

第五章 Python 使用者介面

Python 圖形化使用者介面(Graphical User Interface, GUI) 的工具很多種，從 Web 介面到桌上型的都有，有 GUI 介面讓 Python 程式語言更容易學習；介面更美觀、直覺、專業、好用。

1. Python GUI 簡介

GUI 是一種使用者以圖形化介面方式顯示資訊，並與各種電子設備-例如電腦、手持式裝置與其他設備進行訊息交換與溝通。介面會利用按鈕、選單、文字方塊、清單、圖片等顯示資訊，相較於黑底白字的命令列介面相比，圖形介面視覺上更友善、方便。程式設計師可以設計 GUI 介面讓使用者藉由手指控制滑鼠與鍵盤操作使用者介面，達到與系統交換資訊以及進行運算的功能。

1.1. 最受歡迎的 5 種 Python GUI 工具包

Python 程式語言簡單、好學且工業應用系統開發廣泛，其 GUI 工具包(toolkits)開發很多元，選擇好用的 GUI 開發工具平台是很重要的。根據 2019 年調查 Python GUI 排名前 5 名是 Tkinter, PyQt, KIVY, WxPython 以及 PySide，這些工具包市場已有很多應用包含行動裝置應用、Web 應用以及桌上型應用開發等。以下針對這 5 種 Python GUI 工具包做一個簡介，提供 Python 程式設計師參考。

- Tkinter

Tkinter 由 Tcl 程式語言所開發，可視為 Python 內建的標準 GUI 模組，是目前最常被使用的工具包。Tkinter 的優點是安裝方便與 Python 相容性好；用 Tkinter 開發的 GUI 程式可以在跨平台 - Linux, Windows, Macs 作業系統上執行。不過 Tkinter 沒有方便的視窗可以拖曳物件；需要逐字輸入指令進行設計介面；GUI 元件較少缺乏多樣性；沒有程式設計基礎的使用者較不合適。

- PyQt

PyQt 是用於 Qt 的 Python GUI 工具包，而 Qt(發音同 cute)是 C++ 的跨平台應用程式 GUI 開發工具包。Nokia 於 2008 年收購了 Qt，2012 年 Nokia 將 Qt 賣給了 Digia 公司，以 LGPL(Lesser General Public License 適合於商業授權)與 GPL(General Public License 適合於開源授權)方式對外進行授權。PyQt 開發商是 Riverbank Computing，目前只提供 GPL 方式，不支援 LGPL 授權方

式。簡單的說若程式設計師以 PyQt 開發好的程式要正式對外商業運轉，就要付授權金，若只是學術教學沒有對外商轉的程式就沒有授權金問題。

PyQt5 是目前最新的版本，支援 Python 3.x，並且支援 Windows, Linux, Mac, Android 等作業系統。支援 PyQt 的類別(class)超過 400 個包含 6000 個方法(method)以及豐富的 GUI 元件供介面設計使用。另外，Anaconda 提供設計器(Qt Designer)拖曳 GUI 元件來設計使用者介面。設計好的表單介面程式*.ui 需要經過程式轉換為*.py 的程式。

❖ 註解

LGPL 與 GPL 的差別

兩者同時兩開放源始碼的授權方式，不同點在於(1)GPL 強迫要求所有源自 GPL 授權方式的程式碼進行修改/引用時程式設計師或開發商均必須採用 GPL 方式再授權，也就是被要求一起開放源始碼；(2)程式設計師或開發商可以合法修改/引用 LGPL 授權的函數庫之程式碼來開發商用軟體而且要對外商業運轉，就有支付授權金的問題。

● Kivy

Kivy 是 MIT 授權支援 Python 的 GUI 工具包，可用多點觸控應用軟體開發行動裝置的應用程式。支援多種輸入方式包含鍵盤、滑鼠以及觸控式螢幕，也支援跨平台含 Windows, Linux, Mac, Android 等作業系統，特別是有包含 Web 介面。Kivy 是目標唯一支援 Python 開發行動裝置的工具包，可以在 Kivy 工具包內開發程式並且允許啟動 Web 瀏覽器如 HTML5 做為使用者介面。

● WxPython

WxPython 是 C++ 開發的跨平台 Python GUI 工具包，開發的程式可以在不修改或額外操作之下，在不同的作業系統執行。內含 wxWidgets 函數庫可以提供 Python 物件封裝後使用。wxWidgets 是開源且跨平台的 GUI 工具包，其函數可用來建立 Python 的圖形化使用者介面。

● PySide

PySide 同 PyQt 也是 Qt 發行的 Python GUI 工具包。相較於 PyQt 而言，PySide 支援 Python 的版本較舊，可以使用的 GUI 元件也較少，而且文件說明檔也不多。Qt 推了 PyQt 為何又推出 PySide 呢？2008 年起 Qt 是 Nokia 所有，Py

Qt 的開發商是 Riverbank Computing，Nokia 要求 Riverbank Computing 開放 PyQt 為 LGPL 授權方式，遭 Riverbank Computing 拒絕。Nokia 隨即在 2009 年推出支援 LGPL 授權方式的 PySide。不過，2012 年 Nokia 就將 Qt 賣給了 Digia，PySide 未來的發展仍有一些不確定性。

1.2. 選用 PyQt 開發 Python 程式的理由

Python 近年也出現許多功能強大多樣化的使用者介面工具包，雖然工具包之間各有特色，但是選擇高效能、跨平台、完整豐富的視窗元件、方便好用，這些對初學者來說是重要的考量。因此若以應用廣度與易上手程度來思考，PyQt5 是一款非常適合新手開發 Python 使用者介面的工具包。

- 介面專業

Python 內建的 Tkinter 使用者介面設計的功能較弱。以介面功能的完整度、專業度與細緻度來說，PyQt5 強大很多。PyQt5 可以提供專業的 GUI 程式開發應用程式。

- 元件豐富

提供多樣、完整、豐富的視窗元件，除基本的文字方塊、按鈕、下拉式選單、對話方塊之外，還有繪圖、日曆時間、功能表、表格與多媒體特效等元件。

- 執行效能高

Qt 的底層是以 C++ 撰寫，而 PyQt 是以 Python 語言對 Qt 做封裝。因此，PyQt 的執行效能很高，PyQt 與 Qt 兩者的執行效能是一樣的。以 PyQt 開發 Python 具視覺化的應用程式在學習效率與執行效能上表現都非常好。

- 開發效能高

使用者介面操作簡單適合 Python 初學者，可以藉由 Qt Designer(介面設計整合開發環境, IDE)拖曳元件設計美觀的使用者介面，可以自動產生可執行的程式碼，PyQt 可以提高 Python 程式開發的效能。

1.3. PyQt 開發環境安裝

選擇簡單好用的開發環境，提升讀者的學習效果，PyQt 的安裝環境可以選擇 Windows 作業系統。開發程式時，常要重覆修改程式，重覆執行程式，同時又考慮介面的美觀，提供方便的 GUI 開發環境，可以提高新手的學習動機與成就感。本書採用 Anaconda 內裝的(1)Qt Designer 程式，提供介面設計；(2)pyuic5 程式可以將 *.ui 檔案自動轉成 *.py 程式碼；(3)程式碼編輯器採用的 Spyder。

只要安裝好 Anaconda 就內含 Qt Designer, pyuic5 與 Spyder 見下圖 5_1，就具備了開發 PyQt5 的環境，無需額外的安裝程序。本書採用 Anaconda 的這三個程式開發 Python GUI 程式。Qt Designer 用來設計使用者介面的版面配置；

pyuic5 用以將介面程式轉成 python 程式； Spyder 是程式開發時撰寫程式的編輯器。

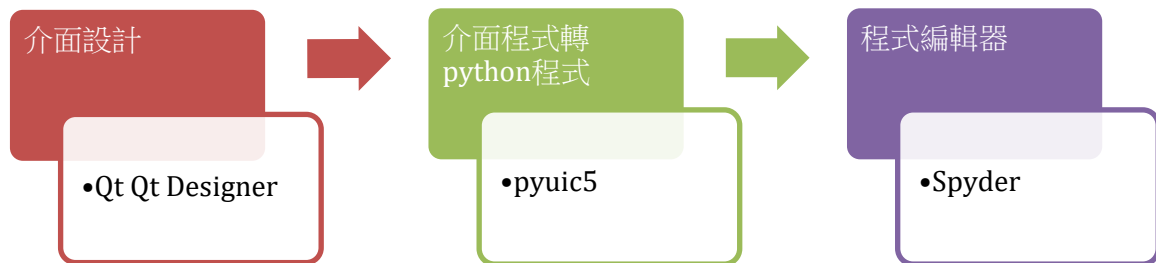


圖 5_1 Anaconda 開發 PyQt5 的環境

1.3.1. 開發環境

開發程式之前，先確認開發環境，本書寫 PyQt5 程式的開發環境整理如下表 5_1。Python 3.x 不下向相容於 Python 2.x，PyQt5 不下向相容於 PyQt4。

表 5_1 PyQt5 開發環境

程式名稱	版本
Windows 作業系統	10
系統類型	64 位元
Python	3.7
Qt Designer	5.9.7
pyuic5	5
Spyder	4.0.0

1.3.2. Qt Designer

Qt Designer 可以方便程式設計師輕鬆拉出專業、美觀的使用者介面版型，省去很多版面配置的程式撰寫時間。安裝 Anaconda 時已一併安裝 Qt Designer，在 Library/bin 資料夾內可以找到 Qt Designer.exe 見下圖 5_2，滑鼠雙擊後即可打開。Qt Designer 讓程式設計師很方便拖曳元件設計視窗版面，設計完成後儲存為 *.ui 檔。

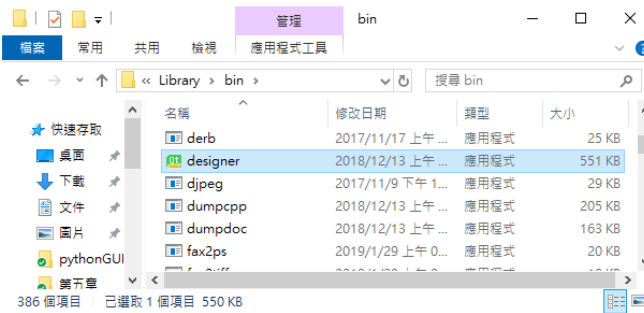


圖 5_2 Qt Designer 所在路徑

Ex5_1 範例: Say Hello 程式

本範例配合 Qt Designer 設計介面的操作步驟說明如下:

1. 新增對話

進入 Qt Designer 畫面或點選新檔案就會進入詢問版面配置對話視窗見下圖 5_3。

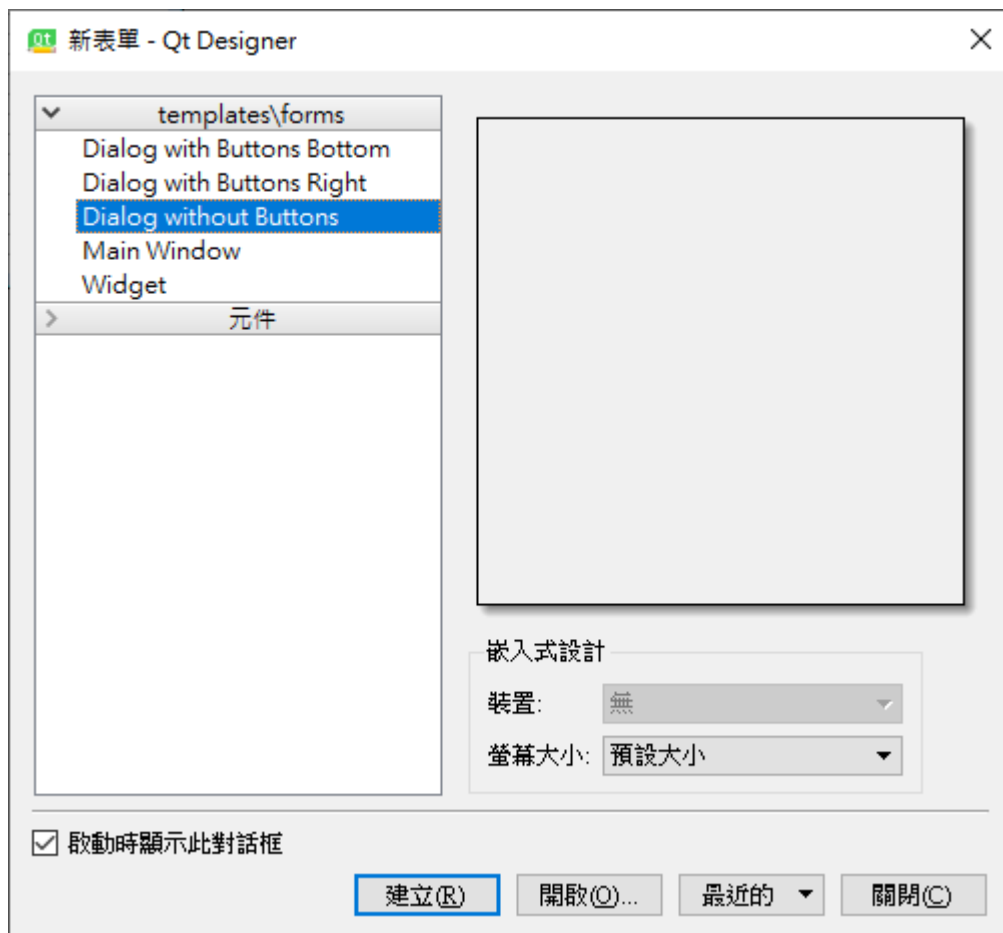


圖 5_3 新表單詢問視窗

2. 選視窗配置版型

選「Dialog without Buttons」按「建立」進入設計視窗見下圖 5_4

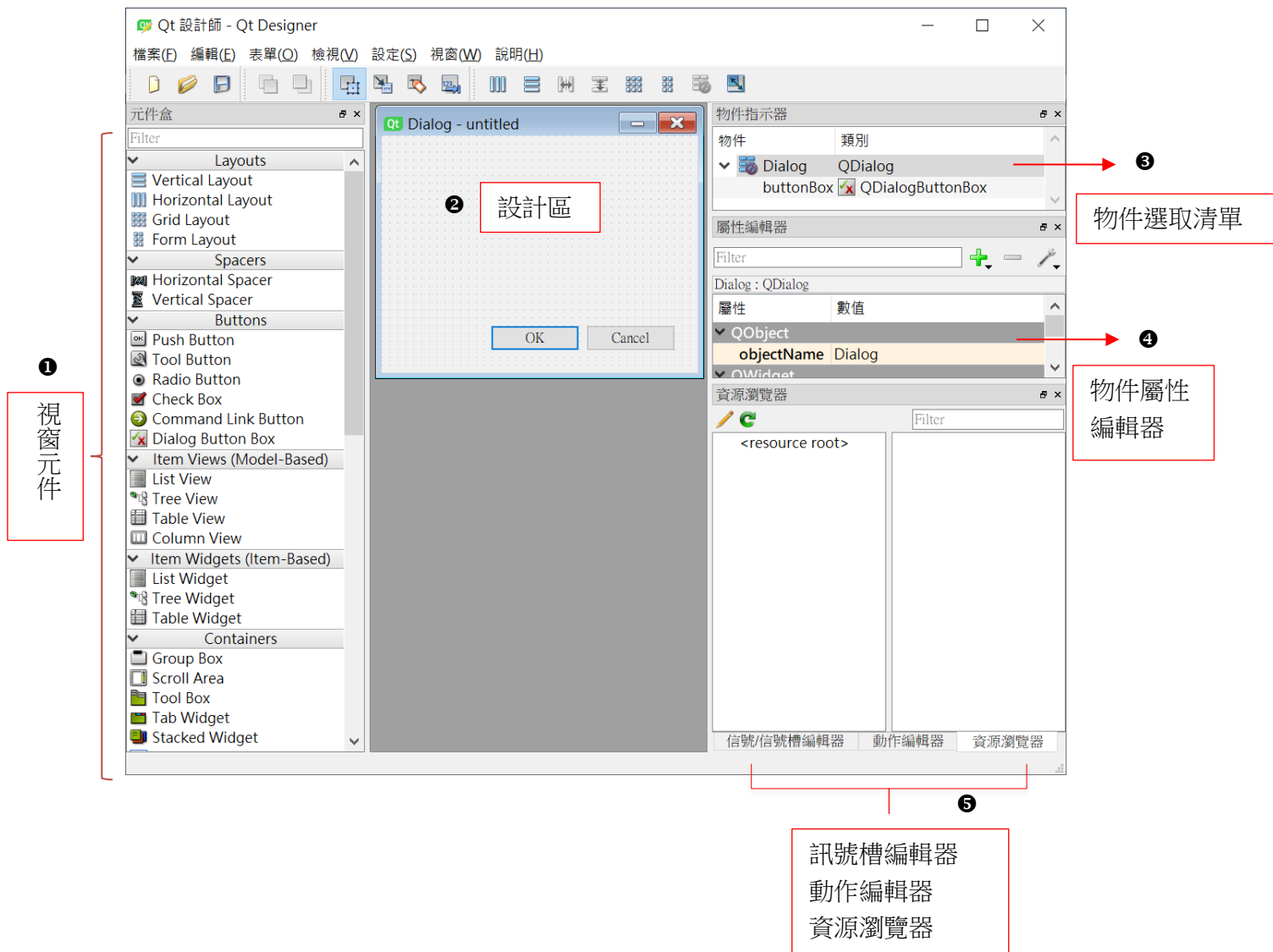


圖 5_4 Qt Designer 視窗功能介紹

❶ 視窗元件

視窗元件 (又稱元件盒)，Qt Designer 提供豐富的元件類型，分為 8 類包含 Layouts (版面配置), Spacers(空間配置), Buttons(按鈕), Item Views(檢視工具), Item Widgets(項目工具), Containers(容器), Input Widgets(輸入工具), Display Widgets(顯示工具)。

❷ 設計區

元件盒的元件可以點選拖曳到設計區，進行介面配置與設計。

❸ 物件選取清單

拖曳元件到設計區進行介面設計，會被收集到物件選取清單視窗。

❹ 屬性編輯器

每一個拖曳到設計區的元件繼承自父類別的屬性，可以針對個別元件的屬性(左邊)例如 **font**，設定其內容值如字型大小等。

⑤ 訊號槽編輯器/資源瀏覽器

訊號槽(signal/slot editor)，可以針對個別的元件設定訊號槽的對應函數並進行編輯見下圖 5_5。資源瀏覽器提供在元件的背景加入圖片或標籤等。



圖 5_5 信號與信號槽編輯器

3. 拖曳元件佈置版面

點選左邊視窗的頁籤「Display Widgets」下的「Label」，拖曳到視窗的適當位置見圖 5_6。

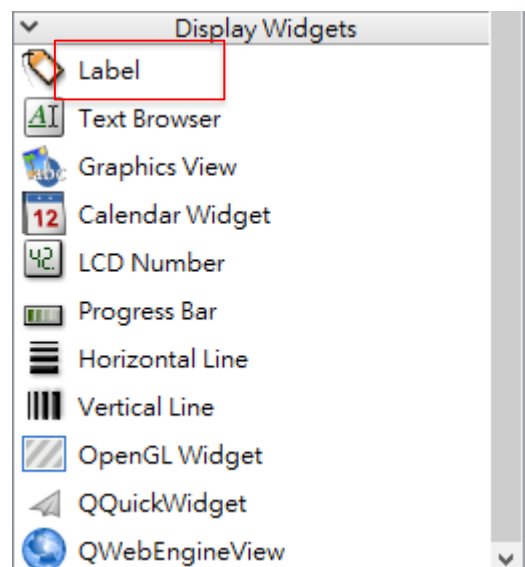


圖 5_6 拖曳 Label 元件

雙擊「Label」可以修改為「Hello PyQt 5」，並拖曳 8 個點的右下角的點進行元件放大縮小的調整，放大後如下圖 5_7。

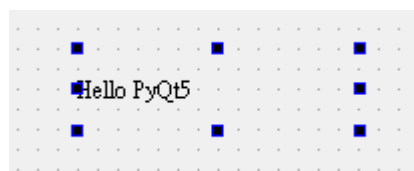


圖 5_7 調整元件大小

4. 改字型大小

點選「Hello PyQt 5」成為選取中的元件為 8 個點，並點選右邊的「屬性編輯器」下的「font」的「...」，進行修改見下圖 5_8，大小改成 16。

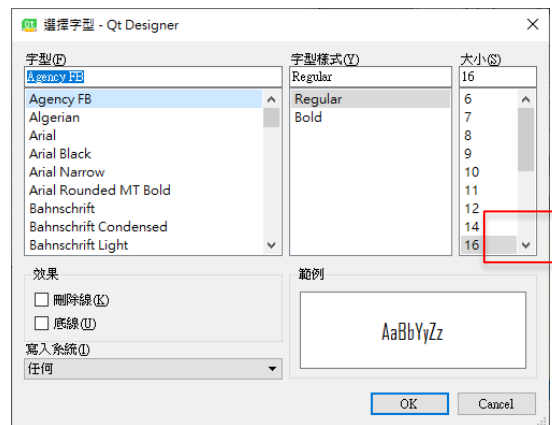


圖 5_8 選擇字型視窗

5. 改視窗標題

點選「屬性編輯器」下的「windowTitle」右邊的內容值，設為「Say Hello」，視窗標題同步顯示修改的內容值見下圖 5_9。

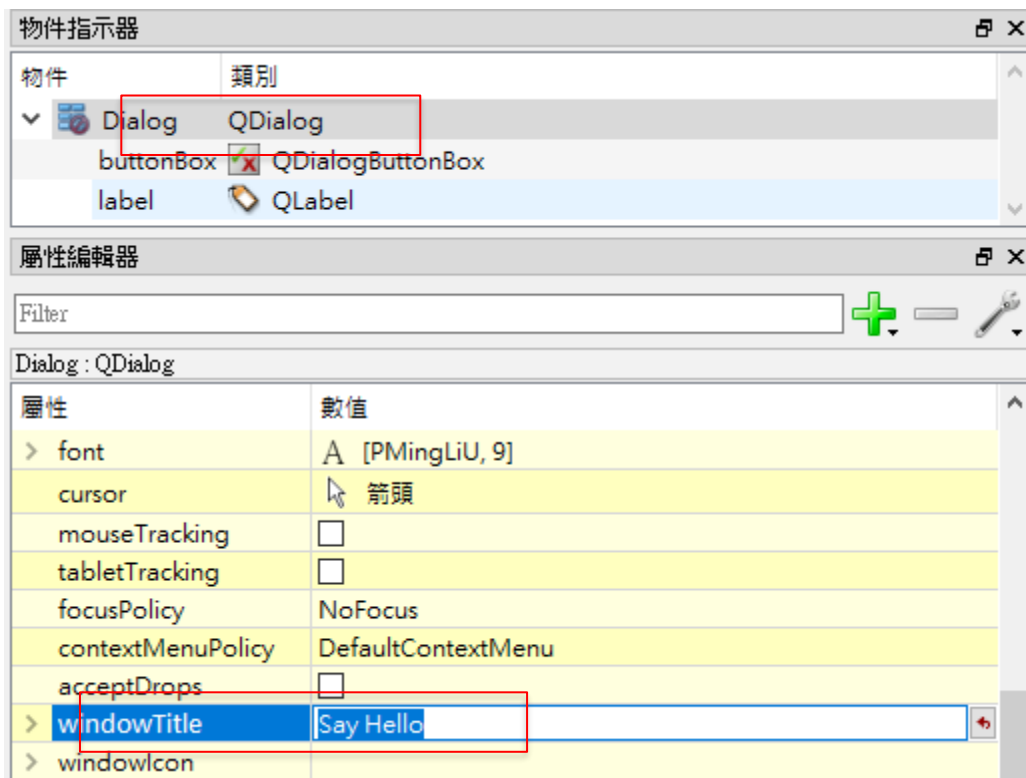


圖 5_9 修改 windowTitle 屬性值

6. 存檔 *.ui

點選檔案下的儲存，出現存檔對話方塊，選完指定路徑後，輸入檔案名稱 **Ex5_1**，副檔名為.ui 見下圖 5_10。

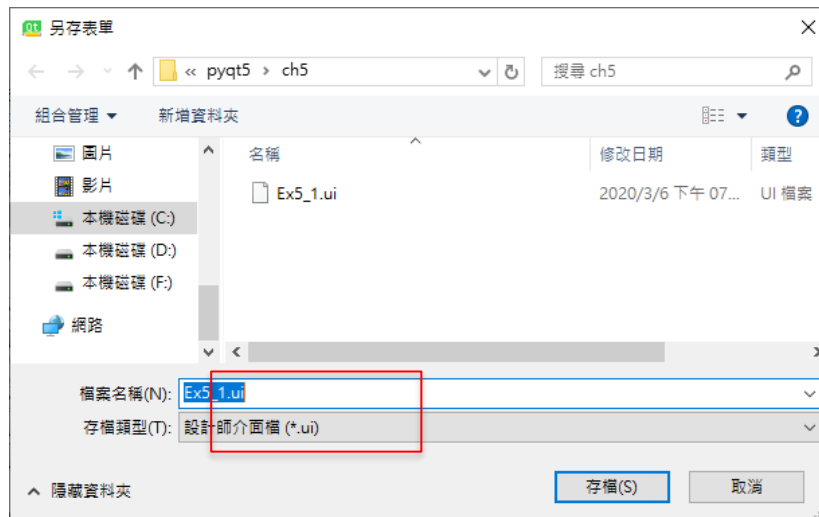


圖 5_10 儲存檔案為 Ex5_1.ui

7. 檢視設計介面

設計畫面結果見下圖 5_11。

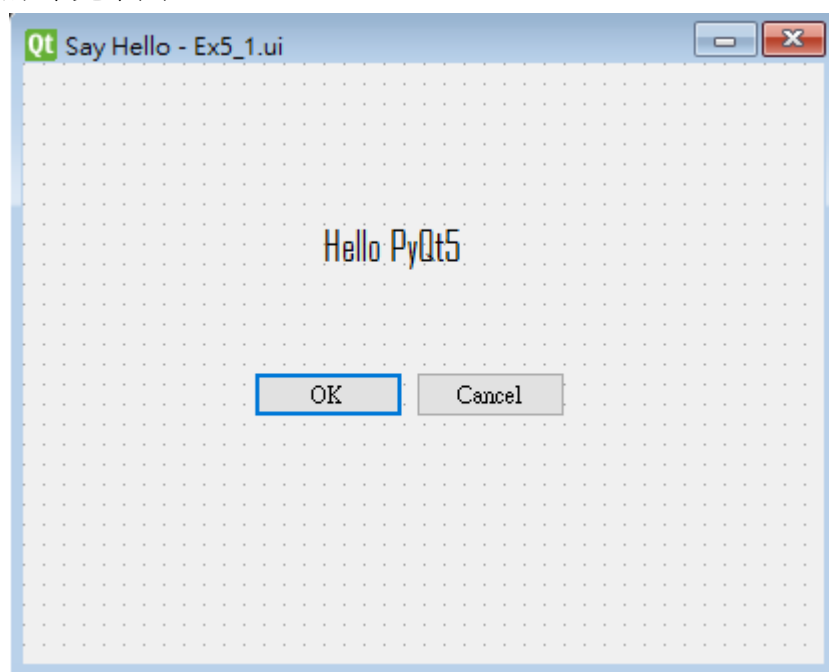


圖 5_11 Say Hello 的設計畫面

● pyuic5

pyuic5 已隨 Anaconda 安裝到本機，pyuic5 程式的功能是將 *.ui 檔案轉成 *.py 檔案。操作步驟如下：

①點選作業系統視窗的左下角的「開始」找 Anaconda 3 資料夾下的 Anaconda Prompt 命令視窗見下圖 5_12。

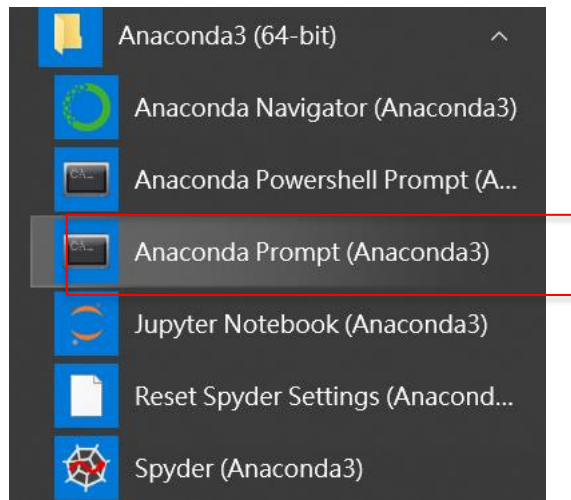


圖 5_12 執行 Anaconda Prompt

②假設 Anaconda 3 是安裝在 c: users\user 資料夾，需要切換到儲存 Ex5_1.ui 檔案的路徑。若 Ex5_1.ui 的檔案儲存在 c:\pyqt5\ch5 資料夾下，必須切換到此路徑，輸入以下指令。

```
cd /d c:\pyqt5\ch5
```

