Fundamentos de Inteligência de Dados

Formação Power BI Analyst

Juliana Mascarenhas

Tech Education Specialist DIO / Owner @Simplificandoredes e @SimplificandoProgramação

Mestre em modelagem computacional | Cientista de dados

@in/juliana-mascarenhas-ds/

Objetivo Geral

Convenções de nomeação

Indentação

DocStrings

Overview sobre as boas práticas de programação em Python com o guia PEP 8.

Espaços em branco

Espaços em branco

Code Layout

Etapa 1

O que é a PEP 8? Qual sua importância?

// Integração com Python

PEP

- Julho 2000
- PEP – Python Enhancement Proposals
- Convenções de acordo com tópicos
- Possui controle de versão

Documentação PEP

Documentação PEP 8

PEP 8 – Guia de Estilo

Objetivo:

- Nomeação de classes e métodos
- Code Layout
- Indentação
- Docstrings - comentários
- ...

"Readability counts."

— *The Zen of Python*

PEP 8 – Legibilidade

O que eu fiz aqui mesmo?

“Code is read much more often than it is written.”

Guido van Rossum

PEP 8 – Legibilidade

- Time de desenvolvimento
- Trabalho em equipe
- Facilidade de entendimento
- Manutenção facilitada

Etapa 2

The Guide Line – Padronização com Python

// Integração com Python

PEP 8 - Nomeando

- Classes
- Variáveis
- Métodos
- Funções

```python
# imports no TOP

class Classe:
    pass

# functions


def soma():
    pass
```

```python
# two spaces before a python class


class ClasseComposta:
    def __init__(self):
        pass
    pass
```

Nomes com significado

“Explicit is better than implicit.”

— *The Zen of Python*

PEP 8 - Nomeando

Type	Naming Convention	Examples
Function	Use a lowercase word or words. Separate words by underscores to improve readability.	`function`, `my_function`
Variable	Use a lowercase single letter, word, or words. Separate words with underscores to improve readability.	`x`, `var`, `my_variable`
Class	Start each word with a capital letter. Do not separate words with underscores. This style is called camel case or pascal case.	`Model`, `MyClass`

PEP 8 - Nomeando

Method	Use a lowercase word or words. Separate words with underscores to improve readability.	`class_method`, `method`
Constant	Use an uppercase single letter, word, or words. Separate words with underscores to improve readability.	`CONSTANT`, `MY_CONSTANT`, `MY_LONG_CONSTANT`
Module	Use a short, lowercase word or words. Separate words with underscores to improve readability.	`module.py`, `my_module.py`
Package	Use a short, lowercase word or words. Do not separate words with underscores.	`package`, `mypackage`

PEP 8 - Nomeando

```
>>> # Not recommended
>>> x = 'John Smith'
>>> y, z = x.split()
>>> print(z, y, sep=', ') 'Smith, John'
```

```
>>> # Recommended
>>> name = 'John Smith'
>>> first_name, last_name = name.split()
>>> print(last_name, first_name, sep=', ') 'Smith, John'
```

PEP 8 - Nomeando

```
# Not recommended
def db(x):
    return x * 2
```

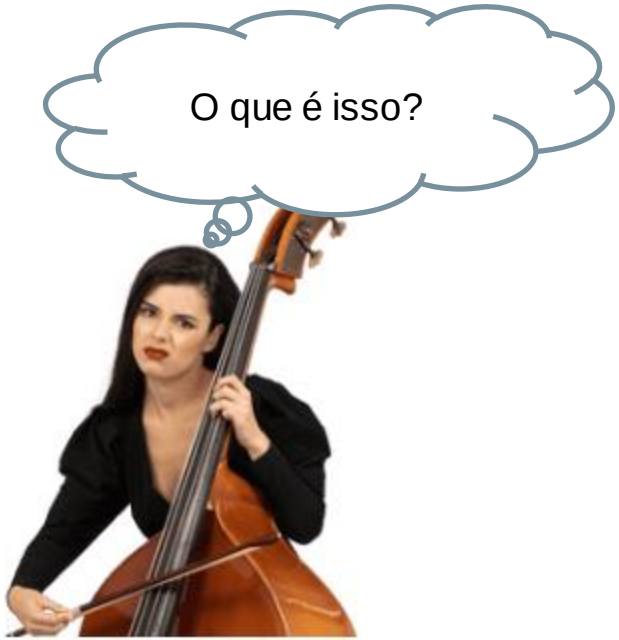

```
# Recommended
def multiply_by_two(x):
    return x * 2
```

