



Python 套件應用

賴璉錡

lclai.t11@o365.fcu.edu.tw

Python檔案操作

- 在Python中，可以使用`open()` 函數來打開一個檔案，並且可以使用不同的模式來讀取或寫入檔案。在檔案操作完成後，要記得使用`close()` 函數來關閉檔案。
- 打開檔案
 - 使用`open()`函數可以打開一個檔案，並且可以指定不同的模式。

```
file = open("abc.txt", "r") # 以讀取模式打開檔案  
file = open("abc.txt", "w") # 以寫入模式打開檔案  
file = open("abc.txt", "a") # 以附加模式打開檔案
```

讀取檔案

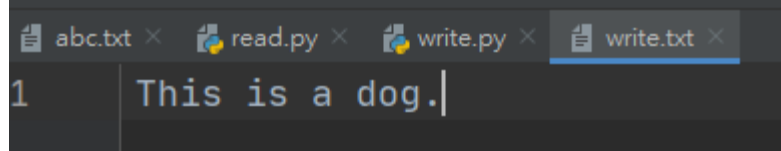
- 使用 `open()` 函數參數帶入 "r" 打開檔案後，可以使用 `read()` 函數來讀取檔案的內容。
- `encoding='utf-8'` : 編碼格式使用 utf-8

```
1 file = open("abc.txt", "r", encoding='utf-8') # 以讀取模式打開檔案
2 content = file.read() # 讀取檔案內容
3 print(content) # 印出檔案內容
4 file.close() # 關閉檔案
```

寫入檔案

- 使用 `open()` 函數參數帶入 "w" 打開檔案後，可以使用 `write()` 函數來寫入檔案的內容。

```
file = open("write.txt", "w") # 以寫入模式打開檔案  
file.write("This is a dog.") # 寫入檔案  
file.close() # 關閉檔案
```

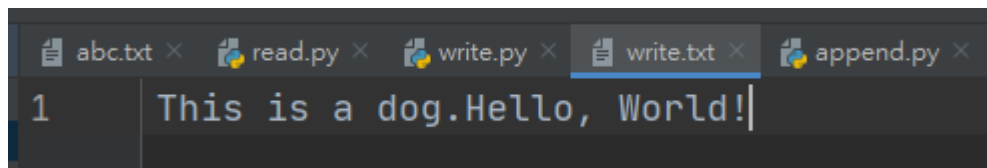


The screenshot shows a code editor with four tabs: 'abc.txt', 'read.py', 'write.py', and 'write.txt'. The 'write.txt' tab is active and shows the text 'This is a dog.' on the first line, with a cursor at the end of the line.

附加到檔案末尾

- 使用 `open()` 函數參數帶入 "a" 打開檔案後，可以使用 `write()` 函數來寫入檔案的內容。

```
file = open("write.txt", "a") # 以附加模式打開檔案  
file.write("Hello, World!") # 附加新的內容到檔案末尾  
file.close() # 關閉檔案
```



The screenshot shows a code editor with several tabs: 'abc.txt', 'read.py', 'write.py', 'write.txt', and 'append.py'. The 'write.txt' tab is active and shows the text 'This is a dog.Hello, World!' on line 1. The text is written in a monospaced font, and the cursor is at the end of the line.

