

商管程式設計 109-1

Coding Style Guide

Agenda

- 簡介
- 規範
 - 註解
 - 空格與空行
 - 命名
 - 雜項
- 檢查與修正

簡介

Python's Official Coding Style Guide

PEP 8 (link)

PEP 8 是做甚麼用的？

- 一個 Python 通用、統一的 coding style 規範
 - 哪裡可以空格，哪裡不應該空格...
 - 就像是寫國文作文，標題空四格，每一段開頭空兩格
- 通用、統一的格式**非常**有助於團隊開發
 - 或者讓你一個月之後還看得懂你自己寫了甚麼
 - 現在養成好習慣，期末報告就不會很痛苦；每個人習慣不同，格式規範就是用來統一大家的習慣的

註解

註解永遠是最重要的

如何寫好註解

- 把程式碼在做甚麼事情，**正確的**交代清楚
 - 尤其是特殊狀況，一定要補充說明
 - 修改 code 的時候，記得把動到的地方的註解一併修改
- **不要寫廢話**
 - 註解不是把程式碼用中文重寫一次

```
for i in range(1, x + 1): # 1 到 x (廢話)
for i in range(1, x + 1): # 編號從 1 開始 (有意義)
```

Comment Example

```
# 這是一段程式碼，只是示範用的。  
# 井字號後要空一格。  
  
#  
# 大家要努力寫程式喔！  
  
a = input() # 井字號前若有其他 code  
b = input() # 就空兩格，以利分辨  
  
c = a/b * 100 # 轉換成百分比  
print(str(c) + '%')
```

空格與空行

縮排與換行

- 標準縮排為四個空格
- 行尾不要多出空格

```
# 一行塞不下時，下一行跟著括號垂直對齊
fib_li = [1, 1, 2, 3, 5, 8, 13, 21,
           34, 55]
my_2d_li = [[1, 2, 3, 4],
             [2, 2, 3, 4],
             [3, 2, 3, 4]]
foo = my_func(var_one, var_two,
               var_three, var_four)
```

```
# 如果覺得對齊括號太浪費空間的話，
# 可以直接全部放在下一行 + 比後面行
# 再多一層縮排進去
foo = long_function_name(
        var_one, var_two,
        var_three, var_four)
bar = 0

def very_very_long_function_name(
        var_one, var_two, var_three,
        var_four):
    print(var_one)
```

縮排

- 標準縮排單位為四個空格

- Sublime 設定：

```
"translate_tabs_to_spaces": true,  
"tab_size": 4
```

- Notepad++ 設定：

偏好設定 程式語言 縮排設定

空行

- 分隔程式邏輯：空一行
- 空行不要有空格 (含 docstring)
 - 容易破壞結構 (尤其是複製貼上的時候)
 - Sublime 設定：trim_trailing_white_space_on_save
 - Notepad++ 設定：在「巨集」裡面

```
for i in range(3):
    if x == 0:
        print(123)
(1)      print(456)
(1)
a_str = input()
```

空格排版

- 等號、比較運算子、邏輯運算子的前後
 - =, [+=, -=, *=,], ==, <, >, !=, <>, <=, >=, in, not in, is, is not, and, or, not
- 運算子(加減乘除等):
 - 同一串中，優先權**最低**的要有空格
 - Ex. 加減和乘除同時出現時，加減要空格，乘除不用

```
my_int = 2*3 + 4*5 - 6/2  
c = (a+b) * (a-b)
```

```
my_int = (1+2) * (1+2 * 3+4)      # NO  
my_int = (1+2) * (1 + 2*3 + 4)    # OK  
my_int = (1+2) * (1+2*3+4)        # OK
```

空格與空行排版

- 不要在行內用連續空格排版
 - 例外：註解對齊時，可以有額外空格
 - 記得還是要空至少兩格
- 適當的利用「空一行」來分段

```
my_int █ = 123 # Bad
```

```
my_int_1 = 123456 # Good  
my_int_2 = 789 █ # Good
```

命名

命名規則

- 變數 : `lower_case_with_underscores`
 - 常數 : `UPPER_CASE_WITH_UNDERSCORES`
- 不管取甚麼名字，總之盡量要能從文字上就看得懂它是甚麼

雜項

其他 (Week 3 進度)

- 檢查 True / False 的時候，不用再 == True

```
# Bad  
if a+b==2 is True:
```

```
# Good  
if a+b == 2:
```

```
# Bad  
if my_bool == False:
```

```
# Good  
if not my_bool:
```

其他 (Week 4 進度)

- 檢查 string 或 list 的開頭或結尾是否符合某字串的時候，不要用 `string[a:]` 或 `string[:b]`，請用 `.startswith()` 或 `.endswith()`
- 當你的 string 長度不足的時候，才不會出現 error

```
# Bad
if my_str[:3] == 'abc':
    return True
```

```
# Good
if my_str.startswith('abc'):
    return True
```

檢查與修正

自動檢查是否符合 PEP 8

- 線上工具
 - Ex. pep8online.com
- pycodestyle (以下在 terminal 直接執行)

```
> pip3 install pycodestyle  
  
> pycodestyle test.py  
test.py:68:11: E221 multiple spaces before operator
```