บทที่ 9

การเขียนโปรแกรมแบบกราฟิก (Graphic User Interface: GUI) ด้วย PySide6

สร้าง GUI ด้วย PySide ตอนที่ 1 : เริ่มต้นโปรแกรมแรก Hello World

สวัสดีครับทุกท่าน บทความนี้จะพาทุกท่านไปรู้จักกับการสร้าง GUI บน Python ด้วย PySide กันครับ PySide เป็นเครื่องมือที่ใช้สร้าง GUI หนึ่งบน Python โดยอาศัย Qt ในการรันโปรแกรมอีกทอดหนึ่ง เหมือน PyQt แต่ PySide ใช้ LGPL ครับ PySide สนับสนุนทั้ง Python 2 , 3 ครับ แต่ผมขออิงบน Python 3 ครับ

การติดตั้ง PySide

- บน Windows ใช้ pip โดยใช้คำสั่งต่อไปนี้ครับ pip install -U PySide หรือสำหรับใครที่อยากโหลดแบบ Binaries https://download.qt-project.org/official_releases/pyside/ ครับ
- บนลินุกซ์สาย Debian ใช้คำสั่ง sudo apt-get install python3-pyside
- Mac OS X เข้าไปที่หน้า http://qt-project.org/wiki/PySide_Binaries_MacOSX
- Linux เข้าไปที่หน้า http://qt-project.org/wiki/PySide_Binaries_Linux

เริ่มต้นโปรแกรมแรก Hello World ด้วย PySide6 ใน Python 3

ตัวอย่างที่ 1

```
#  Import PySide6 classes
import sys
from PySide6.QtWidgets import *

# สร้างโปรแกรมด้วย Qt
app = QApplication(sys.argv)
# สร้าง Label และแสดง
label = QLabel("Hello World")
label.show()
# Enter Qt application main loop
app.exec()
sys.exit()
```

ในตัวอย่างที่ 1 เราได้เขียนโปรแกรมให้แสดงคำว่า "Hello World" โดยใช้ QLabel แล้วแสดงด้วยคำสั่ง label.show()

ผลลัพธ์

~~**ทำไมต้องนำเข้าทั้ง PySide.QtCore และ PySide.QtGui**~~

~~เมื่อเราใช้งาน PySide กับเดสก์ท็อป เราจำเป็นต้องนำเข้า PySide.QtCore และคลาส PySide.QtGui ซึ่งมีหน้าที่สำคัญในการเขียนโปรแกรมด้วย PySide เช่น PySide.QtGui ประกอบด้วยฟังก์ชันสำหรับจัดการกับเครื่องมือในขณะที่ PySide.QtCore มีวิธีการสำหรับการจัดการซิกแนลและสล็อตและการควบคุมโปรแกรม~~

เราสามารถใส่ HTML ลงใน QLabel ได้ ตัวอย่างเช่น

```
label = QLabel("<font color=red size=40>Hello World</font>")
```

ตัวอย่างที่ 2

```
import sys
from PySide6.QtWidgets import *

app = QApplication(sys.argv)

win = QWidget()

win.resize(320, 240) #กำหนดขนาดของหน้าต่างโปรแกรม
win.setWindowTitle("Hello, World!") #กำหนดชื่อตรงหัวโปรแกรม
win.show() #แสดง

app.exec()
sys.exit()
```

ในตัวอย่างที่ 2 นี้เราได้ทำหน้าต่างโปรแกรม โดยมีการกำหนดขนาดของโปรแกรมและชื่อหัวโปรแกรม ครับ

ผลลัพธ์

จากตัวอย่างที่ 2 สามารถปรับเขียนในรูปแบบคลาสได้ ดังนี้

```
import sys
from PySide6.QtWidgets import *

class myApp(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(320, 240)   # กำหนดขนาดของหน้าต่างโปรแกรม
        self.setWindowTitle("Hello, World!")   # กำหนดชื่อตรงหัวโปรแกรม


if __name__ == "__main__":
    app = QApplication(sys.argv)
    win = myApp()
    win.show()
    sys.exit(app.exec())
```