Practice - 九九乘法表

- 將九九乘法表輸出的內容寫入到output.txt
- 換行符號為 `\n`

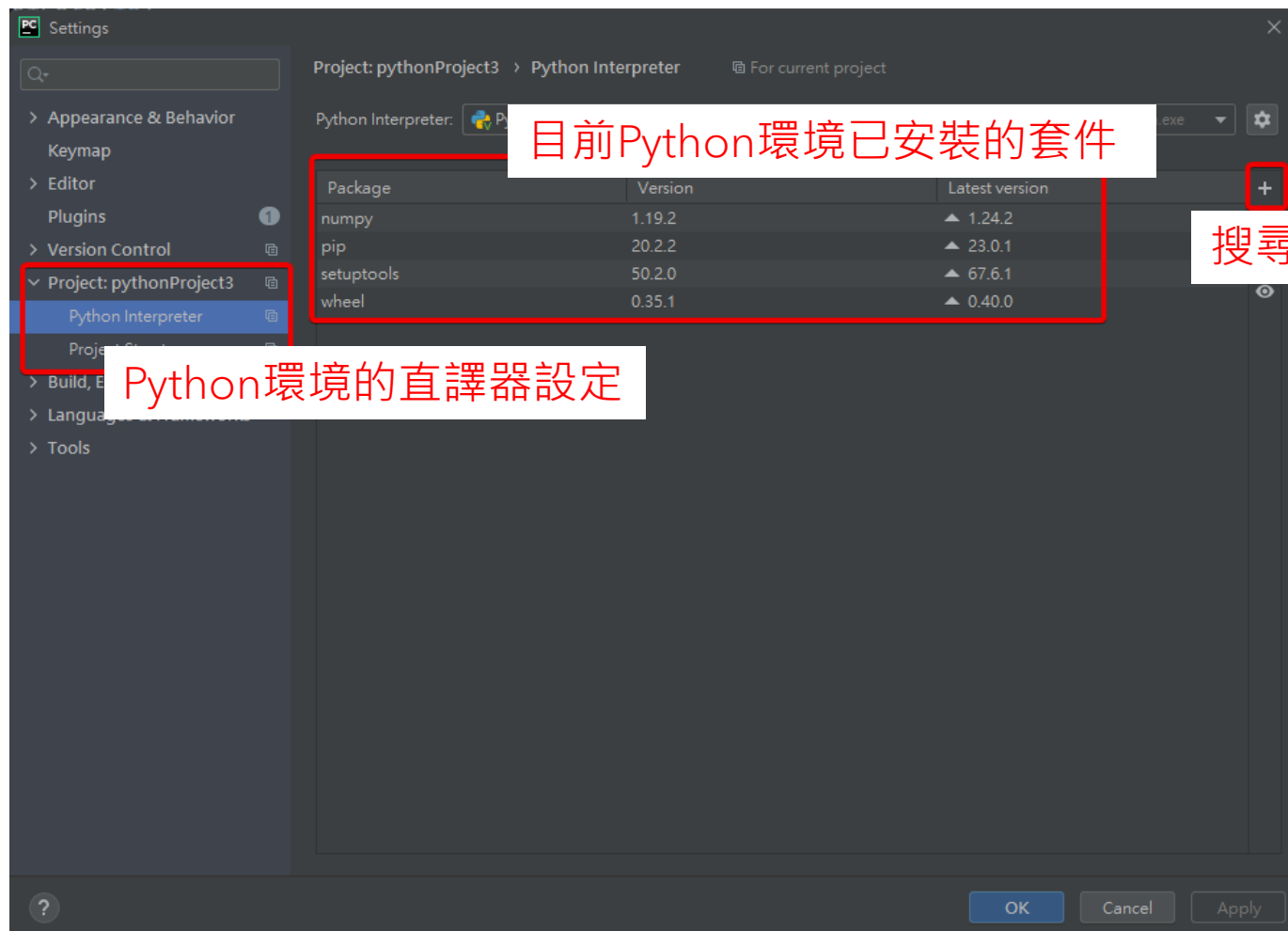
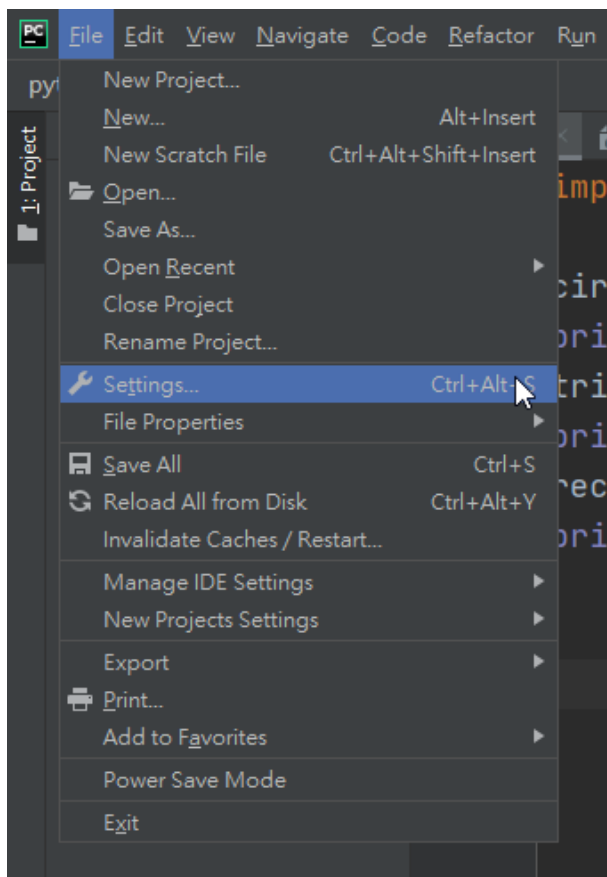
```
file = open("output.txt", "w") # 以寫入模式打開檔案
# 九九乘法表輸出內容
file.close() # 關閉檔案
```

