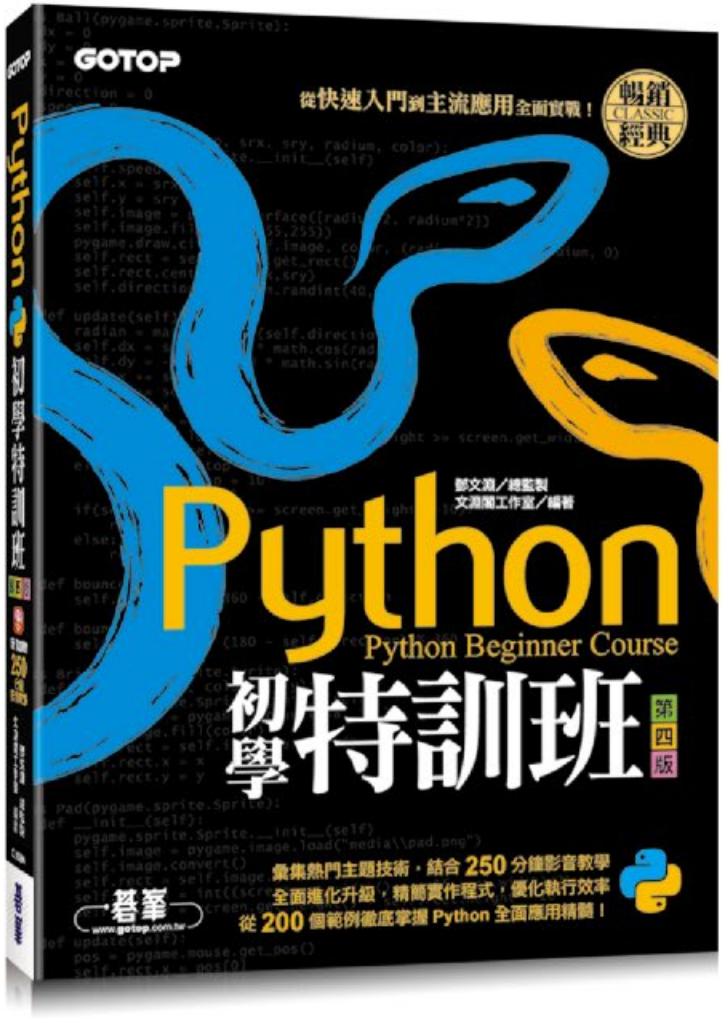


Python

Python Beginner Course

初學特訓班 第四版

CHAPTER

09

實戰：空氣好不好？PM2.5 即時監測顯示器

9.1 Pandas：強大的資料處理模組

Pandas 主要的資料型態有2 種：Series 是一維資料結構，其用法與串列類似；DataFrame 是二維資料結構， 表格即為DataFrame 的典型結構。本書僅說明 DataFrame 的使用方式。

9.1.1 建立 DataFrame 資料

使用Pandas 模組進行資料處理， 首先要匯入Pandas 模組， 官網建議在匯入 Pandas 模組時命名為「pd」，語法為：

```
import pandas as pd
```

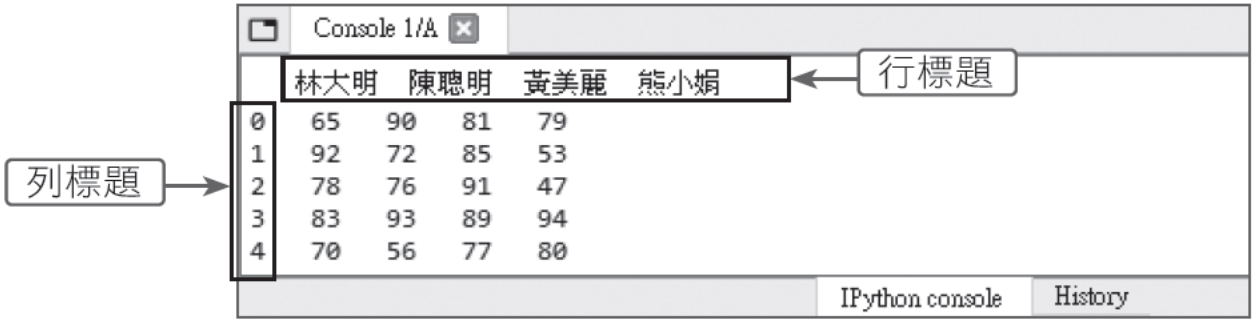
建立 DataFrame 的語法為：

```
資料變數 = pd.DataFrame( 資料型態 )
```

