**PEP 8 Python Style Guide (Clean & Short)**

**✅ Indentation**

* Use **4 spaces** per indentation level.
* ❌ Don’t use tabs.

**✅ Line Length**

* Limit all lines to **79 characters** (for code).
* Limit docstrings/comments to **72 characters**.

**✅ Blank Lines**

* 2 blank lines **between functions & classes** (top-level).
* 1 blank line **between methods inside a class**.
* No extra blank lines inside functions.

**✅ Imports**

* Import **one module per line**.
* Group imports in this order:
  1. Standard library
  2. Third-party
  3. Local app imports
* Example:

import os

import sys

import requests

from myapp import utils

**✅ Naming Conventions**

| **Type** | **Style** | **Example** |
| --- | --- | --- |
| Variables | snake\_case | user\_name |
| Functions | snake\_case | Calculate\_total() |
| Constants | UPPER\_CASE | MAX\_RETRIES |
| Classes | PascalCase | BankAccount |
| Modules/Packages | lowercase | utils.py |

**✅ Spacing**

* No spaces **inside parentheses/brackets/braces**:

func(a, b) # ✅ Good

func( a, b ) # ❌ Bad

* One space **after commas**:

x, y, z = 1, 2, 3 # ✅

* No space **before commas, colons, semicolons**:

if x == 4: print(x, y) # ✅

**✅ String Quotes**

* Use **consistent quotes** in your project:
  + Either 'single' or "double"
* Multi-line strings → triple quotes """ or '''

**✅ Comments**

* Use **# with one space** for inline comments:

# This is a comment

x = x + 1 # Increment x

* Keep comments **short & clear**.

**✅ Docstrings**

* Use **triple double-quotes """**:

def func():

"""Do something useful."""

pass

**✅ Operators**

* Put **spaces around operators**:

x = y + z # ✅ Good

x=y+z # ❌ Bad

* No spaces for keyword arguments:

func(x=1, y=2) # ✅

**✅ Misc**

* Avoid trailing whitespaces.
* End files with a newline.
* Use UTF-8 encoding.

**🏁 Example (PEP 8 Compliant)**

class BankAccount:

"""A simple bank account class."""

def \_\_init\_\_(self, name, balance=0):

self.name = name

self.balance = balance

def deposit(self, amount):

"""Add money to the account."""

self.balance += amount

def withdraw(self, amount):

"""Remove money from the account if funds are available."""

if amount > self.balance:

raise ValueError("Insufficient funds")

self.balance -= amount