Python 套件

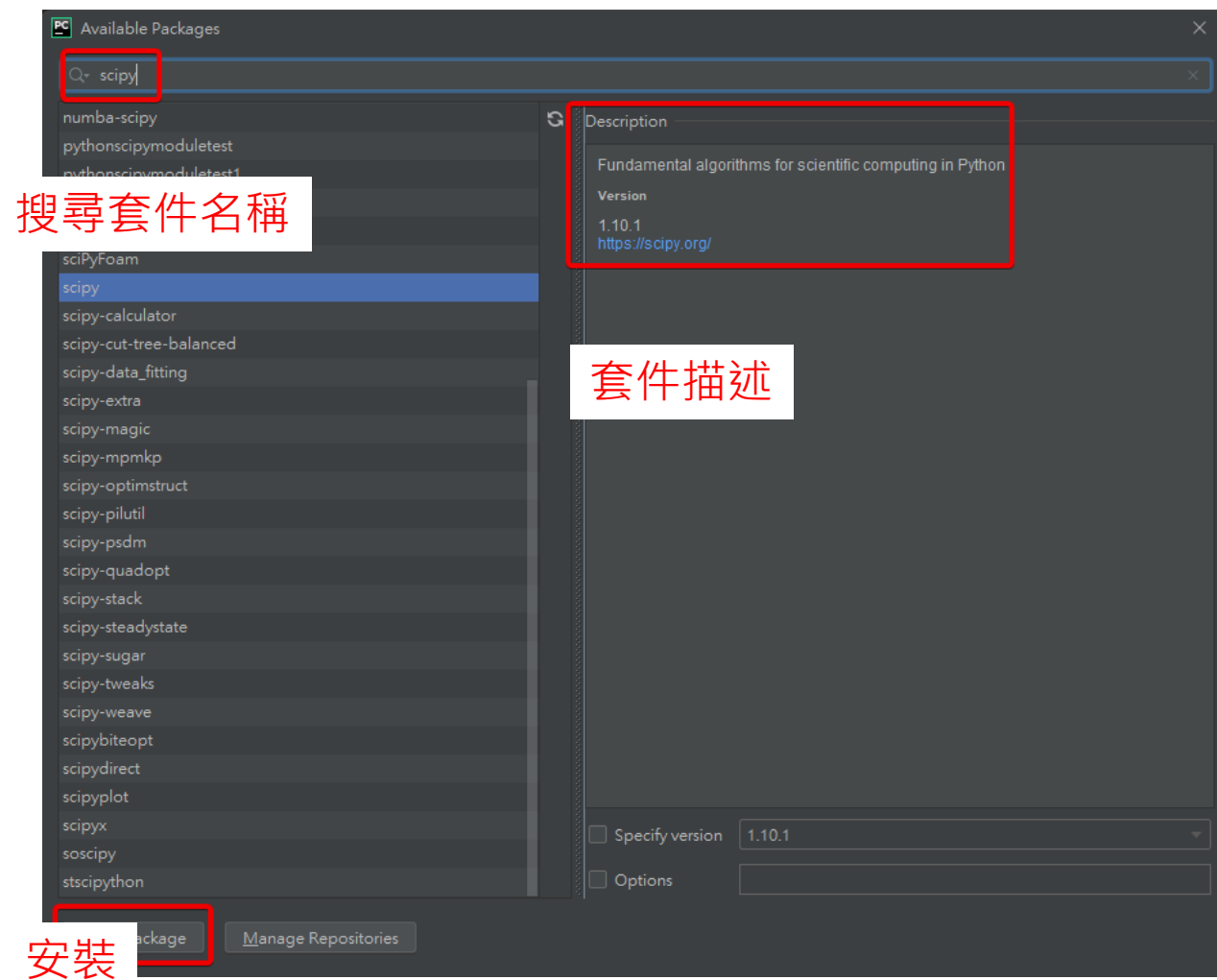
- 透過python中現成的模組與套件，可以減少大量重複性的程式碼，對開發而言更加便利。