說明: cd 是切換路徑的視窗命令，/d 是指定切換到後續的新路徑 c:\pyqt5\ch5，見圖 5_13。

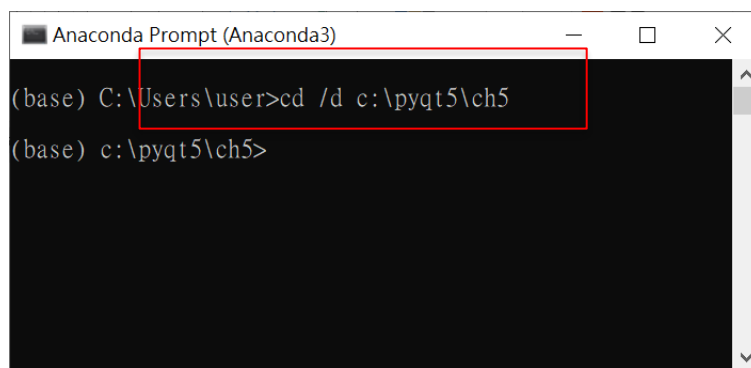


圖 5_13

③輸入 dir 指令顯示目前路徑下的檔案清單見下圖 5_14，此資料夾已經有 Ex5_1.ui 檔案。



```
C:\pyqt5\ch5>dir
磁碟區 C 中的磁碟沒有標籤。
磁碟區序號: 0000-4823

C:\pyqt5\ch5 的目錄

2020/03/06 下午 07:42 <DIR> .
2020/03/06 下午 07:42 <DIR> ..
2020/03/06 下午 07:23      1,938 Ex5_1.ui
                        1,938 位元組
                        2 個目錄 177,205,395,456 位元組可用

C:\pyqt5\ch5>
```

圖 5_14 檢視資料夾內的檔案

8. 將 *.ui 轉成 *.py 檔案

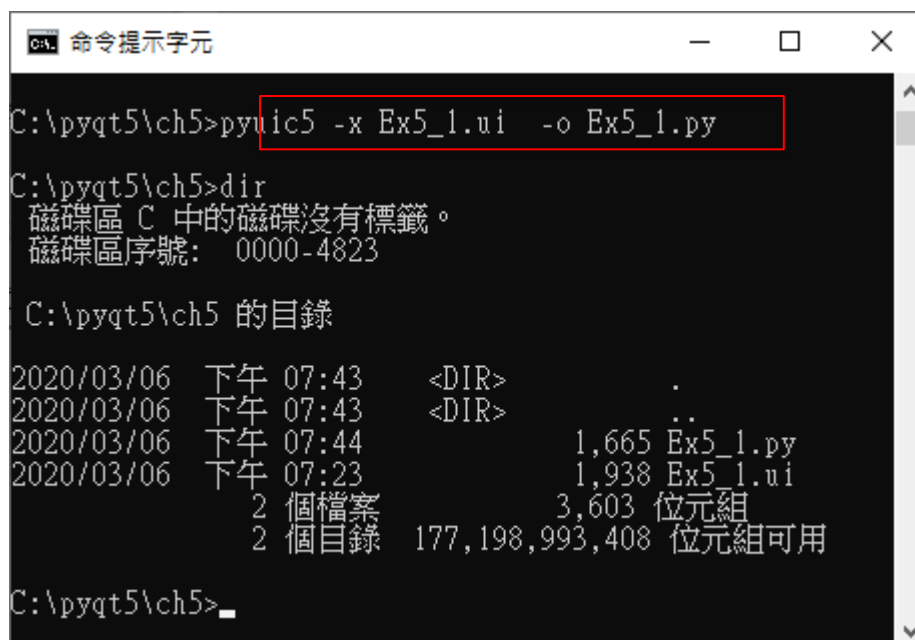
命令視窗輸入轉檔程式 pyuic5 進行轉檔，操作指令語法如下：

pyuic5 (ui 檔名).ui -o (輸出的 py 檔名).py

說明：

- -x 是加入主程式碼 main()，轉成可以直接執行 python 的程式；
- Ex5_1.ui 是輸入的介面檔案名稱；
- -o 是指定要輸出程式；
- Ex5_1.py 是要輸出的.py 檔案名稱；

pyuic5 -x Ex5_1.ui -o Ex5_1.py



```
C:\pyqt5\ch5>pyuic5 -x Ex5_1.ui -o Ex5_1.py

C:\pyqt5\ch5>dir
磁碟區 C 中的磁碟沒有標籤。
磁碟區序號: 0000-4823

C:\pyqt5\ch5 的目錄
2020/03/06 下午 07:43 <DIR> .
2020/03/06 下午 07:43 <DIR> ..
2020/03/06 下午 07:44      1,665 Ex5_1.py
2020/03/06 下午 07:23      1,938 Ex5_1.ui
                2 個檔案      3,603 位元組
                2 個目錄 177,198,993,408 位元組可用

C:\pyqt5\ch5>
```

圖 5_15 執行 pyuic5 轉檔程式

成功在 D:\pyqt5\ch5\資料夾下產生 Ex5_1.py 見下圖 5_16。

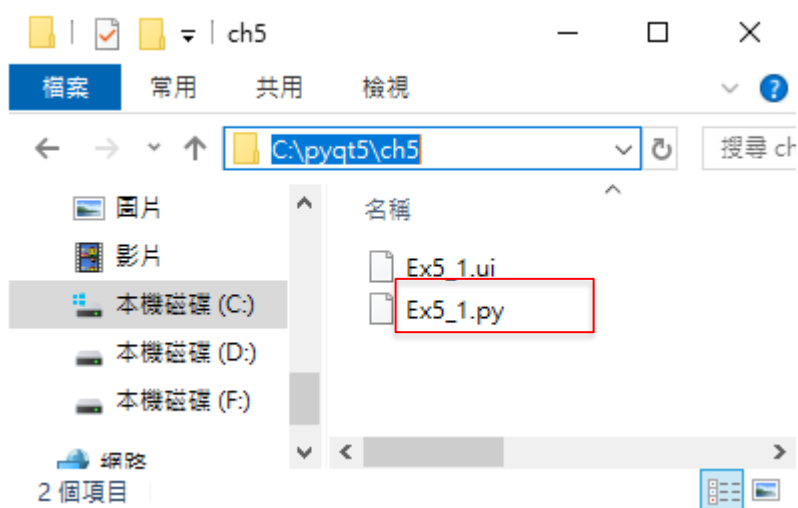


圖 5_16 檢視轉檔成功的檔案

● Spyder

撰寫 python 程式的編輯器很多，各有優缺點，前 10 名推薦的 python 程式編輯器包含 PyDev, Pycharm, 與 Spyder 等. 有些是需要付費的屬於商業版本，本書推薦屬自由軟體的 Anaconda，內裝好的簡易使用的 Spyder。Spyder 可用以編輯 Python 程式，提供程式編輯器視窗、互動式命令視窗、變數追蹤視窗等。支援跨平台(Windows, Macs, Linux)的編輯環境，並且 Spyder 是 PyQt 延伸的套件庫開發出來的。

註: 前 10 名推薦的 python 程式編輯器

<https://noeticforce.com/best-python-ide-for-programmers-windows-and-mac>

9. 啟動 Spyder

轉檔好的 Ex5_1.py，等待進入 Spyder 編輯器進程式撰寫。點選 windows 左下角視窗程式的 Anaconda3 資料夾下的 Spyder 後見下圖 5_17

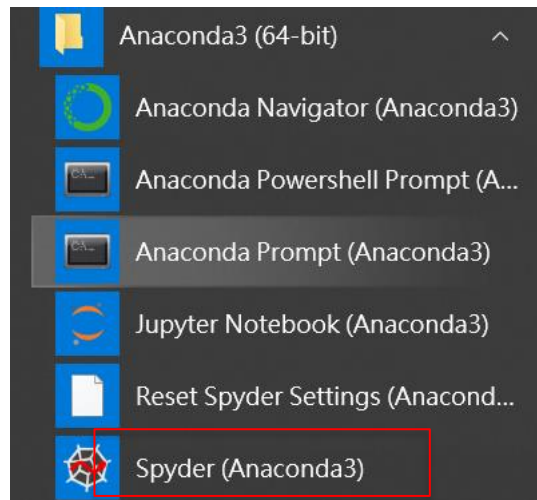


圖 5_17 啟動 Spyder

點選 Spyder 右上角功能表「File」之下的「Open」，開啟 Ex5_1.py 程式見下圖 5_18。

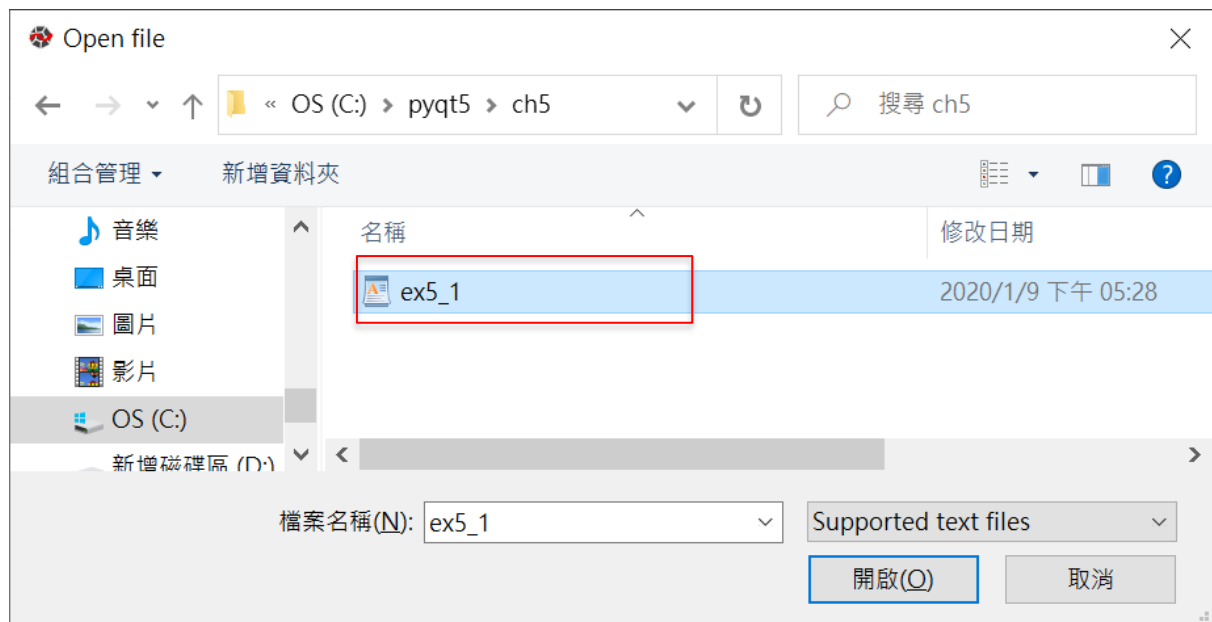


圖 5_18 Spyder 開啟檔案

10. 執行 python 程式

Ex5_1.py 檔案開啟完成見圖 5_19，點選上方綠色三角形按鈕執程式。

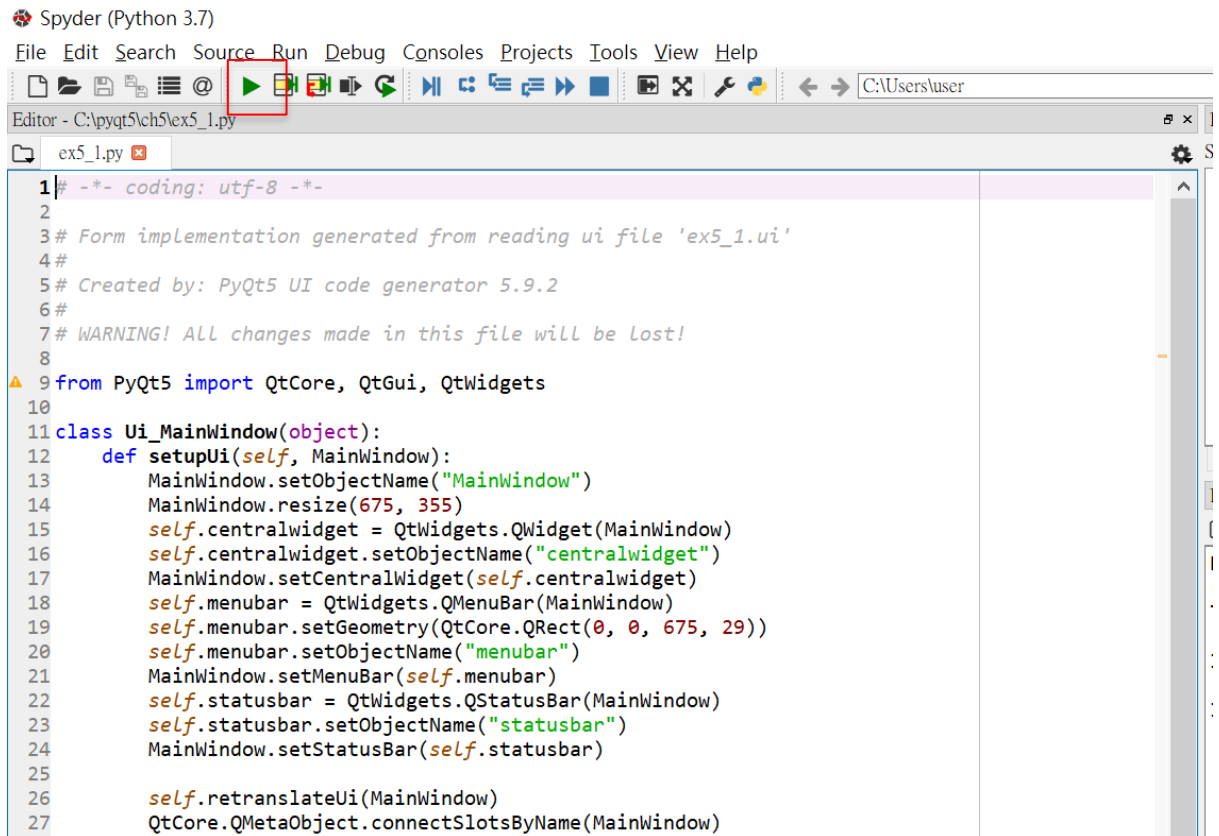


圖 5_19 Spyder 開啟檔案成功

執行結果

Ex5_1.py 程式的執行結果見下圖 5_20。

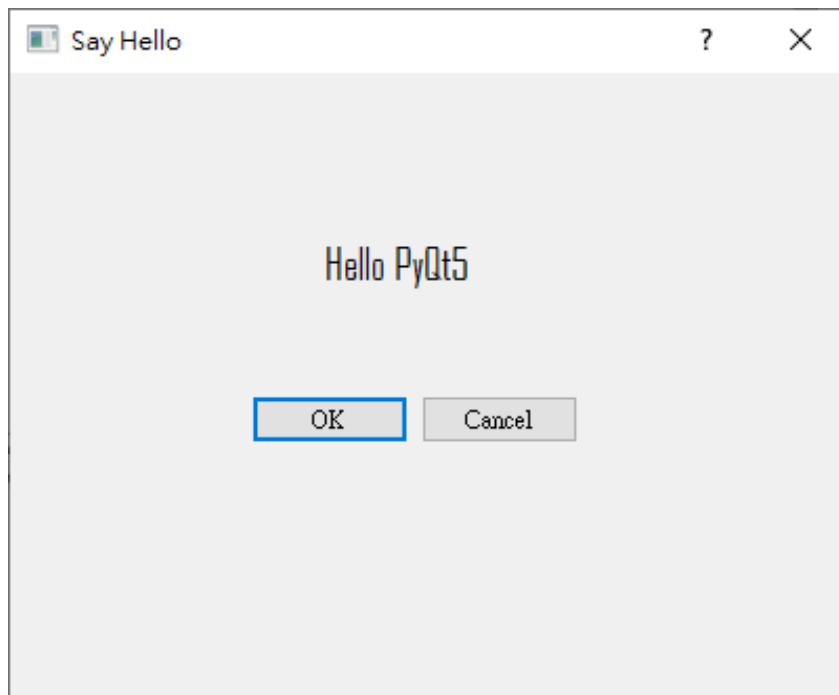


圖 5_20 Say Hello 程式執行畫面

※Spyder 視窗操作說明請參考附錄 A

1.4. GUI 介面程式設計

PyQt5 的元件視窗有，Layouts, Spacers, Buttons, Item views, Item widgets, Containers, Input widgets, Display widgets。本節將針對最常用的元件方法搭配實務案例做介紹。

1.4.1. QMainWindow 主視窗

主視窗提供一個標準的應用程式主畫面。在主視窗內可以配置標籤(QLabel)、文字方塊(QTextEdit)、按鈕(QPushButton)、核取方塊(QCheckBox)、下拉式選單(QComboBox)、日曆時間(QCalendar)、功能表(QMenuBar)等元件。

QMainWindow 繼承自 QWidget 類別，最常用的方法見表 5_2。

表 5_2 主視窗最常用的方法

名稱	說明
.setWindowTitle()	設定視窗標題。
.setStatusTip()	設定間接顯示狀態列訊息，滑鼠必須要滑進主視窗才會顯示提示狀態列字串。

Ex5_2 範例: 設定主視窗背景顏色、視窗標題與狀態列

主視窗是佈置版面的主要區塊，先介紹設定背景顏色再介紹設定狀態列與視窗標題。

(1) 設定背景顏色

①點選主視窗區域的任何位置為作用中(主視窗標題背景為淺藍色)，確認「物件指示器」中反灰的是「MainWindow」見下圖 5_21。

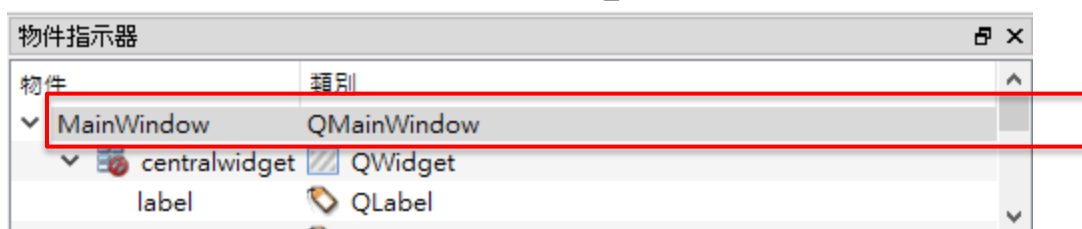


圖 5_21 物件指示器視窗-MainWindow

②點選「屬性編輯器」「QWidget」下的「palette」右側的「繼承」，點選後第一次會出現「變更調色盤」見下圖 5_22。

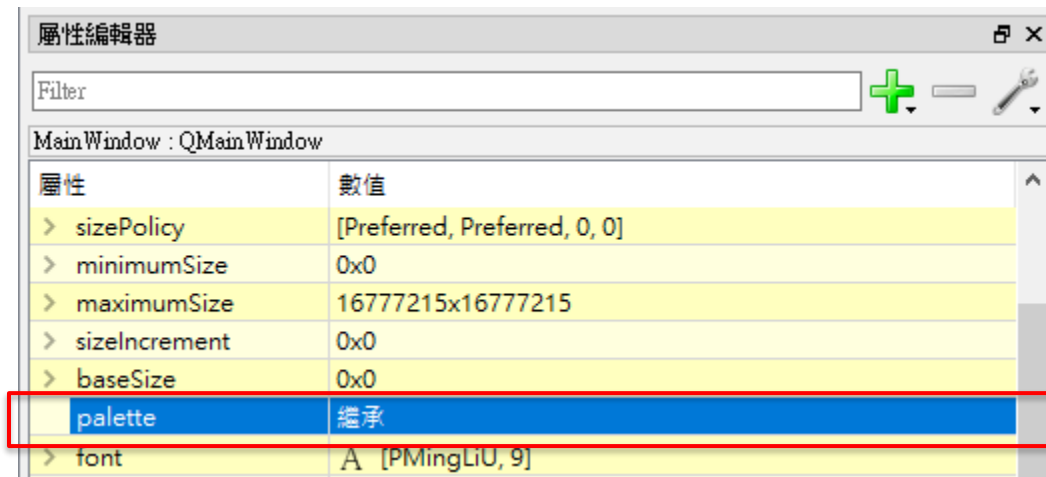


圖 5_22 屬性編輯器的 palette-MainWindow

③點選「變更調色盤」進入 Qt Designer 的「編輯調色盤」視窗，點選「快」右側的水平長條型如下圖 5_23。



圖 5_23 編輯調色盤-MainWindow

④進入 Qt Designer 的選擇顏色視窗，點選調色盤的顏色，點選基本顏色①，或點選右側的調色盤，並選綠色變成黑色十字；②或最右邊的調色長柱；③選好的顏色會顯示淺綠色的顏色垂直色盤；④後按「OK」如下圖 5_24。

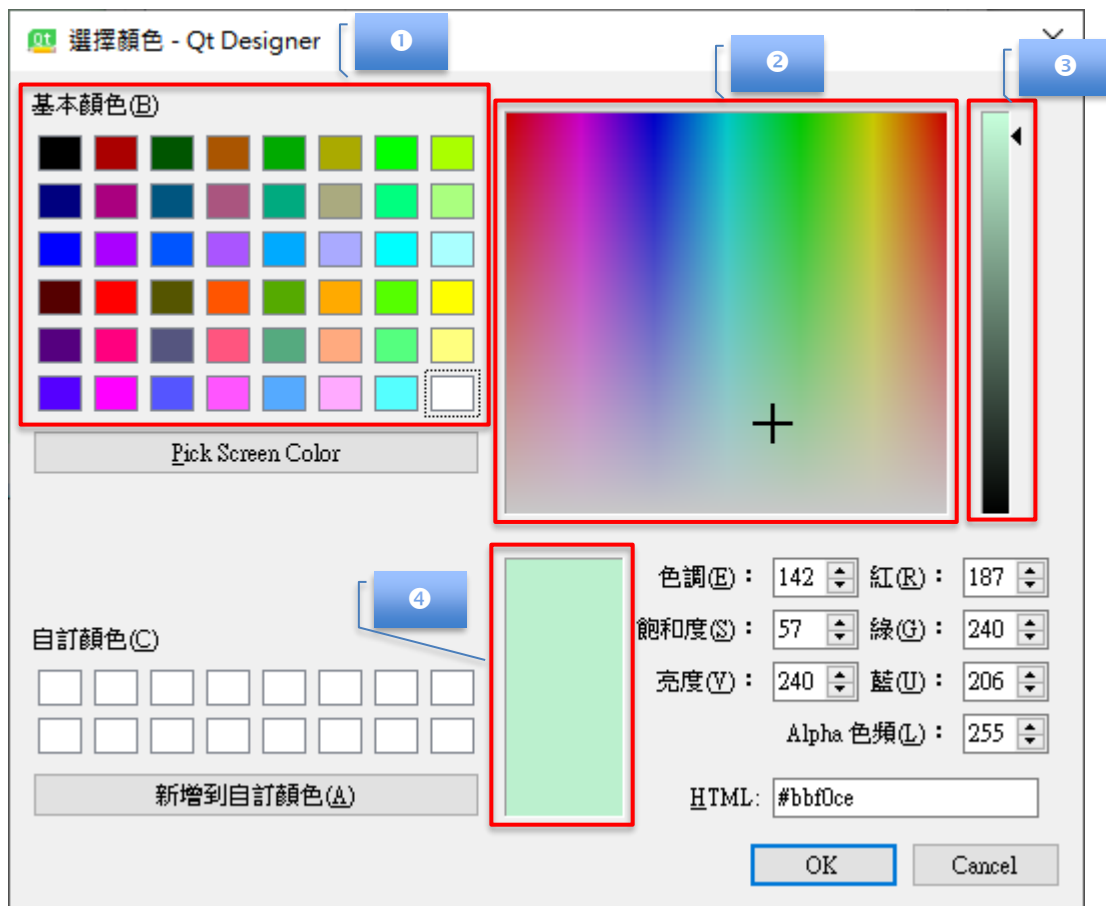


圖 5_24 選擇顏色-MainWindow

顯示結果

儲存檔案名稱為 Ex5_2.ui，在 Qt Designer 的主視窗背景顏色已經設定好見下圖 5_25。

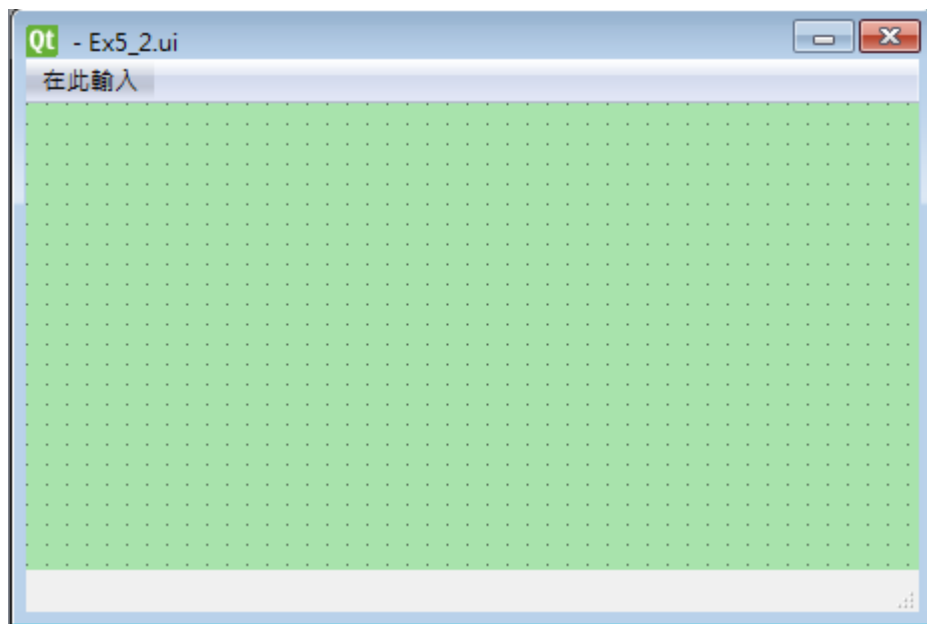


圖 5_25 主視窗設定顏色後的畫面

(2)設定視窗標題

設定視窗標題有兩種方式，❶Qt Designer 介面設定，只需拖曳元件到視窗，自動產生程式碼，介面操作方式很方便；❷手動輸入指令，方便於介面設計之後彈性增添程式碼，這兩種方法二選一即可。