สร้าง GUI ด้วย PySide ตอนที่ 2 : ดึงองค์ประกอบต่าง ๆ มารวมกัน

สวัสดีครับทุกท่าน จากบทความที่แล้วเราได้สร้างโปรแกรม Hello World โดยได้ใช้ PySide ครับ (โปรแกรมระดับตำนาน) ผมได้ยกตัวอย่างทั้ง 2 ตัวอย่าง แต่ถ้าอยากจับทั้งตัวอย่างที่ 1 และ 2 มารวมกันในหน้าต่างเดียวกัน

บทความนี้จะพาทุกท่านไปจับทั้ง 2 ตัวอย่างนั้นมารวมในหน้าต่างเดียวกัน ในการดึงองค์ประกอบต่าง ๆ มารวมกันต้องใช้ QVBoxLayout() ครับ

โค้ดโปรแกรม Hello World ตัวอย่างที่ 3

```
import sys
from PySide6.QtWidgets import *

app = QApplication(sys.argv)
win = QWidget()
win.resize(320, 240) #กำหนดขนาดของหน้าต่างโปรแกรม
win.setWindowTitle("Hello, World!") #กำหนดชื่อตรงหัวโปรแกรม
layout = QVBoxLayout() #ดึง QVBoxLayout() มาใช้งานเพิ่มให้เราสามารถดึงวัตถุมารวมกันได้
win.setLayout(layout) #กำหนดให้ตัวแปร win ซึ่งเป็น QWidget ดึงตัวแปรที่อยู่ใน layout มาแสดง

label = QLabel("Hello World") #กำหนดข้อความ
layout.addWidget(label) #เพิ่มดึงตัวแปร label เข้ามา
win.show() #แสดง
app.exec()
sys.exit()
```

ผลลัพธ์

จากตัวอย่างที่ 3 สามารถปรับเขียนในรูปแบบคลาสได้ ดังนี้

```
import sys
from PySide6.QtWidgets import *

class myApp(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(320, 240)   # กำหนดขนาดของหน้าต่างโปรแกรม
        self.setWindowTitle("Hello, World!")   # กำหนดชื่อตรงหัวโปรแกรม
        layout = QVBoxLayout()
        self.setLayout(layout)

        label = QLabel("Hello World")
        layout.addWidget(label)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    win = myApp()
    win.show()
    sys.exit(app.exec())
```

จะเห็นได้ว่าเราสามารถดึงทั้งองค์ประกอบต่าง ๆ มารวมกันในหน้าต่างเดียวกันได้ครับ หากเราต้องการดึงองค์ประกอบมากกว่าสองขึ้นไปสามารถทำแบบเดียวได้ ดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 4

```
import sys
from PySide6.QtWidgets import *

class myApp(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(320, 240)   # กำหนดขนาดของหน้าต่างโปรแกรม
        self.setWindowTitle("Hello, World!")   # กำหนดชื่อตรงหัวโปรแกรม
        layout = QVBoxLayout()
        self.setLayout(layout)

        label = QLabel("Hello World")
```

```
        label2 = QLabel("สวัสดี")
        label3 = QLabel("เพิ่งเริ่มต้นเรียน PySide -..- จากผู้เขียนบทความ")

        layout.addWidget(label)
        layout.addWidget(label2)
        layout.addWidget(label3)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    win = myApp()
    win.show()
    sys.exit(app.exec())
```

ผลลัพธ์

สำหรับท่านใดที่มีปัญหากับภาษาไทย PySide ให้ท่านตรวจสอบ encoding ของไฟล์กับข้อความว่าเป็น utf-8 หรือไม่ครับ ที่มีปัญหาเพราะ encoding ของไฟล์กับข้อความไม่เหมือนกันครับ

สร้าง GUI ด้วย PySide ตอนที่ 3 : QWebEngineView

สวัสดีทุกท่านครับ บทความชุด สร้าง GUI ด้วย PySide ได้เดินทางมาถึงบทความที่ 3 แล้ว บทความนี้เราจะพูดถึงเรื่อง QWebEngineView ใน PySide ใช้งานเกี่ยวกับการแสดงหน้าเว็บต่าง ๆ มาใช้ในโปรแกรมครับ เนื่องจากเราใช้ PySide ซึ่งอิง Qt อีกทอดหนึ่ง ดังนั้น web browser engine ที่ใช้งานคือ WebEngineWidgets ครับ สำหรับการใช้ QWebEngieneView เราต้อง from PySide.QtWebEngineWidgets import * ด้วยนะครับ