透過 **PyCharm** 安裝套件

透過 PyCharm 安裝套件



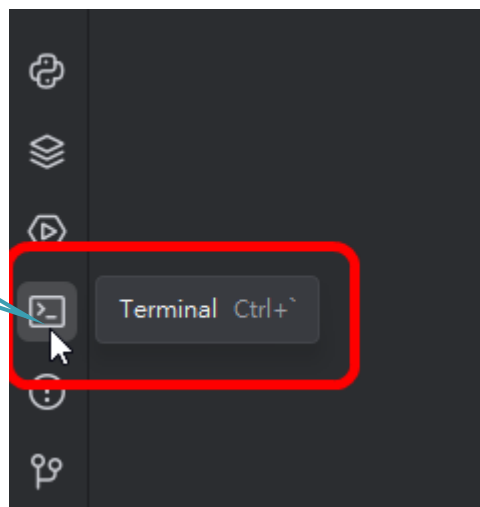
透過 PyCharm 安裝套件



透過「終端機」與「**pip**」安裝套件

啟動終端機

PyCharm左下角可以開啟終端機

A screenshot of the Windows PowerShell terminal window within PyCharm. The window title is 'Terminal Local x + v'. The content shows the Windows PowerShell prompt and output, including copyright information for Microsoft Corporation and a link to download the latest PowerShell. The prompt is '(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1>'.

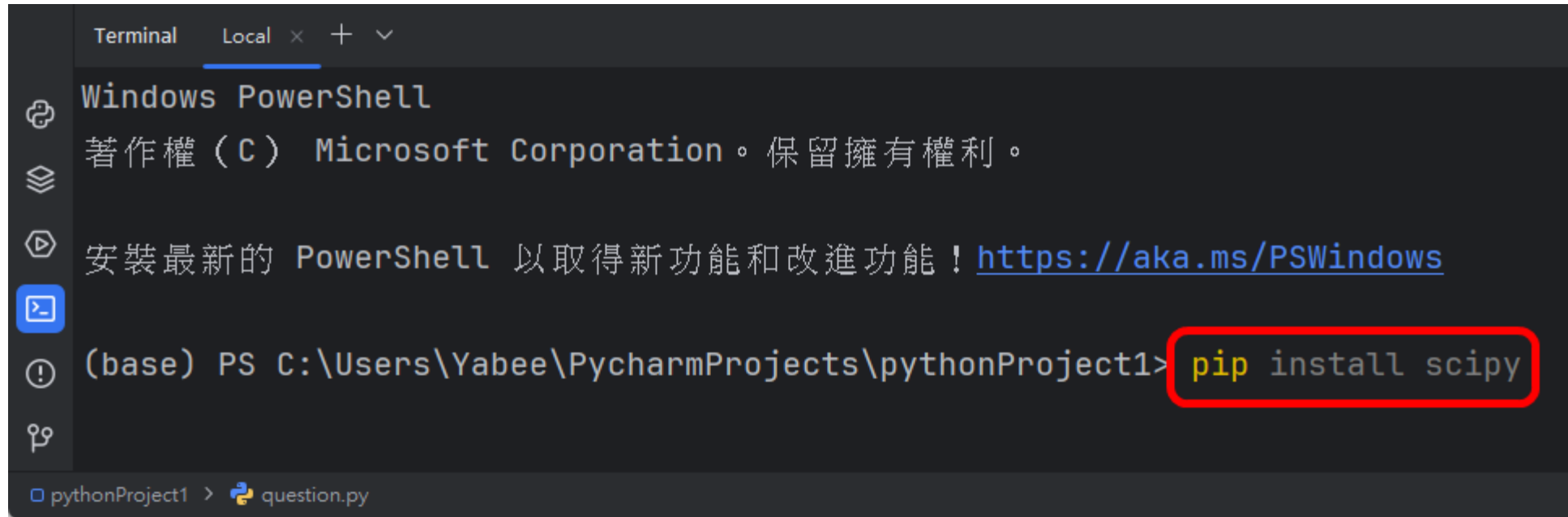
```
Terminal Local x + v
Windows PowerShell
著作權 (C) Microsoft Corporation。保留擁有權利。

安裝最新的 PowerShell 以取得新功能和改進功能！https://aka.ms/PSWindows

(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1>
```

