Python 的 50+ 練習：資料科學學習手冊

> *使用模組包裝函數與類別*

數據交點 | 郭耀仁 yaojenkuo@datainpoint.com

這個章節會登場的保留字與模組

- `import` 保留字。
- `from` 保留字。
- `as` 保留字。
- `datetime` 模組。
- `os` 模組。
- `this` 模組。

複習：Python 提供三種機制供使用者組織程式碼

視應用範疇由小到大依序為：

1. 函數（Function）。
2. 類別（Class）。
3. **模組（Module）**。

複習：如何理解程式碼組織機制的層次

- 數行程式碼可以組織為一個函數。
- 數個函數可以組織為一個類別。
- **數個函數或類別可以組織為一個模組**。
- **數個模組可以組織為一個功能更多的模組**。

關於模組

複習：Python 禪學（The Zen of Python）

- `import` 是 Python 的保留字（Keywords），可以載入模組。
- `this` 是 Python 的一個標準模組，可以印出 Python 禪學。

In [1]:
```
import this
```

```
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

什麼是模組

模組（Module）指的是以檔案或資料夾形式，來組織 Python 的函數以及類別。檔案形式可以對應「數個函數或類別可以組織為一個模組」層次、資料夾形式可以對應「數個模組可以組織為一個功能更多的模組」層次。

模組有三個來源

1. **標準模組**。
2. **來自使用者的定義**。
3. 第三方模組。

目前還沒有載入第三方模組的需求

之後在課程的第三與第四部分「Python 資料科學的基礎」以及「資料科學的應用場景」會說明。

標準模組（Standard libraries）

- 伴隨 Python 直譯器（來自 python.org 的版本）安裝的模組。
- 可以直接載入並使用。
- 有哪些標準模組可以直接載入並使用：https://docs.python.org/3/library

如何載入模組

使用 `import`、`from` 與 `as` 保留字

- 使用 `import` 載入模組。
- 使用 `from module import function/class` 載入模組中特定的函數或類別。
- 使用 `as` 調整模組、函數或者類別的命名。

使用 `import` 載入模組

- 以 `module` 命名使用。
- 使用模組中的函數、類別時都以 `module.function()` 或 `module.class()` 來參照命名。

```
import module

module.function()
object = module.class()
```

In [2]:

```
import os

os.getcwd()
```

Out[2]:

```
'/home/jovyan/07-modules'
```

In [3]:

```
import datetime

first_day_of_2022 = datetime.date(2022, 1, 1)
print(first_day_of_2022)
```

```
2022-01-01
```

使用 `from module import function/class` 載入模組中特定的函數或類別

載入特定函數或類別，就直接以 `function()` 或 `class()` 來參照命名。

```
from module import function
from module import class

function()
object = class()
```

In [4]:
```
from os import getcwd

getcwd()
```

Out[4]:
```
'/home/jovyan/07-modules'
```

In [5]:
```
from datetime import date

first_day_of_2022 = date(2022, 1, 1)
print(first_day_of_2022)
```

```
2022-01-01
```

使用 `as` 調整模組、函數或者類別的命名

```
import module as module_alias
from module import function as function_alias
from module import class as class_alias

module_alias.function()
function_alias()
object = class_alias()
```

In [6]:

```
import os as operating_system
from os import getcwd as get_current_working_directory

print(operating_system.getcwd())
print(get_current_working_directory())
```

```
/home/jovyan/07-modules
/home/jovyan/07-modules
```

In [7]:

```
import datetime as dttm
from datetime import date as dt

first_day_of_2022 = dttm.date(2022, 1, 1)
print(first_day_of_2022)
first_day_of_2022 = dt(2022, 1, 1)
print(first_day_of_2022)
```

```
2022-01-01
2022-01-01
```

如何決定模組載入的形式

- `import module`
- `import module as alias`
- `from module import function/class`
- `from module import function/class as alias`

如何決定模組載入的形式（續）

- 依照模組說明文件（Documentation）中的範例決定。
- 例如：使用 `os` 模組時採用 `import os`
- 例如：使用 `datetime` 模組時採用 `from datetime import function/class`

來源：https://docs.python.org/3/library/os.html#file-object-creation, https://docs.python.org/3/library/datetime.html#examples-of-usage-date

In [8]:
```
import os

os.getcwd()
```

Out[8]:
```
'/home/jovyan/07-modules'
```

In [9]:
```
from datetime import date

