**python语言代码规范**

# **一、简明概述**

## 1、编码

如无特殊情况, 文件一律使用 UTF-8 编码如无特殊情况, 文件头部必须加入#--coding:utf-8--标识

## 2、代码格式

### 2.1、缩进

统一使用 4 个空格进行缩进

### 2.2、行宽

每行代码尽量不超过 80 个字符(在特殊情况下可以略微超过 80 ，但最长不得超过 120)

理由：

这在查看 side-by-side 的 diff 时很有帮助方便在控制台下查看代码太长可能是设计有缺陷

### 2.3、引号

简单说，自然语言使用双引号，机器标示使用单引号，因此 代码里 多数应该使用 单引号

自然语言 使用双引号 "..."  
例如错误信息；很多情况还是 unicode，使用u"你好世界"机器标识 使用单引号 '...'  
例如 dict 里的 key正则表达式 使用原生的双引号 r"..."文档字符串 (docstring) 使用三个双引号 """......"""

### 2.4、空行

模块级函数和类定义之间空两行；类成员函数之间空一行；

class A:

def \_\_init\_\_(self):

pass

def hello(self):

pass

def main():

pass

可以使用多个空行分隔多组相关的函数函数中可以使用空行分隔出逻辑相关的代码

### 2.5、编码

文件使用 UTF-8 编码文件头部加入#-\*-conding:utf-8-\*-标识

## 3、import 语句

import 语句应该分行书写

# 正确的写法import osimport sys

# 不推荐的写法import sys,os

# 正确的写法from subprocess import Popen, PIPE

import语句应该使用 absolute import

# 正确的写法from foo.bar import Bar

# 不推荐的写法from ..bar import Bar

import语句应该放在文件头部，置于模块说明及docstring之后，于全局变量之前；import语句应该按照顺序排列，每组之间用一个空行分隔

import osimport sys

import msgpackimport zmq

import foo

导入其他模块的类定义时，可以使用相对导入

from myclass import MyClass

如果发生命名冲突，则可使用命名空间

import bar

import foo.bar

bar.Bar()

foo.bar.Bar()

## 4、空格

在二元运算符两边各空一格[=,-,+=,==,>,in,is not, and]:

# 正确的写法

i = i + 1

submitted += 1

x = x \* 2 - 1

hypot2 = x \* x + y \* y

c = (a + b) \* (a - b)

# 不推荐的写法

i=i+1

submitted +=1

x = x\*2 - 1

hypot2 = x\*x + y\*y

c = (a+b) \* (a-b)

函数的参数列表中，,之后要有空格

# 正确的写法def complex(real, imag):

pass

# 不推荐的写法def complex(real,imag):

pass

函数的参数列表中，默认值等号两边不要添加空格

# 正确的写法def complex(real, imag=0.0):

pass

# 不推荐的写法def complex(real, imag = 0.0):

pass

左括号之后，右括号之前不要加多余的空格

# 正确的写法

spam(ham[1], {eggs: 2})

# 不推荐的写法

spam( ham[1], { eggs : 2 } )

字典对象的左括号之前不要多余的空格

# 正确的写法

dict['key'] = list[index]

# 不推荐的写法

dict ['key'] = list [index]

不要为对齐赋值语句而使用的额外空格

# 正确的写法

x = 1

y = 2

long\_variable = 3

# 不推荐的写法

x = 1

y = 2

long\_variable = 3

## 5、换行

Python 支持括号内的换行。这时有两种情况。

1. 第二行缩进到括号的起始处

foo = long\_function\_name(var\_one, var\_two,

var\_three, var\_four)

1. 第二行缩进 4 个空格，适用于起始括号就换行的情形

def long\_function\_name(

var\_one, var\_two, var\_three,

var\_four):

print(var\_one)

使用反斜杠换行，二元运算符+.等应出现在行末；长字符串也可以用此法换行

session.query(MyTable).\

filter\_by(id=1).\

one()

print 'Hello, '\

'%s %s!' %\

('Harry', 'Potter')

禁止复合语句，即一行中包含多个语句：

# 正确的写法

do\_first()

do\_second()

do\_third()

# 不推荐的写法

do\_first();do\_second();do\_third();

if/for/while一定要换行：

# 正确的写法

if foo == 'blah':

do\_blah\_thing()

# 不推荐的写法

if foo == 'blah': do\_blash\_thing()

### 6、docstring

docstring 的规范中最其本的两点：

所有的公共模块、函数、类、方法，都应该写 docstring 。私有方法不一定需要，但应该在 def 后提供一个块注释来说明。docstring 的结束"""应该独占一行，除非此 docstring 只有一行。

"""Return a foobar

Optional plotz says to frobnicate the bizbaz first.

