Atul FYIT 96

PEP 8 Python

PEP 8 in Python | What is the purpose of PEP 8 in Python?

In this tutorial, we will learn what PEP-8 is and how we can use it in Python coding. We will discuss the guidilines for using PEP in Programming - this tutorial is aimed at beginners to intermediate. We will also discuss the benefits of using PEP 8 while Coding

What is PEP?

The PEP is an abbreviation form of Python Enterprise Proposal. Writing code with the proper logic is a key factor of Programming, but many other important factors can affect's the code's quality. The developers coding style makes the code much reliable, and very developer should keep in that Python Strictly follows the way of order and format of the String.

Adaptive a nice coding style makes the code more readable. The code becomes easy for end user

PEP 8 is a document that provides various guidelines to write the readable in Python. PEP 8 describes how the developer can write beautiful code. It was officially written in 2001 by Guido van Rossum Barry Warsaw, and Nick Coghlan. The main aim of PEP is to enhance the readability and consistency of code

Why PEP 8 is Important?

PEP 8 enhances the readability of the Python code but why is readability so Important? Let's understand the concept.

Creator of Python, Guido van Rossum said, "code is much more often than it is written" The code can be written in a few minutes, a few hours, or a whole day but once we have written the code, we will never to read the code again and again

At this point, we must have an idea of why we wrote the particular line in the code The code should reflect the meaning of each line That's why readability is so much important

We will describe few important guidelines for writing effective code that can be read by others as well

Naming Convention

When we write the code, we need to assign name to many thing such as variables functions, classes, packages and a lot more things, selecting a proper name will save time and energy. When can easily recall represents Developers should avoid choosing inappropriate names

The naming Convention in Python is slightly messy, but there are certain conventions that we can follow. Lets' see the following naming convention

Example ⇒ Single lowe case

```
a= 10
```

Single apper case letter

```
A= 10
```

Lowe case

```
var = 10
```

Lower_case_with under scores

```
number_of_apple = 5
```

upper case

var = 6

UPPER_CASE_WITH_UNDESCORES

NUM_OF_CARS = 20

Capitalized Words (or Camel Case)

Number of Books = 100

Name Style

Below is the table that specifies some of the common naming styles in Python Consider the following table

Type	Naming Convention	Examples
Function	We shoud use the lower case words or separates words by the under scores	my function my_function
variable	We should use a lower case letter, words, or separates words to enhance the readability	a, var variable_name

class	The first letter of class name should be capitalized; use camel case. Do not separate with the Underscore	My class, form Model
Constant	we should use a short uppercase letter, words, or separate words to enhance the readability	MY CONSTANT, CONSTANT MY_CONSTANT
Method	We should use a lowercase letter, words or separates words to enhance readability	class_method method
Module	We should use a lowecase letter, words, or seprates words to enhance the readability. Do not Seprate words with the under score	Package my package

Above are some common naming convention that are useful to beautify the python code. For additional improvement, we should choose the name carefully.

Code Layout

The code layout defines how much the code is readable In this section, we will learn how to use white space to improve code readability.

Identination

Unlike other programming languages, the indentation is used to define the code block in Python. The indentations are the important part of the python programming language and it determines the level of lines of code. Generally, we use the 4 space for indentation. Lets understand the following example

```
x=5
if x==5
Print ("x is larger than5')
```

In the above example, the indented print statement will get executed if the condition of if statement is true. This indetation defines the code block and and tells us

Atul FYIT-96

DATE

What statements execute when a function is called or condition trigger

Tabs vs. Space

We can also use the tabs to prade the consecutive spaces to indicate the indentation, but whitespaces are the most preferable. Python 2 allows the mixing of tabs and spaces but we will an erre in python 3.

Indentation following Line Break

It is essential to use indentation when using line continuations to keep the line to fewer than 79 characters. It provides the flexibility to determining between two lines of codes and a single line of code that extends two lines. Let's understand the following example

example

```
## correct way

## Aligned with opening delimiter
obj = func_name(argument_one, argument_two,
                argument_three, argument
    _four):
```

We can use the following structure

```
## first line doesn't has any argument
## we add4 spaces from the second line to
discrimin

def function_name(
        argument_one, argument_two argument
    three,
        argument_four):
    Print(argument_two)
## 4 space indentation to add a level
foo = long_function_name(
    var_one, var_two
    var_three, var_four)
```

use docstring

Python provides the two types of document string or docstring- single line and multiple lines. we use the triple quotes to define a single line or multiline quotes. Basically these are used to describe the function or particular Program, Lets understand the following example

```
def add(a, b):
    """This a simple add method"""

    """This is
    a

    Simple add program to add
    the two numbers"""
```

DATE

Should a Line Break Before or After a Binary operator?

The lines break befor or after a binary operation is a tradinational approach. But it affects the readability extensively because the operators are scattered across the different screens and each operator is kept away from its operand and onto the previous line Lets understand the following example

Example
wrong
operators sit for away from their operands

```
marks = (english_marks+
         math_marks
         (scine_marks- biology_marks)+
         physics_marks
```

As we can see in the example it seems quite messy to read. we can solve such types of problems by using the following structure

Example
Correct
easy to match operator with ~~example~~ operands

```
Total_marks = (English_marks
              +math_marks
              +(science_marks - biology_marks)
              + physics_marks
```

Python allows us to break line before or after a binary operator, as long as the convention is consistent locally.

Importing module
We should import the modules in the separates line as follows

```
import pygame
import os
import sys
```

wrong
```
import sys.os
```

we can also use the following approach.
```
from subprocess import Popen, PIPE
```

import statement should be written at the top of the file or just after any module comment Absolute Importsare the recommended because they are more readable and tend to be better behaved.

```
import my pkg.sibling
from my pkg import sibling
from my pkg.sibling import example
```