PEP8: Guía de Estilo para Python

PEP8 es un conjunto de directrices de estilo para el código Python. Fue creado para promover la coherencia y legibilidad en el código Python. PEP8 no es una norma obligatoria, pero se recomienda ampliamente seguirlo para mejorar la calidad del código.

```
# Heading 1

## Heading 2

This is `inline code`.

```c
// fenced code block
void foo() {
    printf("bar");
}
```

Lists

- **bold**
- *italic*

Checklists

- [ ] TBD
- [x] Done

{
  "year": 2020,
  "sucks": true
}
```

Importancia de PEP8

1 Legibilidad

El código legible es más fácil de entender y mantener, lo que facilita la colaboración entre los desarrolladores.

2 Consistencia

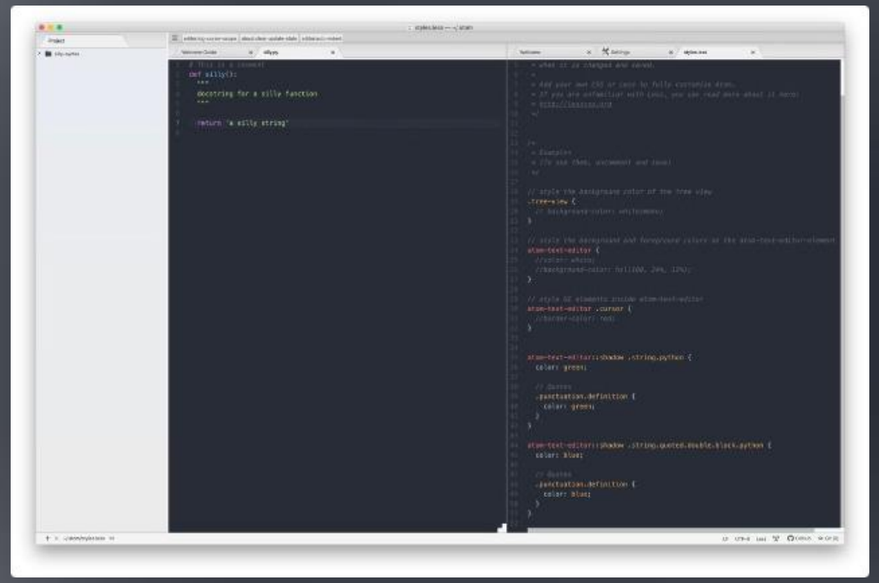
PEP8 garantiza que el código Python se vea y funcione de manera similar en diferentes proyectos.

3 Mantenimiento

El código consistente es más fácil de modificar y depurar, lo que reduce el tiempo y los costes de desarrollo.

4 Eficiencia

Seguir las directrices de PEP8 puede ayudar a evitar errores comunes y a mejorar la eficiencia del código.



Convenciones de Estilo

Sangrías

Utiliza 4 espacios para cada nivel de sangría.

No utilices tabulaciones.

Espaciado

Utiliza un espacio en blanco alrededor de operadores como `=` y `+`.

Utiliza dos líneas en blanco para separar funciones y clases.

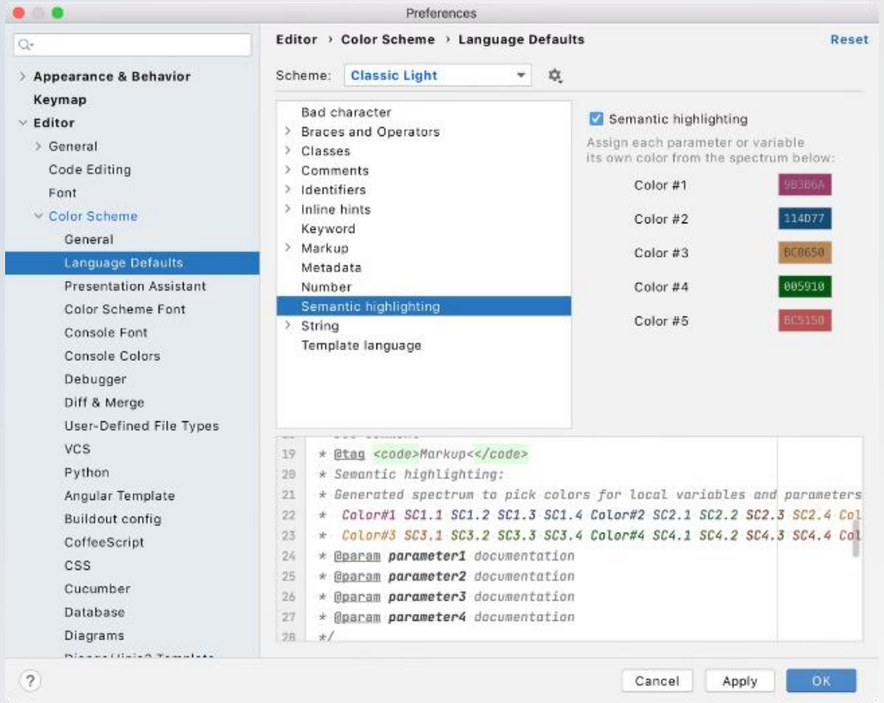
Longitud de Líneas

Limita las líneas de código a 79 caracteres.

Utiliza barras invertidas (`\`) para continuar líneas largas.

