Cheatography

Python PEP8 style guide Cheat Sheet

by jmds via cheatography.com/84942/cs/20012/

Naming conventions

Never use l, O, or I single letter names as these can be mistaken for 1 and 0, depending on typeface	O = 2 # This may look like you're trying to reassign 2 to zero
Function	function, my_function
Variable	x, var, my_variable
Class	Model, MyClass
Method	class_method, method
Constant	CONSTANT, MY_CONSTANT, MY_LONG_CONSTANT
Module	module.py, my_module.py
Package	package, mypackage

Maximum Line Length and Line Breaking

PEP 8 suggests lines should be limited to 79 characters. This is because it allows you to have multiple files open next to one another, while also avoiding line wrapping.

Python will assume line contin uation if code is contained within parent heses, brackets, or braces:

```
def functi on( arg _one, arg_two,
                          arg _three, arg_four):
        return arg_one
```

If it is impossible to use implied contin uation, then you can use backsl ashes to break lines instead:

```
from mypkg import example1, \
        exa mple2, example3
# Recomm ended
total = (first _va riable
                + second _va riable
                - third_ var iable)
# Not Recomm ended
total = (first _va riable +
                  sec ond _va riable -
                  thi rd_ var iable)
```

Identation

Use 4 consecutive spaces to indicate indentation.

Prefer spaces over tabs.

Comments

Limit the line length of comments and docstrings to 72 characters.	Use complete sentences, starting with a capital letter.	Make sure to update comments if you change your code.

Block Comments

Indent block comments to the same level as the code they describe.	Start each line with a # followed by a single space.	Separate paragraphs by a line containing a single #.

Inline Comments

Use inline comments sparingly.

Write inline comments on the same line as the statement they refer to.

Separate inline comments by two or more spaces from the statement.

Start inline comments with a # and a single space, like block comments.

Don’t use them to explain the obvious.

When to Avoid Adding Whitespace

The most important place to avoid adding whitespace is at the end of a line. This is known as trailing whitespace

Immediately inside parentheses, brackets, or braces:

Before a comma, semicolon, or colon:

Before the open parenthesis that starts the argument list of a function call:

Before the open bracket that starts an index or slice:

Between a trailing comma and a closing parenthesis:

To align assignment operators:

immediately inside brackets, as well as before commas and colons.

Programming Recommendations

Don’t compare boolean values to True or False using the equivalence operator.	Use the fact that empty sequences are falsy in if statements.
Use is not rather than not ... is in if statements.	Don’t use if x: when you mean if x is not None:

Use .startswith() and .endswith() instead of slicing.

Code Layout

Surround top-level functions and classes with two blank lines

Surround method definitions inside classes with a single blank line.

Use blank lines sparingly inside functions to show clear steps.

Indentation Following Line Breaks

There are two styles of indentation you can use.

The first of these is to align the indented block with the opening delimiter:

An alternative style of indentation following a line break is a hanging indent. This is a typographical term meaning that every line but the first in a paragraph or statement is indented.

Indentation Following Line Breaks 2

```
def function(arg_one, arg_two,
                          arg _three, arg_four):
        return arg_one
x = 5
if (x > 3 and
       x < 10):
        pri nt(x)
x = 5
if (x > 3 and
       x < 10):
       # Both conditions satisfied
        pri nt(x)
x = 5
if (x > 3 and
               x < 10):
        pri nt(x)
# hanging indent
var = function(
        arg _one, arg_two,
        arg _three, arg_four)
```

Where to Put the Closing Brace

PEP 8 provides two options for the position of the closing brace in implied line continuations:

1 - Line up the closing brace with the first non-wh ite space character of the previous line:

```
list_o f_n umbers = [
       1, 2, 3,
       4, 5, 6,
       7, 8, 9
```

Where to Put the Closing Brace (cont)

```
>    ]
```

2 - Line up the closing brace with the first character of the line that starts the construct:

```
list_of_numbers = [
   1, 2, 3,
   4, 5, 6,
   7, 8, 9
]
```

Documentation Strings

Surround docstrings with three double quotes on either side, as in """This is a docstring""".

Write them for all public modules, functions, classes, and methods.

Put the """ that ends a multiline docstring on a line by itself:

For one-line docstrings, keep the """ on the same line:

Whitespace Around Binary Operators

Surround the following binary operators with a single space on either side:

Assignment operators (=, +=, -=, and so forth)	Comparisons (==, !=, >, <. >=, <=) and (is, is not, in, not in)	Booleans (and, not, or)
When = is used to assign a default value to a function argument, do not surround it with spaces.	def function(default_paramet-er=5):	
y = x**2 + 5	z = (x+y) * (x-y)	

```
if x>5 and x%2==0:
# Treat the colon as the operator with lowest priority
list[x+1 : x+2]
```

In an extended slice, both colons must be surrounded by the same amount of whitespace

list[3:4:5]	list[x+1 : x+2 : x+3]
The space is omitted if a slice parameter is omitted	list[x+1 : x+2 :]

Cheatography

Python PEP8 style guide Cheat Sheet

by jmds via cheatography.com/84942/cs/20012/

When to Ignore PEP 8

If complying with PEP 8 would break compatibility with existing software

If code surrounding what you're working on is inconsistent with PEP 8

If code needs to remain compatible with older versions of Python

By **jmds**

cheatography.com/jmds/

Not published yet.

Last updated 10th July, 2019.