● Qt Designer 介面設定

使用 Qt Designer 設定主視窗介面的操作步驟如下：

❶點選 Ex5_2.ui 的主視窗，確認「物件指示器」MainWindow 物件有反灰狀態如下圖 5_26。

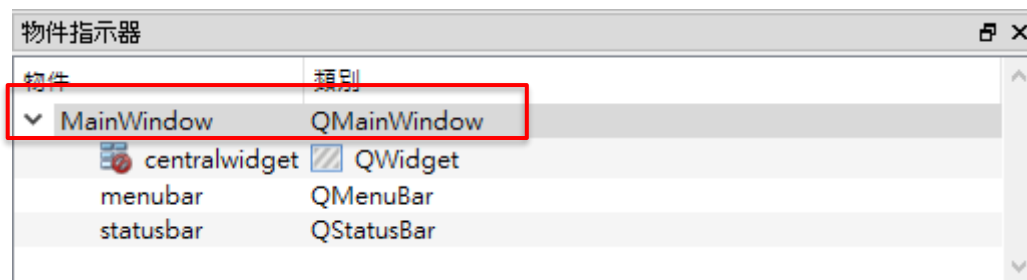


圖 5_26 物件指示器-MainWindow

❷點選「屬性編輯器」下的「QWidget」下的「windowTitle」並輸入「設定主視窗標題」見下圖 5_27。存檔 Ex5_2.ui。

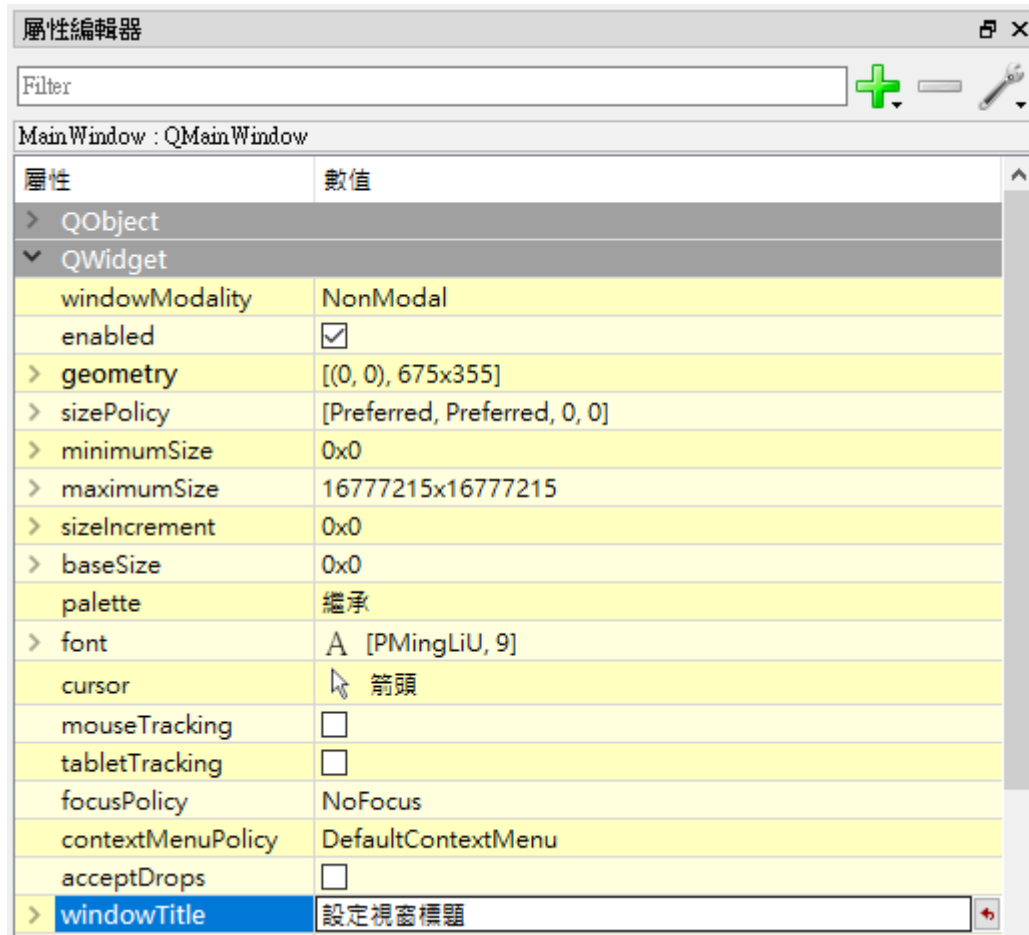


圖 5_27 屬性編輯器-MainWindow

- 手動輸入指令

Spyder 開啟 Ex5_2.py 程式碼見表 5_3，#註解後的指令不執行，並於第 163 列，藉由.setWindowTitle()方法手動輸入以下指令，設定視窗主題與帶入字串"設定視窗標題"，執行結果與 Qt Designer 介面設定會是一樣的，見圖 5_28。

163	MainWindow.setWindowTitle("設定視窗標題") ###新增此列
-----	---

(3) 設定視窗狀態列

主視窗的左下角底端可以設定狀態列，可用以提示使用者訊息，請於第 164 列新增以下程式碼，以.setStatusTip()方法代入字串"這是狀態列提示"，其他程式碼的說明請見 5_3 的註解。

164	MainWindow.setStatusTip("這是狀態列提示") ###新增此列
-----	--

表 5_3 Ex5_2.py 程式碼

1	# -*- coding: utf-8 -*-
2	
3	# Form implementation generated from reading ui file 'Ex5_2.ui'

4	#
5	# Created by: PyQt5 UI code generator 5.9.2
6	#
7	# WARNING! All changes made in this file will be lost!
8	
9	from PyQt5 import QtCore, QtGui, QtWidgets
10	
11	class Ui_MainWindow(object):
12	def setupUi(self, MainWindow): ###定義使用者介面 setupUi 函數()
13	MainWindow.setObjectName("MainWindow")
14	MainWindow.resize(675, 355)
15	palette = QtGui.QPalette()
16	brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
17	brush.setStyle(QtCore.Qt.SolidPattern)
...	...
151	MainWindow.setPalette(palette) ###設定調色盤
...	...
163	MainWindow.setWindowTitle("設定視窗標題") ###新增此列
164	MainWindow.setStatusTip("這是狀態列提示") ###新增此列
165	self.retranslateUi(MainWindow)
166	QtCore.QMetaObject.connectSlotsByName(MainWindow)
167	
168	def retranslateUi(self, MainWindow): ###定義 etranslateUi 函數
169	pass ###不做任何動作，維持程式結構完整性
170	
171	
172	if __name__ == "__main__": ###執行 python 程式
173	import sys ###引用系統套件
174	app = QtWidgets.QApplication(sys.argv) ###執行 QApplication 的方法產生 app
175	MainWindow = QtWidgets.QMainWindow()
176	ui = Ui_MainWindow()
177	ui.setupUi(MainWindow)
178	MainWindow.show() ###顯示主視窗
179	sys.exit(app.exec_()) ###執行 app

執行結果

按下 Spyder 的 F5 執行 Ex5_2.py 程式時會出現以下畫面，「設定視窗標題」出現在右上角。滑鼠移到主視窗的中間空白處就會在左下角底部出現「這是狀態列提示」字串見下圖 5_28。

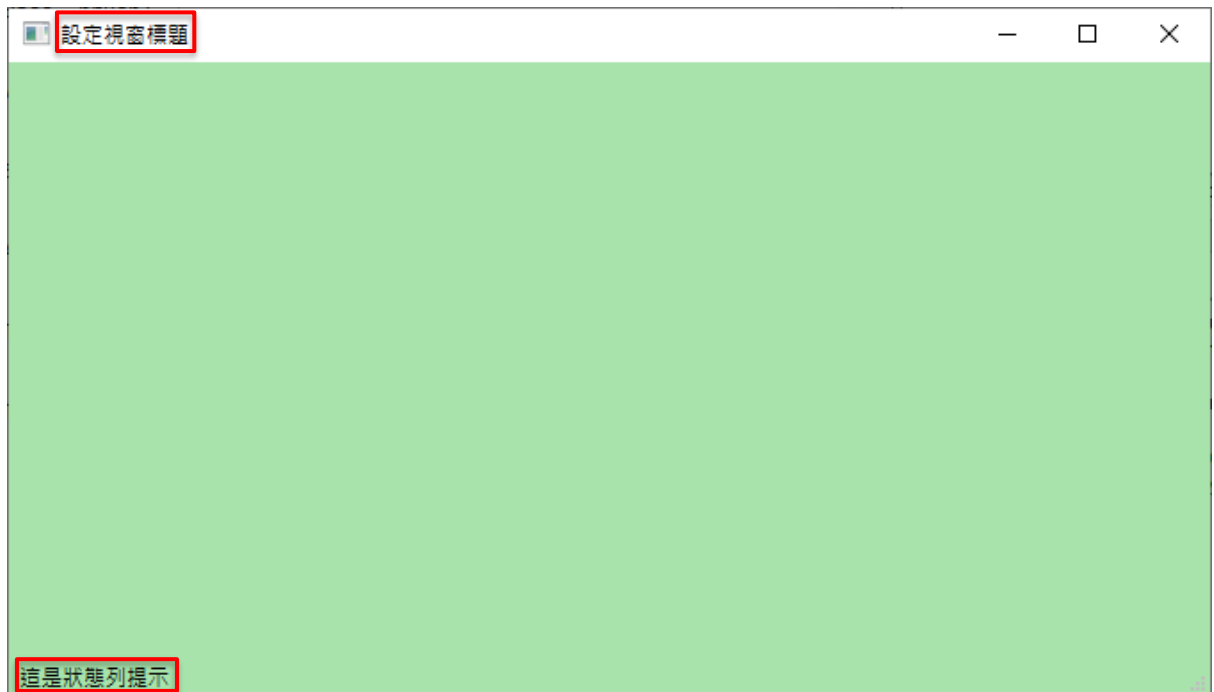


圖 5_28 設定主視窗背景顏色

1.4.2. QLabel 標籤

標籤(label)元件是繼承自 `QLabel` 類別，label 元件可以顯示不提供編輯的文字、圖片、動畫、超連結等。常用的方法包含取值、設定值、改變字型與對齊格式等見下表 5_4。

表 5_4: 標籤最常用的方法

名稱	說明
<code>.text()</code>	取出標籤的字串內容。
<code>.setText()</code>	設定字串到標籤。
<code>.selectedText()</code>	回傳使用者選取標籤的字串內容。
<code>.setPixmap()</code>	設定圖片到標籤。
<code>.setAlignment()</code>	設定標籤內容值的對齊方式： <code>Qt.AlignRight</code> : 靠右對齊 <code>Qt.AlignLeft</code> : 靠左對齊 <code>Qt.AlignCenter</code> : 水平置中對齊 <code>Qt.AlignVCenter</code> : 垂直置中對齊 <code>Qt.AlignJustify</code> : 左右間距對齊 <code>Qt.AlignTop</code> : 置頂端對齊 <code>Qt.AlignBottom</code> : 置底端對齊

Ex5_3: 實務案例-標準體重之標籤設計

本範例延用第 4 章 BMI 標準體重的範例，原來的做法由 Jupyter 指令視窗 (IPython console) 的使用者介面，改由 PyQt5 的介面設計，本範例重點在標籤

設計(1)拖曳標籤；(2)更改標籤寬度與輸入字串；(3)加入圖片與超連結；(4)更改標籤字型與顏色背景顏色。操作步驟如下：

(1) 拖曳標籤

本範例以幾個標籤進行介面設計，由上而下依序第一個標籤元件名稱預設為 label；第二到五個標籤元件名稱預設分別為 label_2, label_3, label_4, label_5，本書的元件名稱一律用 Qt Designer 預設，不另外取新名稱見下圖 5_29，檔案名稱儲存為 Ex5_3_1.ui。

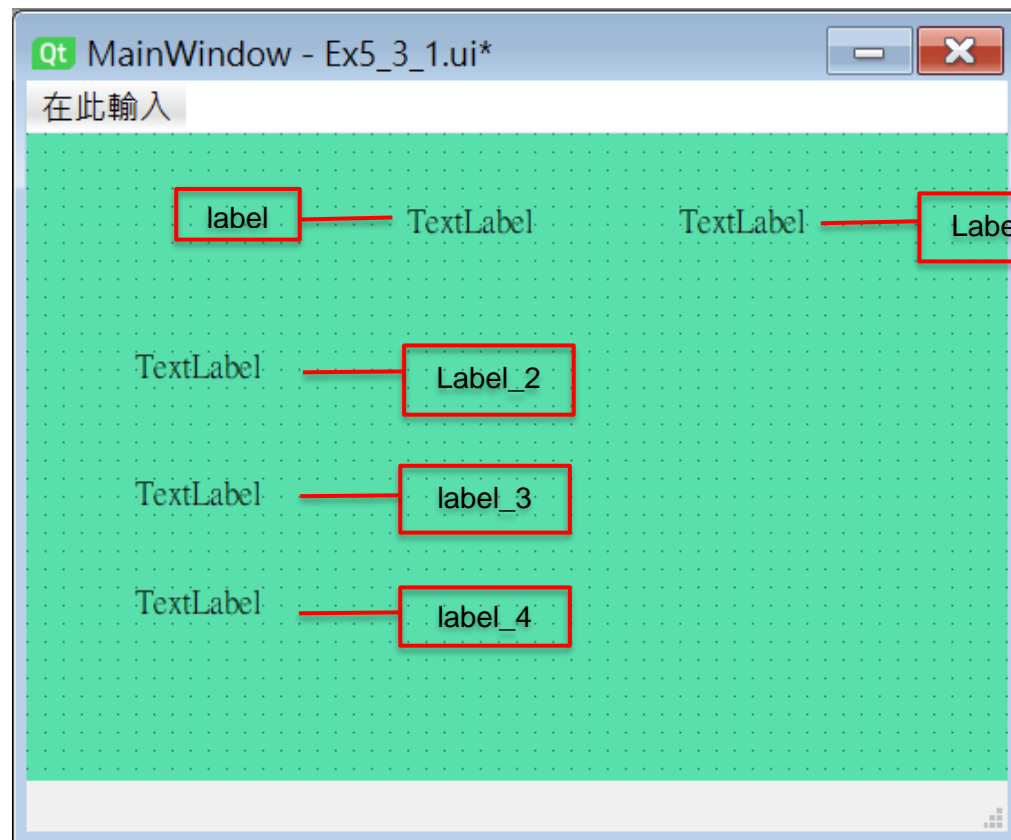
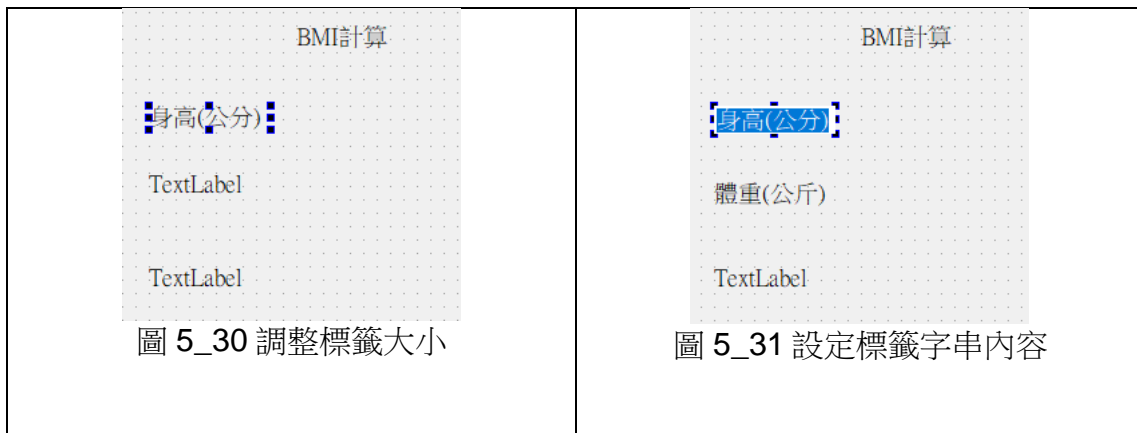


圖 5_29 拖曳標籤

(2)更改標籤寬度與輸入字串

接下來要對標籤進行設定顯示出來的字串，label 顯示字串設定為「BMI 計算」；label_2, label_3, label_4, label_5 顯示的字串分別設定為「身高(公分)」，「體重(公斤)」，「結果」，「點我說明」。若標籤寬度不夠顯示字串，可以點選此元件變成選取狀態的 8 個點後，上下左右拖曳調整元件大小見左下圖 5_30；雙擊標籤變成反藍可以輸入新的字串內容見右下圖 5_31。





label_5 顯示「結果」的標籤要向右拉長一些，之後要顯示標準體重計算的說明文字，文字說明約 10 個中文字。四個標籤顯示的字串設定完成後儲存為 Ex5_3_2.ui 轉成 Ex5_3_2.py。

執行結果

Ex5_3_2.py 在 Spyder 視窗執行結果見下圖 5_32。

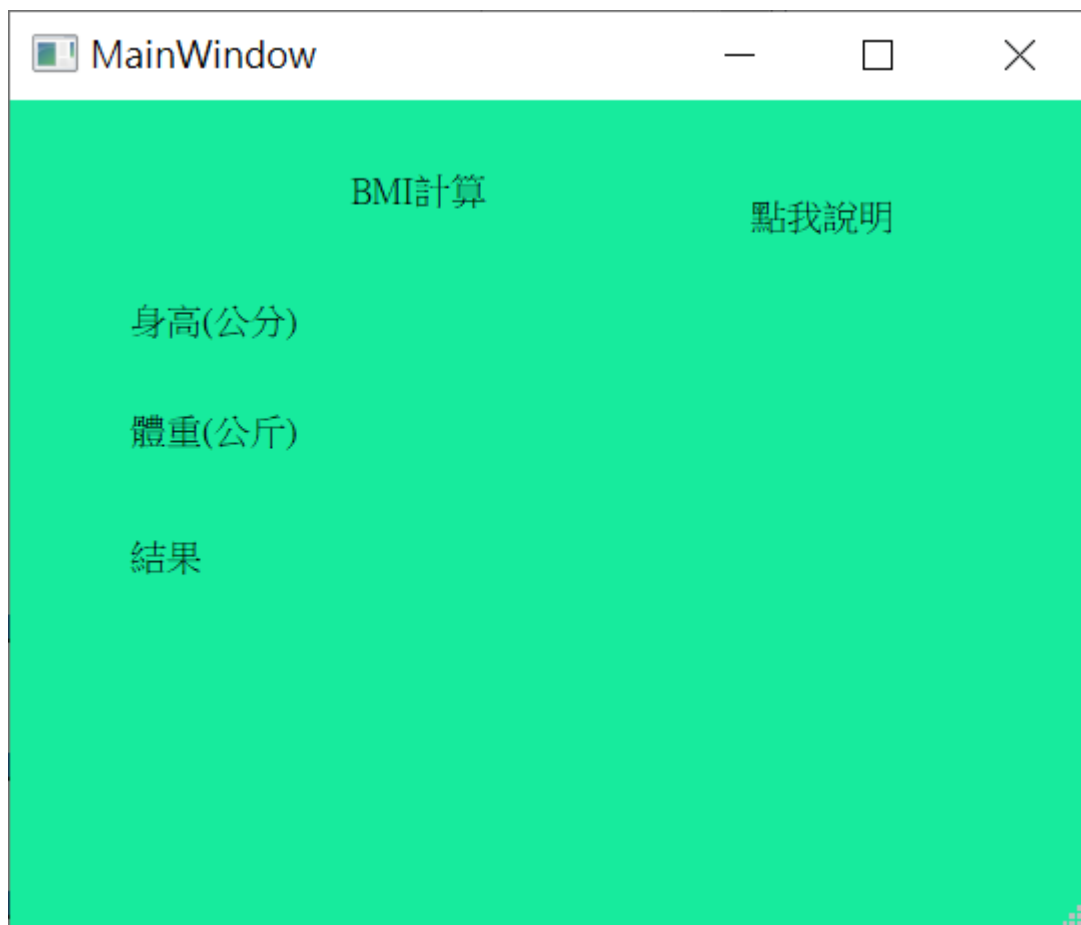


圖 5_32 BMI 標籤設計執行畫面

(3)加入圖片與超連結

本範例要試範將 label「BMI 計算」的文字改為圖示以及將 label_5「點我說明」加入一個超連結。

❶加入圖片

標籤可以插入文字也可以插入圖片，操作步驟如下：

①點選標題「BMI 計算」變成 8 個點見下圖 5_33。

②選擇視窗右側「屬性編輯器」中 QLabel 下「pixmap」見下圖 5_33。

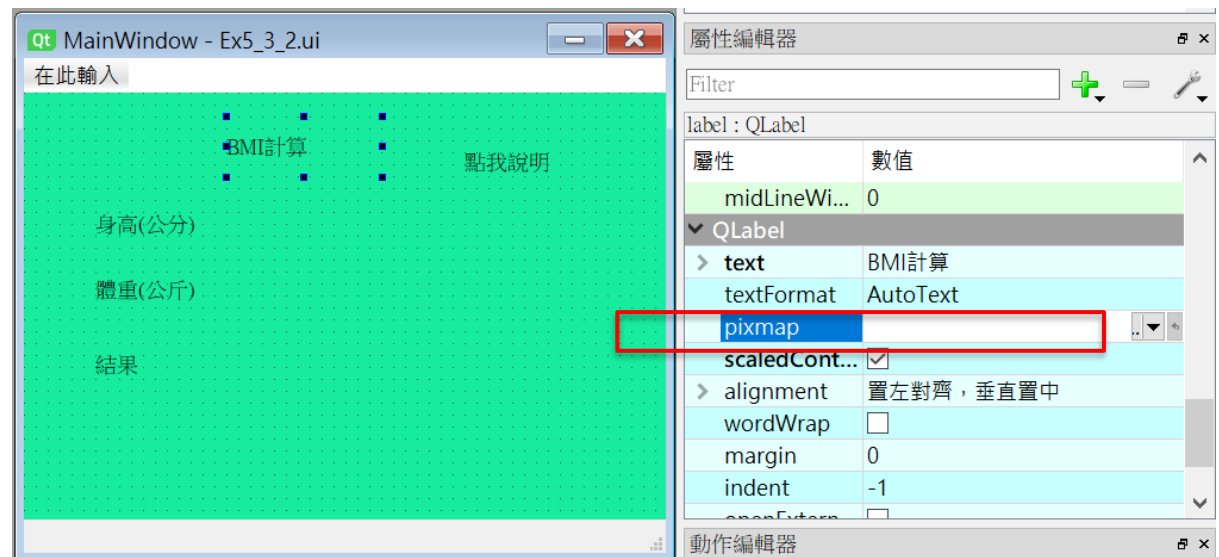


圖 5_33 label 的 pixmap 屬性

③點選 pixmap 的右邊下接式選單中「…」右側的下三角型，並點選「選擇檔案」見下圖 5_34。

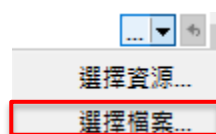


圖 5_34

④進入「選擇像素圖」視窗後，可以改變路徑，選擇 pyqt5\ch5 資料夾，點選 LabelPlot.png 後按「開啟」見下圖 5_35。

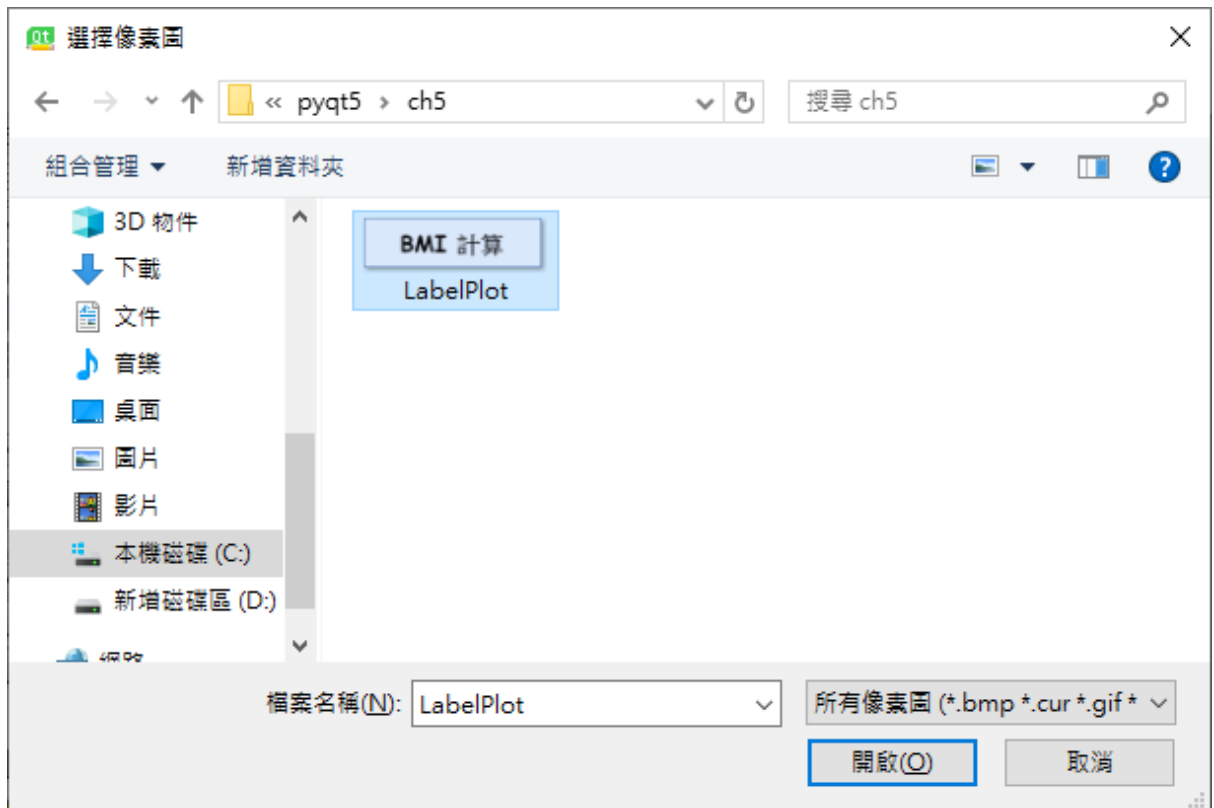


圖 5_35 label 載入圖片

⑨回到「屬性編輯器」中 QLabel 下勾選「scaledContents」右邊的核取方塊為打勾選取狀態見下圖 5_36，即可隨著元件大小改變時，會動態調整圖片大小以配適標籤元件的長與寬，若沒有打勾，圖片不會隨標籤元件調整大小，而且可能會被切邊。

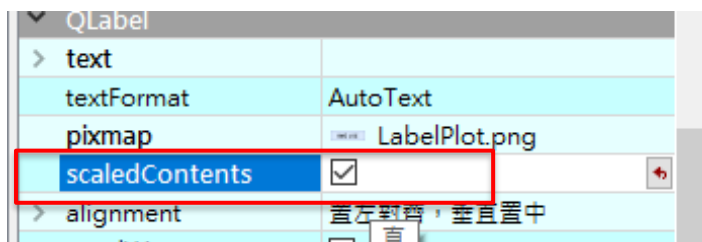


圖 5_36 label 的 scaledContents 屬性

⑩將圖片拖曳大一些，因為有將 scaledContents 打勾，圖片與 label 會一起調整拖曳的大小，介面視覺比較清楚。

執行結果

將檔案儲存為 Ex5_3_3.ui 並轉成 Ex5_3_3.py 並在 spyder 執行。pyqt5 調整元件大小很方便，依需求將拖曳圖片大小，視覺上比較美觀如下圖 5_37。