「資料」可有多種型態：第一種資料型態是以擁有相同數目元素串列的字典建立 DataFrame 資料，例如建立一個4 位學生，每人有 5 科成績的 DataFrame，資料變數名稱為 df：

```
df = pd.DataFrame( {"林大明":[65,92,78,83,70], "陳聰明":[90,72,76,93,56],\
    "黃美麗":[81,85,91,89,77], "熊小娟":[79,53,47,94,80] } )
```

建立的 DataFrame 如下圖：以字典「鍵」做為行標題。(<dataframe1.py>)

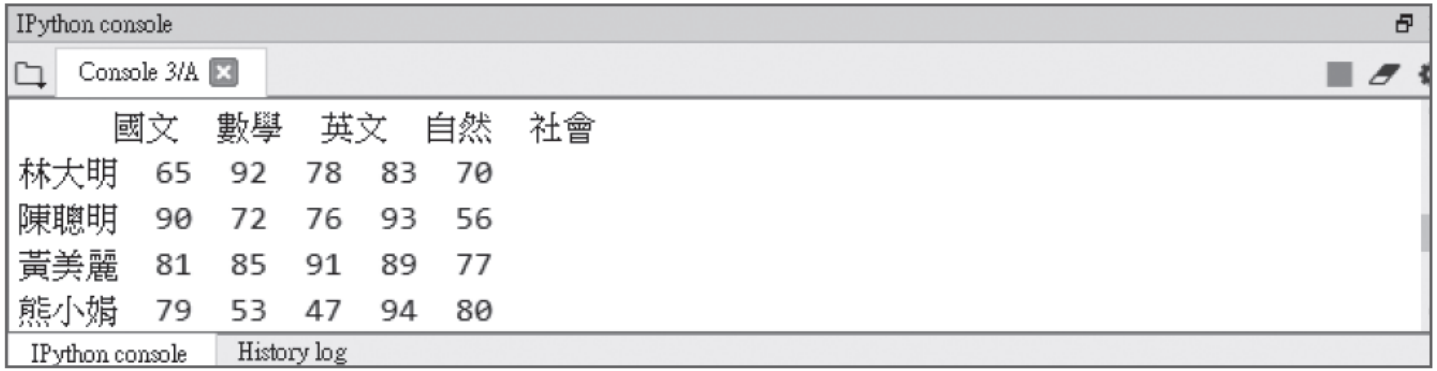


第二種資料型態是自行指定行及列標題：

```
資料變數 = pd.DataFrame( 資料 [, columns= 行標題串列 , index= 列標題串列 ])
```

例如建立一個4 位學生，每人有5 科成績的DataFrame，資料變數名稱為df，行標題為科目名稱，列標題為學生姓名：(<dataframe2.py>)

```
datas = [[65,92,78,83,70], [90,72,76,93,56], [81,85,91,89,77], [79,53,47,94,80]]
indexs = ["林大明", "陳聰明", "黃美麗", "熊小娟"]
columns = ["國文", "數學", "英文", "自然", "社會"]
df = pd.DataFrame(datas, columns=columns,  index=indexs)
```



