Python Style Guide

Yan Zeng, Version 1.0, last updated: 2019-11-23

# Abstract

Notes on Python style guide, based on [1, pp. 218-221], [2] and [3].

# Useful Functions and Attributes

* **dir**: it lists all the attributes of an object (and therefore all functions, classes, variables, and so on of a module) [1, p. 218]
* **help**: it displays docstring among many other things. [1, pp. 219-220]
* **\_\_all\_\_**: it defines the public interface of a module so that if you use “from [module name] import \*”, you get only the functions listed in the \_\_all\_\_ variable. [1, p. 219]
* **\_\_doc\_**: docstring of a Python object. [1, p. 220]

# Docstring

Always use the three double-quote “”” format for docstrings. A docstring should be organized as a summary line (one physical line) terminated by a period, question mark, or exclamation point, followed by a blank line, followed by the rest of the docstring starting at the same cursor position as the first quote of the first. [2] 3.8.1.

# Imports

Use import statements for packages and modules only, not for individual classes or functions. Import each module using the full pathname location of the module. [2] 2.2, 2.3.

Imports should be on separate lines. Imports are always put at the top of the file, just after any module comments and docstrings and before module global and constants. Imports should be grouped from most generic to least generic. [2] 3.13

# Module

Files should start with a docstring describing the contents and usage of the module. [2] 3.8.2.

Module level “dunders” such as \_\_all\_\_, \_\_author\_\_, \_\_version\_\_, etc. should be placed after the module docstring but before any import statements except “from \_\_future\_\_” imports. [3]

# Class

Classes should have a docstring below the class definition describing the class. If your class has public attributes, they should be documented here in an Attribute section and follow the same formatting as a function’s Arg section. [2] 3.8.4.

# Function

Docstring should contain the following attributes when applicable: [2] 3.8.3.

* **Args**: List each parameter by name. A description should follow the name and be separated by a colon and a space. The description should include required type(s) if the code does not contain a corresponding type annotation.
* **Returns**: Describe the type and semantics of the return value.
* **Raises**: List all exceptions that are relevant to the interface.

# Guidelines derived from Guido’s recommendations

A screenshot of a cell phone

Description automatically generated