first_day_of_2022 = date(2022, 1, 1)
print(first_day_of_2022)
```

```
2022-01-01
```

關於路徑

模組存在的兩種形式：檔案或資料夾

1. 檔案的名稱為模組名，副檔名為 `.py`，例如檔案 `module.py` 就是命名為 `module` 的模組。
2. 資料夾的名稱為模組名，可以收納檔案形式的模組，例如資料夾 `library` 就是命名為 `library` 的模組。

為什麼模組名稱可以被認得

- Python 會搜尋標準模組的安裝路徑中是否有檔名為 `datetime.py`、`os.py`、`this.py` 或資料夾名稱 `this`、`os`、`datetime`（事實上這裡舉例的模組形式均為檔案名稱）。
- 如果沒有，就會產生找不到模組錯誤（`ModuleNotFoundError`）。
- 如果有，就會依據指令載入裡面全部或特定的函數與類別。

In [10]:
```
# There is a this.py, but no that.py
try:
    import that
except ModuleNotFoundError as error_message:
    print(error_message)
```

```
No module named 'that'
```

什麼是路徑

> *路徑是一段用來標註檔案或資料夾在電腦中位置的文字。*

來源：https://en.wikipedia.org/wiki/Path_(computing)

In [11]:
```
import os

os.getcwd() # path: current working directory
```

Out[11]:
```
'/home/jovyan/07-modules'
```

路徑的起點（根目錄）

- Unix(macOS/Linux): `/`
- Windows: `C:\`
- 課程所使用的 JupyterLab 根目錄為 `/`

預設的路徑（家目錄）

- Unix(macOS/Linux): `/Users/username`
- Windows: `C:\Users\username`
- 課程所使用的 JupyterLab 家目錄為 `/home/jovyan`

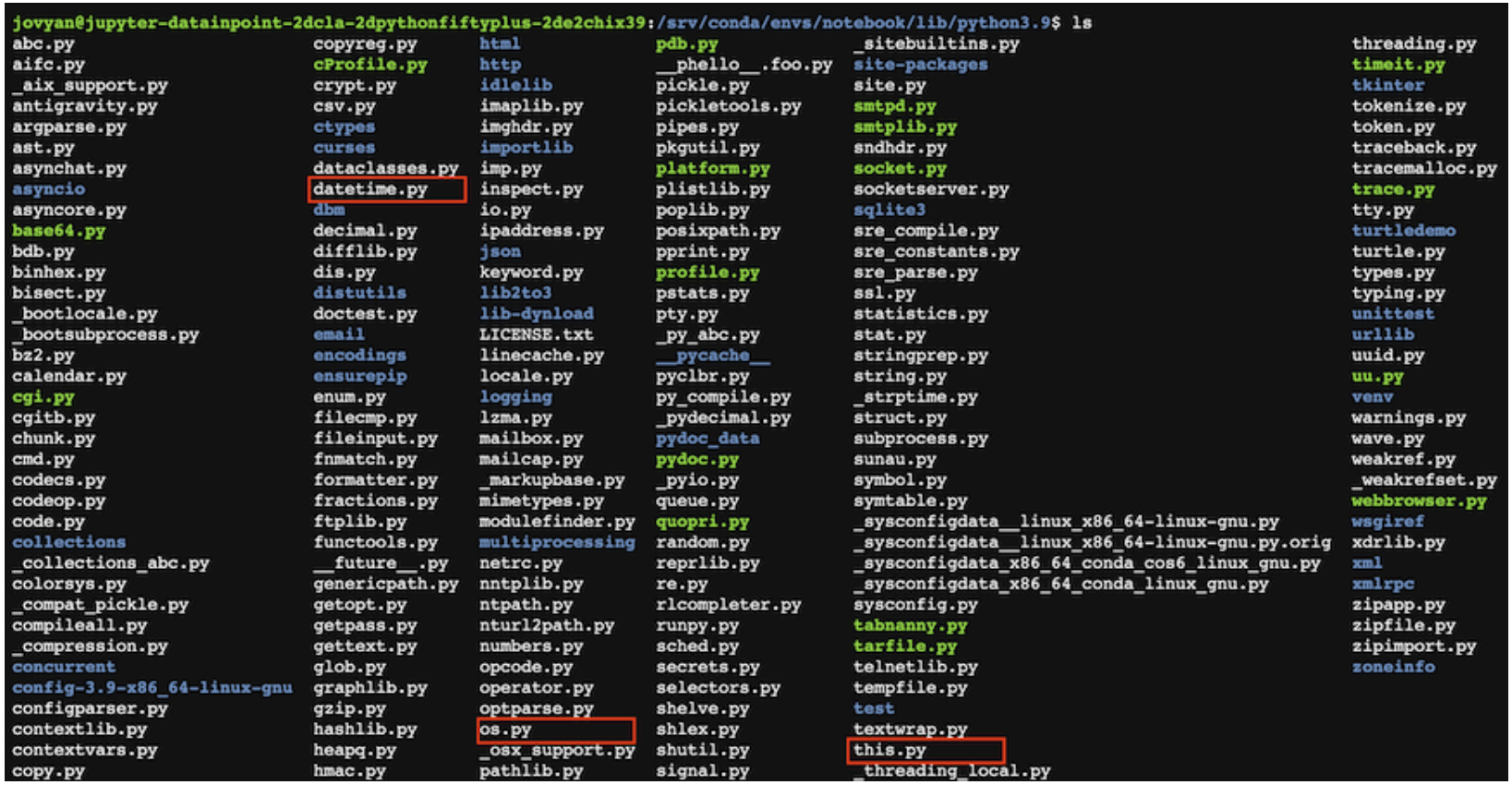
兩種路徑的標註方式

1. 絕對路徑：從根目錄開始標註，例如
   `/home/jovyan/mymodules/simplecalculator.py`
2. 相對路徑：從工作目錄（Current working directory）開始標註，例如
   `mymodules/simplecalculator.py`

標準模組的安裝路徑

- 我們課程所使用的 JupyterLab 標準模組安裝在 `/srv/conda/envs/notebook/lib/python3.9`
- 點選上方選單的 File -> New -> Terminal 開啟終端機。
- 輸入 `cd /srv/conda/envs/notebook/lib/python3.9` 後按下 Enter
- 輸入 `ls` 後按下 Enter

可以看到 `datetime.py`、`os.py`、`this.py`



自行定義模組

如何定義檔案形式的模組

- 將函數、類別撰寫在副檔名為 `.py` 的檔案中。
- 檔案名稱就是模組命名，其中的函數、類別都以 `module.function()` 或 `module.class` 來參照命名。

在 `/home/jovyan` 新增 `simplecalculator.py`

In [ ]:

```python
# simplecalculator.py
class SimpleCalculator:
    """
    This class creates a simple calculator that has 2 attributes and 1 method.
    """
    def __init__(self, a, b):
        self.a = a
        self.b = b
    def __repr__(self):
        out = "a: {}, b: {}".format(self.a, self.b)
        return out
    def add(self):
        out = self.a + self.b
        return out
```

在 `/home/jovyan` 新增一個空白筆記本看看是否能載入自行定義模組 `simplecalculator`

```
import simplecalculator

simple_calculator = simplecalculator.SimpleCalculator(55, 66)
simple_calculator.add()
```

如何定義資料夾形式的模組

- 將函數、類別撰寫在副檔名為 `.py` 的檔案中。
- 將副檔名為 `.py` 的檔案存在資料夾中。
- 資料夾名稱就是模組命名，其中的函數、類別都以 `library.module.function()` 或 `library.module.class` 來參照命名。

在 `/home/jovyan` 新增 `mymodules/simplecalculator.py`

In [ ]:

```python
# simplecalculator.py
class SimpleCalculator:
    """
    This class creates a simple calculator that has 2 attributes and 1 method.
    """
    def __init__(self, a, b):
        self.a = a
        self.b = b
    def __repr__(self):
        out = "a: {}, b: {}".format(self.a, self.b)
        return out
    def add(self):
        out = self.a + self.b
        return out
```

在 `/home/jovyan` 新增一個空白的筆記本看看是否能載入自行定義模組 `mymodules` 中的 `simplecalculator`

```
from mymodules import simplecalculator

simple_calculator = simplecalculator.SimpleCalculator(55, 66)
simple_calculator.add()
```

重點統整

- 模組（Module）指的是以檔案或資料夾形式，來組織 Python 的函數以及類別。
- 模組有三個來源
    - 標準模組。
    - 來自使用者的定義。
    - 第三方模組。

重點統整（續）

- 使用 `import`、`from` 與 `as` 保留字載入模組全部的函數、類別或者特定函數、類別。
- 依照模組說明文件（Documentation）中的範例決定載入形式。
- 定義檔案形式的模組：將函數、類別撰寫在副檔名為 `.py` 的檔案中。
- 定義資料夾形式的模組：將函數、類別撰寫在副檔名為 `.py` 的檔案中並將該檔案存在資料夾中。