pip 指令

```
$ pip install <套件名稱>
```



The screenshot shows a Windows PowerShell terminal window. The title bar indicates it's a 'Local' terminal. The window content includes the standard PowerShell startup text: 'Windows PowerShell', '著作權 (C) Microsoft Corporation。保留擁有權利。', and a message about installing the latest PowerShell with a link to <https://aka.ms/PSWindows>. The current prompt is '(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1>'. The command 'pip install scipy' is being typed at the prompt, with 'pip' highlighted in yellow and the entire command enclosed in a red rectangular box. The bottom status bar shows the current directory as 'pythonProject1' and the active file as 'question.py'.

```
Terminal Local x + v
Windows PowerShell
著作權 (C) Microsoft Corporation。保留擁有權利。
安裝最新的 PowerShell 以取得新功能和改進功能！https://aka.ms/PSWindows
(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1> pip install scipy
```

pip 指令

- 套件已安裝

```
Terminal Local x + v
安裝最新的 PowerShell 以取得新功能和改進功能！ https://aka.ms/PSWindows
(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1> pip install scipy
Requirement already satisfied: scipy in c:\users\yabee\anaconda3\lib\site-packages (1.9.1)
Requirement already satisfied: numpy<1.25.0,>=1.18.5 in c:\users\yabee\anaconda3\lib\site-packages (from scipy) (1.21.5)
(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1>
```

- 套件安裝完成

```
Collecting tzdata
  Downloading tzdata-2024.2-py2.py3-none-any.whl (346 kB)
  346.6/346.6 kB ? eta 0:00:00
Requirement already satisfied: typing-extensions>=4 in c:\users\yabee\anaconda3\lib\site-packages (from asgiref<4,>=3.6.0->django) (4.3.0)
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.8.1 django-4.2.16 sqlparse-0.5.2 tzdata-2024.2
(base) PS C:\Users\Yabee\PycharmProjects\pythonProject1>
```

已安裝的套件，包含相依的套件

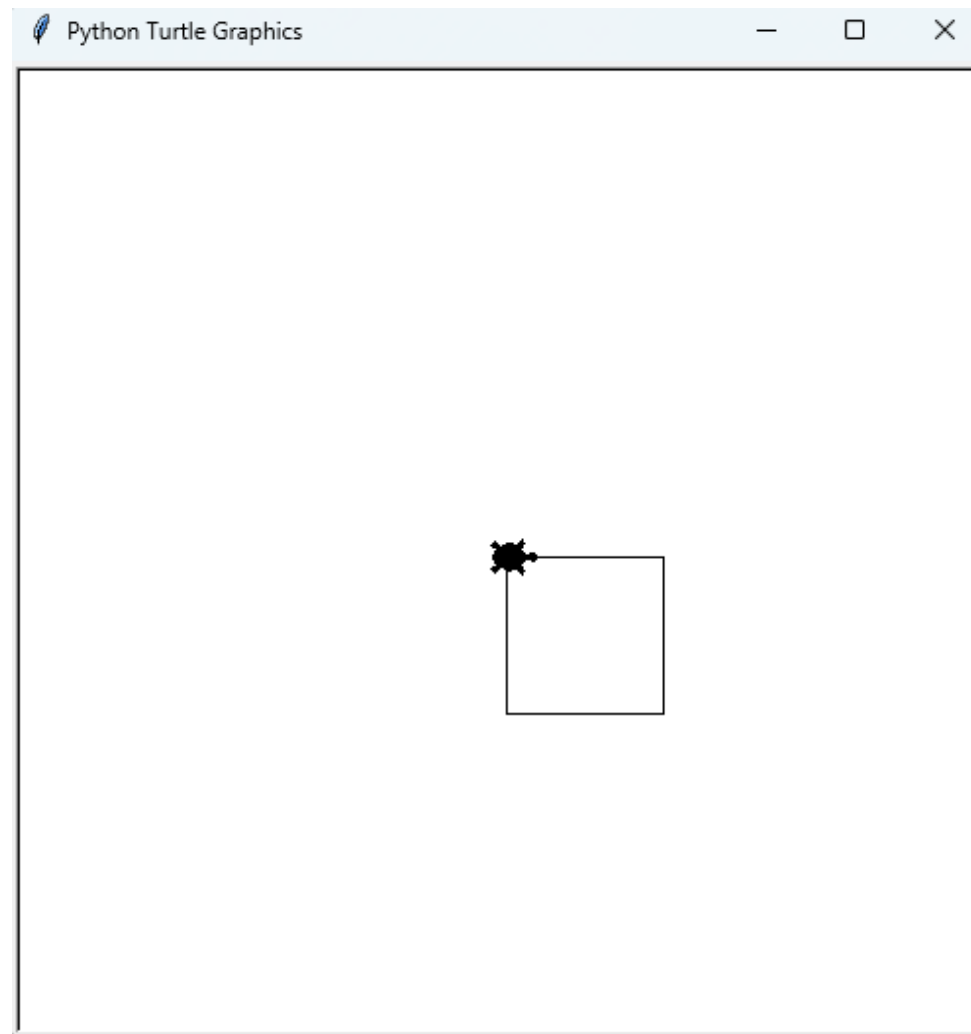
Python標準函式庫

- Python 的 Windows 安裝檔基本上包含整個標準函式庫，且通常也包含許多附加的組件^[1]。
- 常用函式庫
 - random
 - math
 - os
 - os.path