Etapa 2

The Guide Line – Code Layout & Tópicos Relacionados

// Integração com Python

Code Layout




```python
class Classe:
    pass
# functions
def soma():
    pass
# two spaces before a python class or method
class ClasseComposta:
    def __init__(self):
        inicio = 'Bem-vindo'
        return self.inicio
    pass
def function_compost():
    pass
def function_sum(number_one, number_two):
    total = number_one + number_two
    return total
```

“Beautiful is better than ugly.”

— *The Zen of Python*

PEP 8 – Code Layout

```python
def function_compost():
    pass


def function_sum(number_one, number_two):
    total = number_one + number_two
    return total
```

```python
class Classe:
    pass

# functions


def soma():
    pass
```

```python
# two spaces before a python class or method


class ClasseComposta:
    def __init__(self):
        inicio = 'Bem-vindo'
        return self.inicio
    pass
```

“Beautiful is better than ugly.”

— *The Zen of Python*

PEP 8 – Code Layout

- Blank lines
- Espaçamento
- Tamanho máximo
- Quebra de linha

"Beautiful is better than ugly."
— *The Zen of Python*

PEP 8 - Indentação

Define o agrupamento das linhas de código e a lógica da aplicação

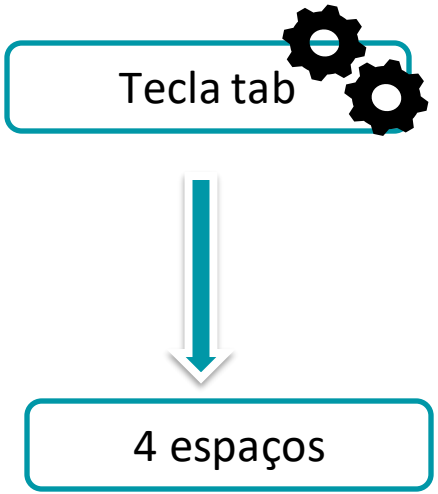
```
x = 3
if x > 5:
    print('x is larger than 5')
```

“There should be one—and preferably only one—obvious way to do it.”

— *The Zen of Python*

PEP 8 – Tab e Espaço

- Esqueça a tecla Tab
- 4 espaços



Pode misturar?

```
code.py: inconsistent use of tabs and spaces in indentation
```

PEP 8 – Tab e Espaço

Limite = 79 caracteres

- Quebra estruturas e métodos

```
def function(
        arg_one,
        arg_two,
        arg_three,
        arg_four):
    return arg_one
```

```
def function( arg_one, arg_two,
arg_three, arg_four):
    return arg_one
```

PEP 8 – Testando o Código

Shell

```
$ python2 -t code.py
code.py: inconsistent use of tabs and spaces in indentation
```

Shell

```
$ python2 -tt code.py
  File "code.py", line 3
    print(i, j)
              ^
TabError: inconsistent use of tabs and spaces in indentation
```

Fonte: https://realpython.com/python-pep8/#naming-conventions

PEP 8 – Quebrando linhas

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

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

PEP 8 – Quebrando linhas

```python
# Correct:
import os
import sys
```

```python
# Wrong:
import sys, os
```

It's okay to say this though:

```python
# Correct:
from subprocess import Popen, PIPE
```

Etapa 1

The Guide Layout - Comentários no código

// Integração com Python

Comentários no código

- Comentários tanto quanto necessário
- Objetivo: facilitar o entendimento
- Código bem escrito também é documentação

> “If the implementation is hard to explain, it’s a bad idea.”
>
> — *The Zen of Python*

Comentários no código

- Limite da linha = 79 caracteres
- Sentenças completas iniciando com letra maíscula
- Atualização de comentários

> “If the implementation is hard to explain, it’s a bad idea.”
>
> — *The Zen of Python*

Blocos de comentários

Python

```
for i in range(0, 10):
    # Loop over i ten times and print out the value of i, followed by a
    # new line character
    print(i, '\n')
```

"If the implementation is hard to explain, it's a bad idea."