修改行、列標題

如果建立DataFrame 時沒有設定行、列標題，或者程式執行過程中需修改行、列標題，例如前面範例中學生更改姓名，可使用修改行、列標題命令修改行、列標題。

修改行標題的語法為：

```
df.columns = 行標題串列
```

修改列標題的語法為：

```
df.index = 列標題串列
```

9.1.2 取得DataFrame 資料

取得行資料

取得一個行資料的語法為：

```
df[ 行標題 ]
```

若要取得2 個以上行資料則需以2 個中括號包圍行標題，語法為：

```
df[[ 行標題 1, 行標題 2, ……]]
```

也可以使用行資料進行邏輯運算來取得資料，例如取得數學科成績80 分以上(含) 的所有學生成績：

```
df[df.數學 >= 80]
```

df.values 取得資料

「df.values」可取得全部資料，是一個二維串列，執行結果為：

(<datatake2.py>)

```
[ [65 92 78 83 70]
 [90 72 76 93 56]
 [81 85 91 89 77]
 [79 53 47 94 80] ]
```

取得第2 位學生陳聰明成績的語法為：

```
df.values[1]
```

執行結果：

```
[90 72 76 93 56]
```

取得第2 位學生陳聰明的英文成績 (第3 個科目) 的語法為：

```
df.values[1][2]
```

執行結果為「76」。

df.loc：以行、列標題取得資料

使用 df.loc 的語法為：

```
df.loc[ 列標題 , 行標題 ]
```

例如取得學生陳聰明的所有成績：(<datatake3.py>)

```
df.loc[" 陳聰明 ", :]
```

取得學生陳聰明的數學科成績：

```
df.loc[" 陳聰明 "][" 數學 "]
```

取得學生陳聰明、熊小娟的所有成績：

```
df.loc[(" 陳聰明 ", " 熊小娟 "), :]
```

取得學生陳聰明、熊小娟的數學、自然科成績：

```
df.loc[(" 陳聰明 ", " 熊小娟 "), (" 數學 ", " 自然 ")]
```

取得學生陳聰明到熊小娟的數學科到社會科成績：

```
df.loc["陳聰明":"熊小娟", "數學":"社會"]
```

取得從頭到黃美麗的學生，他們的數學科到社會科成績：

```
df.loc[:"黃美麗", "數學":"社會"]
```

取得從陳聰明到最後的學生，他們的數學科到社會科成績：

```
df.loc["陳聰明":, "數學":"社會"]
```

df.iloc：以行、列位置取得資料

「df.iloc」是以行、列位置取得資料，語法為：

```
df.iloc( 列位置 , 行位置 )
```

例如取得陳聰明 (第2 位學生) 的所有成績：(<datatake4.py>)

```
df.iloc[1, :]
```

取得學生陳聰明的數學科 (第2 個科目) 成績：

```
df.iloc[1][1]
```

取得最前或最後數列資料

如果要取得最前面幾列資料，可使用 head 方法，語法為：

```
df.head([n])
```

若要取得最後面幾列資料，則使用 tail 方法，語法為：

```
df.tail([n])
```

9.1.3 修改及排序DataFrame 資料

修改 DataFrame 資料

要修改DataFrame 的資料非常簡單，只要於前一節中取得的資料項目設定指定值即可。例如修改陳聰明的數學成績為 91：(<datamodify1.py>)

```
df.loc["陳聰明"]["數學"] = 91
```

或修改陳聰明的所有成績皆為 80：

```
df.loc["陳聰明", :] = 80
```

排序DataFrame 資料

Pandas 提供2 種方法對DataFrame 資料排序。

第1 種是根據資料數值排序，語法為：

```
資料變數 = df.sort_values(by= 行標題 [, ascending= 布林值 ])
```

- **行標題**：做為排序值的行標題。
- **布林值**：可省略，True 表示遞增排序 (預設值)，False 表示遞減排序。

例如以數學成績做遞減排序，並將結果存於df1 中：(<datasort1.py>)

```
df1 = df.sort_values(by=" 數學 ", ascending=False)
```

第2 種是根據行、列標題排序，語法為：

```
資料變數 = df.sort_index(axis= 行列數值 [, ascending= 布林值 ])
```

■ **行列數值**：0表示依列標題排序，1 表示依行標題排序。

例如按照列標題遞增排序，並將結果存於df2 中：

```
df2 = df.sort_index(axis=0)
```

9.1.4 刪除DataFrame 資料

Pandas 使用drop 刪除DataFrame 資料，語法為：

```
資料變數 = df.drop( 行標題或列標題 [, axis= 行列數值 ])
```

