Python PyQt6 Class 1



**from PyQt6 import QtWidgets (py –m pip install PyQt6)**

**import sys**

#這行程式碼導入了 Python 的 sys 模組，這個模組用來處理與 Python 解釋器及系統交互的#功能。這裡 sys.argv 用來接收命令行引數，這是 QApplication 需要的。

**app = QtWidgets.QApplication(sys.argv)**

#這行程式碼創建了一個 QApplication 的實例 app，並傳入 sys.argv，它通常是用來處理命令#行參數的。QApplication 需要在創建任何其他 GUI 控件之前初始化，它是管理整個應用程式#事件的核心。

**Form = QtWidgets.QWidget()**

**Form.setWindowTitle('oxxo.studio') # 設定標題**

**Form.resize(320, 240) # 設定長寬尺寸**

**Form.setStyleSheet('background:#fcc;') # 使用網頁 CSS 樣式設定背景**

**Form.show()**

**sys.exit(app.exec())**

#這行程式碼啟動應用程式的事件循環。app.exec() 會讓應用程式進入事件循環，等待並處理來#自使用者的操作（例如點擊、鍵盤輸入等）。sys.exit() 會確保應用程式退出時正確清理資源並#退出。

**QLabel 位置設定**

透過下列 QLabel 方法，可以將 QLabel 元件定位到指定的位置：

| **方法** | **參數** | **說明** |
| --- | --- | --- |
| move() | x, y | 設定 QLabel 在擺放的父元件中的 xy 座標，x 往右為正，y 往下為正，尺寸根據內容自動延伸。 |
| setGeometry() | x, y, w, h | 設定 QLabel 在擺放的父元件中的 xy 座標和長寬尺寸，x 往右為正，y 往下為正，如果超過長寬尺寸，預設會被裁切無法顯示。 |
| setContentsMargins() | left, top, right, bottom | QLabel 的邊界寬度。 |

**QLabel 文字設定**

透過下列常用的 QLabel 方法，可以設定 QLabel 中的文字樣式 ( 設定字體需要搭配 QtGui，設定對齊要搭配 QtCore，需要額外載入對應模組 )：

| **方法** | **參數** | **說明** |
| --- | --- | --- |
| setWordWrap() | bool | 是否換行，預設 Fasle 不換行，設定 True 換行。 |
| setAlignment() | QtCore.Qt.AlignmentFlag | 對齊方式，預設QtCore.Qt.AlignmentFlag.AlignCenter，可設定QtCore.Qt.AlignmentFlag.AlignRight、QtCore.Qt.AlignmentFlag.AlignLeft |
| setFont() | QtGui.QFont() | 文字樣式設定，需搭配 QtGui.QFont()。 |

label1 = QtWidgets.QLabel(Form)

label1.setText('hello world, how are you?')

label1.move(50, 50)

label2 = QtWidgets.QLabel(Form)

label2.setText('hello world, how are you?')

label2.setGeometry(50, 80, 100, 100)

from PyQt6 import QtWidgets, QtGui, QtCore

label = QtWidgets.QLabel(Form)

label.setText('hello world, how are you?')

label.setGeometry(30, 30, 100, 100)

label.setContentsMargins(0,0,0,0) # 設定邊界

label.setWordWrap(True) # 可以換行

label.setAlignment(QtCore.Qt.AlignmentFlag.AlignCenter) # 對齊方式

font = QtGui.QFont() # 建立文字樣式元件

font.setFamily('Verdana') # 設定字體

font.setPointSize(20) # 文字大小

font.setBold(True) # 粗體

font.setItalic(True) # 斜體

font.setStrikeOut(True) # 刪除線

font.setUnderline(True) # 底線

label.setFont(font) # 設定文字樣式

**QLabel 圖片設定**

如果要在 QLabel 裡加入圖片，需要先使用 **QtGui.QImage()** 方法讀取圖片，接著使用 setPixmap() 方法加入圖片，詳細步驟可以參考下方程式碼：

from PyQt6 import QtWidgets, QtGui

label = QtWidgets.QLabel(Form)

label.setGeometry(20, 20, 200, 150)

img = QtGui.QImage('mona.jpg') # 讀取圖片

label.setPixmap(QtGui.QPixmap.fromImage(img)) # 加入圖片

**如果會使用網頁 CSS 語法，就能透過 setStyleSheet() 設定 QLabel 樣式**，在設計樣式上也較為***彈性好用***，下方的程式碼執行後，會套用 CSS 樣式語法，實現一個黑色虛線外框的 QLabel。

from PyQt6 import QtWidgets, QtGui

import sys

label = QtWidgets.QLabel(Form)

label.setText('hello world, how are you?')

label.setGeometry(20, 20, 200, 150)

label.setWordWrap(True) # 設定可以換行

label.setStyleSheet('''

background:#fff;

color:#f00;

font-size:20px;

font-weight:bold;

border:2px dashed #000;

padding:20px;

text-align:center;

''')

三個點（'''）在 Python 中是用來表示多行字串的開始和結束。它們通常用於包含多行字串格式（像是 HTML 或 CSS 這類格式）

**QPushButton設定**

下方的程式碼執行後會放入兩個 QPushButton，一個使用 move() 定位在 (50,30) 位置，另外一個使用 setGeometry() 方法定位在 (50,60) 的位置並設定大小為 100x50。

btn1 = QtWidgets.QPushButton(Form)

btn1.setText('按鈕 1')

btn1.move(50,30) # 移動到 (50,30)

btn2 = QtWidgets.QPushButton(Form)

btn2.setText('按鈕 2')

btn2.setGeometry(50,60,100,50) # 移動到 (50,60)，大小 100x50

調整 setStyleSheet() 內容，也能做出類似網頁按鈕 hover 的樣式效果，下方的程式碼執行後，當滑鼠移動到按鈕上方，按鈕就會變色。

btn = QtWidgets.QPushButton(Form)

btn.setText('按鈕')

btn.setGeometry(50,50,100,50)

btn.setStyleSheet('''

QPushButton {

font-size:20px;

color: #f00;

background: #ff0;

border: 2px solid #000;

}

QPushButton:hover {

color: #ff0;

background: #f00;

}

''')

**QPushButton 點擊事件**

使用**clicked.connect(fn)**方法可以設定 QPushButton 的點擊事件，該方法表示「**點擊按鈕時，會執行 fn 函式**」，下方的程式碼執行後，點擊按鈕會執行 show 函式，show 函式會不斷地將變數 a 增加 1，再透過 QLabel 顯示數字。

a = 0

def show():

global a

a = a + 1

label.setText(str(a)) # 更新 QLabel 內容

label = QtWidgets.QLabel(Form)

label.setText('0')

label.setStyleSheet('font-size:20px;')

label.setGeometry(50,30,100,30)

btn = QtWidgets.QPushButton(Form)

btn.setText('增加數字')

btn.setGeometry(50,60,100,30)

btn.clicked.connect(show) # 點擊時執行 show 函式