圖 5_37 標籤載入圖片執行結果

②加入超連結

標籤可以插入超連結網址，點選後可以連結到對外的指定網頁，操作步驟如下：

①點選標題「點我說明」變成 8 個點。

②選擇視窗右側「屬性編輯器」中 QLabel 下的「text」右邊的「…」見下圖 5_38。

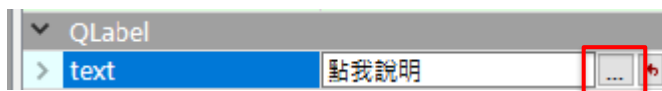


圖 5_38 label 的 text 屬性

③進入「編輯文字」視窗，點選右上角的地球連結的圖示見下圖 5_39。

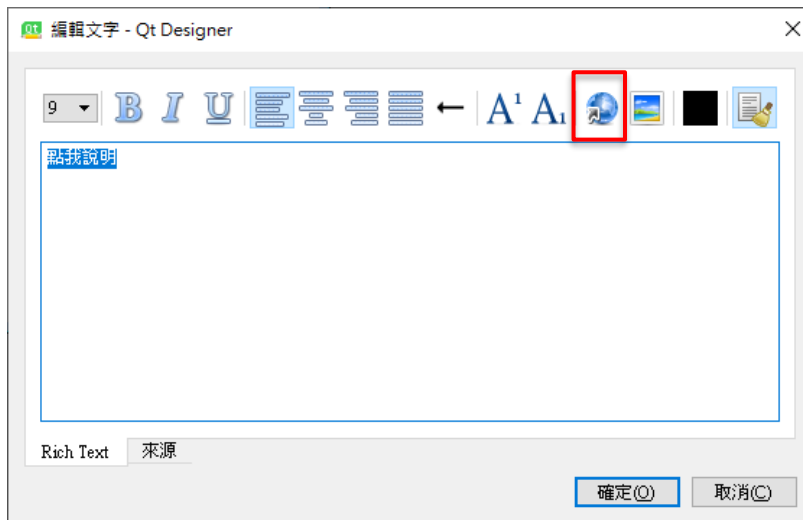


圖 5_39 label 的 text 屬性載入超連結

④進入「插入連結」視窗，並將以下網址(衛生福利部國民健康署官網)貼到網址欄位見下圖 5_40。

https://health99.hpa.gov.tw/OnlinkHealth/Onlink_BMI.aspx

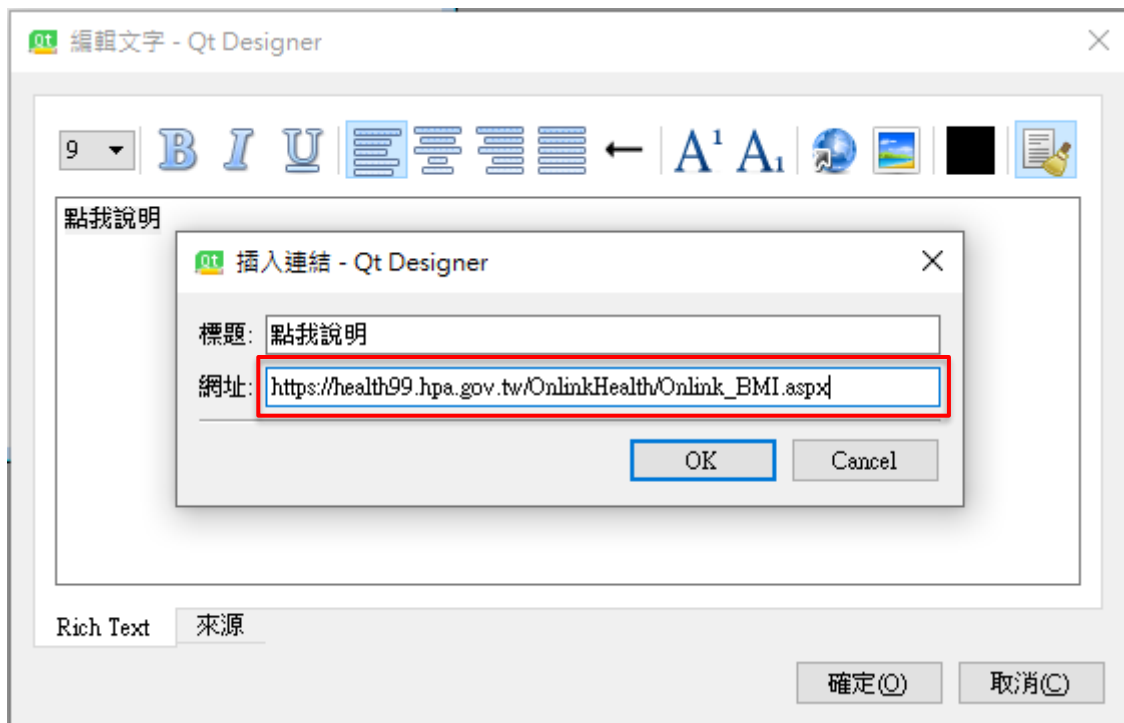


圖 5_40 label 貼入超連結

⑤按「OK」後再按「確定」回到主畫面，「點我說明」會變成藍色加底線的超連結形式。

⑥回到「屬性編輯器」中 Qlabel 下勾選「openExternalLinks」為選取狀態見下圖 5_41，即可在程式執行時點選「點我說明」開啟視窗連結到外部網站。

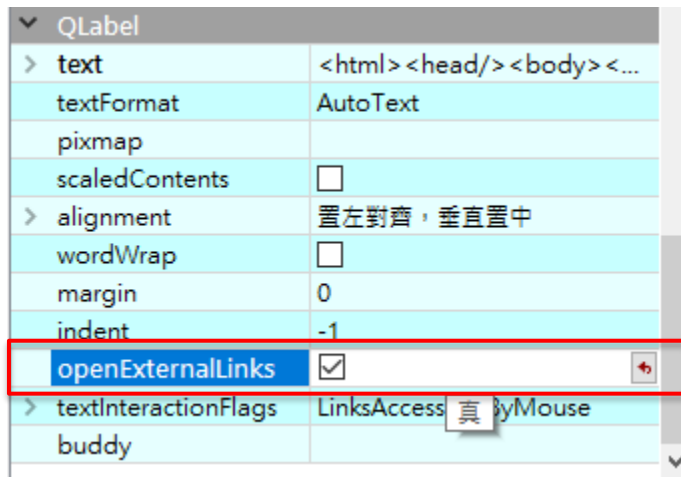


圖 5_41 勾選 OpenExternalLinks 屬性

將檔案儲存為 Ex5_3_4.ui 並轉成 Ex5_3_4.py 並在 spyder 執行測試看執行結果。

(4)設定元件字型與顏色

Qt Designer 可以提供使用者設定標籤元件的字型與顏色。

①回到主視窗，確認標籤是選取狀態，點選「屬性編輯器」，「Qwidget」下的「font」右側的「…」見下圖 5_42。

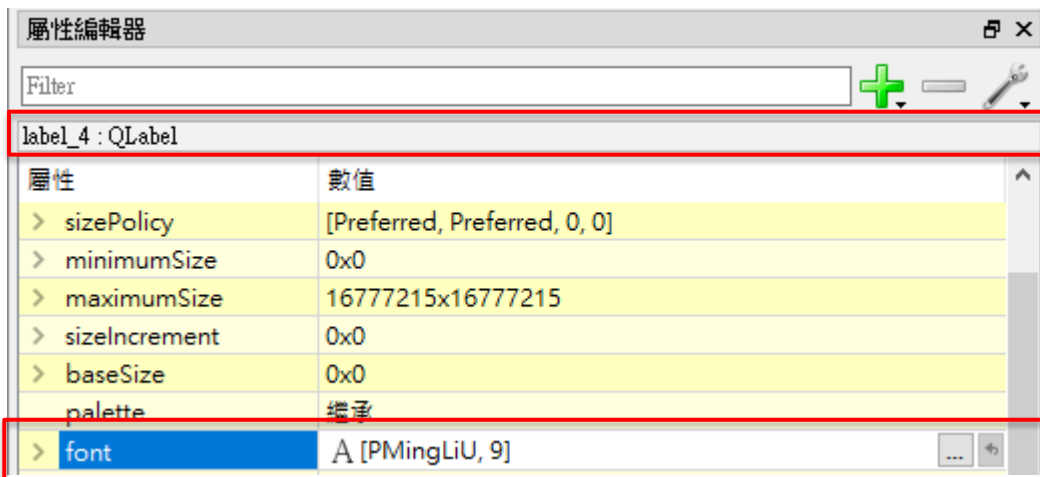


圖 5_42 label 的 font 屬性

進入 Qt Designer 的「選擇字型」視窗，字型選擇「標楷體」；字型樣式選「Regular」；大小選「12」，後按「OK」如下圖 5_43。

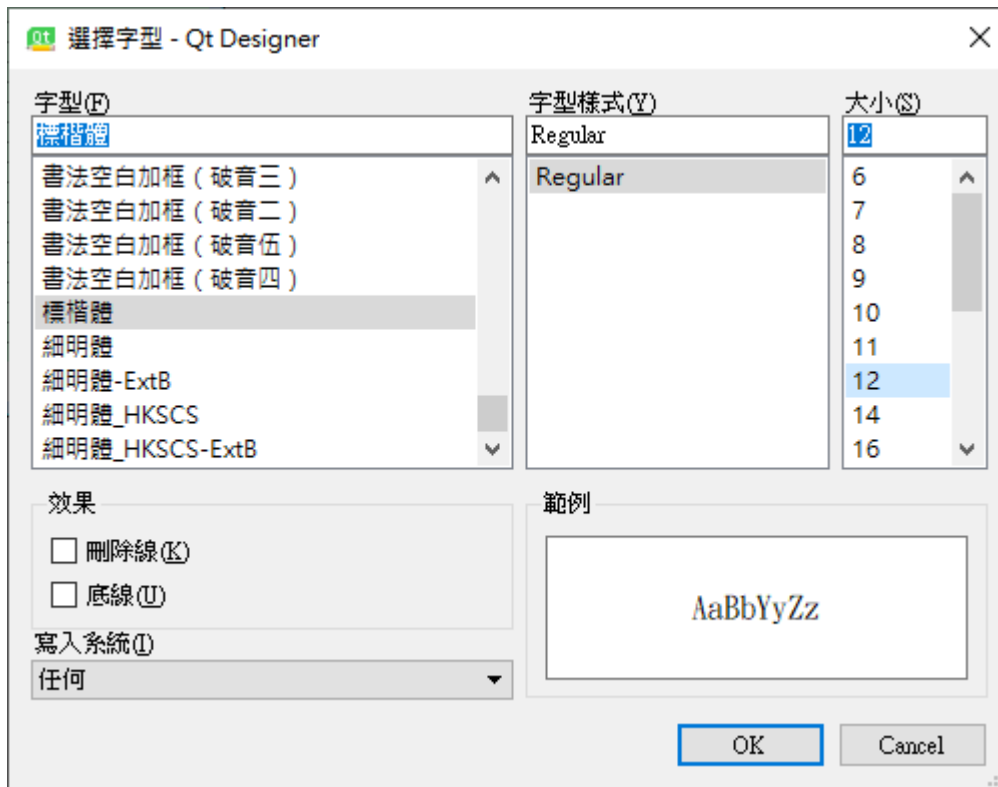


圖 5_43 label 元件進行字型設定

②設定標籤字型的顏色，與設定視窗背景顏色的操作流程類似，點選要修改標籤為 8 個點，再選「palette」進入編輯調色盤，再雙擊「WindowText」右邊的長方型見圖 5_44，進入調色盤並點選藍色同前範見圖 5_24，後按「OK」。

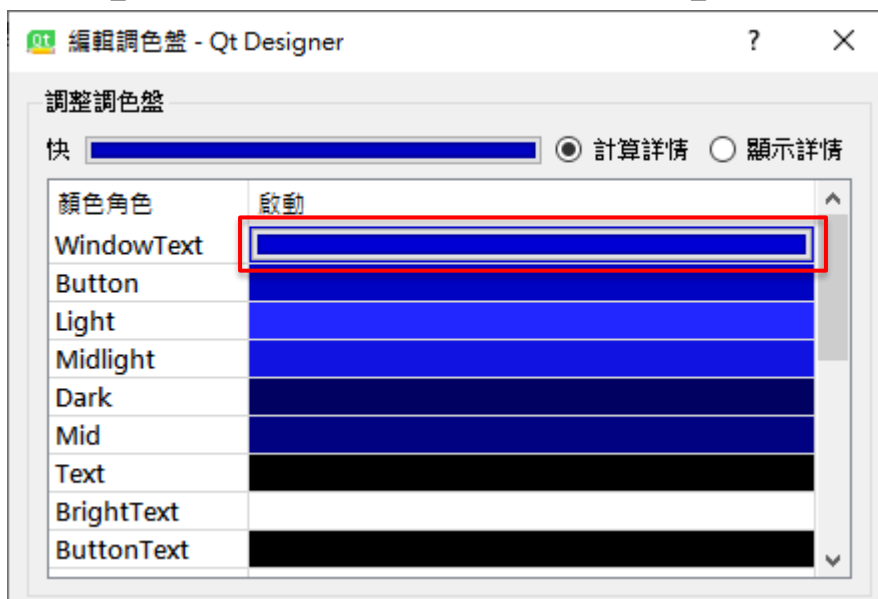


圖 5_44 label 調整

將檔案儲存為 Ex5_3_5.ui 並轉成 Ex5_3_5.py 並在 spyder 執行。

執行結果

在 spyder 執行 Ex5_3_5.py 程式，視窗會出現「點我說明」的超連結見下圖 5_45，點選後即會進入衛生福利部國民健康署的網站見下圖 5_46。



圖 5_45 BMI 標籤設計執行結果

BMI 測試



說明：

世界衛生組織建議以身體質量指數 (Body Mass Index, BMI) 來衡量肥胖程度，其計算公式是以體重 (公斤) 除以身高 (公尺) 的平方。國民健康署建議我國成人BMI應維持在18.5 (kg/m²) 及24 (kg/m²) 之間，太瘦、過重或太胖皆有礙健康。研究顯示，體重過重或是肥胖 (BMI≥24) 為糖尿病、心血管疾病、惡性腫瘤等慢性疾病的主要風險因素；而過瘦的健康問題，則會有營養不良、骨質疏鬆、猝死等健康問題。

【BMI 測試】

身高(cm)	成人肥胖定義	身體質量指數(BMI)(kg/m ²)	腰圍(cm)
	體重過輕	BMI<18.5	
	健康體位	18.5<=BMI<24	
	體位異常	過重：24<=BMI<27 輕度肥胖：27<=BMI<30 中度肥胖：30<=BMI<35 重度肥胖：BMI>=35	男性：>= 90 公分 女性：>= 80 公分

※ BMI = $\frac{\text{體重(公斤)}}{\text{身高}^2(\text{公尺}^2)}$

延伸閱讀：運動情形男女有別

您的BMI

圖 5_46 衛生福利部國民健康署的網站

1.4.3. QLineEdit 文字方塊

文字方塊分單列(QLineEdit)與多列(QTextEdit)兩種，本文介紹第一種。單列只能輸入一行文字於文字方塊中；最常用的方法包含取值、設定值、設定最長字串、設定字型與對齊格式等見下表 5_5。

表 5_5: QLineEdit 文字方塊最常用的方法

名稱	說明
.text()	取出文字方塊的字串內容。
.setText()	設定字串到文字方塊。
.selectAll()	全選文字方塊的字串內容。
.setFocus()	設定文字方塊為焦點，即為選取狀態。
.clear()	清除文字方塊的字串內容。
.setValidator()	設定使用者在文字方塊輸入的資料型態限制，並協助驗證是否符合輸入的限制條件如下： QIntValidator: 限制輸入整數資料型別。 QDoubleValidator: 限制輸入浮點數資料型別。
.setAlignment()	設定文字方塊內容值的對齊方式： Qt.AlignRight: 靠右對齊 Qt.AlignLeft: 靠左對齊 Qt.AlignCenter: 水平置中對齊 Qt.AlignVCenter: 垂直置中對齊 Qt.AlignJustify: 左右間距對齊 Qt.AlignTop: 置頂端對齊 Qt.AlignBottom: 置底端對齊

Ex5_4 實務案例-標準體重之文字方塊設計

文字方塊元件方便讓使用者輸入資料，本案例的設計重點(1)拖曳文字方塊；(2)設定文字方塊的字型；(3)設定文字方塊字型的顏色與背景顏色，操作步驟如下：
①點選左邊元件盒的「Input Widgets」下的「Line Edit」如下圖 5_47。

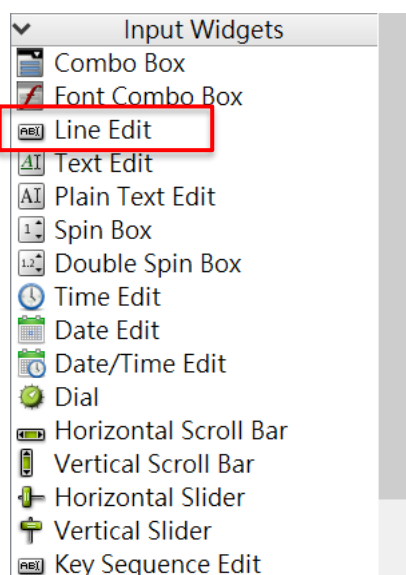


圖 5_47 拖曳文字方塊

②拖曳兩個 Line Edit 元件到主視窗，並點選為 8 個點的選取狀態下，向下拖曳放大文字方塊如下圖 5_48。

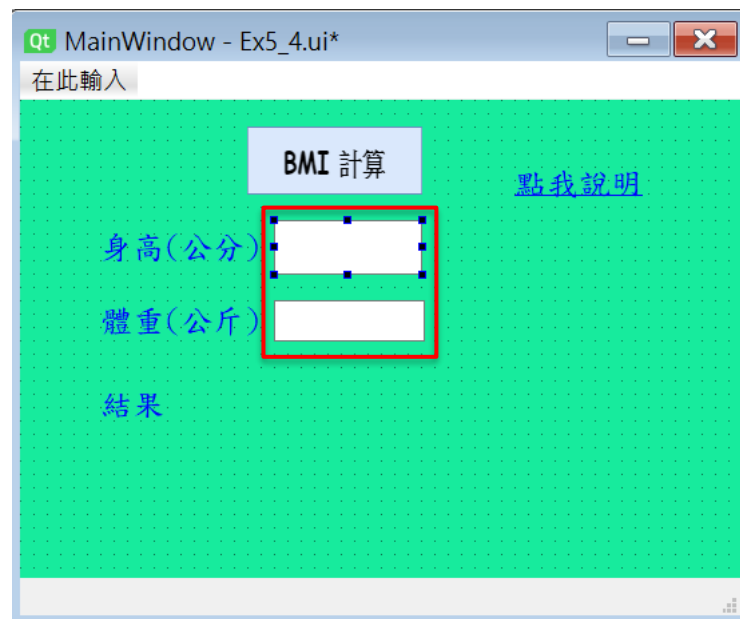


圖 5_48 調整文字方塊大小

③點選要更改大小的文字方塊變成 8 個點後，再點選視窗右側「屬性編輯器」中 QWidget 下的「font」右邊的「...」見下圖 5_49。

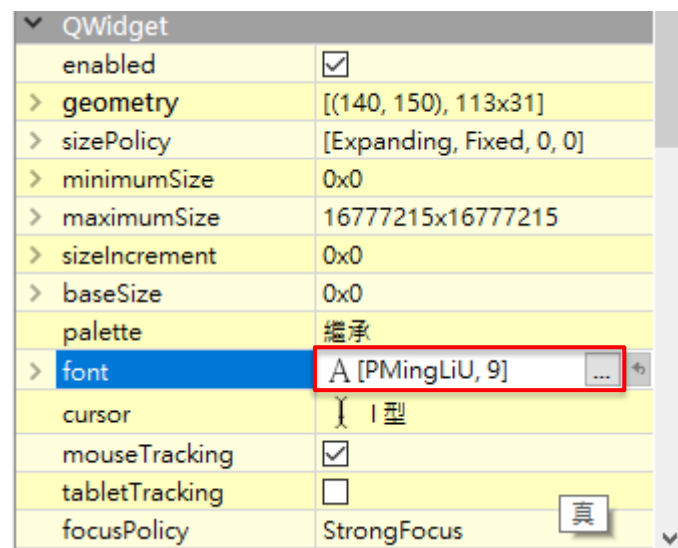


圖 5_49 點選文字方塊的 font 屬性

④進入「選擇字型」視窗，字型選擇「Times New Roman」；字型樣式選擇「Regular」；大小選擇「12」，再點選「OK」，見下圖 5_50。將檔案儲存為 Ex5_4.ui 並轉成 Ex5_4.py。

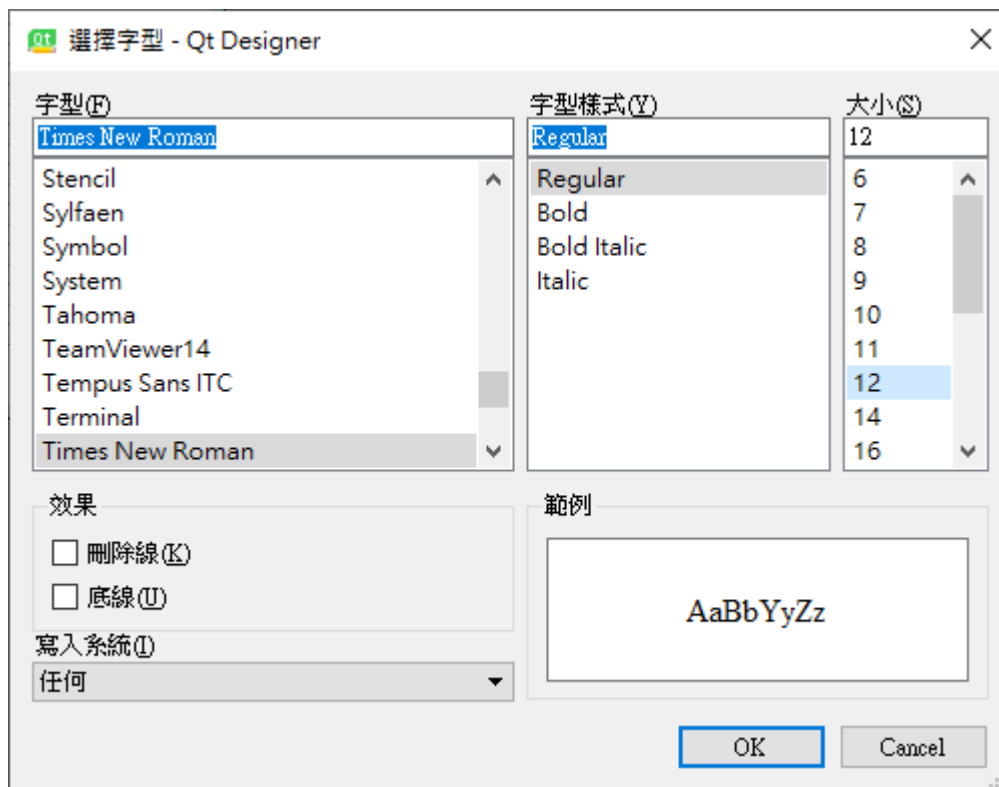


圖 5_50 文字方塊的選擇字型視窗

執行結果

spyder 執行 Ex5_4.py 的結果如下圖 5_51。可以在文字方塊中輸入數字 165 與 55，但還沒寫程式，因此還不能做計算。

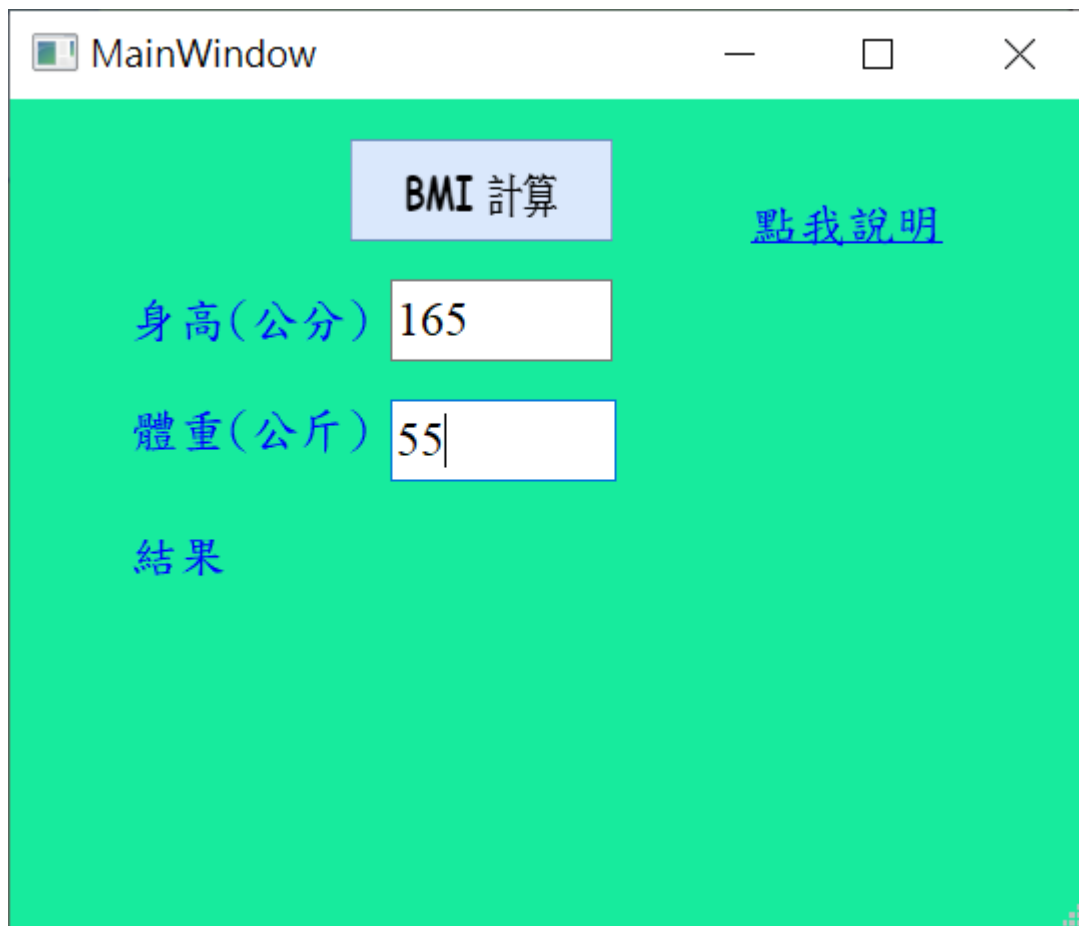


圖 5_51 文字方塊輸入模式的執行結果

1.4.4. QPushButton 按鈕

按鈕(Button)是所有元件當中最廣泛使用的，幾乎每個 GUI 介面透過標籤與文字方塊等元件收集資料後，必須要設計一個按鈕(PushButton)，將使用者輸入的資料藉由按鈕發出訊號觸發相對應的處理事件或程式進行運算。按鈕元件細分 QPushButton, RadioButton, CheckBox 三種，三者同屬繼承自 AbstractButton 類別。本節先介紹 QPushButton，其常見的方法見表 5_6。