■ **行列數值**：0表示依列標題排序(預設值)，1 表示依行標題排序。

例如刪除陳聰明(列標題) 的成績：(<datadrop1.py>)

```
df1 = df.drop("陳聰明")  #axis 參數可省略
```

刪除數學科(行標題) 成績：

```
df2 = df.drop("數學", axis=1)
```

若刪除的行或列超過1 個，需以串列做為參數，例如刪除數學科及自然科成績：

```
df3 = df.drop(["數學", "自然"], axis=1)
```

如果刪除的行或列項目很多且連續，可使用刪除「範圍」方式處理。刪除連續列的語法為：

```
資料變數 = df.drop(df.index[開始數值:結束數值][, axis=行列數值])
```

執行結果會刪除「開始數值」到「結束數值 - 1」列，例如刪除第2 列到第4 列 (陳聰明、黃美麗、熊小娟) 成績：

```
df4 = df.drop(df.index[1:4])
```

刪除連續行的語法為：

```
資料變數 = df.drop(df.columns[開始數值:結束數值][, axis=行列數值])
```

例如刪除第2 行到第4 行 (數學、英文、自然) 成績：

```
df5 = df.drop(df.columns[1:4], axis=1)
```

9.1.5 匯入資料

自行製作Pandas 的DataFrame 資料是件非常繁瑣的工作，通常是將資料存於統計軟體Excel 或資料庫中，再將資料匯入Pandas。另一種情況是擷取網頁中成千上萬的表格資料，匯入Pandas 成為DataFrame。

Pandas 常用的匯入資料方法有：

方法	說明
read_csv	匯入表格式文字資料 (*.csv)
read_excel	匯入 Microsoft Excel 資料 (*.xlsx)
read_sql	匯入 SQLite 資料庫資料 (*.sqlite)
read_json	匯入 Json 格式文字資料 (*.json)
read_html	匯入網頁中表格資料 (*.html)

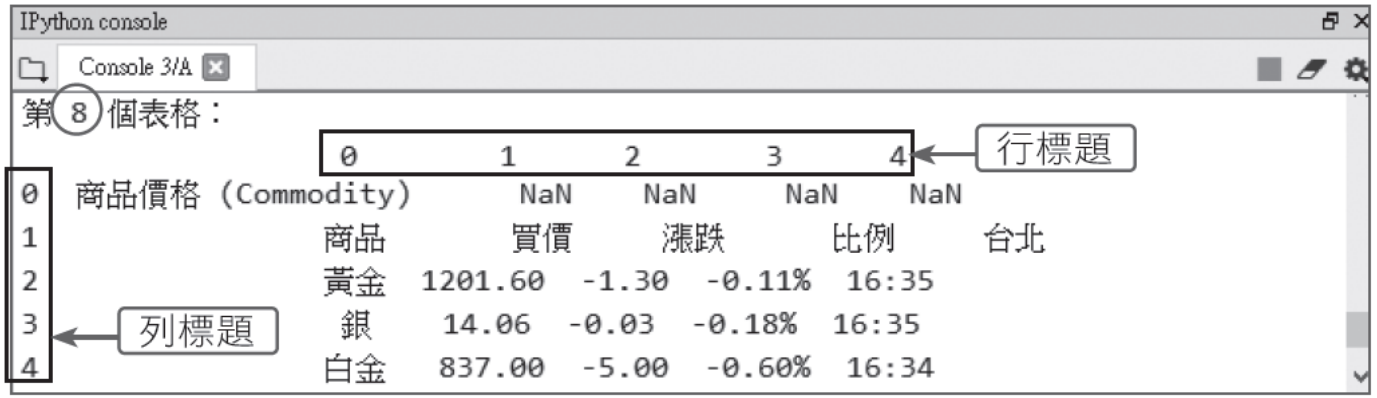
Pandas 的read_html 方法會使用html5lib 模組，在Anaconda Prompt 中以下列命令安裝：

```
conda install html5lib
```

