**Lab Exercise 6 – PyQT Widgets**

Creating widgets in PyQt is a fundamental aspect of building graphical user interfaces (GUIs) using Python. PyQt provides a wide range of widgets like buttons, labels, text fields, and more. In this tutorial, we'll create a PyQt application with various widgets and demonstrate how to work with them.

**Prerequisites:**

Before starting this tutorial, make sure you have PyQt5 installed. You can install it using pip:

pip install PyQt5

**Step 1: Creating a Basic PyQt Application**

We'll start by creating a simple PyQt application that displays a main window.

import sys

from PyQt5.QtWidgets import QApplication, QMainWindow

app = QApplication(sys.argv)

window = QMainWindow()

window.setWindowTitle('PyQt Widgets Tutorial')

window.setGeometry(100, 100, 400, 300) # (x, y, width, height)

window.show()

sys.exit(app.exec\_())

This code initializes a PyQt application, creates a main window, sets its title, size, and position, and finally, shows the window.

**Step 2: Adding Widgets to the Main Window**

Now, let's add some widgets to the main window. We'll add a label, a button, and a text input field.

import sys

from PyQt5.QtWidgets import QApplication, QMainWindow, QLabel, QPushButton, QLineEdit, QVBoxLayout, QWidget

app = QApplication(sys.argv)

window = QMainWindow()

window.setWindowTitle('PyQt Widgets Tutorial')

window.setGeometry(100, 100, 400, 300)

# Create a central widget to hold other widgets

central\_widget = QWidget()

window.setCentralWidget(central\_widget)

# Create widgets

label = QLabel('Enter your name:')

name\_input = QLineEdit()

greet\_button = QPushButton('Greet')

# Create a layout for widgets

layout = QVBoxLayout()

layout.addWidget(label)

layout.addWidget(name\_input)

layout.addWidget(greet\_button)

# Set the layout for the central widget

central\_widget.setLayout(layout)

window.show()

sys.exit(app.exec\_())

In this code, we've added three widgets: a label, a text input field, and a button. We've also created a layout (a vertical layout) to arrange these widgets.

**Step 3: Adding Functionality to Widgets**

Now, let's add functionality to the button. When the button is clicked, it will display a greeting message in the window's title.

import sys

from PyQt5.QtWidgets import QApplication, QMainWindow, QLabel, QPushButton, QLineEdit, QVBoxLayout, QWidget

def greet():

name = name\_input.text()

if name:

greeting = f'Hello, {name}!'

else:

greeting = 'Hello, World!'

window.setWindowTitle(greeting)

app = QApplication(sys.argv)

window = QMainWindow()

window.setWindowTitle('PyQt Widgets Tutorial')

window.setGeometry(100, 100, 400, 300)

central\_widget = QWidget()

window.setCentralWidget(central\_widget)

label = QLabel('Enter your name:')

name\_input = QLineEdit()

greet\_button = QPushButton('Greet')

greet\_button.clicked.connect(greet)

layout = QVBoxLayout()

layout.addWidget(label)

layout.addWidget(name\_input)

layout.addWidget(greet\_button)

central\_widget.setLayout(layout)

window.show()

sys.exit(app.exec\_())

Now, when you enter your name in the text input field and click the "Greet" button, it will change the window's title to greet you.

**Step 4: Running the PyQt Application**

Save the script with a .py extension, and then run it using Python:

python your\_script\_name.py

You should see the PyQt application with the widgets. Experiment with different widgets and their properties to create more complex GUIs.