— *The Zen of Python*

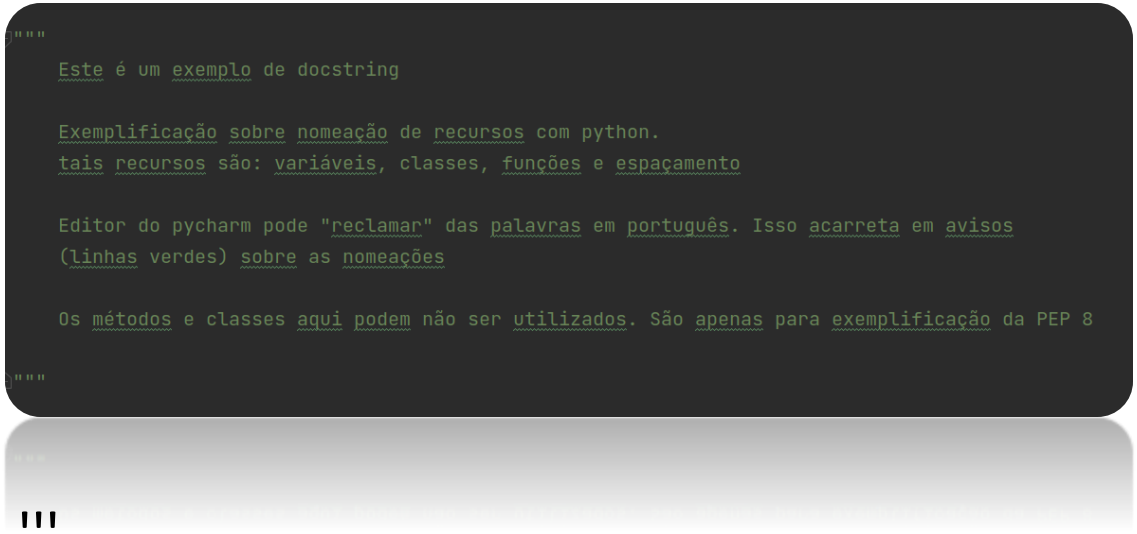
Blocos de comentários

```python
empty_list = []  # Initialize empty list

x = 5
x = x * 5  # Multiply x by 5
```

Não explique o óbvio!

Docstring



- Caracteres: """ ou '''
- Dentro e classes, funções e início de programas
- Escritos para módulos públicos

Etapa 1

The Guide Line - Espaços em Branco em Expressões

// Integração com Python

Espaçamento

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

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

```
# Correct:
if x == 4: print(x, y); x, y = y, x
```

```
# Wrong:
if x == 4 : print(x , y) ; x , y = y , x
```

Não exagere nos espaços!

```
# Correct:
foo = (0,)
```

```
# Wrong:
bar = (0, )
```

Python

```
# Recommended
if x>5 and x%2==0:
    print('x is larger than 5 and divisible by 2!')
```

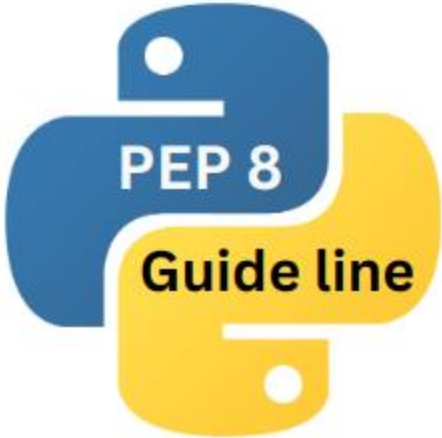
Quando evitar?

Poucos espaços

- Difícil de ler

Esparso demais

- Muitos espaços



Dúvidas?

> Fórum/Artigos

> Comunidade Online (Discord)

Para saber mais

Referências principais:

- https://peps.python.org/pep-0008/
- https://peps.python.org/pep-0257/
- https://pypi.org/project/pylint/
- https://pypi.org/project/flake8/

https://github.com/julianazanelatto