程式碼：**readhtml1.py**

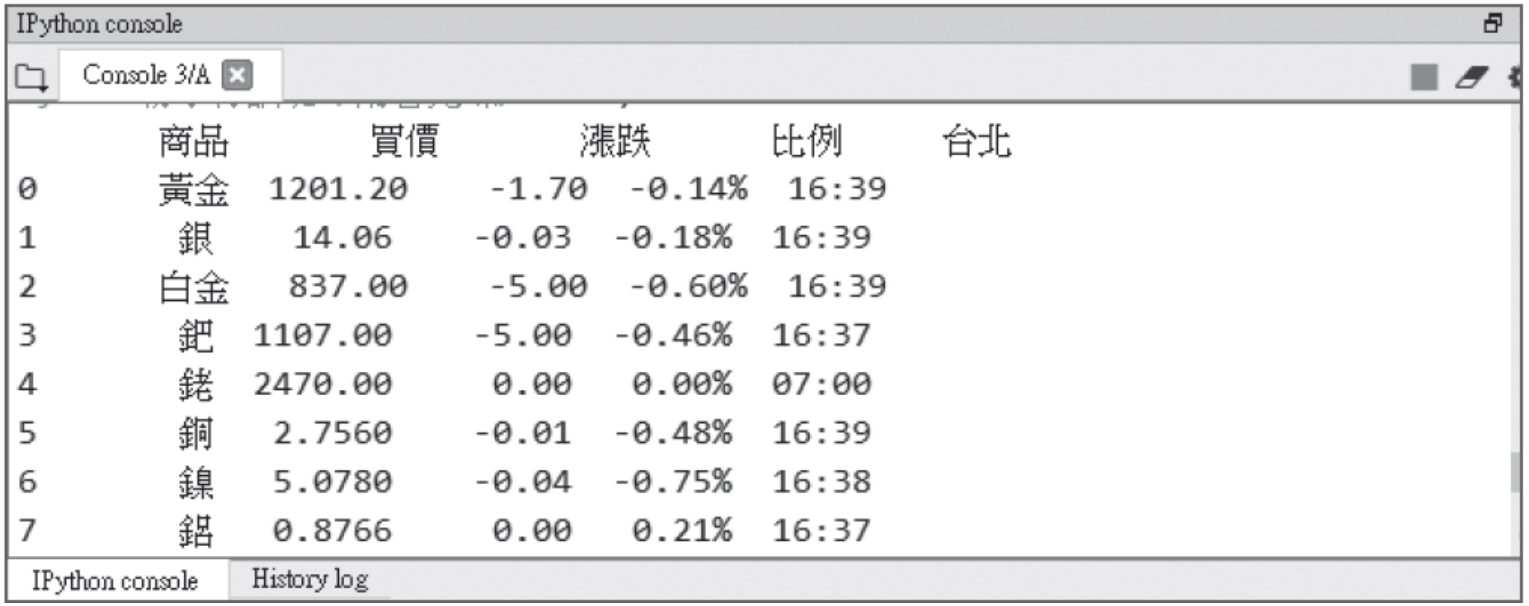
```
1 import pandas as pd
2 tables = pd.read_html("http://www.stockq.org/market/commodity.php")
3 n = 1
4 for table in tables:
5     print("第 " + str(n) + " 個表格：")
6     print(table.head())
7     print()
8     n += 1
```