表 5_6: QPushButton 最常用的方法

名稱	說明
.isEnabled()	檢測 QPushButton 是否為啟用狀態。
.isDown()	檢測 QPushButton 是否被按下。
.setIcon()	設定 QPushButton 的圖示。
.setText()	設定字串到 QPushButton。
.text()	取出 QPushButton 的字串內容。

Ex5_5 實務案例-標準體重之按鈕介面設計

本案例的設計重點是(1)拖曳按鈕(Push Button)到視窗；(2)按鈕的文字與背景顏色設定，操作步驟如下：

①點選左邊「元件盒」的「Buttons」下的「Push Button」如下圖 5_52。

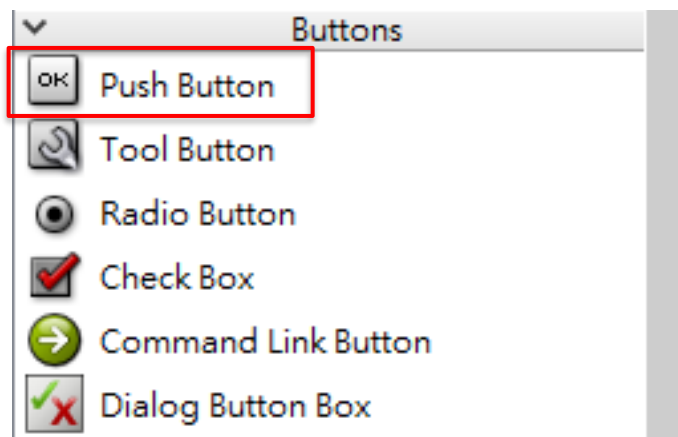


圖 5_52 按鈕元件

②拖曳到視窗見下圖成 8 個點如下圖 5_53。

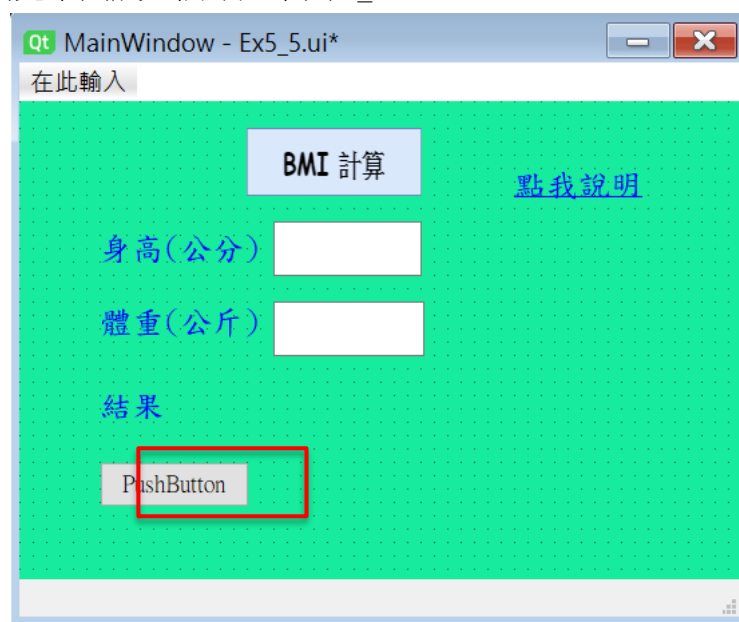


圖 5_53 拖曳按鈕到視窗

③確認有點選按鈕元件成 8 個點後，再點選右側「屬性編輯器」視窗中 QAbstractButton 下的「text」右邊的「...」見下圖 5_54。

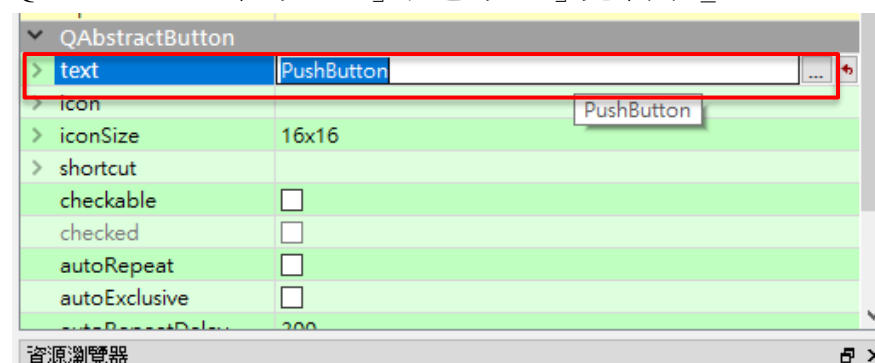


圖 5_54 PushButton 的 text 屬性

④進入「編輯文字」視窗，輸入「計算」，再點選「OK」，見下圖 5_55。

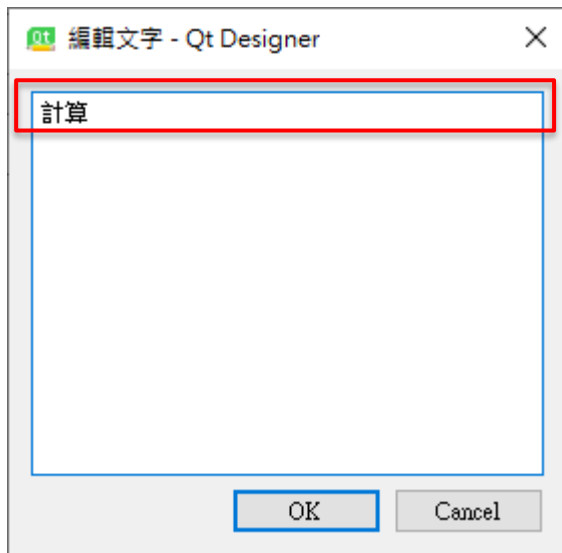


圖 5_55 編輯按鈕中顯示的文字

⑥設定按鈕背景顏色，點選按鈕元件成 8 個點後，再點選右側「屬性編輯器」視窗中 QWidget 下的「styleSheet」右邊的「...」見下圖 5_56。

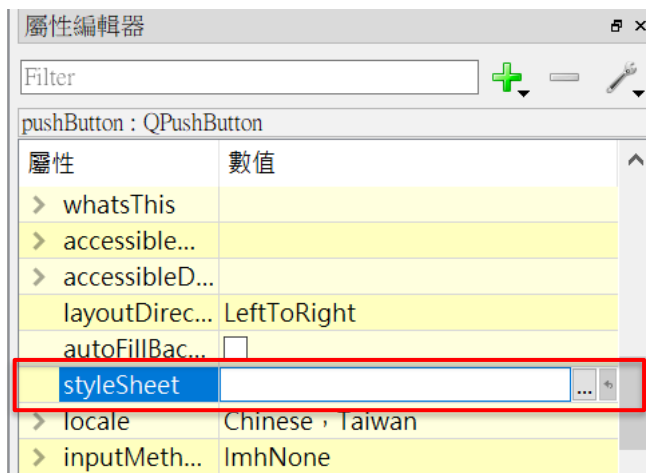


圖 5_56 點選 styleSheet 屬性

⑦進入「編輯樣式表」視窗後，點選「新增顏色」下的「color」見下圖 5_57，進入 Qt Designer「選擇顏色」視窗見圖 5_58 可以設定字型的顏色，建議設定為藍色。按下「OK」。

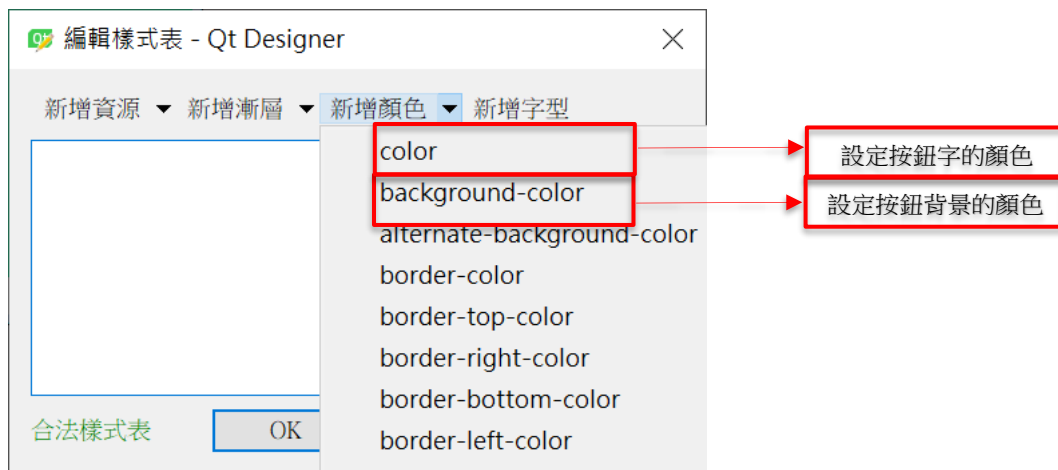


圖 5_57 新增樣式顏色

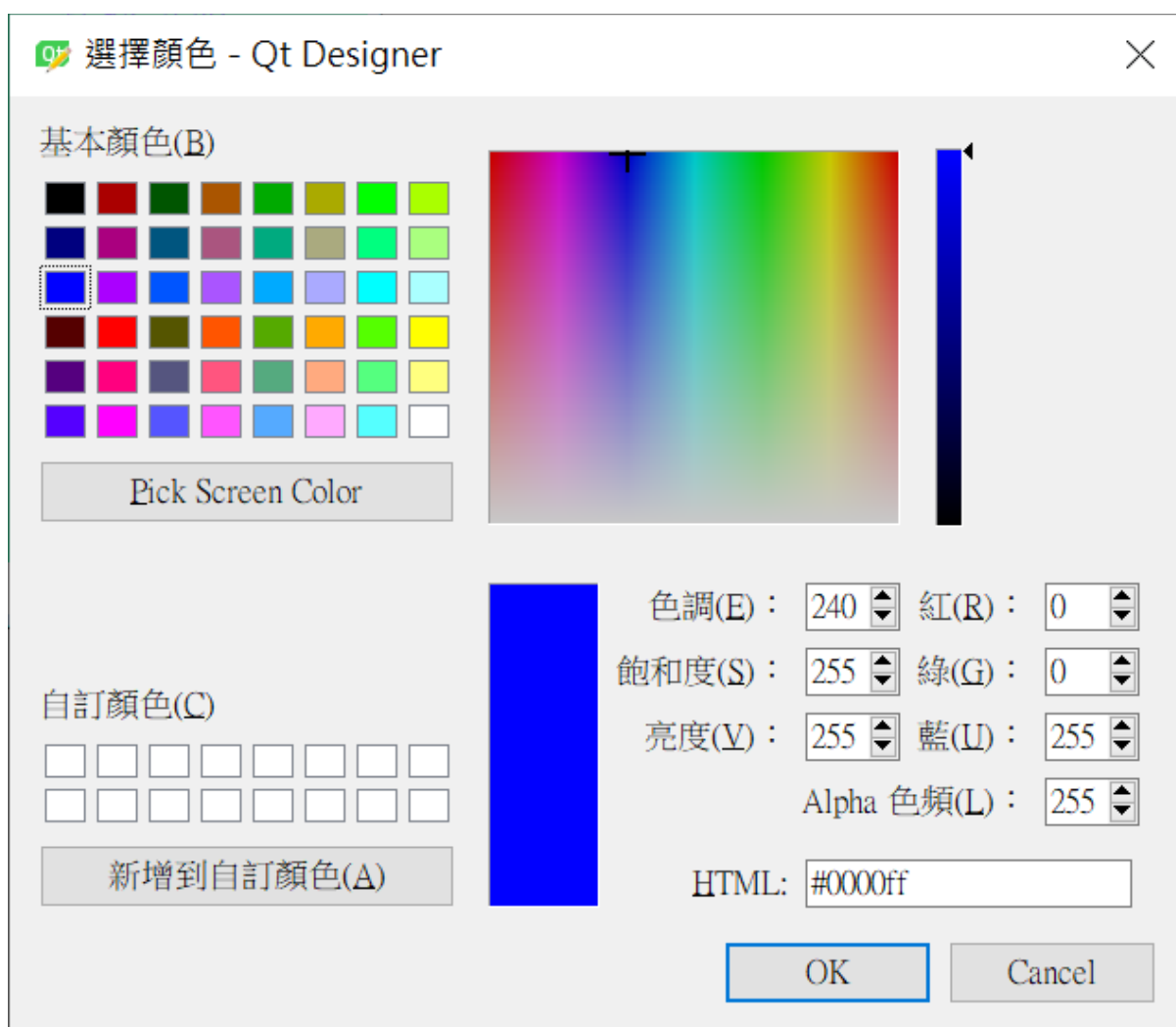


圖 5_58 設定按鈕的字型顏色

⑦回到「編輯樣式表」見圖 5_57，點選「background-color」選擇淺藍色後按下「OK」，樣式指令自動填好見圖 5_59。

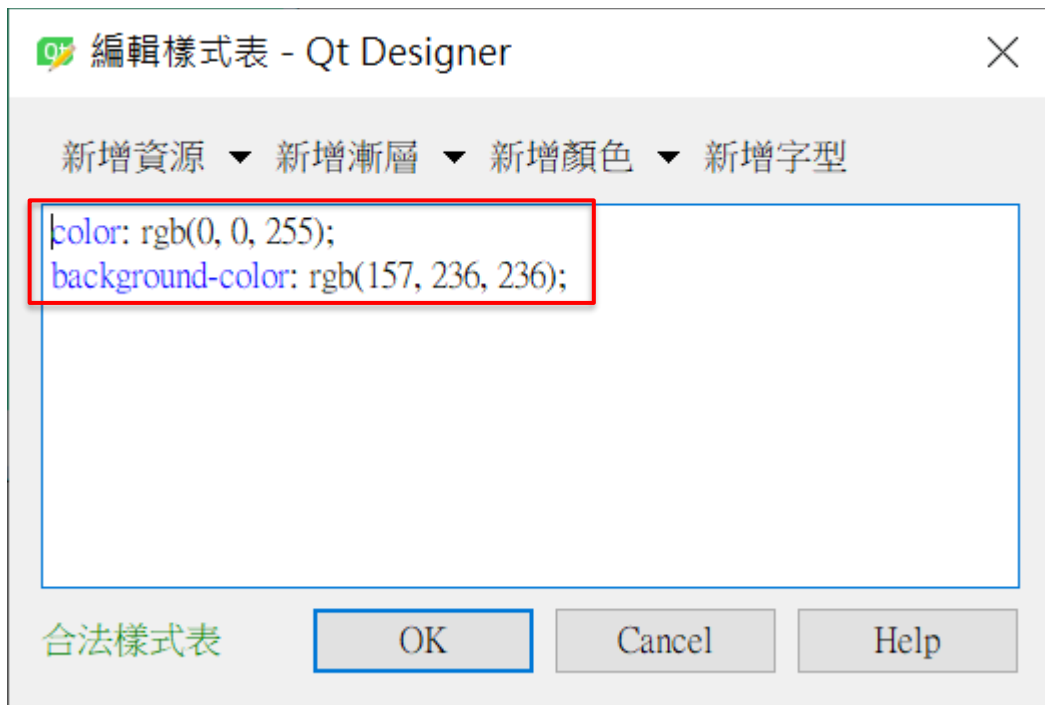


圖 5_59 背景樣式指令自動完成

⑧回到「編輯樣式表」，點選「新增字型」進入「選擇字型」視窗，選擇「標楷體」，字的大小選「12」後按下「OK」，字型設定指令自動填好見圖 5_60。

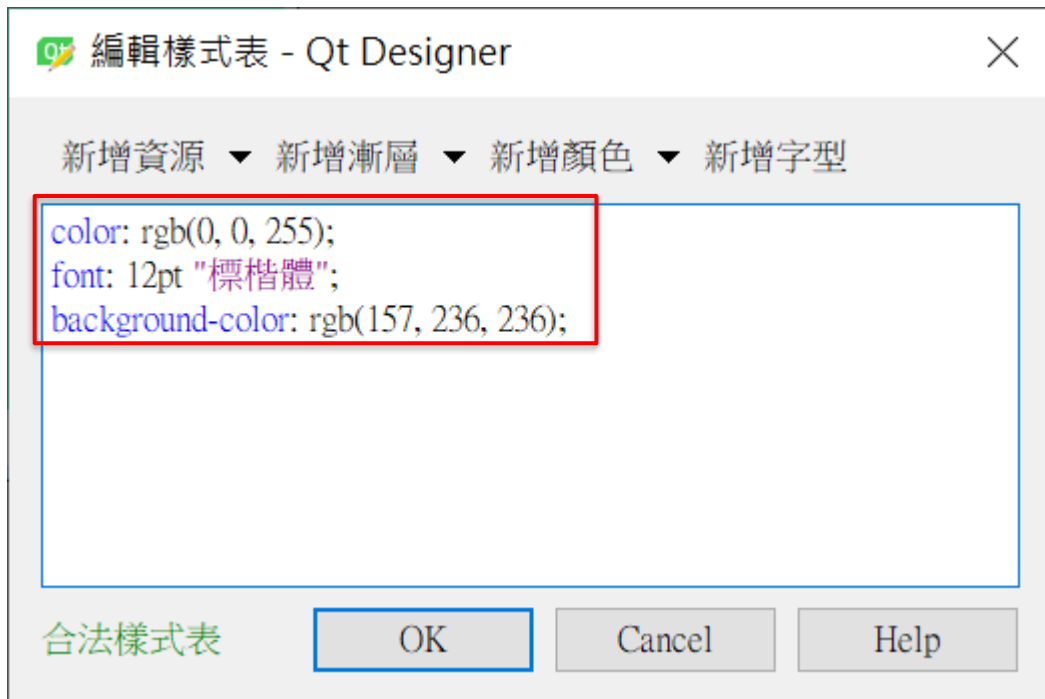


圖 5_60 字型樣式指令自動完成

⑨回到主視窗，點選主視窗的任何空白區域，點選「屬性編輯器」將「windowTitle」屬性改為「BMI 測試」就完成了介面設計階段見圖 5_61。將檔案儲存為 Ex5_5.ui 並轉成 Ex5_5.py。

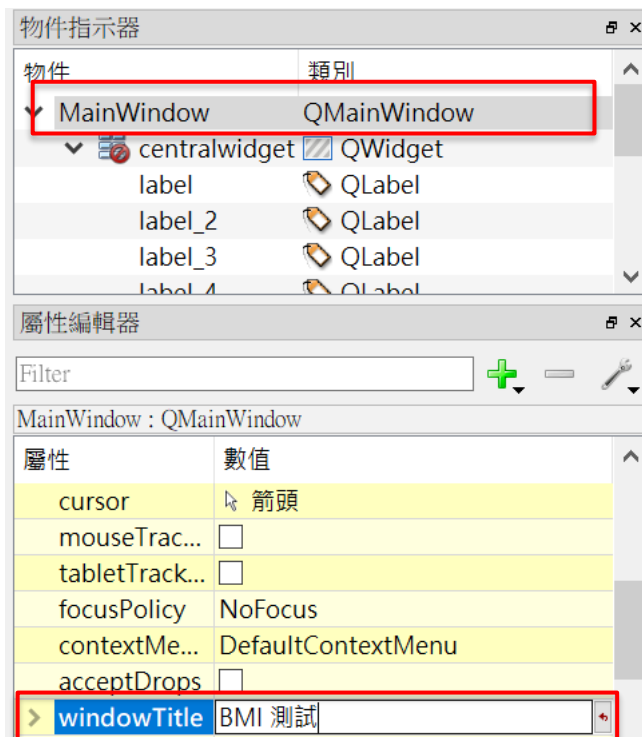


圖 5_61 主視窗標題設定為 BMI 測試

執行結果

在 spyder 執行 Ex5_5.py 結果如下圖 5_62。但還沒寫程式，因此還不能計算 BMI 值。



圖 5_62 BMI 計算的按鈕格式設計完成

Ex5_6 範例: 實務案例-標準體重之程式設計

BMI 計算的程式處理流程包含幾個步驟: 介面設定、按鈕事件、取值、計算、顯示。當使用者按下按鈕後，啟動按鈕事件需要設計一個自訂函數處理取值、計算、顯示三個功能，見圖 5_63。BMI 計算公式如下:

$$\text{BMI} = \text{體重(公斤)} / \text{身高}^2(\text{公尺}^2)$$

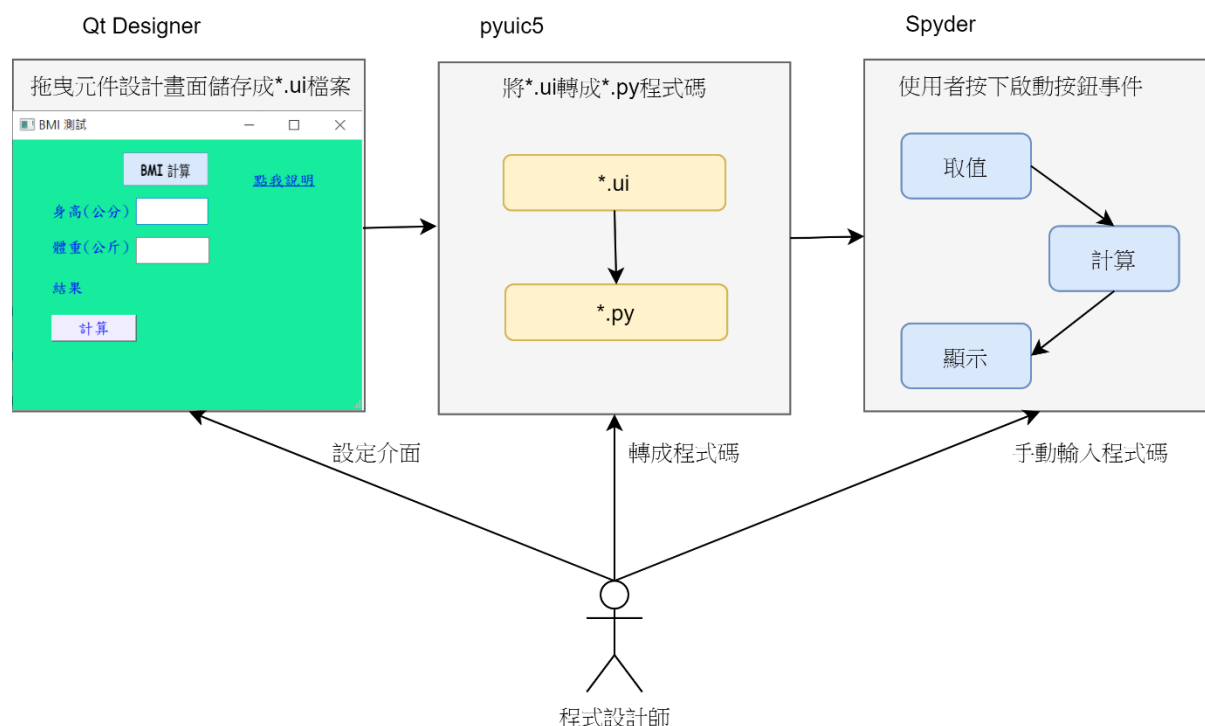


圖 5_63 BMI 程式設計處理流程

① 設定介面

介面設定是由 pyqt5 設計，由 Ex5_5.ui 另儲新檔 Ex5_6.ui，再轉成 Ex5_6.py，已完成介面設計的程式碼。本節補充說明程式碼編排在 `setupUi()` 自訂函數內容與功能。本範例用到的元件清單整理見下表 5_7，其中，`setupUi()` 自訂函數主要功能是設定主視窗以及元件的樣式，見 Ex5_6.py 程式碼第 12 到 450 列。

表 5_7 範例元件清單

元件名稱	功能	格式設定摘要
Main Window	設定主視窗格式。	palette: 背景設淺綠色。 Window Title: BMI 測試。
label	設定標題標籤元件格式。	pixmap: 載入 LabelPlot.png
label_2	設定身高標籤元件格式。	font: 大小 12；標楷體。 palette: 字型顏色設深藍色。
label_3	設定體重標籤元件格式。	
label_4	設定結果標籤元件格式。	
label_5	設定點我說明標籤元件格式。	text: 載入外部網址。 OpenExternalLinks 打勾
lineEdit	設定空白文字方塊元件格式。	font: 大小 14；Times New Roman；預設黑色。
lineEdit_2	設定空白文字方塊元件格式。	
pushButton	設定按鈕格式。	font: 大小 12；標楷體； palette: 字型顏色設深藍色。

② 設定元件顯示在外的字串

`retranslateUi` 自訂函數，以 `.setWindowTitle()` 方法設定主視窗的標題「BMI 測試」。以 `.setText()` 方法設定標籤顯示的字串「身高(公分)」、「體重(公斤)」，

「結果」、以及「點我說明」的超連結網址。同樣也以 `setText()` 方法設定按鈕顯示的字串「計算」。這些程式碼在 **Qt Designer** 時已經隨介面設定完畢，藉由 **pyuic5** 轉檔而成，程式碼已自動產生，見表 5_8 第 452 到 459 列。

③設定啟動按鈕事件

在 `retranslateUi` 自訂函數的最後一列可以設定當使用者按下按鈕時啟動 `onClick` 事件見表 5_8 第 461 列。

④按鈕內的計算功能

當使用者在介面上按下按鈕即可啟動執行 `onClick` 事件，`onClick` 事件的功能主要有三個部份：取值、計算以及顯示。

i. 取值

採用 `text()` 方法將使用者在 `lineEdit` 與 `lineEdit_2` 兩個元件輸入的文字取出分別指派給 `height`(身高公分)與 `weight`(體重公斤)兩個變數。其中，身高輸入是公分，而 BMI 公式要的是公尺，因此公分單位轉換公尺要除以 100。自文字方塊的資料預設是字串形態，在做算術運算前必須要先使用 `float()` 函數轉成數值資料，轉成整數也可以。見表 5_8 第 465 到 466 列。

ii. 計算

計算的步驟有兩部份(1)計算 BMI；(2)判斷 BMI 值落入那個範圍。

(1) 計算 BMI

將 `height` 與 `weight` 變數代入公式，體重除以身高平方，將公式轉為程式碼並把執行結果指派給 `BMI` 見表 5_8 第 467 列。

(2) 判斷 BMI

判斷 BMI 值小於 18.5，顯示"體重過輕囉，多吃點！"

BMI 介於 18.5 至 24 之間，顯示"體重剛剛好，繼續保持！"

BMI 大於 24，顯示"體重有點過重囉，少吃多運動！"

並將判斷結果指派給 `msg`，見表 5_8 第 468 到 473 列。

iii. 顯示

顯示的處理步驟(1)將判斷結合併說明的字串 (2)顯示合併後的字串。

(1) 計算好的 BMI 取小數點 2 位，再加上說明文字與判斷結果的字串 `msg`，將字串合併的結果指派給 `output_text` 見表 5_8 第 474 列。

(2) 並將判斷結果 `output_text` 以 `setText()` 方法顯示於 `label_4` 見表 5_8 第 475 列。

執行結果

在 spyder 執行 Ex5_6.py，在使用者介面輸入身高(公分)165、體重(公斤)55，按下計算按鈕，判斷結果顯示在視窗見圖 5_64。



圖 5_64 BMI 含按鈕計算程式的執行結果

表 5_8 Ex5_6 範例部份程式碼

...	...
452	def retranslateUi(self, MainWindow):
453	_translate = QtCore.QCoreApplication.translate
454	MainWindow.setWindowTitle(_translate("MainWindow", "BMI 測試"))
455	self.label_2.setText(_translate("MainWindow", "身高(公分)"))
456	self.label_3.setText(_translate("MainWindow", "體重(公斤)"))
457	self.label_4.setText(_translate("MainWindow", "結果"))
458	self.label_5.setText(_translate("MainWindow", "<html><head><body><p>點我說明</p></body></html>\""))
459	self.pushButton.setText(_translate("MainWindow", "計算"))
460	###設定當使用者按下按鈕時啟動 onClick 事件
461	self.pushButton.clicked.connect(self.onClick)
462	###onClick 自訂函數
463	def onClick(self):
464	_translate = QtCore.QCoreApplication.translate
465	height = float(self.lineEdit.text())/100
466	weight = float(self.lineEdit_2.text())
467	BMI = weight / height ** 2
468	if BMI < 18.5:
469	msg = "體重過輕囉，多吃點！"

470	elif BMI >= 18.5 and BMI < 24:
471	msg = "體重剛剛好，繼續保持！"
472	elif BMI >= 24 :
473	msg = "體重有點過重囉，少吃多運動！"
474	output_text = "BMI 為：" + str(round(BMI,2)) + "，" + msg
475	self.label_4.setText(_translate("Dialog", output_text))
...	...