"""

"""Oneline docstring"""

# 二、注释

## 1、注释

### 1.1、块注释

“#”号后空一格，段落件用空行分开（同样需要“#”号）

# 块注释

# 块注释

#

# 块注释

# 块注释

### 1.2、行注释

至少使用两个空格和语句分开，注意不要使用无意义的注释

# 正确的写法

x = x + 1 # 边框加粗一个像素

# 不推荐的写法(无意义的注释)

x = x + 1 # x加1

### 1.3、建议

在代码的关键部分(或比较复杂的地方), 能写注释的要尽量写注释比较重要的注释段, 使用多个等号隔开, 可以更加醒目, 突出重要性

app = create\_app(name, options)

# =====================================

# 请勿在此处添加 get post等app路由行为 !!!

# =====================================

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

## 2、文档注释（Docstring）

作为文档的Docstring一般出现在模块头部、函数和类的头部，这样在python中可以通过对象的**doc**对象获取文档.  
编辑器和IDE也可以根据Docstring给出自动提示.

文档注释以 """ 开头和结尾, 首行不换行, 如有多行, 末行必需换行, 以下是Google的docstring风格示例

# -\*- coding: utf-8 -\*-"""Example docstrings.

This module demonstrates documentation as specified by the `Google Python

Style Guide`\_. Docstrings may extend over multiple lines. Sections are created

with a section header and a colon followed by a block of indented text.

Example:

Examples can be given using either the ``Example`` or ``Examples``

sections. Sections support any reStructuredText formatting, including

literal blocks::

$ python example\_google.py

Section breaks are created by resuming unindented text. Section breaks

are also implicitly created anytime a new section starts.

"""

不要在文档注释复制函数定义原型, 而是具体描述其具体内容, 解释具体参数和返回值等

# 不推荐的写法(不要写函数原型等废话)def function(a, b):

"""function(a, b) -> list"""

... ...

# 正确的写法def function(a, b):

"""计算并返回a到b范围内数据的平均值"""

... ...

对函数参数、返回值等的说明采用numpy标准, 如下所示

def func(arg1, arg2):

"""在这里写函数的一句话总结(如: 计算平均值).

这里是具体描述.

参数

----------

arg1 : int

arg1的具体描述

arg2 : int

arg2的具体描述

返回值

-------

int

返回值的具体描述

参看

--------

otherfunc : 其它关联函数等...

示例

--------

示例使用doctest格式, 在`>>>`后的代码可以被文档测试工具作为测试用例自动运行

>>> a=[1,2,3]

>>> print [x + 3 for x in a]

[4, 5, 6]

"""

文档注释不限于中英文, 但不要中英文混用文档注释不是越长越好, 通常一两句话能把情况说清楚即可模块、公有类、公有方法, 能写文档注释的, 应该尽量写文档注释

# 三、命名规范

## 1、模块

模块尽量使用小写命名，首字母保持小写，尽量不要用下划线(除非多个单词，且数量不多的情况)

# 正确的模块名import decoderimport html\_parser

# 不推荐的模块名import Decoder

## 2、类名

类名使用驼峰(CamelCase)命名风格，首字母大写，私有类可用一个下划线开头

class Farm():

pass

class AnimalFarm(Farm):

pass

class \_PrivateFarm(Farm):

pass

将相关的类和顶级函数放在同一个模块里. 不像Java, 没必要限制一个类一个模块.

## 3、函数

函数名一律小写，如有多个单词，用下划线隔开

def run():

pass

def run\_with\_env():

pass

私有函数在函数前加一个下划线\_

class Person():

def \_private\_func():

pass

## 4、变量名

变量名尽量小写, 如有多个单词，用下划线隔开

if \_\_name\_\_ == '\_\_main\_\_':

count = 0

school\_name = ''

常量采用全大写，如有多个单词，使用下划线隔开

MAX\_CLIENT = 100

MAX\_CONNECTION = 1000

CONNECTION\_TIMEOUT = 600

### 5、常量

常量使用以下划线分隔的大写命名

MAX\_OVERFLOW = 100

Class FooBar:

def foo\_bar(self, print\_):

print(print\_)