**Python程式設計：模組**

**目錄**

[1. 基本概念 1](#_Toc207547267)

[2. 建立模組檔案 1](#_Toc207547268)

[3. 在其他程式中應用自己建立的函式 2](#_Toc207547269)

[3.1 方法一：導入整個模組 import 模組名稱 2](#_Toc207547270)

[3.2 方法二：從模組中導入特定函式 from ... import ... 2](#_Toc207547271)

[3.3 方法三：使用 as 給模組或函式取一個「別名」 2](#_Toc207547272)

# 基本概念

想像一下，您正在寫一個越來越大的程式。起初您把所有的功能、類別都寫在同一個 .py 檔案裡，但很快這個檔案就變得數千行長，難以閱讀、修改和維護。模組 (Module) 就是為了解決這個問題而誕生的。

在 Python 中，任何一個 .py 檔案都可以被視為一個模組。將相關的函式 (functions)、類別 (classes) 和變數組織在一個獨立的檔案中，以達到程式碼的組織化與重複使用 (reuse)。另外，下面所教的引入方式，是適用於module跟class歐

一個模組就像一個「工具箱」。您可以建立一個專門放「美食相關」工具的 makefood.py 工具箱，再建立一個專門放「銀行相關」工具的 banks.py 工具箱。當您需要用到某個工具時，只需要「匯入 (import)」對應的工具箱即可，而不需要把所有工具都散落在同一個地方。

# 建立模組檔案

我們來建立一個名為 makefood.py 的檔案。這個檔案本身就是我們的模組。

# makefood.py

def make\_icecream(toppings):

    """ 列出製作冰淇淋的配料 """

    print("製作冰淇淋的配料如下：")

    for topping in toppings:

        print("--- ", topping)

# 定義製作飲料的函式

def make\_drink(size, drink):

    """ 輸入飲料規格與種類，然後輸出飲料 """

    print("--- 客製化飲料 ---")

    print("尺寸：", size)

    print("飲料：", drink)

# 在其他程式中應用自己建立的函式

## 方法一：導入整個模組 import 模組名稱

├── makefood.py

└── main.py

# main.py

import makefood

# 呼叫函式時，必須在前面加上 "模組名稱."

makefood.make\_icecream(['草莓醬', 'OREO餅乾', '巧克力碎片'])

makefood.make\_drink('large', 'coke')

這種方式的好處是，程式碼非常清楚，任何人一看就知道 make\_icecream 這個函式是來自 makefood 模組，不會搞混。

## 方法二：從模組中導入特定函式 from ... import ...

# main.py

from makefood import make\_icecream, make\_drink

# 因為已經明確導入，可以直接呼叫函式名稱

make\_icecream(['草莓醬', 'OREO餅乾', '巧克力碎片'])

make\_drink('large', 'coke')

## 方法三：使用 as 給模組或函式取一個「別名」

有時候模組名稱太長，或是導入的函式名稱與現有程式碼衝突，我們可以用 as 來取一個好記又簡短的別名。

* 語法 (模組別名)：import 模組名稱 as 別名
* 語法 (函式別名)：from 模組名稱 import 函式名稱 as 別名

# main.py

import makefood as m  # 將 makefood 取一個簡短的別名 m

m.make\_icecream(['草莓醬'])

m.make\_drink('small', 'tea')