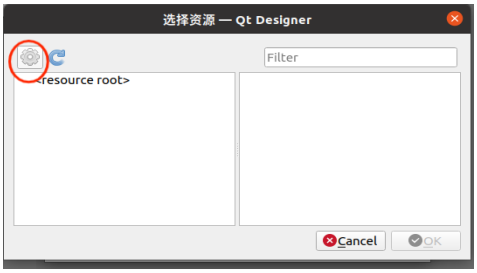
标签可以帮助我们在页面中打入标签。



也可以右键点击标签，点击改变样式表，在添加资源中添加背景图片，使标签充当一个画布。

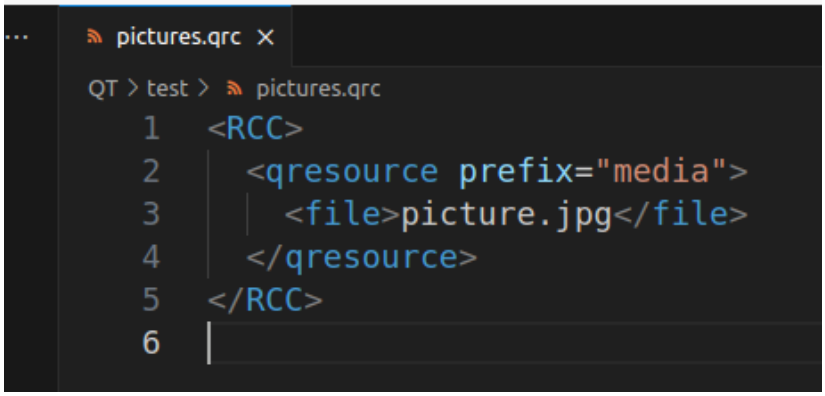
首先需要将标签上的文字删除，再右键点击标签，选择改变样式表，点击添加资源右边下拉菜单里的 background-image。

点击齿轮。



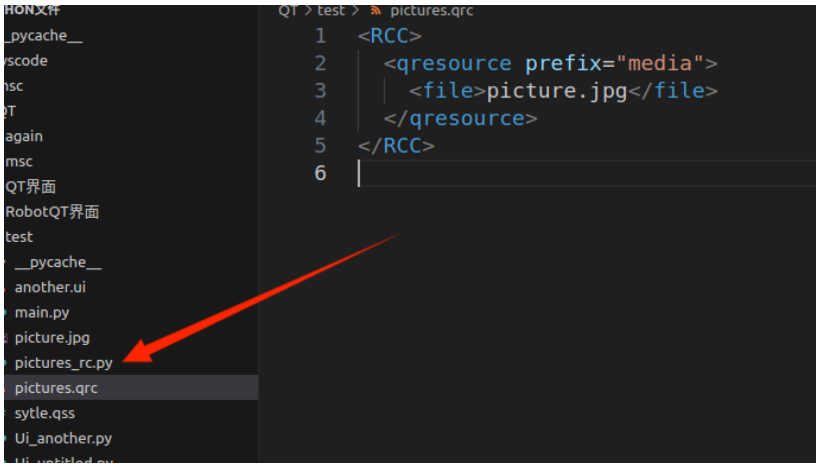
点击蓝色的“打开资源文件”按钮，找到需要的图片文件。

QT 编辑器只能显示.qrc 后缀的文件，所以我们需要先编写一个 qrc 文件




```
<RCC>

<qresource prefix="media">

<file>picture.jpg</file>

</qresource>

</RCC>
```

再右键 qrc 文件点击 PYQT：Compile Form，会自动生成一个 py 文件



（3）Stacked Widget 分页工具

可以创建多个页面。

右上角的小三角只会在编辑时显示，运行时不会显示。

可以通过按钮绑定页码实现页面更换。

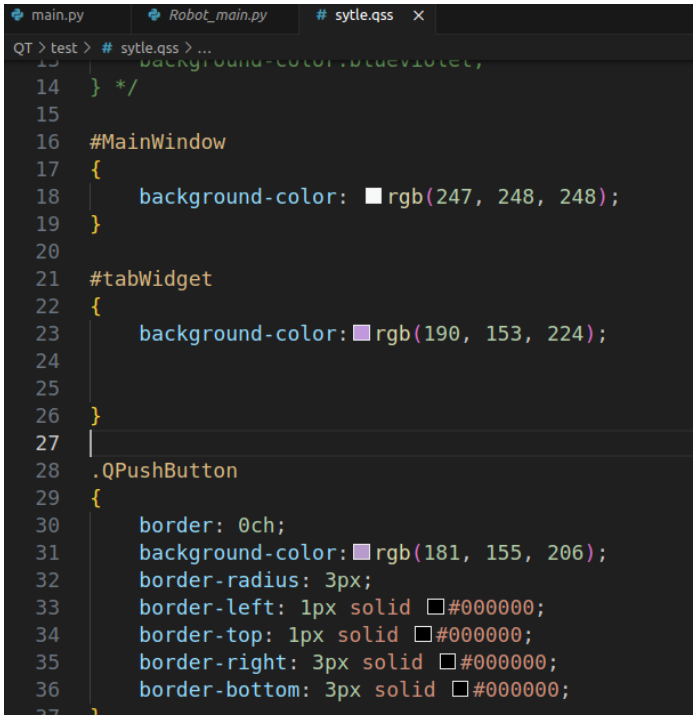
```python
import sys
from PyQt5.QtWidgets import QApplication,QMainWindow,QVBoxLayout
from PyQt5 import QtWidgets
from Ui_untitled import Ui_MainWindow as hello_ui

class Main_Window(QtWidgets.QMainWindow, hello_ui):

    def __init__(self):
        super(Main_Window, self).__init__()
        self.setupUi(self)
        self.buttoncommand()

    def buttoncommand(self):
        self.pushButton_2.clicked.connect(self.goto1)
        self.pushButton_3.clicked.connect(self.goto2)

    def goto1(self):
        self.stackedWidget.setCurrentIndex(0)

    def goto2(self):
        self.stackedWidget.setCurrentIndex(1)
```

```
def goto1(self):

self.stackedWidget.setCurrentIndex(0)

def goto2(self):

self.stackedWidget.setCurrentIndex(1)
```

四.　界面美化

1.添加样式文件

创建一个 style 文件夹，并创建一个 style.qss 的样式文件，类似于 css 文件
.加类名可以更改整个类的样式
#加对象名可以更改某个对象的样式



2.在主文件中添加更改样式代码

主文件中可以通过.setStyleSheet 的方法改变对象样式