1.4.5. QMessageBox 訊息對話方塊

訊息對話方塊 **QMessageBox** 提供可能經過運算後的訊息與使用者溝通，讓使用者點選定義好的標準按鈕，訊息回饋給程式判斷相對應的後續處理準則。標準含圖示的對話方塊版型見表 5_9 以及最常用的對話方塊方法見表 5_10。

表 5_9: 含圖示的對話方塊標型





圖示	名稱	說明
	question	詢問問題含圖示的視窗對話方塊。
	information	報告訊息含圖示的視窗對話方塊。
	critical	提示發生錯誤含圖示的視窗對話方塊。
	warning	警告訊息含圖示的視窗對話方塊。

表 5_10 QMessageBox 最常用的方法

名稱	說明
.setTitle()	設定對話方塊視窗的標題。
.setText()	設定對話方塊的顯示字串。
question(QWidget parent, title, text, buttons, defaultButton)	顯示問答對話方塊，參數說明如下： parent : 指定父類別名稱； title : 對話方塊標題字串； text : 對話方塊詢問文字； buttons : 顯示一個以上的標準按鈕，預設是左邊第一個 Ok 按鈕。 defaultButton : 預設選取的標準按鈕，規則是左邊第一個按鈕。
information(QWidget parent, title, text, buttons, defaultButton)	訊息對話方塊，參數同上。
critical(QWidget parent, title, text, buttons, defaultButton)	提示錯誤對話方塊，參數同上。
warning(QWidget parent, title, text, buttons, defaultButton)	提醒對話方塊，參數同上。
about(QWidget parent, title,	關於對話方塊，單一按鈕，參數同上。

text)	
-------	--

Ex5_7 範例: 訊息對話方塊

設計五個按鈕測試五種對話方塊，主視窗設計五個按鈕，分別顯示的字串為訊息對話方塊、問答對話方塊、錯誤對話方塊、提醒對話方塊及關於對話方塊。存檔為 Ex5_7.ui 後再轉檔為 Ex5_7.py。開啟 Ex5_7.py 後輸入以下程式碼。

①匯入 QMessageBox 套件

於程式開頭匯入 QMessageBox 套件見表 5_11 第 10 列。

10	from PyQt5.QtWidgets import QMessageBox
----	---

②加入各按鈕啟動的連結程式

在 retranslateUi 自訂函數最後加入各按鈕啟動的連結程式。以 information 為例，當使用者按下訊息對話方塊的按鈕(pushButton)時，啟動 information_msg 自訂函數。

見表 5_11 第 44 列，其他按鈕啟動程式設定見 5_11 第 45 到 48 列。

44	self.pushButton.clicked.connect(self.information_msg)
----	---

③設計各按鈕執行對話方塊的自訂函數

以 information 為例，問答對話方塊的自訂函數中，先呼叫 QMessageBox()(類別)後指派 msgBox 變數(物件實體)，代入.information()方法見見 5_11 第 50 到 52 列，其他按鈕自訂函數設定見 5_11 第 53 到 64 列。

50	def information_msg(self):
51	msgBox = QMessageBox()
52	QMessageBox.information(msgBox, 'Information', "訊息對話方塊")

執行結果

在 spyder 執行 Ex5_7.py，畫面見圖 5_65，當使用者按下訊息對話方塊、問答對話方塊、錯誤對話方塊、提醒對話方塊及關於對話方塊按鈕時，分別會出現不同的對話方塊見圖 5_66 到圖 5_70。



圖 5_65 訊息對話方塊執行結果

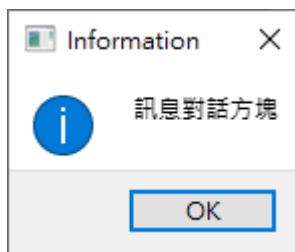


圖 5_66 訊息對話方塊

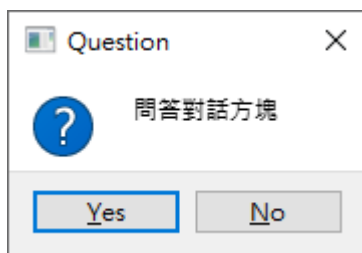


圖 5_67 問答對話方塊

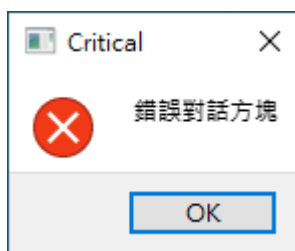


圖 5_68 錯誤對話方塊

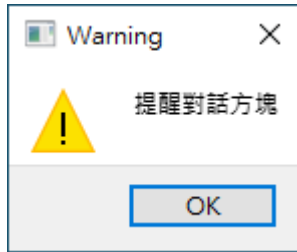


圖 5_69 提醒對話方塊

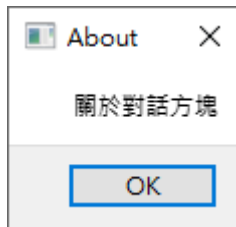


圖 5_70 關於對話方塊

表 5_11 Ex5_7 範例部份程式碼

...	...
10	from PyQt5.QtWidgets import QMessageBox
...	...
43	###加入各按鈕啟動的連結程式
44	self.pushButton.clicked.connect(self.information_msg)
45	self.pushButton_2.clicked.connect(self.question_msg)
46	self.pushButton_3.clicked.connect(self.critical_msg)
47	self.pushButton_4.clicked.connect(self.warning_msg)
48	self.pushButton_5.clicked.connect(self.about_msg)
49	###設計各按鈕執行訊息對話方塊的自訂函數
50	def information_msg(self):
51	msgBox = QMessageBox()
52	QMessageBox.information(msgBox, 'Information', "訊息對話方塊")
53	def question_msg(self):
54	msgBox = QMessageBox()
55	QMessageBox.question(msgBox, 'Question', "問答對話方塊")
56	def warning_msg(self):
57	msgBox = QMessageBox()
58	QMessageBox.warning(msgBox, 'Warning', "提醒對話方塊")
59	def critical_msg(self):
60	msgBox = QMessageBox()
61	QMessageBox.critical(msgBox, 'Critical', "錯誤對話方塊")
62	def about_msg(self):
63	msgBox = QMessageBox()
64	QMessageBox.about(msgBox, 'About', "關於對話方塊")
...	...

1.4.6. QComboBox 下拉式選單

下拉式選單(QComboBox)是結合按鈕與彈出清單的功能，提供給使用者將選項隱藏只留當前(或預設)選項，點選後會彈出可供選擇的清單，節省視窗版面顯

示的空間。組合清單的選項可以供編輯、清除。常用的方法包含取出使用者選到的項目、回傳選單的項目個數等見表 5_12。

表 5_12: QComboBox 最常用的方法

名稱	說明
.currentText()	取出被選取的項目。
.currentIndex()	取出被選取的索引。
.addItem()	從現有選單中新增項目。
.Clear()	清除選單中的所有項目。
.count()	取出選單所有項目的個數。

ComboBox 元件的操作步驟如下：

①點選左邊元件盒的「Input Widgets」下的「Combo Box」如下圖 5_71。

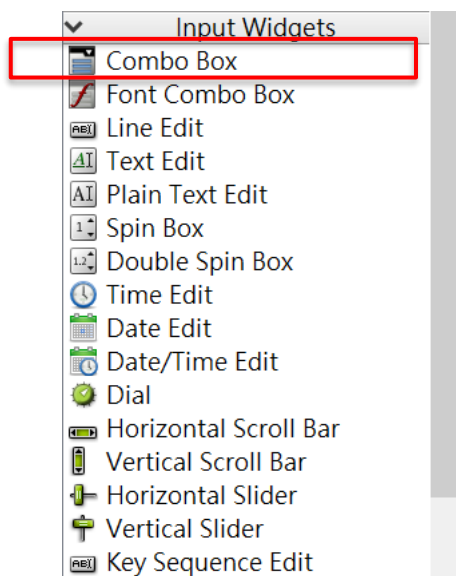


圖 5_71 comboBox 元件

②雙擊 comboBox 元件後進入「編輯下拉式選單」視窗見圖 5_72，按左下角的綠色「+」號會出現新增項目，輸入 1 按下 **Enter** 鍵完成新增。按左下角的紅色「-」號即會刪除項目，請輸入 1, 2, 3, 4, 5 共五個選項後按下「OK」完成選項輸入。

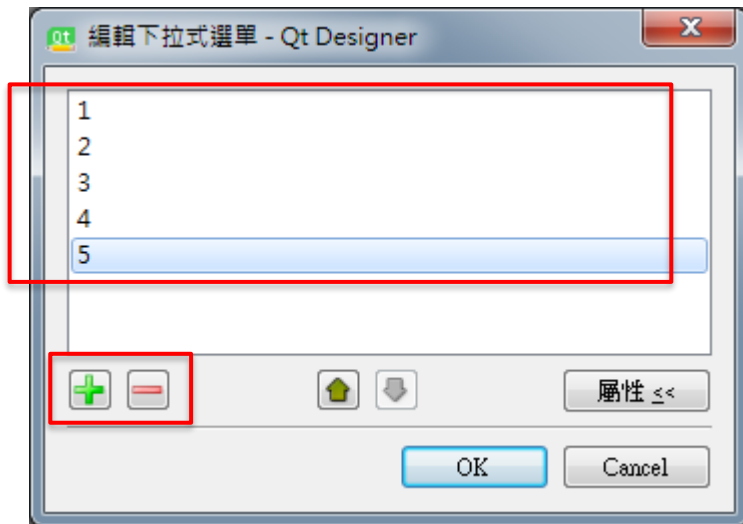


圖 5_72 編輯下拉式選單新增項目

③回到主視窗，comboBox 會出現一個只顯示第一個選項 1 的下拉式選單見下圖 5_73，儲存檔案為 comboBox.ui。

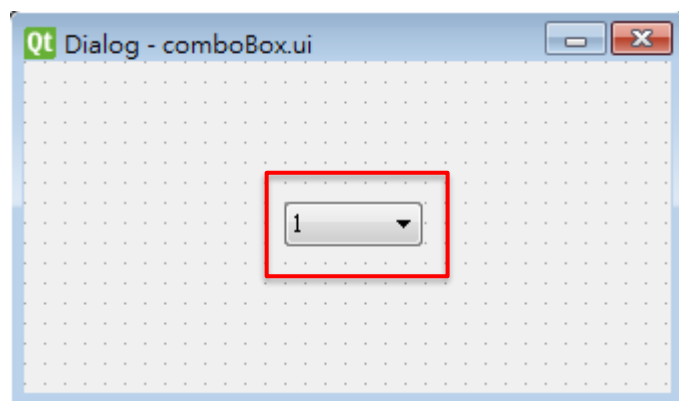


圖 5_73 完成下拉式選單

Ex5_8 銀行存款試算-整存整付

銀行存款是較常見的財務試算方式，尤其是整存整付。整存整付的意思是指投資人將本金(pv)一次存入，約定存入期數(n)與利率(i)，到期後本金再加上利息(本利和)一次提取。計算公式如下：

$$pv \times (1 + i)^n$$

假設存款金額輸入 10000；利率(%) 3；期數(年)選 5，意思是整筆存入銀行做定期存款，存入金額為 10000 元，年利率 3%，存 5 年，到期後本金加利息(本利和)是多少。本例另外加入檢查輸入的資料是否為數值資料以及新增訊息對話方塊的提醒功能，其他處理流程若同前例則省略，說明如下：

①介面設計

新開一個檔案，設計的新元件清單見下表 5_13。檔案儲存為 Ex5_8.ui 的介面設計如下圖 5_74。

表 5_13 整存整付範例設計的元件清單

元件名稱	功能	格式設定摘要
Main Window	設定主視窗格式。	palette: 背景設淺藍色。 Window Title: 財務試算。
label	設定「整存整付」標籤元件格式。	font: 大小 18；標楷體； palette: 字型顏色設深藍色。
label_2	設定「存款金額」標籤元件格式。	font: 大小 14；標楷體； palette: 字型顏色設深藍色。
label_3	設定「利率(%)」標籤元件格式。	font: 大小 14；標楷體； palette: 字型顏色設深藍色。
label_4	設定「期數」標籤元件格式。	font: 大小 14；標楷體； palette: 字型顏色設深藍色。
label_5	設定「本利和」標籤元件格式。	font: 大小 14；標楷體； palette: 字型顏色設紫色。
lineEdit	設定空白的文字方塊元件格式，提供使用者輸入存款金額。	font: 大小 12；Times New Roman；黑色 (預設)。
lineEdit_2	設定空白的文字方塊元件格式，提供使用者輸入利率(%)。	font: 大小 12；Times New Roman；黑色 (預設)。
comboBox	設定下拉式選單元件格式，提供使用者輸入期數。	進入「編輯下拉式選單」新增 1, 2, 3, 4, 5 五個項目。 palette: 字型顏色設藍色。
pushButton	設定「試算」按鈕格式。	font: 大小 14；標楷體。 palette: 字型顏色設藍色。



圖 5_74 整存整付介面設計

② 按鈕內的計算功能

將檔案名稱儲存為 **Ex5_8.ui**，再轉成 **Ex5_8.py**，其他與前例相同設定，本例即略過說明。按鈕的程式碼要撰寫在 **onClick** 事件內，說明如下。

i. 取值

本範例需要使用者輸入的資料有三項如下：

(1) 存款金額

採用 `.text()` 方法將使用者在 `lineEdit` 元件輸入的文字取出，必須要先使用 `int()` 函數轉成整數資料再指派給 `pv` 變數，見表 5_14 第 297 列。

(2) 利率(%)

採用 `.text()` 方法將使用者在 `lineEdit_2` 元件輸入的文字取出，必須要先使用 `int()` 函數轉成整數資料再指派給 `i` 變數。利率(%)文字方塊，使用者輸入的是整數，須再除以 100 才是百分比，見表 5_14 第 298 列。

(3) 期數

使用 `.currentText()` 方法取出 `comboBox` 中使用者點選的期數，指派給變數 `n`。

輸入的期數必須經 `int()` 函數轉成整數資料，見表 5_14 第 299 列。

ii. 檢查

(1) 下拉式選單

期數可以不必做數值資料檢查，因為 **comboBox** 的選單是預設好的為數值選項，因此可以不必檢查是否為非數值資料。

(2)輸入的值是否為數值資料

開放性的文字方塊讓使用者輸入的資料包含存款金額與利率(%)，有可能輸入非數值的資料例如字母等其他字元，就無法算術運算並會發生程式的錯誤。再者，取出文字方塊的資料預設是字串資料型別，可以引用第三章的字串資料型別中 **.isdigit()** 字串方法可以用來檢查輸入的字串是否為數值資料。檢查 **pv** 內容值是字串的數值資料，而且 **i** 的內容值是字串的數值資料見表 5_14 第 300 列。若兩者的檢查結果同時都是為真(True)則可以進行算術運算計算本利和；反之為否(False)，即進入 **else** 指令區見表 5_14 第 307 列，則引用前節的訊息對話方塊中的警告(Warning)訊息對話方塊用以提醒使用者「請輸入數值資料」，見表 5_14 第 308 到 309 列。

iii. 計算

$$\text{本利和} = \text{存款金額} * (1 + \text{利率}(\%))^{\text{期數}}$$

取次方可用 ****** 將本利和的金額指派給 **fv** 見表 5_14 第 304 列如下：

$$fv = pv * ((1+i) ** n)$$

iv. 顯示

本範例顯示的功能分兩個部份 (1)顯示本利和的結果，見表 5_14 第 305 到 306 列；(2)顯示輸入的資料為非數值資料時出現警告對話方塊，見表 5_14 第 308 到 309 列。

表 5_14 Ex5_8 範例部份程式碼

...	...
291	###設定當使用者按下按鈕時啟動 onClick 事件
292	self.pushButton.clicked.connect(self.onClick)
293	###onClick 自訂函數
294	def onClick(self):
295	from PyQt5.QtWidgets import QMessageBox
296	_translate = QtCore.QCoreApplication.translate
297	pv = self.lineEdit.text()
298	i = self.lineEdit_2.text()
299	n = self.comboBox.currentText()
300	if (pv.isdigit() and i.isdigit()):
301	pv = int(pv)
302	i = int(i) / 100

303	<code>n = int(n)</code>
304	<code>fv = pv * ((1+i) ** n)</code>
305	<code>output_text = "本利和 = " + str(round(fv,2)) + "元"</code>
306	<code>self.label_5.setText(_translate("Dialog", output_text))</code>
307	<code>else:</code>
308	<code>msgBox = QMessageBox()</code>
309	<code>QMessageBox.warning(msgBox, '提醒', "請輸入數值資料")</code>
...	...

執行結果

在 spyder 執行 Ex5_8.py，為了驗證使用者是否輸入正確的數值資料，本程式分兩個部份測試：

(1) 輸入正確的資料

存款金額輸入 10000；利率(%) 3；期數(年)選 5，執行結果的本利和 = 11592.74 元，見下圖 5_75。

圖 5_75 輸入正確的資料執行結果

(2) 輸入非數值的資料

假設在存款金額輸入 abc 或利率(%)輸入-3 見下圖 5_76，非數值資料後按計算，會顯示警告對話方塊，提醒要輸入數值資料，使用者按下「OK」即會回到主視窗見下圖 5_77。

圖 5_76 輸入非數值的資料執行結果

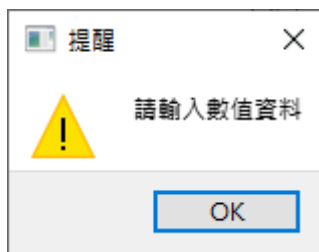


圖 5_77 顯示 Warning 對話方塊

1.4.7. QCheckBox 核取方塊

核取方塊(QCheckBox)是提供多選，屬於按鈕之一，一次可以從多個選項中選出多個，即多選多。點選選項核取方塊會變打勾即為選取；選項沒有打勾即為沒有選取。可以使用 `isChecked()` 方法檢測是否有被選到。QCheckBox 常見的方法見表 5_16。

表 5_16: QCheckBox 最常用的方法

名稱	說明
<code>.isEnabled()</code>	檢測 CheckBox 是否為啟用狀態。
<code>.setChecked()</code>	設定 CheckBox 為選取狀態(實心的)，若回傳為 True 表示是選取狀態。
<code>.isChecked()</code>	檢測 CheckBox 是否被選取狀態。被選中(實心)回傳 True；沒有選中(空心)回傳 False。
<code>.setText()</code>	設定字串到 CheckBox 元件。
<code>.text()</code>	取出 CheckBox 的字串內容。

CheckBox 元件的操作步驟如下：

①點選左邊「元件盒」的「Buttons」下的「Check Box」如下圖 5_80。

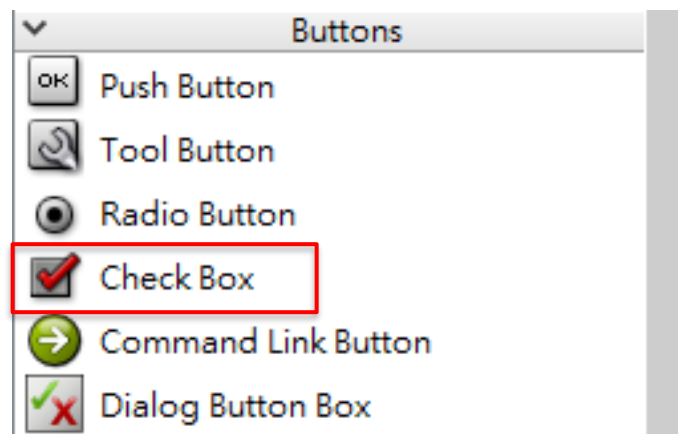


圖 5_80 核取方塊元件

②拖曳 2 個核取方塊到視窗，直接點選元件成 8 個點即可分別輸入顯示的字串「巧克力可可」與「芋頭西米露」見下圖 5_81，儲存檔案為 checkbox.ui。

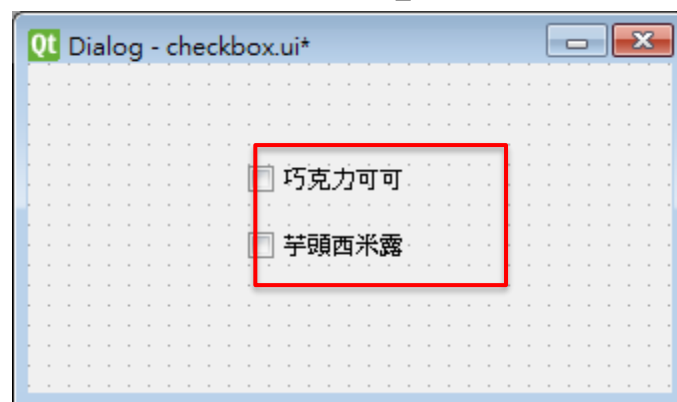


圖 5_81 拖曳按鈕到視窗

1.4.8. QRadioButton 選項按鈕

選項按鈕(QRadioButton)是提供單選的選項按鈕，一次只能從多個選項中選出一個。點選選項按鈕變成實心的即為選取；選項為空心的即為沒有選取狀態。

QRadioButton 常見的方法見表 5_15。

表 5_15: QRadioButton 最常用的方法

名稱	說明
.isEnabled()	檢測 RadioButton 是否為啟用狀態。
.setCheckable()	設定 RadioButton 為可以選取狀態，才能讓使用者用改變選項為選取或不選取狀態，若為 True 表示可以改變的狀態。
.setChecked()	設定 RadioButton 為選取狀態。
.isChecked()	檢測 RadioButton 是否被選取狀態。被選中(實心)回傳 True；沒有選中(空心) 回傳 False。

.setText()	設定字串到 RadioButton 。
.text()	取出 RadioButton 的字串內容。

Radio Button 元件的操作步驟如下：

①點選左邊「元件盒」的「Buttons」下的「Radio Button」如下圖 5_78。

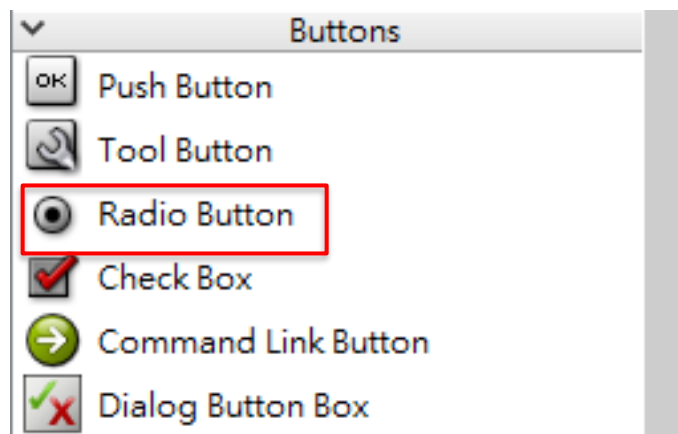


圖 5_78 選項按鈕元件

②拖曳 2 個選項按鈕到視窗，直接點選元件成 8 個點即可分別輸入顯示的字串「現金」與「Line Pay」見下圖 5_79，儲存檔案為 **radiobutton.ui**。

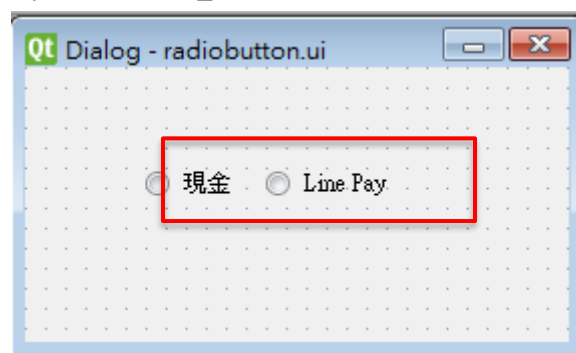


圖 5_79 拖曳按鈕到視窗

1.1.1. **QGroupBox** 群組盒

QGroupBox 群組盒是最常見的容器類別元件，用於將相關主題的元件框架在一起進行版面設計。例如，可以將付款方式包含「現金」與「Line Pay」兩個核取方塊群組在一起，提高介面主題的辨識度。**QGroupBox** 常見的方法見表 5_17。

表 5_17: **QGroupBox** 最常用的方法

名稱	說明
.checkable()	設定 GroupBox 左上角會出現一個核取方塊，方便使用者點選後可以選取或取消元件的狀態。見下圖 5_xx 與 5-xx
.checked()	設定 GroupBox 內的所有元件為可以已選取狀態或為未選取

狀態。

GroupBox 元件的操作步驟如下：

①點選左邊元件盒的「Containers」下的「Group Box」如下圖 5_82。

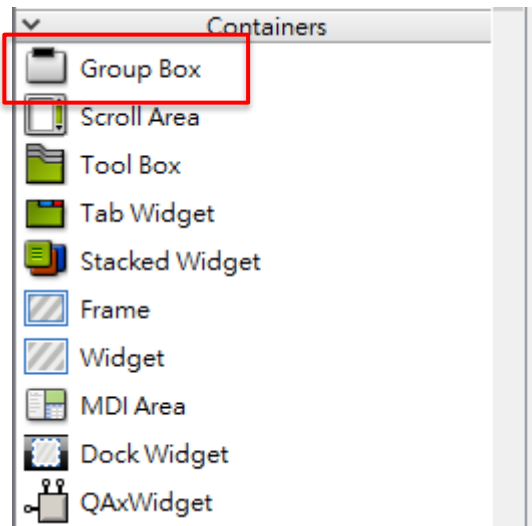


圖 5_82 GroupBox 元件

②拖曳 groupBox 元件將「現金」與「Line Pay」元件群組起來見下圖 5_83。

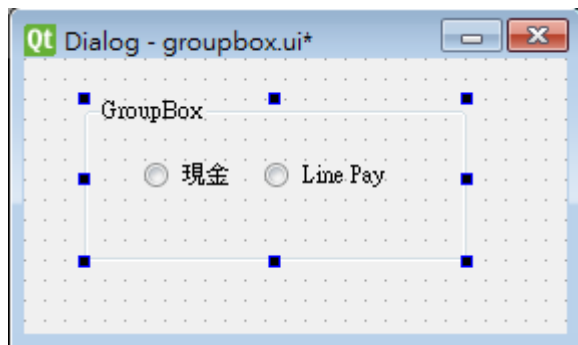


圖 5_83 拖曳群組盒將元件群組起來

③點選 GroupBox 元件成 8 個點後，再點選右側「屬性編輯器」視窗中 QGroupBox 下的「checkable」，點選為勾選狀態，會連同 checked 也一併選取見下圖 5_84。

