遵循**PEP 8** 风格指南

《Python Enhancement Proposal #8》（8 号Python 增强提案）又叫PEP 8，它是针对Python 代码格式而编订的风格指南。尽管可以在保证语法正确的前提下随意编写Python 代码，但是，采用一致的风格来书写可以令代码更加易懂、更加易读。采用和其他Python 程序员相同的风格来写代码，也可以使项目更利于多人协作。即便代码只会由你自己阅读，遵循这套风格也依然可以令后续的修改变得容易一些。

PEP 8 列出了许多细节，以描述如何撰写清晰的Python 代码。它会随着Python 语言持续更新。大家应该把整份指南都读一遍（http://www.python.org/dev/peps/pep-0008）。下面列出几条绝对应该遵守的规则。

**空白**：Python 中的空白（whitespace）会影响代码的含义。Python 程序员使用空白的时候尤其在意，因为它们还会影响代码的清晰程度。

1. 使用 space（空格）来表示缩进，而不要用 tab（制表符）。
2. 和语法相关的每一层缩进都用 4 个空格来表示。
3. 每行的字符数不应超过 79。
4. 对于占据多行的长表达式来说，除了首行之外的其余各行都应该在通常的缩进 级别之上再加4 个空格。
5. 文件中的函数与类之间应该用两个空行隔开。
6. 在同一个类中，各方法之间应该用一个空行隔开。
7. 在使用下标来获取列表元素、调用函数或给关键字参数赋值的时候，不要在两 旁添加空格。
8. 为变量赋值的时候，赋值符号的左侧和右侧应该各自写上一个空格，而且只写 一个就好。

**命名**：PEP 8 提倡采用不同的命名风格来编写Python 代码中的各个部分，以便在阅读代码时可以根据这些名称看出它们在Python 语言中的角色。

1. 函数、变量及属性应该用小写字母来拼写，各单词之间以下划线相连，例如， lowercase\_underscore。
2. 受保护的实例属性，应该以单个下划线开头，例如，\_leading\_underscore。
3. 私有的实例属性，应该以两个下划线开头，例如，\_\_double\_leading\_underscore。
4. 类与异常，应该以每个单词首字母均大写的形式来命名，例如，CapitalizedWord。
5. 模块级别的常量，应该全部采用大写字母来拼写，各单词之间以下划线相连， 例如，ALL\_CAPS。
6. 类中的实例方法（instance method），应该把首个参数命名为self，以表示该对象自身。
7. 类方法（class method）的首个参数，应该命名为 cls，以表示该类自身。

**表达式和语句**：《The Zen of Python》（Python 之禅）中说：“每件事都应该有直白的做法，而且最好只有一种。”PEP 8 在制定表达式和语句的风格时，就试着体现了这种思想。

1. 采用内联形式的否定词，而不要把否定词放在整个表达式的前面，例如，应该 写if a is not b 而不是if not a is b。
2. 不要通过检测长度的办法（如 if len(somelist) == 0）来判断 somelist 是否为 [] 或'' 等空值，而是应该采用if not somelist 这种写法来判断，它会假定：空值将自动评估为False。
3. 检测 somelist 是否为 [1] 或 'hi' 等非空值时，也应如此，if somelist 语句默认会把 非空的值判断为True。
4. 不要编写单行的 if 语句、for 循环、while 循环及 except 复合语句，而是应该把 这些语句分成多行来书写，以示清晰。
5. import 语句应该总是放在文件开头。
6. 引入模块的时候，总是应该使用绝对名称，而不应该根据当前模块的路径来 使用相对名称。例如，引入bar 包中的foo 模块时，应该完整地写出from bar import foo，而不应该简写为import foo。
7. 如果一定要以相对名称来编写import语句，那就采用明确的写法：from.import foo。
8. 文件中的那些 import 语句应该按顺序划分成三个部分，分别表示标准库模块、 第三方模块以及自用模块。在每一部分之中，各import 语句应该按模块的字母顺序来排列。

**要点**

1. 当编写 Python 代码时，总是应该遵循 PEP 8 风格指南。
2. 与广大 Python 开发者采用同一套代码风格，可以使项目更利于多人协作。
3. 采用一致的风格来编写代码，可以令后续的修改工作变得更为容易。