การดึงหน้าเว็บเพจโดยอ้างอิงที่อยู่เว็บเพจมาแสดง

มีการใช้งานดังนี้ครับ

```
web.load(QUrl("ที่อยู่เว็บไซต์"))
```

ตัวอย่างที่ 5

```
import sys
from PySide6.QtWidgets import *
from PySide6.QtCore import QUrl
from PySide6.QtWebEngineWidgets import QWebEngineView

class myWeb(QWebEngineView):
    def __init__(self):
        super().__init__()
        self.load(QUrl("http://www.google.co.th")) #ดึงหน้าเว็บเพจมาแสดง

if __name__ == "__main__":
    app = QApplication(sys.argv)
    web = myWeb()
    web.show()
    sys.exit(app.exec())
```

ผลลัพธ์

ถ้าเราต้องการกำหนด HTML ให้ QWebEngineView แสดง มีหลักการดังนี้ครับ

web_view.setHtml('''HTML CODE''')

ตัวอย่างที่ 6

```
import sys
from PySide6.QtWidgets import *
from PySide6.QtWebEngineWidgets import QWebEngineView

class myWeb(QWebEngineView):
    def __init__(self):
        super().__init__()
        self.setHtml('''<html>
        <head>
```

```
        <title>ทดสอบ</title>
        </head>
        <body>
        <h1>Hello, World!</h1>
        <hr />
   ทดสอบการแสดงผล HTML ใน QWebView
        </body>
        </html>''')

if __name__ == "__main__":
    app = QApplication(sys.argv)
    web = myWeb()
    web.show()
    sys.exit(app.exec())
```

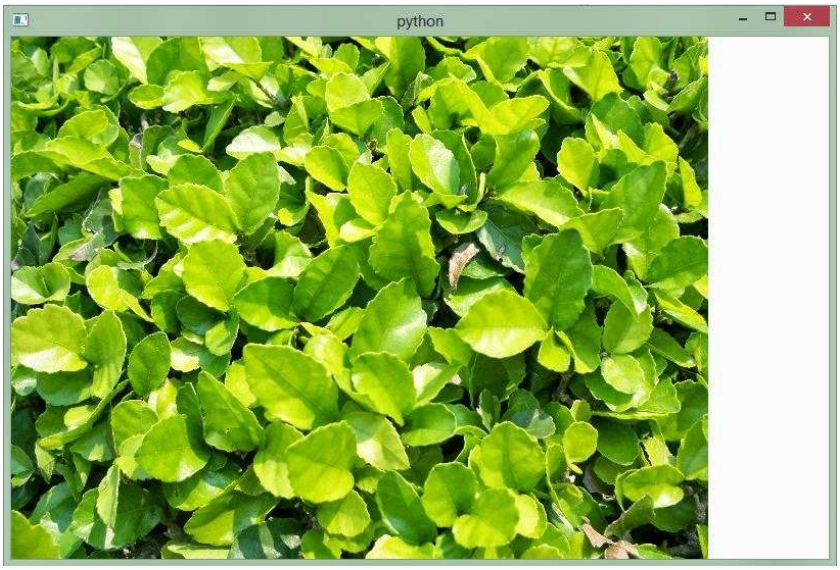
ผลลัพธ์

นอกจากใช้แสดงหน้าเว็บแล้ว QWebEngineView สามารถแสดงรูปภาพได้ โดยใช้ QWebEngineView.setContent ดังตัวอย่างต่อไปนี้

ตัวอย่างที่ 7

```
import sys
from PySide6.QtWidgets import *
from PySide6.QtWebEngineWidgets import QWebEngineView

class myWeb(QWebEngineView):
    def __init__(self):
        super().__init__()
        img = open('101_0336.png', 'rb').read() #ดึงไฟล์ 101_0336.png เข้ามาแล้วอ่าน
        #ดึงตัวแปร img ที่เก็บไฟล์รูปภาพมาแสดง โดยมีการกำหนด setContent เป็น image/png
        self. setContent(img,"image/png")

if __name__ == "__main__":
    app = QApplication(sys.argv)
    web = myWeb()
    web.show()
    sys.exit(app.exec())
```