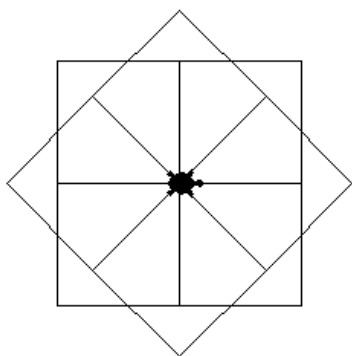
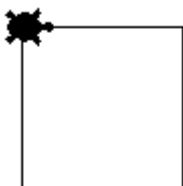
[1][Python 標準函式庫 \(Standard Library\) – Python 3.11.3 說明文件](#)

Practice - turtle

- Turtle模組是一種簡易繪圖程式，操作方式為載入turtle模組，開啟一個畫布(screen)，就能開始控制指標(turtle)移動方向(上、下、左、右)和繪圖應用(下筆、停筆、變換畫筆顏色)等變化應用，所有的控制動作都是使用指令語法操作。



Practice - turtle



```
import turtle
```

```
screen = turtle.Screen() # 注意S是大寫喔!
```

```
screen.setup(500, 500) # 設定畫布(視窗)長寬
```

```
turtle.pensize() # 線條的寬度
```

```
turtle.pencolor() # 線條的顏色
```

```
turtle.shape("turtle") # 筆尖的形狀，有"arrow"、"turtle"、"circle"、"square"
```

```
turtle.hideturtle() # 隱藏筆尖形狀，
```

```
turtle.speed(3) # 畫筆的移動速度，範圍是0-10，數字越大
```

```
# 移動
```

```
turtle.goto(-10, 22) # 將畫筆移動到(-10, 22)座標處
```

```
turtle.forward(40) # 向畫筆現在的方向移動40像素長
```

```
turtle.backward(40) # 向畫筆現在的相反方向移動40像素長
```

```
turtle.right(90) # 順時針右轉 90°
```

```
turtle.left(90) # 逆時針左轉 90°
```

```
screen.exitonclick() # 按一下才關閉畫面
```

若輸入數值大於10 或小於0.5 則速度設為0。速度字串與速度值的對應關係如下:

- "fastest": 0 最快
- "fast": 10 快
- "normal": 6 正常
- "slow": 3 慢
- "slowest": 1 最慢

kivy - The Open Source Python App Development Framework.

- <https://kivy.org/>

Kivy has been built to be easy to use, cross-platform and fast.

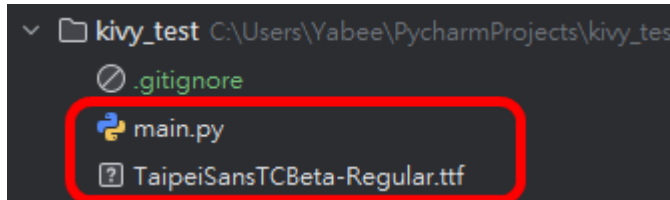
With a single codebase, you will be able to deploy apps on Windows, Linux, macOS, iOS and Android.