執行結果：



範例：擷取網頁原物料商品行情表格資料

以read_html 方法擷取網頁原物料商品行情表格資料，並移除前2 列資料，然後重新設定行、列標題。



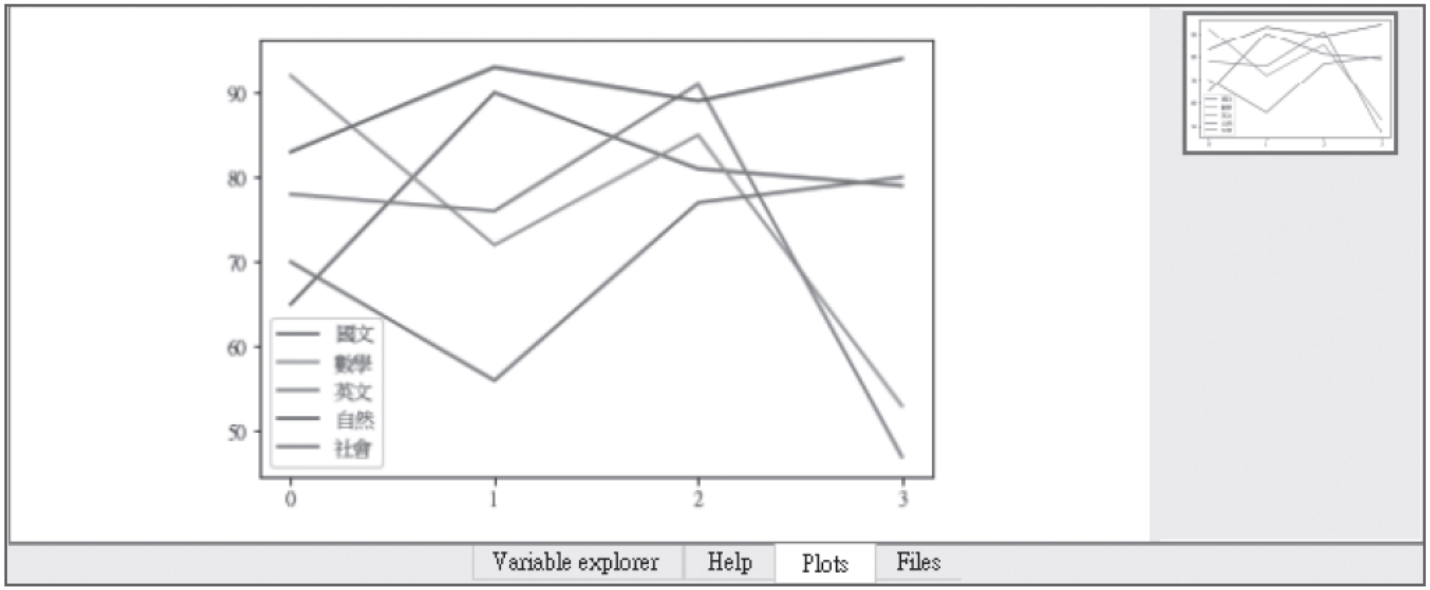
9.1.6 繪製線形圖

Pandas 繪製圖形功能的語法為：

```
df.plot()
```

範例：繪製學生成績線形圖

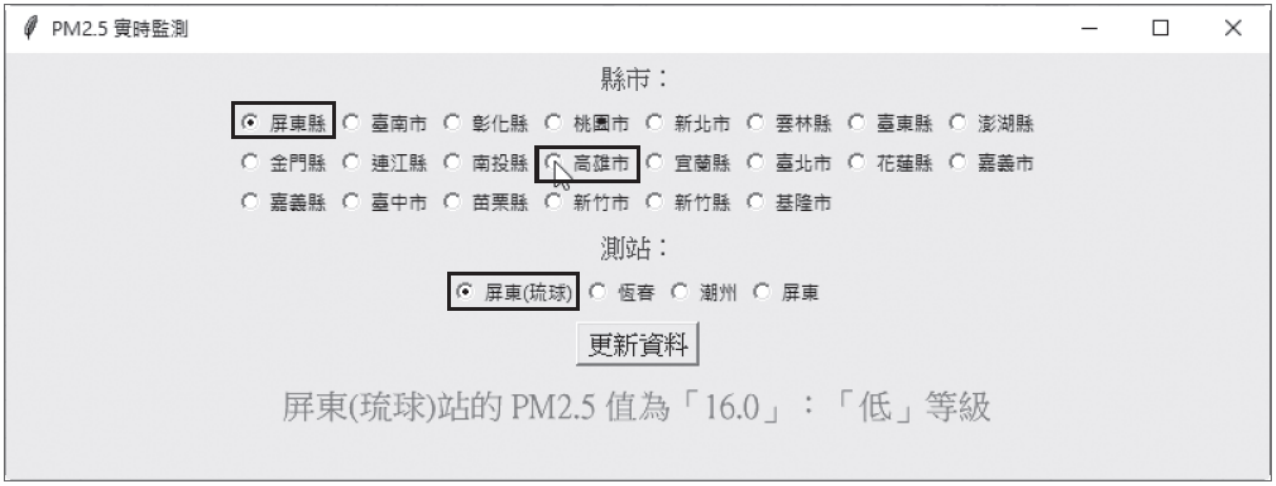
以DataFrame 資料繪製線形統計圖。

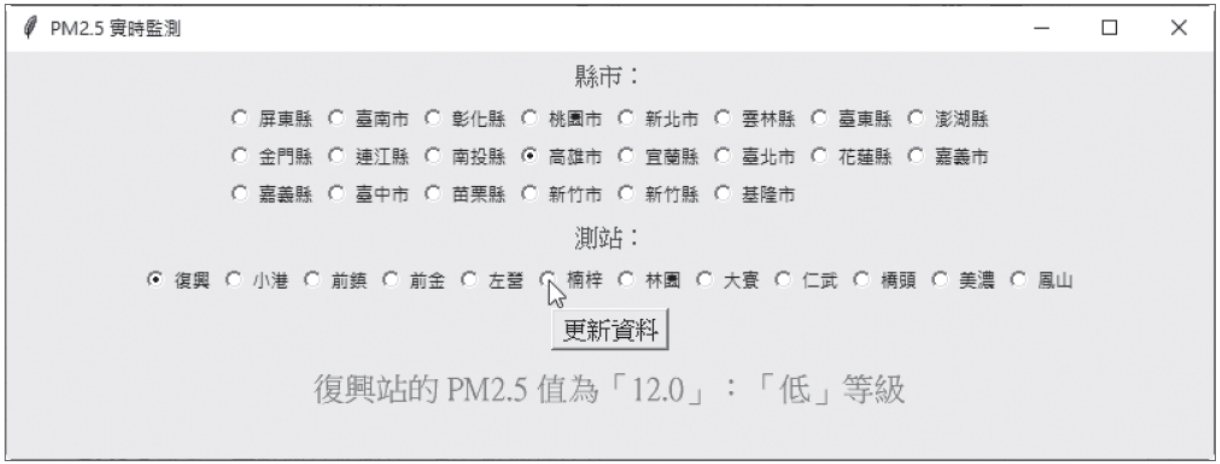


9.2 實戰：PM2.5 即時監測顯示器

本專題將撰寫即時監測 PM2.5 資料的應用程式，直接使用開放平台資料，隨時都可取得最新監測資料。

9.2.1 應用程式總覽




PM2.5 實時監測
縣市：
屏東縣 臺南市 彰化縣 桃園市 新北市 雲林縣 臺東縣 澎湖縣
金門縣 連江縣 南投縣 高雄市 宜蘭縣 臺北市 花蓮縣 嘉義市
嘉義縣 臺中市 苗栗縣 新竹市 新竹縣 基隆市
測站：
復興 小港 前鎮 前金 左營 楠梓 林園 大寮 仁武 橋頭 美濃 鳳山
更新資料
復興站的 PM2.5 值為「12.0」：「低」等級


PM2.5 實時監測
縣市：
屏東縣 臺南市 彰化縣 桃園市 新北市 雲林縣 臺東縣 澎湖縣
金門縣 連江縣 南投縣 高雄市 宜蘭縣 臺北市 花蓮縣 嘉義市
嘉義縣 臺中市 苗栗縣 新竹市 新竹縣 基隆市
測站：
復興 小港 前鎮 前金 左營 楠梓 林園 大寮 仁武 橋頭 美濃 鳳山
更新資料
楠梓站的 PM2.5 值為「2.0」：「低」等級