ผลลัพธ์



สร้าง GUI ด้วย PySide ตอนที่ 4 : ทำปุ่มกัน

สวัสดีทุกท่านครับ บทความนี้ผมจะพาทุกท่านไปทำปุ่มบน GUI ด้วย PySide ครับ การที่เราจะสร้างปุ่มกดบนบน GUI ด้วย PySide เราต้องใช้โมดูลของ PySide นั้นคือ QPushButton
โดยมีลักษณะการใช้งานดังตัวอย่างต่อไปนี้ครับ

ตัวอย่างที่ 8

```
import sys
from PySide6.QtWidgets import *

class myApp(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(320, 240)   # กำหนดขนาดของหน้าต่างโปรแกรม
        self.setWindowTitle("Hello, World!")   # กำหนดชื่อตรงหัวโปรแกรม
        layout = QVBoxLayout()
        self.setLayout(layout)

        hello = QPushButton("Hello world!")   # กำหนดข้อความในปุ่ม
        hello.resize(100, 30)   # กำหนดขนาดของปุ่ม
        layout.addWidget(hello)


if __name__ == "__main__":
    app = QApplication(sys.argv)
    win = myApp()
    win.show()
    sys.exit(app.exec())
```

ผลลัพธ์

จากตัวอย่างที่ 8 พอเราคลิกที่ปุ่ม Hello world! แล้วพบว่าไม่มีอะไรเกิดขึ้น หากเราต้องการให้มีการกระทำเกิดขึ้นหลังจากกดปุ่มเช่น ปิดโปรแกรมโดยมีลักษณะการใช้งานดังตัวอย่างต่อไปนี้ครับ

ตัวอย่างที่ 9

```
import sys
from PySide6.QtCore import *
from PySide6.QtGui import *
from PySide6.QtWidgets import *

app = QApplication(sys.argv)
quit = QPushButton("Quit") #กำหนดข้อความในปุ่ม
quit.resize(75, 30) #กำหนดขนาดของปุ่ม
quit.setFont(QFont("Times", 18,QFont.Bold))
QObject.connect(quit,SIGNAL("clicked()"),
app, SLOT("quit()"))

quit.show()
sys.exit(app.exec_())
```

ผลลัพธ์

จากตัวอย่างที่ 9 เมื่อแรคลิกที่ปุ่ม Quit พบว่า ได้มีการปิดโปรแกรมครับ เพราะเราได้มีการตั้งค่าที่โค้ด

```
QObject.connect(quit,SIGNAL("clicked()"),app, SLOT("quit()"))
```

ได้มีการเชื่อมต่อกับวัตถุ quit ซึ่งเป็นปุ่มแล้วเราได้กำหนดเงื่อนไข SIGNAL("clicked()") เมื่อคลิกแล้วให้มีการกระทำกับวัตถุ app โดยใช้โค้ดคำสั่ง SLOT("quit()")) คือปิดหน้าต่างโปรแกรมครับ

ข้อควรรู้

เราสามารถกำหนด Font ตัวอักษร PySide ได้โดยใช้

```
setFont(QFont("ชื่อ Font", ขนาด, ถ้าต้องการกำหนดตัวหนาให้ใส่ QFont.Bold เพิ่ม)) #ตัวอย่างเช่น
setFont(QFont("ชื่อ Font เช่น Times", ขนาดเช่น 18,QFont.Bold))
```