Practice: 待辦事項

1. 建立 kivy 專案
2. 規劃畫面，上半部為已儲存待辦事項，下半部為輸入框與按鈕
3. 新增待辦事項功能
4. 移除待辦事項功能
5. 修改待辦事項功能

Practice: 待辦事項

- 下載字型檔
 - [中文字型檔連結](#)
- 建立 main.py 檔案



```
from kivy.app import App
from kivy.uix.boxlayout import BoxLayout
from kivy.uix.label import Label
from kivy.core.text import LabelBase
from kivy.resources import resource_add_path
import os

# 引用字體檔案
resource_add_path(os.path.abspath('TaipeiSansTCBeta-Regular.ttf'))
# 將kivy預設的字體替換成指定的中文字體
LabelBase.register('Roboto', 'TaipeiSansTCBeta-Regular.ttf')

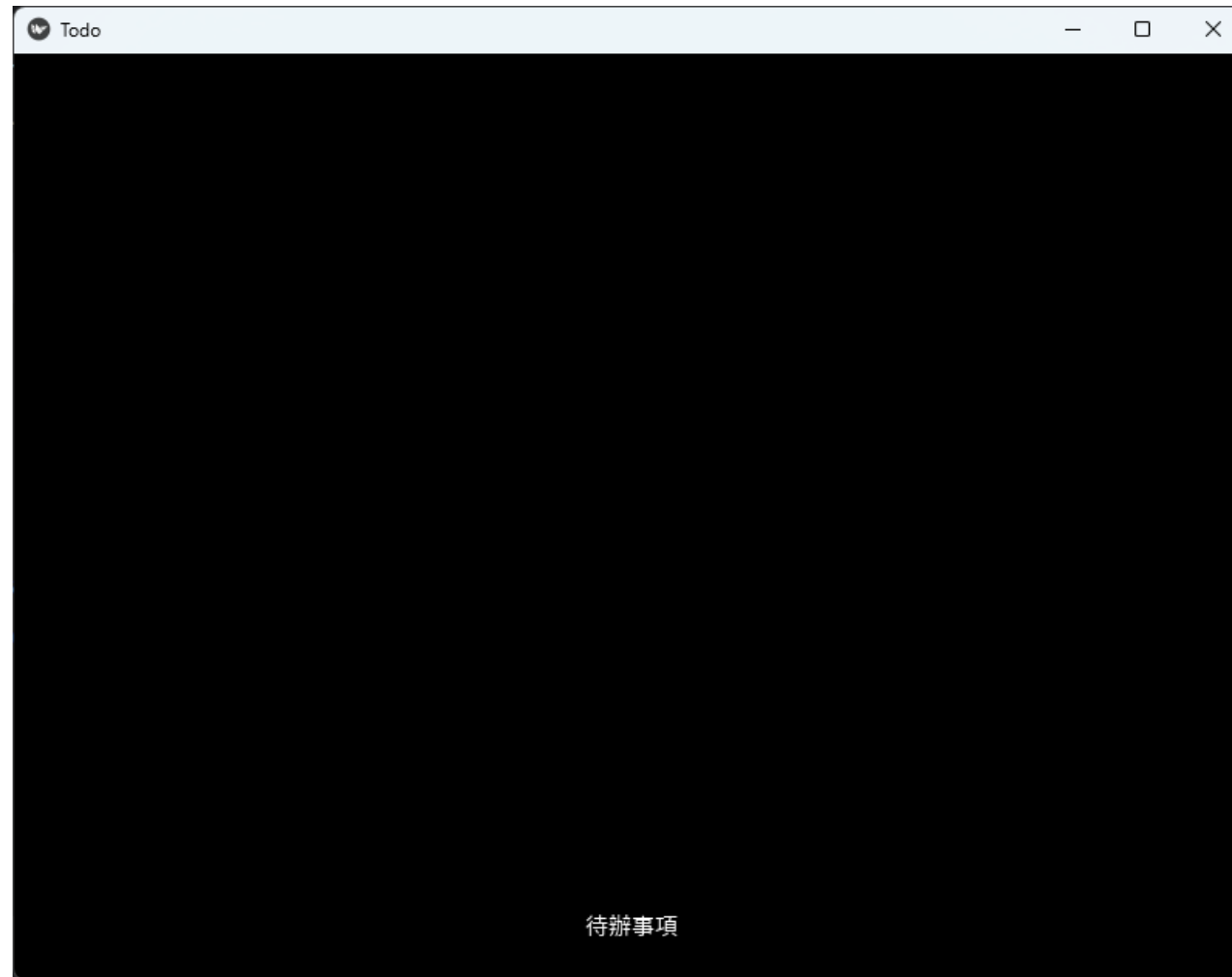
class TodoApp(App):
    def build(self):
        layout = BoxLayout(orientation='vertical', padding=10, spacing=10)

        title = Label(text='待辦事項', size_hint_y=None, height=50)
        layout.add_widget(title)

        return layout

if __name__ == '__main__':
    TodoApp().run()
```

