【Python】import 用法

在 Python 中，`import` 是用來引入其他模組、套件或特定功能的關鍵字，讓你可以重複利用現有的程式碼，避免重複撰寫功能。以下是 `import` 的詳細說明及常見用法：

---

1. `import` 的基本概念

`import` 用於引入一個 Python 模組或套件，讓你可以使用其中的函數、類別或變數。

模組與套件

- **模組**：是一個 Python 檔案（以 `.py` 結尾），其中包含定義的函數、類別或變數。
  - 範例：`math` 模組、你自己撰寫的 `mymodule.py`。
- **套件**：是一個包含多個模組的目錄，且該目錄下有 `__init__.py` 文件。
  - 範例：`os` 套件、`requests` 套件。

---

2. `import` 的使用方式

2.1 匯入整個模組

- 語法：

```
import module_name
```

- 範例：

```
import math

print(math.sqrt(16))  # 使用 math 模組中的 sqrt 函數
```

- 特點：需要使用 `模組名稱.功能` 的方式來存取模組內容。

2.2 匯入模組的特定部分

- 語法：

```
from module_name import specific_function_or_variable
```

- 範例：

```
from math import sqrt

print(sqrt(16))  # 直接使用 sqrt 函數，無需加上模組名稱
```

- 特點：只匯入需要的部分，節省記憶體，但可能引發命名衝突。

2.3 匯入模組並重新命名

- 語法：

```
import module_name as alias
```

- 範例：

```
import numpy as np

arr = np.array([1, 2, 3])
print(arr)
```

- 特點：透過別名縮短模組名稱，讓程式碼更簡潔。

2.4 匯入模組所有內容

- 語法：

```
from module_name import *
```

- 範例：

```
from math import *

print(sqrt(16)) # 可以直接使用 math 中的所有功能
```

- 特點：
  - 匯入所有內容，但不推薦，因為可能導致命名衝突。
  - 建議明確列出需要匯入的內容（用 `__all__` 控制）。

2.5 匯入套件中的子模組

- 語法：

```
from package_name import submodule_name
```

- 範例：

```
from os import path

print(path.exists("example.txt")) # 使用 os.path 模組中的 exists 函數
```

---

3. Python 搜尋模組的順序

當執行 `import` 時，Python 按照以下順序尋找模組：

1. **內建模組**：Python 標準庫中的模組，例如 `math`、`os`。
2. **當前目錄**：程式執行時所在的目錄。
3. **PYTHONPATH**：環境變數中指定的路徑。
4. **全域安裝的目錄**：例如 `site-packages`。

如果找不到模組，會拋出 `ModuleNotFoundError`。

---

4. 常見的 import 模式比較

匯入方式	使用方式	優點	缺點
`import module_name`	`module_name.function()`	清楚來源，避免命名衝突	使用時需要加上模組名稱
`from module_name import func`	`func()`	使用簡單，僅匯入需要的內容	可能導致命名衝突
`import module_name as alias`	`alias.function()`	模組名稱簡潔，程式碼更易閱讀	增加了別名學習的成本
`from module_name import *`	`function()`	簡單直接，適用於了解所有內容的情況	容易命名衝突，降低可讀性

---

5. 自訂模組的匯入

5.1 自訂模組

假設你有一個名為 `mymodule.py` 的檔案，內容如下：

```
# mymodule.py
def greet(name):
    return f"Hello, {name}!"
```

你可以在同目錄下使用：

```
import mymodule

print(mymodule.greet("Alice"))  # 輸出: Hello, Alice!
```

5.2 結構化套件

假設你有以下檔案結構：

```
myproject/
├── main.py
└── mypackage/
    ├── __init__.py
    ├── module1.py
    └── module2.py
```

`mypackage/__init__.py` **的內容：**

```
from .module1 import func1
from .module2 import func2

__all__ = ["func1", "func2"]
```

使用：

```
from mypackage import func1

func1()
```

---

6. 結論

- `import` **是 Python 中的核心功能，用於模組的重複利用。**
- **掌握不同的匯入方式，可以讓你的程式碼更高效、更清晰。**
- **盡量避免使用** `from module import *`**，選擇明確的匯入方式，提升程式碼可讀性與維護性。**

🕓修訂版本 #1
★由 treeman 建立於 23 ﷺ 2025 11:38:46
🖉由 treeman 更新於 23 ﷺ 2025 11:39:21