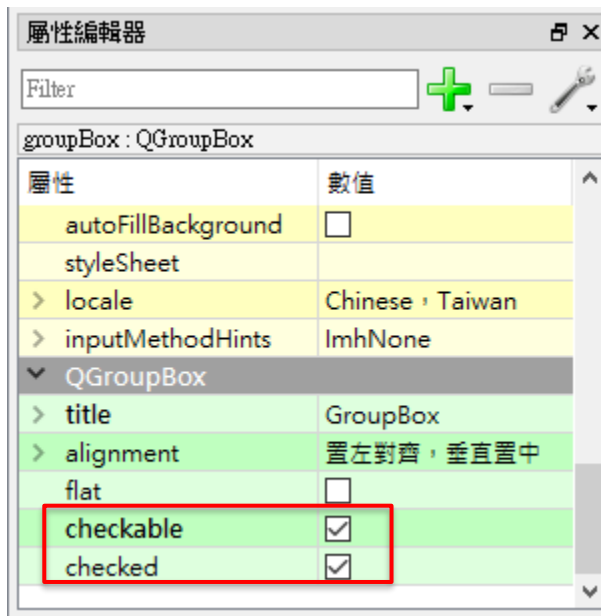


圖 5_84 設定 groupBox 的 checkable 屬性為選取狀態

④若 **chekable** 為選取狀態(點選為有作用)見下圖 5_85；沒有選取狀態(點選為無作用)見下圖 5_83，儲存檔案為 **groupbox.ui**。



圖 5_85 元件設為可以選取狀態

1.1.2. QSlider 滑動條

QSlider 是 Widgets 類的輸入元件，以滑動條的方式提供使用者點選輸入狀態值。滑動條有分垂直(Vertical Slider)與水平(Horizontal Slider)的元件樣式，使用者在滑動條移動點選其中的位置，可以將位置轉換為合法的範圍值(預先設定好：最大、最小與間隔數值)。QSlider 常見的方法見表 5_18。

表 5_18: QSlider 最常用的方法

名稱	說明
.setValue()	設定滑動條的值。
.value()	取出滑動條的值。
.setMinimum()	設定滑動條的最大值。
.setMaximum()	設定滑動條的最小值。
setSingleStep()	設定滑動條的增減量值，遞增量值或遞減量值。

Slider 元件的操作步驟如下：

①點選左邊元件盒的「Input Widgets」下的「Horizontal Slider」如下圖 5_86。

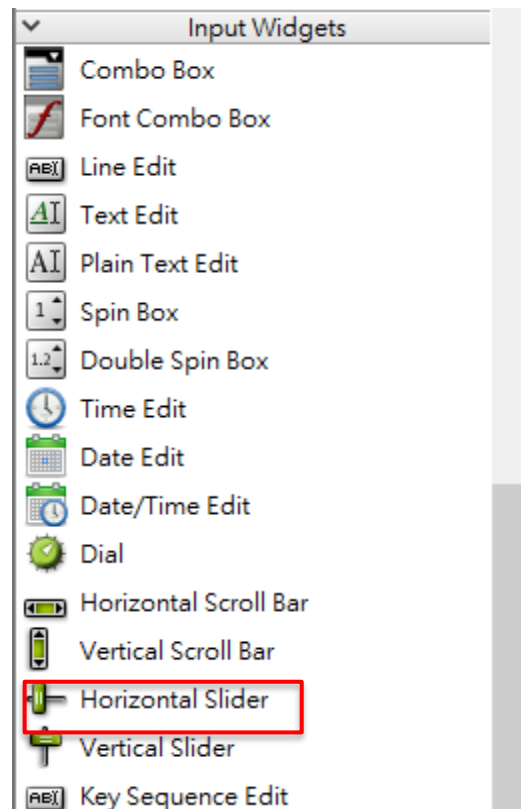


圖 5_86 Horizontal Slider 元件

②先拖曳「Horizontal Slider」元件到視窗後，點選「HorizontalSlider」元件成 8 個點，見下圖 5_87。再點選「屬性編輯器」視窗的「QabstractSlider」下，測試 minimum 設為 1；maximum 設為 5；singleStep 設為 1 見下圖 5_88。

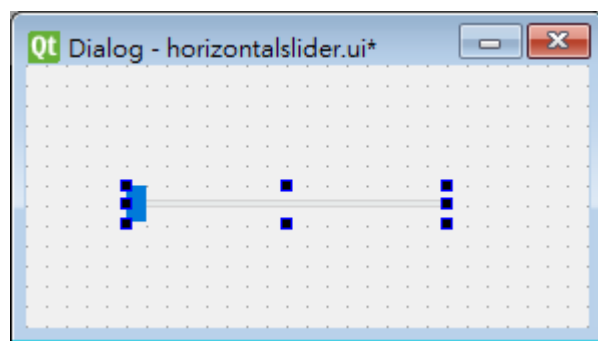


圖 5_87 點選 Horizontal Slider 元件為選取狀態



圖 5_88 設定最大最小範圍

③儲存檔案為 `horizontalslider.ui` 見下圖 5_89。

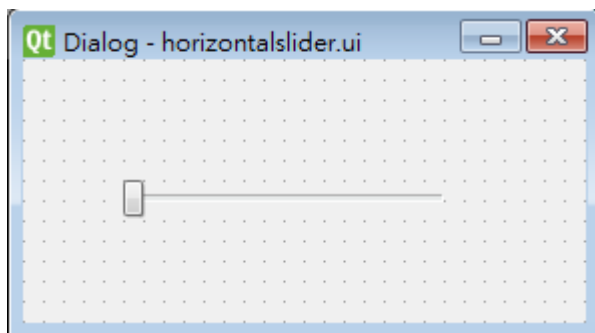


圖 5_89 儲存為 `horizontalslider.ui`

Ex5_9 「茶好喝」飲料點餐結帳試算

茶好喝飲料點餐結帳試算，功能有幾個特點：(1) 推薦熱門飲品(包含價格)，當使用者點選飲品的核取方塊為勾選時，數量的文字方塊才能變更為可以輸入狀態；(2)數量的文字方塊的 `enable` 屬性取消勾選，設定為不可輸入狀態；(3)使用者若沒有點選飲品就無法輸入數量；(4)冰量表提供客戶選擇含冰量；(5)付款方式有現金與 **Line Pay** 兩種選擇方式。本範例除有設計到基本元件(標籤、文字方塊、按鈕)外，額外用到核取方塊，選項按鈕、滑動條、群組盒等。

● 推薦飲品

品名	價格
黃金梅子綠茶	\$45
夏威夷綠茶	\$45
龍眼花蜜茶	\$50
品鑽咖啡	\$40
巧克力可可	\$40
芋頭西米露	\$50

● 冰量表

正常 微冰 少冰 溫飲 熱飲
1 2 3 4 5

客戶可以購買多種飲品，以及填入數量，點選完冰量表與付款方式後，再點選結帳按鈕即可算出消費金額。介面設計與結帳計算的程式設計流程說明如下：

①介面設計

新開一個檔案，設計的元件清單見下表 5_19。

表 5_19 整存整付範例設計的元件清單

元件名稱	功能	格式設定摘要
Main Window	設定主視窗。	palette: 背景設淺紫色。 Window Title: 茶好喝 飲料菜單。
Label	設定「熱門飲品推薦」標籤元件。	font: 大小 20；標楷體；黑色 (預設)。
label_2	設定「單價」標籤元件。	font: 大小 16；標楷體；黑色(預設)。 enable: 取消打勾，設定為不可選取狀態。
label_3	設定「數量」標籤元件。	
checkBox	設定「黃金梅子綠茶」核取方塊元件。	
checkBox_2	設定「夏威夷綠茶」核取方塊元件。	
checkBox_3	設定「龍眼花蜜茶」核取方塊元件。	
checkBox_4	設定「品鑽咖啡」核取方塊元件。	
checkBox_5	設定「巧克力可可」核取方塊元件。	font: 大小 16；標楷體；黑色 (預設)。
checkBox_6	設定「芋頭西米露」核取方塊元件。	
label_4	設定「\$45」標籤元件(黃金梅子綠茶的單價)。	
lable_5	設定「\$45」標籤元件(夏威夷綠茶的單價)。	
lable_6	設定「\$50」標籤元件格式(龍眼花蜜茶的單價)。	
lable_7	設定「\$40」標籤元件(品鑽咖啡的單價)。	
lable_8	設定「\$40」標籤元件(巧克力可可的單價)。	font: 大小 16；Times New Roman；黑色 (預設)。 enabled 取消勾選，表示設為 False。
lable_9	設定「\$55」標籤元件(芋頭西米露的單價)。	
lineEdit	設定文字方塊元件(黃金梅子綠茶的數量)。	
lineEdit_2	設定文字方塊元件(夏威夷綠茶的數量)。	
lineEdit_3	設定文字方塊元件格式(龍眼花蜜茶的數量)。	
lineEdit_4	設定文字方塊元件(品鑽咖啡的數量)。	
lineEdit_5	設定文字方塊元件(巧克力可可的數量)。	補充說明見② groupBox 的框線設定。
lineEdit_6	設定文字方塊元件(芋頭西米露的數量)。	
groupBox	設定「冰量表」群組盒元件格式。	
lable_10	設定「正常」標籤元件(第 1 級的冰量表)。	
lable_11	設定「微冰」標籤元件(第 2 級的冰量表)。	
lable_12	設定「去冰」標籤元件(第 3 級的冰量表)。	
lable_13	設定「溫飲」標籤元件(第 4 級的冰量表)。	font: 大小 9；標楷體；黑色 (預設)。
lable_14	設定「熱飲」標籤元件(第 5 級的冰量表)。	
horizontalSlider	設定滑動條元件。	minimum: 1 minimax: 5 singleStep: 1
groupBox_2	設定「付款方式」群組盒元件。	補充說明見② groupBox 的框線設定。
radioButton	設定「現金」選項按鈕元件。	font: 大小 9；標楷體；黑

radioButton_2	設定「Line Pay」選項按鈕元件。	色 (預設)。
lable_15	設定「結帳金額」標籤元件格式，顯示點餐後的合計金額。	font: 大小 14；標楷體；黑色 (預設)。
pushButton	設定「計算」按鈕，計算點餐後的合計金額並顯示。	font: 大小 16；標楷體；黑色 (預設)。

lineEdit 的 Qt Designer 元件設計時 enable 須事先設定為沒有勾選狀態見下圖 5_90，表示預設為不可輸入狀態。

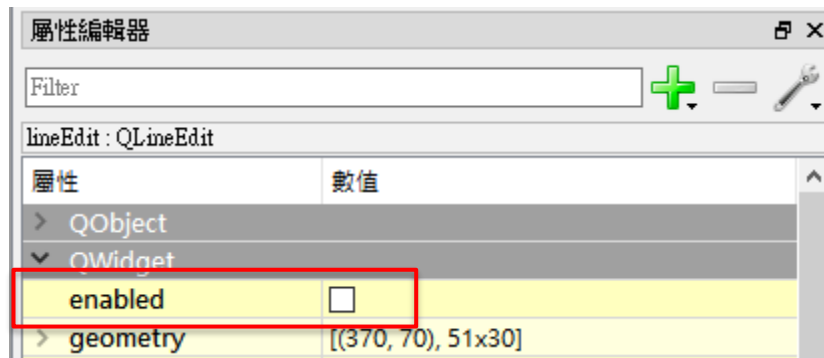


圖 5_90 文字方塊的 enable 屬性取消勾選

設計介面如下圖 5_91，將檔案名稱儲存為 Ex5_9.ui。



圖 5_91 「茶好喝」使用者介面

本範例引用的元件包含 label, lineEdit, radioButton, checkBox, groupBox, horizontalSlider, pushButton 等共約 34 個。多數的操作方式同前面的範例，本例說明省略，只針對特別的使用方法 groupBox 框線顏色與按鈕內的程式設計做說明。

◎ groupBox 的框線設定

groupBox 的框線顏色設定同其他元件的操作，即點選 groupBox 成 8 個點後，再點選右側「屬性編輯器」視窗中「QWidget」下的「styleSheet」右邊的「...」，點選「新增顏色」下拉式選單後有發現「border-color」屬性(其功能是設定 groupBox 的所有元件的邊框顏色)，但沒有 border 這個屬性(其功能是設定單純 groupBox 的外框)，就無法單獨設定 groupBox 的外框格式。解決的辦法是 PyQt5 已將元件的格式設定結合 CSS 樣式設定。因此，當視窗找不到合適的樣

式可以設定時，可以測試使用 CSS 語法的其他屬性。若要針對 groupBox 元件的框線設定 CSS 可以新增以下指令後按下「OK」，樣式指令自動填好見圖 5_92。

```
#groupBox {  
border: 1px solid;  
border-color: rgb(255, 170, 255);  
}
```

其中，groupBox 當作 CSS 的 selector (選擇器)，前面加「#」號；border 屬性是設定外框的樣式，線的寬度 1px，實線(solid)；border-color 邊框顏色設 rgb 函數或藉由調色盤點選顏色均可。groupBox_2 設定方式同 groupBox。

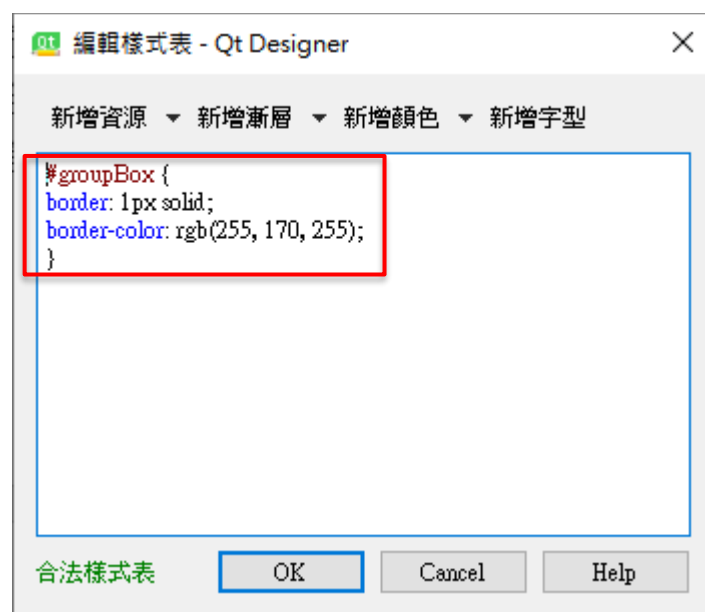


圖 5_92 groupBox 外框樣式編輯

※其他 CSS 屬性的設定可以參考以下網站。

<https://www.1keydata.com/css-tutorial/tw/border.php>

③按鈕內的計算功能

將 Ex5_9.ui 轉成 Ex5_9.py，其他與前例相同設定，本例即略過說明。僅就按鈕啟動 onClick()事件內的程式碼做說明。

i. 判斷

本範例的設計是當程式執行時，使用者勾選飲料名稱的核取方塊時，lineEdit 元件(數量)才會被設定為可輸入狀態。因此，checkBox 元件需要設計一個啟動 chk_onClick 事件的連結程式見第 405 列如下，其他元件啟動連結程式見表 5_20 第 406 到 413 列。

405	self.checkBox.clicked.connect(self.chk_onClick)
-----	---

相對應要設計一個 chk_onClick 自訂函數，並判斷當 checkBox 被選取時，即 .isChecked()方法傳回 True 見 416 列，就將 lineEdit 的 enable 屬性設為 True 見 417 列，此時數量的文字方塊改為可以輸入模式；否則當 .isChecked()方法傳回 False 見 418 列，表示 checkBox 沒有被選取時，lineEdit 的 enable 屬性設為 False 見第 419 列，就不提供輸入數量。checkBox_2 元件到 checkBox_6 的處理類似見表 5_20 第 420 到 449 列。

414	###chk_onClick 自訂函數
415	def chk_onClick(self):
416	if self.checkBox.isChecked():
417	self.lineEdit.setEnabled(True)
418	else:
419	self.lineEdit.setEnabled(False)

另外，付款方式內有現金(radioButton)與 Line Pay(radioButton_2)兩個選項按鈕。需要設計一個啟動 rdb_onClick 與 rdb2_onClick 事件的連結程式見第 411 到 412 列。

411	self.radioButton.clicked.connect(self.rdb_onClick)
412	self.radioButton_2.clicked.connect(self.rdb2_onClick)

當 radioButton 按點擊時，就會啟動 rdb_Click 事件，並藉由 .setChecked()方法將 radioButton 設定為選取狀態(實心)第 452 列，同時將 radioButton_2 設定為取消選取狀態(空心)見第 453 列。rdb2_onClick 事件的處理相同見表 5_20 第 455 到 457 列。

450	###rdb_onClick 自訂函數
451	def rdb_onClick(self):
452	self.radioButton.setChecked(True)
453	self.radioButton_2.setChecked(False)

ii. 取值

本範例需要使用者勾選飲品名稱、輸入飲品的數量、點選的冰量表與付款方式，才能顯示結果與計算結帳金額。

(1) 取出冰量表

最出冰量表的值，需要幾個步驟:

①設定冰量表的對應值

ice 變數為字典資料型別，負責對應冰量表的值。1：「正常」,2：「微冰」,3：「去冰」,4：「溫飲」,5：「熱飲」。其中，數字 1-5 為鍵，中文字為量表為值，配對好放入字典中。前後使用大括弧 { }；鍵與值成對以逗號隔開；值的部份要用字串前後以雙引號括弧起來，見第 462 列。

462	ice = {1: "正常", 2: "微冰", 3: "去冰", 4: "溫飲", 5: "熱飲"}
-----	---

②取出冰量表的值

.value()方法取出使用者在 horizontalSlider 元件點選的值(1 到 5 之間)並指派 temp 變數，見第 464 列。

464	temp= self.horizontalSlider.value()
-----	-------------------------------------

③取出字典資料

字典資料型別的用法是輸入鍵可以取出值，因此 ice[temp] (要用中括弧一對)輸入 temp 可以取出冰量表 ice 的對應值。以 str()將取出冰量表的值轉成字串，再與字串"冰量表 ="相加，見第 465 列。

465	icetemp = "冰量表 =" + str(ice[temp]) #取出 ice 字典鍵為 temp 的內容值
-----	---

(2) 取出付出方式

取出使用者點選的付款方式，須先由 .isChecked()方法判斷 radioButton(現金)選項按鈕是否被選取狀態，若為 True 則表示使用者選擇「現金」為付款方式見第 467 列，則將"付款方式 = 現金"字串指派給 pay 變數見第 468 列；否則見第 469 列，將"付款方式 = Line Pay"指派給 pay 變數見第 470 列。

467	if self.radioButton.isChecked():
468	pay = "付款方式 = 現金"
469	else:
470	pay = "付款方式 = Line Pay"

iii. 計算結帳金額

.isChecked()方法檢查 checkBox 是否有被選取，為 True 表示被選取見第 474 列。則以 .text()方法取出使用者在 lineEdit 輸入的飲品數量轉成整數後指派給 num 變數見第 475 列。將數量乘上單價即為單項飲品的金額累加並指派給 total 變數見第 476 列。

474	if self.checkBox.isChecked():
475	num = int(self.lineEdit.text())
476	total += 45 * num

其他的 checkBox_2 到 checkBox_6 的處理方式相同，將該項的金額累計給 total 見表 5_20 第 478 到 496 列。

iv. 顯示

完成所有 checkBox 的檢查計算累計的結帳金額(total)後轉成字串，使用 round 函數四捨五入取小點以下 2 位後再轉成字串。加上付款方式字串 pay 與冰量表字串 icetemp，三段字串相加後指派給 output_text 見第 497 列。

497	output_text = pay + ", " + icetemp + ", 共" + str(round(total,2)) + "元"
-----	--

以.setText()方法將 output_text 設定給 label_15 顯示結帳金額見第 498 列。

498	self.label_15.setText(_translate("Dialog", output_text))
-----	--

表 5_20 Ex5_8.py 範例部份程式碼

...	...
404	###設定當使用者按下按鈕時啟動 onClick 事件
405	self.checkBox.clicked.connect(self.chk_onClick)
406	self.checkBox_2.clicked.connect(self.chk2_onClick)
407	self.checkBox_3.clicked.connect(self.chk3_onClick)
408	self.checkBox_4.clicked.connect(self.chk4_onClick)
409	self.checkBox_5.clicked.connect(self.chk5_onClick)
410	self.checkBox_6.clicked.connect(self.chk6_onClick)
411	self.radioButton.clicked.connect(self.rdb_onClick)
412	self.radioButton_2.clicked.connect(self.rdb2_onClick)
413	self.pushButton.clicked.connect(self.bnt_onClick)
414	###chk_onClick 自訂函數
415	def chk_onClick(self):
416	if self.checkBox.isChecked():
417	self.lineEdit.setEnabled(True)
418	else:
419	self.lineEdit.setEnabled(False)
420	###chk2_onClick 自訂函數
421	def chk2_onClick(self):
422	if self.checkBox_2.isChecked():

423	self.lineEdit_2.setEnabled(True)
424	else:
425	self.lineEdit_2.setEnabled(False)
426	###chk3_onClick 自訂函數
427	def chk3_onClick(self):
428	if self.checkBox_3.isChecked():
429	self.lineEdit_3.setEnabled(True)
430	else:
431	self.lineEdit_3.setEnabled(False)
432	###chk4_onClick 自訂函數
433	def chk4_onClick(self):
434	if self.checkBox_4.isChecked():
435	self.lineEdit_4.setEnabled(True)
436	else:
437	self.lineEdit_4.setEnabled(False)
438	###chk5_onClick 自訂函數
439	def chk5_onClick(self):
440	if self.checkBox_5.isChecked():
441	self.lineEdit_5.setEnabled(True)
442	else:
443	self.lineEdit_5.setEnabled(False)
444	###chk6_onClick 自訂函數
445	def chk6_onClick(self):
446	if self.checkBox_6.isChecked():
447	self.lineEdit_6.setEnabled(True)
448	else:
449	self.lineEdit_6.setEnabled(False)
450	###rdb_onClick 自訂函數
451	def rdb_onClick(self):
452	self.radioButton.setChecked(True)
453	self.radioButton_2.setChecked(False)
454	###rdb2_onClick 自訂函數
455	def rdb2_onClick(self):
456	self.radioButton_2.setChecked(True)
457	self.radioButton.setChecked(False)
458	###bnt_onClick 自訂函數
459	def bnt_onClick(self):
460	_translate = QtCore.QCoreApplication.translate
461	#設定冰量表的為 ice 的字典資料型別，對應值，1 對應為正常, 2 對應為微冰...
462	ice = {1: "正常", 2: "微冰", 3: "去冰", 4:"溫飲", 5:"熱飲"}
463	#取出 horizontalSlider 使用者點選的值(1-5)指派給 temp 變數

464	temp= self.horizontalSlider.value()
465	icetemp = "冰量表 = " + str(ice[temp]) #取出 ice 字典鍵為 temp 的內容值
466	#判斷 radioButton 若為選取狀態則指派 pay 的字串
467	if self.radioButton.isChecked():
468	pay = "付款方式 = 現金"
469	else:
470	pay = "付款方式 = Line Pay"
471	#檢查 checkBox 若為選取狀態就取出對應右邊文字方塊輸入值並轉成整數後乘上單價後累加到 total
472	total = 0
473	num = 0
474	if self.checkBox.isChecked():
475	num = int(self.lineEdit.text())
476	total += 45 * num
477	
478	if self.checkBox_2.isChecked():
479	num = int(self.lineEdit_2.text())
480	total += 45 * num
481	
482	if self.checkBox_3.isChecked():
483	num = int(self.lineEdit_3.text())
484	total += 50 * num
485	
486	if self.checkBox_4.isChecked():
487	num = int(self.lineEdit_4.text())
488	total += 40 * num
489	
490	if self.checkBox_5.isChecked():
491	num = int(self.lineEdit_5.text())
492	total += 40 * num
493	
494	if self.checkBox_6.isChecked():
495	num = int(self.lineEdit_6.text())
496	total += 55 * num
497	output_text = pay + ", " + icetemp + ", 共" + str(round(total,2)) + "元"
498	self.label_15.setText(_translate("Dialog", output_text))
...	...

執行結果

點選黃金梅子綠茶、龍眼花蜜茶、巧克力可可後，數量的文字方塊立刻轉為可以輸入狀態，分別輸入 4, 3, 2；選量冰量表為去冰；與付款方式為現金，再按結帳按鈕，即會顯示: 付款方式 = 現金, 冰量表 = 去冰, 共 410 元見下圖 5_93。



茶好喝 飲料菜單

熱門飲品推薦

	單價	數量
<input checked="" type="checkbox"/> 黃金梅子綠茶	\$45	4
<input type="checkbox"/> 夏威夷綠茶	\$45	
<input checked="" type="checkbox"/> 龍眼花蜜茶	\$50	3
<input type="checkbox"/> 品鑽咖啡	\$40	
<input checked="" type="checkbox"/> 巧克力可可	\$40	2
<input type="checkbox"/> 芋頭西米露	\$55	

冰量表

正常 微冰 去冰 溫飲 熱飲

付款方式

☒ 現金 ☐ Line Pay

付款方式 = 現金, 冰量表 = 去冰, 共410元

結帳

圖 5_93 茶好喝結帳的執行結果

附錄 A

Spyder 操作介面說明

前節已針對 Spyder 執行程式的功能做一個概述，本節將針對 Spyder 的環境與其他功能做一個詳細的介紹。Spyder 為 Anaconda 內建 Python 專用的編譯器，其介面簡單、適合數據分析使用。

步驟 1. 搜尋 Spyder

在 windows 的搜尋功能中，搜尋 Spyder



圖 4-1. 搜尋 Spyder

步驟 2. 開啟 Spyder 畫面

Spyder 開啟後，預設畫面如圖 4-2.