สามารถใช้ได้ทั้ง QLabel,QPushButton และต่าง ๆ ที่ใช้ตัวอักษรครับ

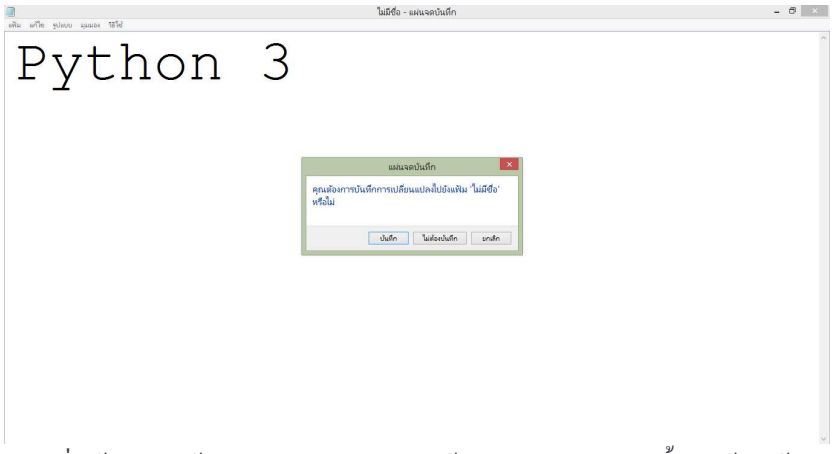
จากตัวอย่างที่ 9 สามารถปรับเขียนในรูปแบบคลาสได้ ดังนี้

```
import sys
from PySide6.QtWidgets import *
from PySide6.QtGui import *
```

```
class myApp(QWidget):
    def __init__(self):
        super().__init__()
        self.resize(320, 240)   # กำหนดขนาดของหน้าต่างโปรแกรม
        self.setWindowTitle("Hello, World!")   # กำหนดชื่อตรงหัวโปรแกรม

        btn = QPushButton("Quit", self)
        btn.resize(75, 30)
        btn.setFont(QFont("Times", 16, QFont.Bold)) # ต้อง import QtGui
        btn.move(80, 50)

        btn.clicked.connect(self.btn_clicked)

    def btn_clicked(self):
        QApplication.quit()


if __name__ == "__main__":
    app = QApplication(sys.argv)
    win = myApp()
    win.show()
    sys.exit(app.exec())
```

สร้าง GUI ด้วย PySide ตอนที่ 6: Message Box

สวัสดีทุกท่านครับ บทความนี้ผมจะพูดถึงเรื่อง Message Box ใน PySide ครับ Message Box คือ กล่องข้อความ เป็นกล่องข้อความโต้ตอบกับผู้ใช้แมื่อมีการกระทำเกิดขึ้นครับ ตัวอย่างเช่น คุณกำลังพิมพ์เอกสารในโปรแกรมพิมพ์เอกสารแล้วคุณไปกด X โดยที่ไม่ได้บันทึกจะขึ้นข้อความตอบโต้ว่า "คุณต้องการบันทึกไหม"



ประมาณนี้ครับ เรามาเริ่มต้นเขียนโค้ดกันเลยนะครับ ในการใช้ Message Box นั้นเราต้องใช้ QMessageBox ครับ มีลักษณะการใช้งานตามตัวอย่างต่อไปนี้ครับ

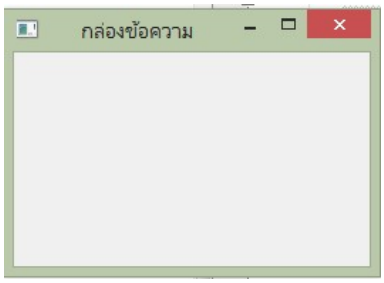
ตัวอย่างที่ 10

```
import sys
from PySide6.QtWidgets import *

class myMessageBox(QWidget):
     def __init__(self, parent=None):
         QWidget.__init__(self, parent)
         self.setGeometry(300, 300, 250, 150)
         self.setWindowTitle('กล่องข้อความ') #ชื่อหัวของโปรแกรม

     def closeEvent(self, event): #กำหนดการกระทำเมื่อมีความต้องการปิดโปรแกรม
         reply = QMessageBox.question(self, 'Message',
             "คุณแน่ใจนะว่าคุณต้องการปิดโปรแกรม?", QMessageBox.Yes, QMessageBox.No)
         if reply == QMessageBox.Yes:
             event.accept() #ยันยืนการกระทำคือปิดโปรแกรมครับ
         else:
             event.ignore() #ไม่สนใจการกระทำ คือ ไม่ปิดโปรแกรมครับ


app = QApplication(sys.argv)
qb = myMessageBox() #อ้างอิงคลาส MessageBox
qb.show() #แสดงผล
sys.exit(app.exec())
```

อธิบาย QMessageBox.question กำหนดข้อความประโยคคำถาม แล้วกำหนดให้มี QMessageBox.Yes นั้นคือปุ่ม Yes ส่วน QMessageBox.No นั้นคือปุ่ม No ครับ

ผลลัพธ์

เมื่อยังไม่ได้คลิก X เพื่อปิดโปรแกรม



เมื่อคลิก X เพื่อปิดโปรแกรม