Practice: 待辦事項



Practice: 待辦事項

- 規劃畫面，上半部為已儲存待辦事項，下半部為輸入框與按鈕

待辦事項

新增

Practice: 待辦事項

```
class TodoList(BoxLayout):
    def __init__(self, **kwargs):
        super().__init__(**kwargs)
        self.orientation = 'vertical'
        self.spacing = 10
        self.padding = 10

        # 上半部分：已儲存的待辦事項區域
        self.tasks_layout = BoxLayout(orientation='vertical', size_hint_y=0.7)
        self.scroll = ScrollView()
        self.tasks_container = BoxLayout(orientation='vertical', size_hint_y=None)
        self.tasks_container.bind(minimum_height=self.tasks_container.setter('height'))

        self.scroll.add_widget(self.tasks_container)
        self.tasks_layout.add_widget(self.scroll)

        # 下半部分：新增待辦事項區域
        self.input_layout = BoxLayout(orientation='horizontal', size_hint_y=0.3, spacing=10)
        self.task_input = TextInput(multiline=False)
        self.add_button = Button(text='新增', size_hint_x=0.3)

        self.input_layout.add_widget(self.task_input)
        self.input_layout.add_widget(self.add_button)

        # 將所有元件加入主布局
        self.add_widget(Label(text='待辦事項', size_hint_y=0.1))
        self.add_widget(self.tasks_layout)
        self.add_widget(self.input_layout)
```

Practice: 待辦事項

- 新增待辦事項功能

```
class TaskWidget(BoxLayout):
    def __init__(self, text, **kwargs):
        super().__init__(**kwargs)
        self.orientation = 'horizontal'
        self.size_hint_y = None
        self.height = 40
        self.spacing = 5

        # 待辦事項文字
        self.task_label = Label(text=text, size_hint_x=0.7)
        self.add_widget(self.task_label)

        # 刪除按鈕
        delete_button = Button(text='刪除', size_hint_x=0.3)
        self.add_widget(delete_button)
```

```
add_button.bind(on_press=self.add_task)
```

```
def add_task(self, instance):
    text = self.task_input.text.strip()
    if text:
        task_widget = TaskWidget(text)
        self.tasks_container.add_widget(task_widget)
        self.task_input.text = "" # 清空輸入框
```


Practice: 待辦事項

- 刪除待辦事項功能

```
class TaskWidget(BoxLayout):
    def __init__(self, text, parent_list, **kwargs):
        super().__init__(**kwargs)
        self.orientation = 'horizontal'
        self.size_hint_y = None
        self.height = 40
        self.spacing = 5

        # 保存對父列表的引用
        self.parent_list = parent_list

        # 待辦事項文字
        self.task_label = Label(text=text, size_hint_x=0.7)
        self.add_widget(self.task_label)

        # 刪除按鈕
        delete_button = Button(text='刪除', size_hint_x=0.3)
        delete_button.bind(on_press=self.on_delete)
        self.add_widget(delete_button)

    def on_delete(self, instance):
        """當刪除按鈕被點擊時，呼叫父列表的刪除方法"""
        self.parent_list.delete_task(self)
```

```
class TodoList
```

```
def add_task(self, instance):
    text = self.task_input.text.strip()
    if text:
        task_widget = TaskWidget(text, parent_list=self)
        self.tasks_container.add_widget(task_widget)
        self.task_input.text = " # 清空輸入框"
```

```
def delete_task(self, task_widget):
    """刪除指定的待辦事項"""
    self.tasks_container.remove_widget(task_widget)
```

Practice: 待辦事項

- 修改待辦事項功能

```
class TaskWidget(BoxLayout):  
    ...  
    # 保存對父列表的引用和文字內容  
    self.parent_list = parent_list  
    self.text = text  
  
    # 修改按鈕  
    delete_button = Button(text='修改', size_hint_x=0.2)  
    delete_button.bind(on_press=self.on_edit)  
    self.add_widget(delete_button)  
  
    ...  
    def on_edit(self, instance):  
        """當編輯按鈕被點擊時，呼叫父列表的編輯方法"""  
        self.parent_list.edit_task(self)
```

```
class TodoList
```

```
def edit_task(self, task_widget):  
    """開始編輯指定的待辦事項"""  
    self.task_input.text = task_widget.text  
    self.add_button.text = '確認修改'  
    self.task_input.focus = True # 自動聚焦到輸入框
```

作業: 修改後的儲存

- 練習的程式碼，修改文字後儲存會變成新增，請修好這個功能，使其修改後按下按鈕可以正常修改，而新增時按下按鈕會正常新增。

待辦事項

123

修改 刪除

123456

確認修改

待辦事項

123

123456

修改 刪除

修改 刪除

按下「確認修改」後會變成新增，
以及按鈕不會變回新增

確認修改

提示: 可以設定一個`self.editing_widget`來記錄當下修改中的widget