**Lab Exercise 23- QTimer and QProgress in PyQT**

Here's an example with good design using PyQt5. The program creates a simple GUI with a progress bar and a start button. Clicking the start button initiates a process that increments the progress bar until it reaches the maximum value.

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QPushButton, QProgressBar

from PyQt5.QtCore import QTimer

class Example(QWidget):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.initUI()

def initUI(self):

self.setWindowTitle('Progress Bar Example')

self.setGeometry(100, 100, 300, 150)

# Create a progress bar and set its initial value to 0

self.progress = QProgressBar(self)

self.progress.setGeometry(50, 50, 200, 25)

self.progress.setValue(0)

# Create a start button

self.button = QPushButton('Start', self)

self.button.setGeometry(100, 90, 100, 30)

self.button.clicked.connect(self.startProgressBar)

# Set the layout

layout = QVBoxLayout()

layout.addWidget(self.progress)

layout.addWidget(self.button)

self.setLayout(layout)

def startProgressBar(self):

self.timer = QTimer(self)

self.timer.timeout.connect(self.updateProgress)

self.timer.start(100) # milliseconds

def updateProgress(self):

cur\_val = self.progress.value()

max\_val = self.progress.maximum()

if cur\_val >= max\_val:

self.timer.stop()

else:

self.progress.setValue(cur\_val + 1)

if \_\_name\_\_ == '\_\_main\_\_':

app = QApplication(sys.argv)

ex = Example()

ex.show()

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

**Explanation:**

* The necessary modules from PyQt5 are imported to create the GUI.
* The Example class is defined, which inherits from QWidget.
* In the initUI method, the window title and geometry are set. The progress bar and the start button are created and positioned accordingly.
* The startProgressBar method is called when the button is clicked. It creates a QTimer, connects its timeout signal to the updateProgress method, and starts the timer with a specified interval.
* The updateProgress method is called at every timeout interval of the timer. It checks the current value of the progress bar and stops the timer if the current value reaches the maximum value. Otherwise, it increments the value of the progress bar by 1.
* Finally, in the main block, the application is executed, and the GUI window is displayed.

This code creates a simple and intuitive UI with a clear separation between the progress bar and the start button.