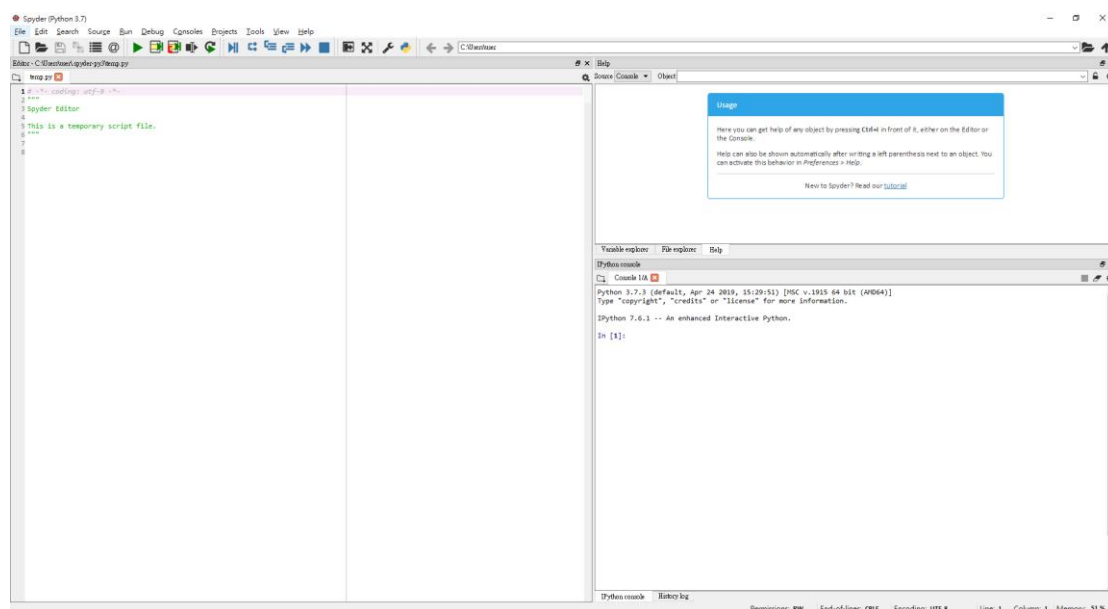













圖 4-2 預設畫面

步驟 3. 常用功能列介紹



Spyder 之功能有很多如圖 4-3.，以下介紹一些較常用之功能，除錯部分的用語，到後續章節才會說明，前面讀者會使用前五項功能即可



圖 4-3.功能列

- ：開啟新檔案
- ：瀏覽之前的存檔
- ：存檔，將當前檔案的更動儲存
- ：全部存檔，將所有檔案的更動儲存
- ：執行，執行當前檔案的程式碼，預設快捷鍵為 **F5**
- ：除錯執行，將程式碼進入除錯程序
- ：逐行執行，除錯時，每一行都走訪
- ：執行到下一個方法(method)或函數(function)結束
- ：執行到下一個回傳(return)
- ：執行到下一個斷點(breakpoint)
- ：偏好設定(Preferences)

步驟 4. 開啟新檔案

點擊  開啟新檔案，並且點擊  儲存檔案，會出現以下畫面，本範例將檔案命名為 **Hello World**，並且儲存在桌面，如圖 4-4。

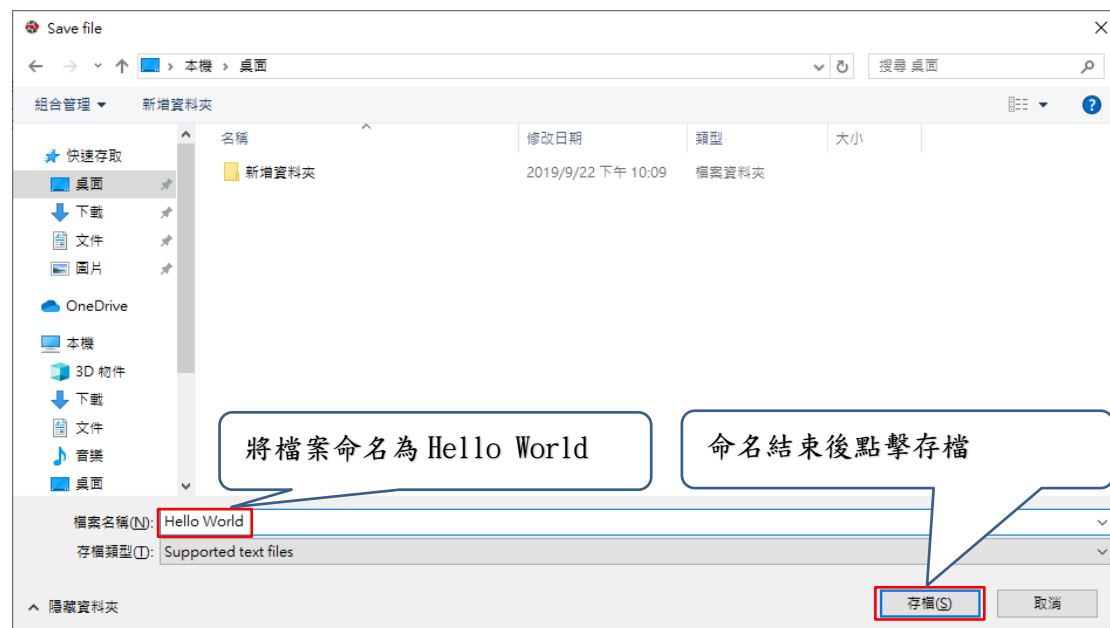


圖 4-4. 開啟新檔案

步驟 5. 檔案內容介紹

完成開啟新檔案後，Spyder 會預設一些資訊，如圖 4-5。程式第 1 行為此表示此檔案為 **utf-8** 編碼的註解，第 2 行到第 6 行為此檔案作者的註解，

註解是什麼？

如圖 4-5，在 **python** 中，“**#**” 號為單行註解(第 1 行)，以成對的三個單引號或是成對的三個雙引號圍起來的程式碼為大量註解(第 2 到 6 行)，註解程式不會執行，但是在編譯程式中，註解非常重要，無論是自己練習，或是在大型合作的專案，都可以幫助自己或隊友快速了解上一次撰寫的程式，降低溝通成本，否則等於做兩遍工，導致事倍功半，所以養成完整註解的習慣，有助於程式的學習喔！

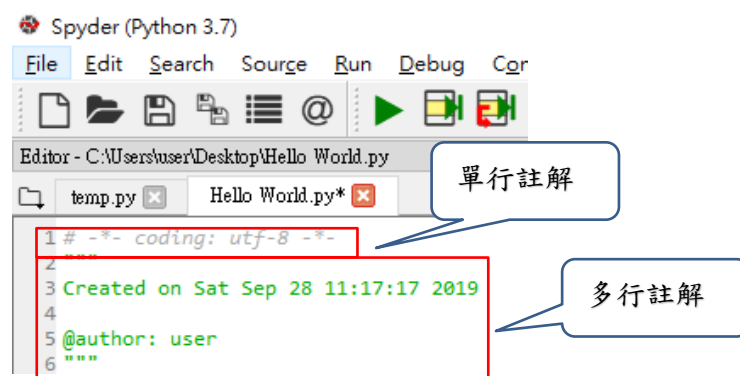


圖 4-5. 開啟檔案後的預設資訊

Spyder 註解小技巧：

步驟 A. 將要註解的程式反白起來，如圖 4-A.

```
9 print(123)
10 print(456)
11 print(789)
```

圖 4-A 選取要註解的部分

步驟 B. 按下 **Ctrl + 4**，即可完成多行註解如圖 4-B.

```
9 #=====
10 # print(123)
11 # print(456)
12 #=====
13 print(789)
```

圖 4-B. 完成註解

步驟 C. 若要取消註解，上一次註解的內容反白起來，如圖 4-B.，將輸入游標放置在註解的內容中也就是第 10 行或第 11 行，按下 **Ctrl + 5**，即可完成取消註解，如圖 4-C.

```
9 print(123)
10 print(456)
11 print(789)
```

圖 4-C. 取消註解

步驟 6. 編譯程式

開啟檔案後便可開始編譯程式，在第 7 行輸入 `print("Hello World")`，並按下 F5 執行程式

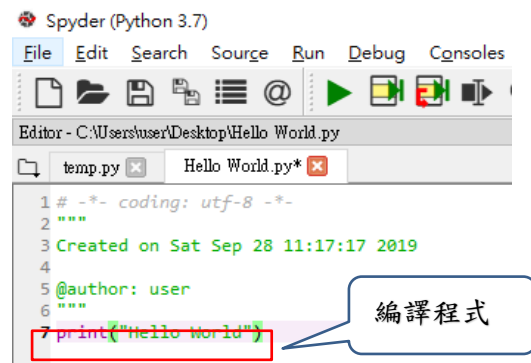


圖 4-6. 編譯程式

步驟 7. 執行設定

第一次執行會跳出設定視窗，本書建議直接使用預設的即可，點擊 run，如圖 4-7.

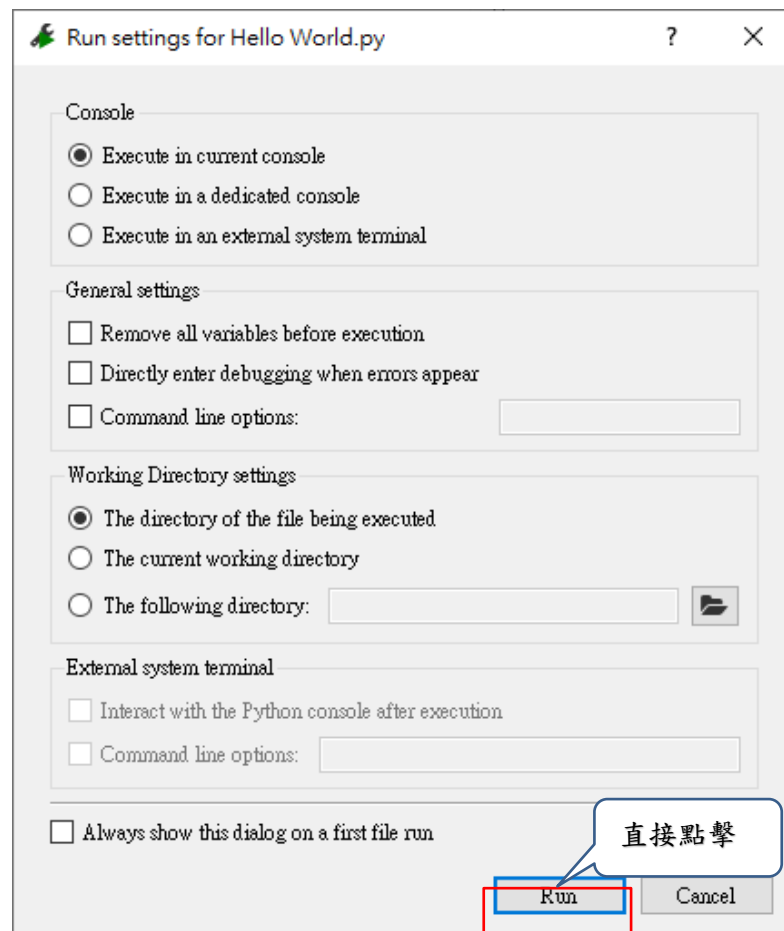
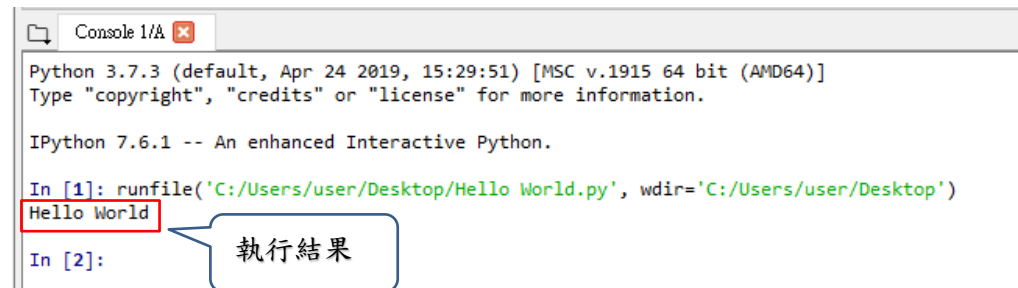


圖 4-7. 執行設定

步驟 8. 印出 Hello World

執行結束後，即可在畫面右下角的視窗看到結果



The screenshot shows the IPython console window in Spyder. The title bar is 'Console 1/A'. The text in the console is as follows:

```
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.6.1 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/user/Desktop/Hello World.py', wdir='C:/Users/user/Desktop')
Hello World
In [2]:
```

A red box highlights the output 'Hello World'. A blue callout bubble with the text '執行結果' (Execution Result) points to the output.

圖 4-8. 執行結果

附錄 4-A : Spyder 顏色設定

步驟 1. 進入設定

要將 Spyder 改成黑色背景，點擊  進入偏好設定，並點擊 Syntax coloring，如圖 A-1。

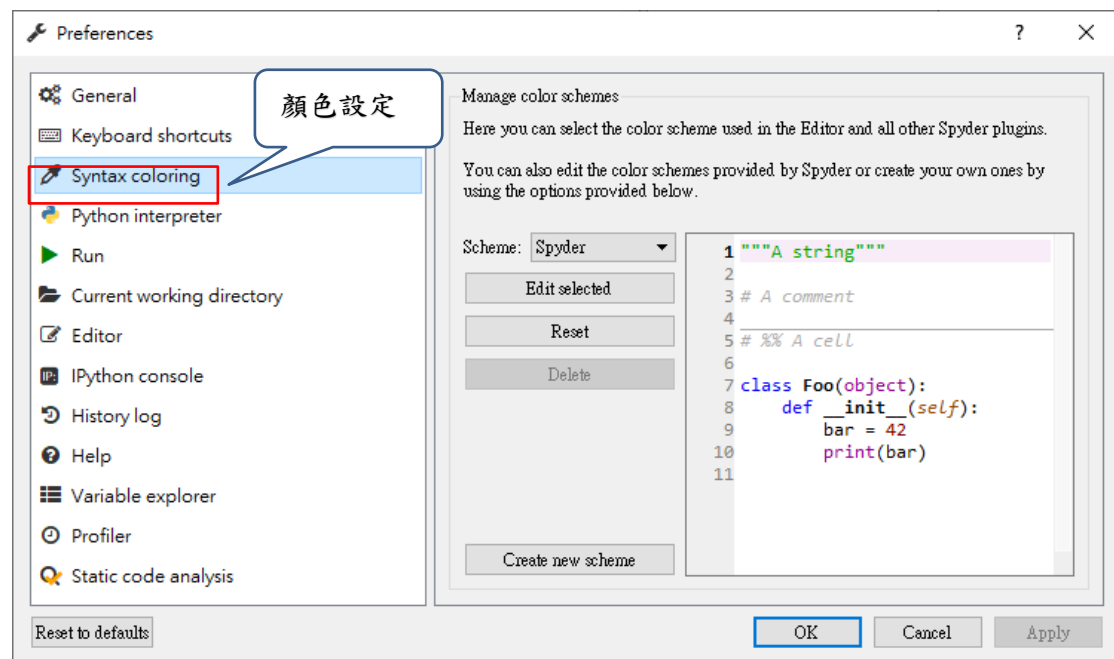


圖 A-1. 進入顏色設定

步驟 2. 選擇顏色

選擇 **Spyder Dark**，將背景變成黑色色調，其餘色調也可依使用者喜好挑選，如圖 A-2。

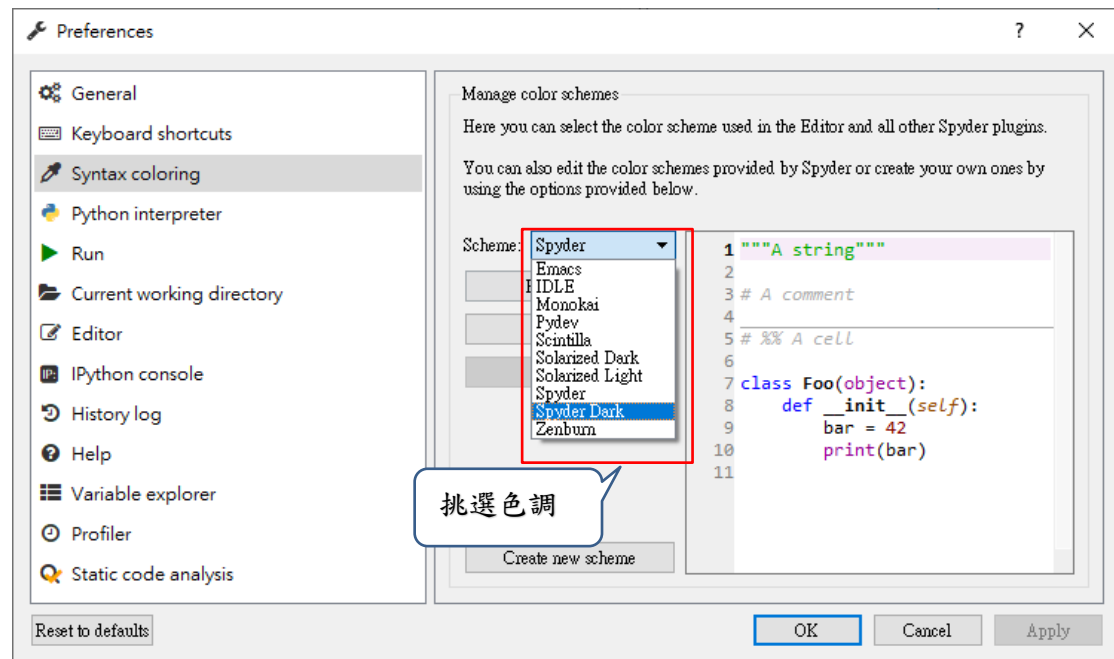


圖 A-2. 調整色調

步驟 3. 細節設定

色調選擇完成後，可以點擊 **Edit selected** 選擇關鍵字顏色，以註解為例，將其從灰色調整為淡藍色，如圖 A-3.，除了變更顏色以外，也可以將其調整為粗體(B)以及斜體(I)

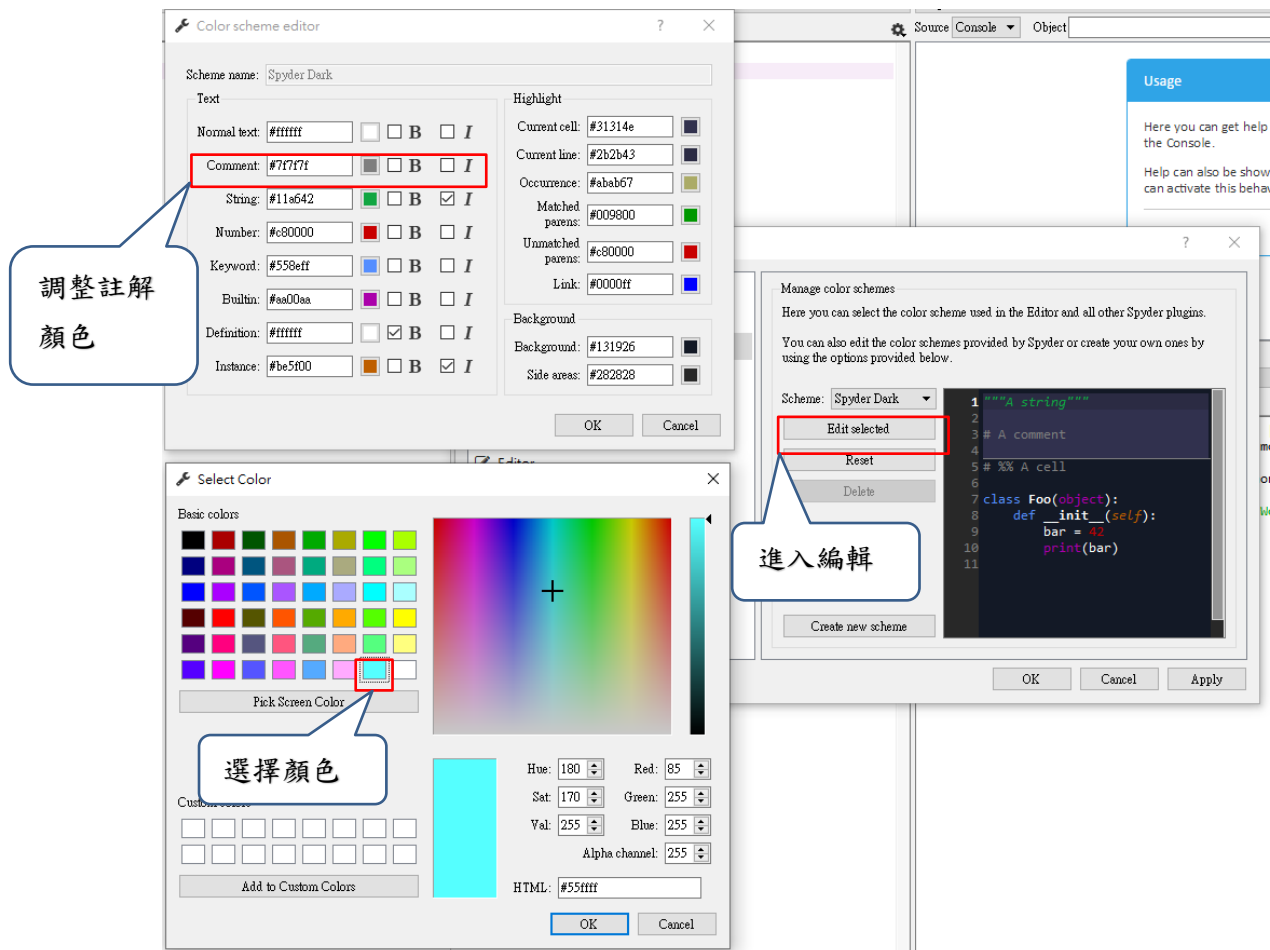


圖 A-3. 細節設定

步驟 4. 完成畫面
完成後如圖 A-4.

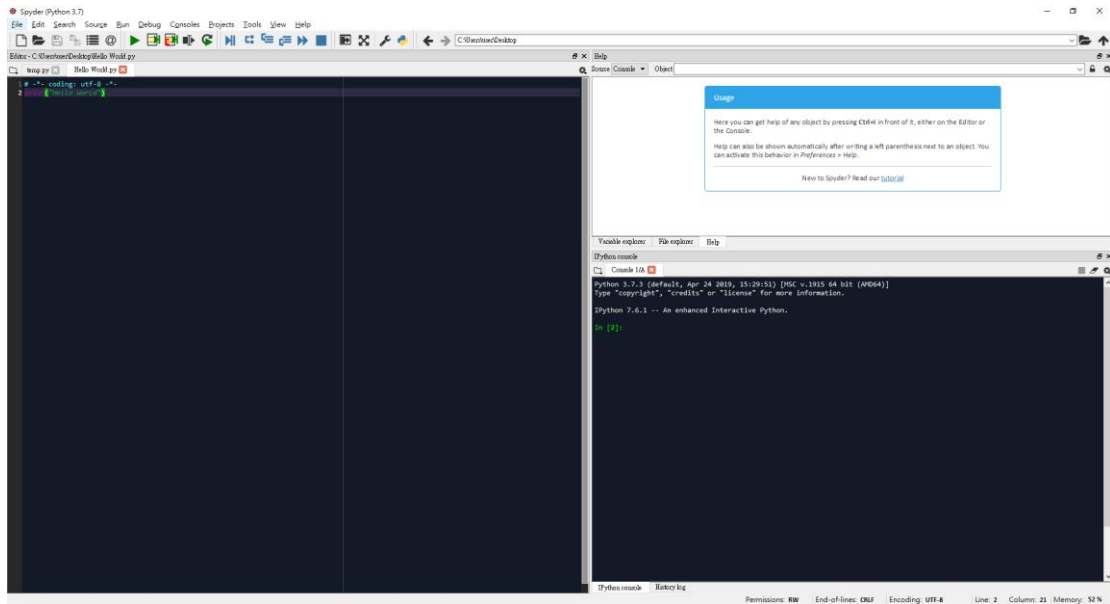


圖 A-4. 更改顏色後的畫面

附錄 B. 設定縮排樣式

步驟 1. 進入設定

點擊  進入偏好設定，並點擊 **Editor**，如圖 B-1.，若要看到定位點(tab)以及空格(space)符號，將 **Show blank spaces** 勾選起來，有些檔案是使用 tab 進行縮排，而有些則是使用四個空格(space)進行縮排，兩者不能混用，否則將會出錯，建議使用者將此功能打開，以防此問題以及方便對齊縮排

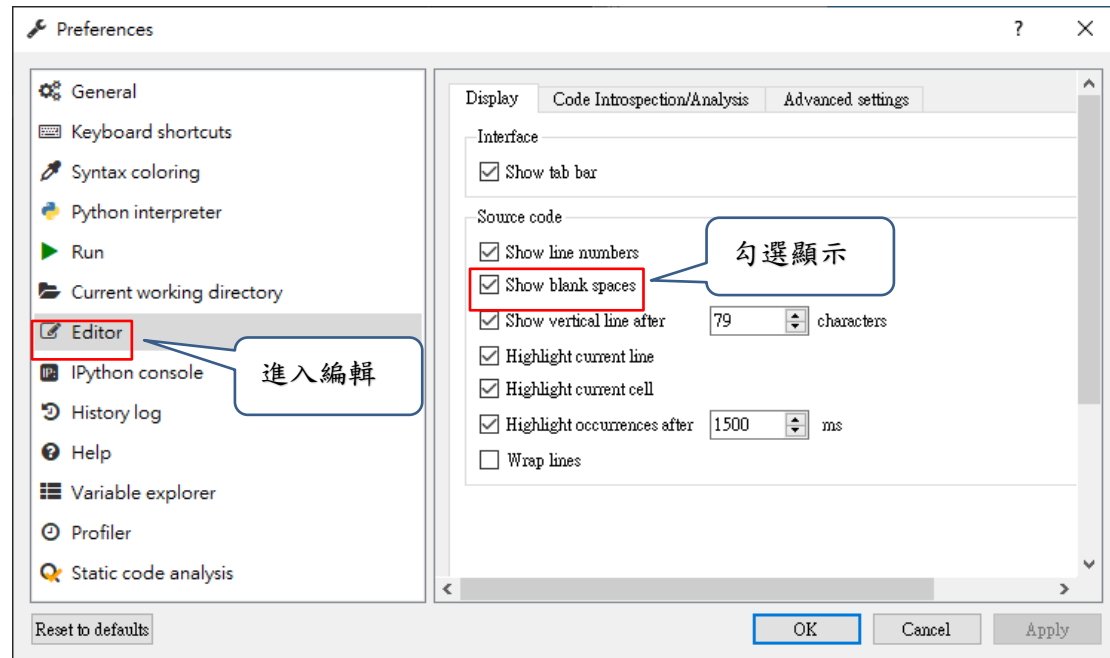


圖 B-1. Editor 設定

步驟 2. 設定預設縮排符號

使用者也可以設定縮排的符號，預設為四個空格(space)，本書建議將符號設為定位點符號(tab)，以方便觀察縮排對齊，如圖 B-2.

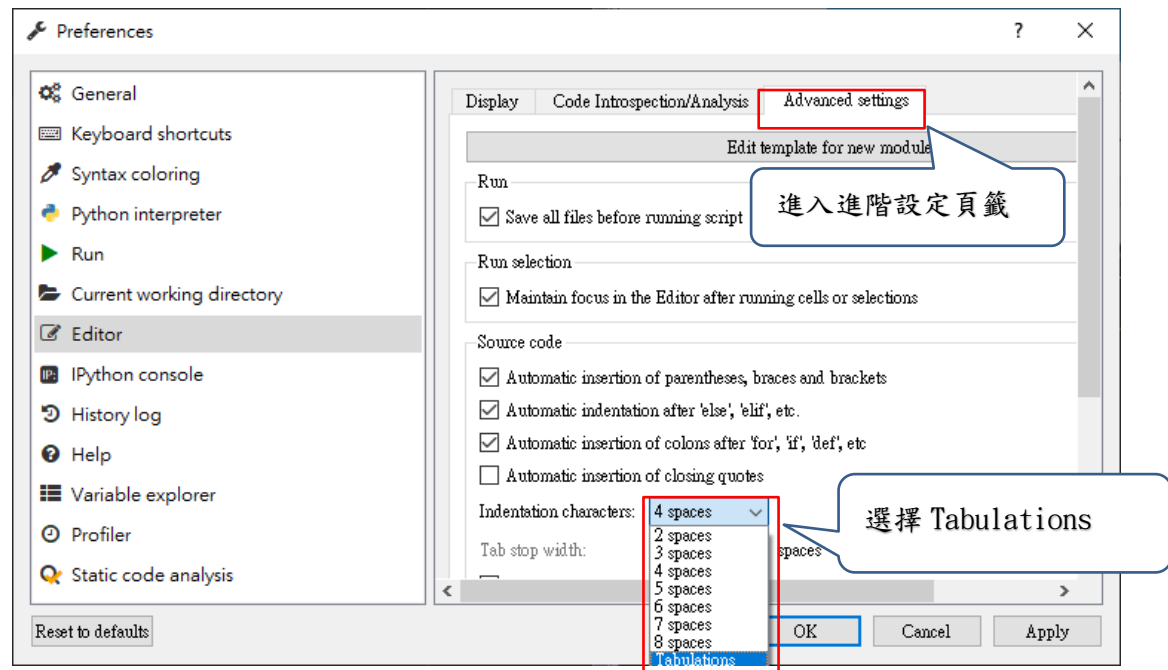


圖 B-2. 縮排符號設定

步驟 3. 完成畫面

如圖 B-3.，定位點(tab)符號以箭頭表示，空格以“點”表示



圖 B-3. 完成設定