Nomenclatura de Variables y Funciones

Tipo	Ejemplo	Descripción
Variable	my_variable	En minúsculas, separadas por guiones bajos.
Función	my_function	En minúsculas, separadas por guiones bajos.
Constante	MY_CONSTANT	En mayúsculas, separadas por guiones bajos.
Clase	MyClass	CamelCase, comenzando con una letra mayúscula.



Longitud de Líneas

1

Límites

Mantén las líneas de código a 79 caracteres.

2

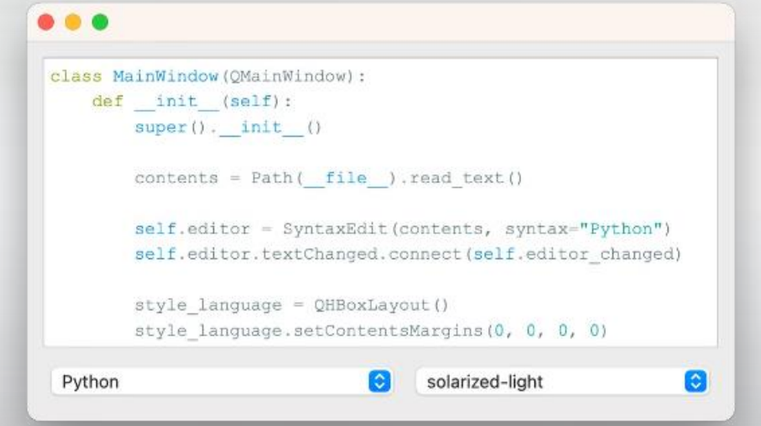
Continuación

Usa barras invertidas (`\`) para continuar líneas largas.

3

Paréntesis

Utiliza paréntesis para dividir líneas largas.



```python
from dataclasses import dataclass


@dataclass
class Point:
    x: int
    y: int


def where_is(point):
    match point:
        case Point(x=0, y=0):
            print("Origin")
        case Point(x=0, y=y):
            print(f"Y={y}")
        case Point(x=x, y=0):
            print(f"X={x}")
        case Point():
            print("Somewhere else")
        case _:
            print("Not a point")

```

Espaciado y Sangrías

Sangrías

Utiliza 4 espacios para cada nivel de sangría.

Espaciado

Utiliza un espacio en blanco alrededor de operadores como `=` y `+`.

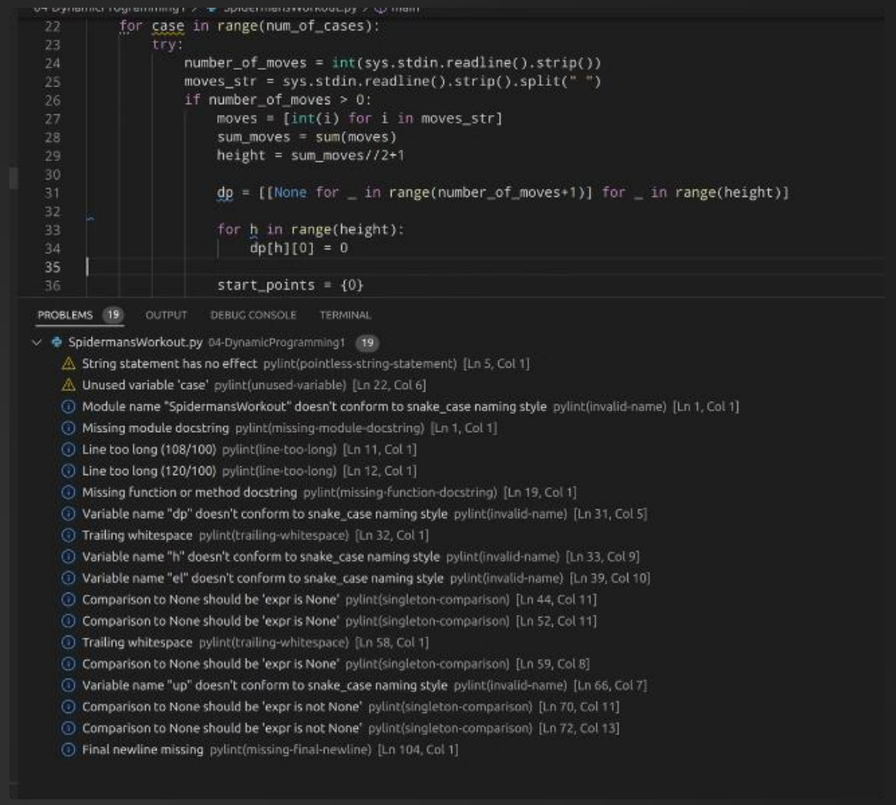
Líneas en Blanco

Utiliza dos líneas en blanco para separar funciones y clases.

Comentarios

Utiliza espacios en blanco para separar el código de los comentarios.

Documentación y Comentarios



Docstrings

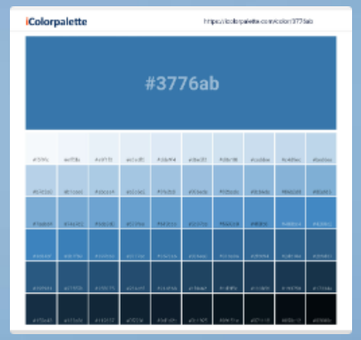
Las docstrings documentan funciones, clases y módulos. Se utilizan para generar documentación.

Comentarios en Línea

Los comentarios en línea se utilizan para explicar el código en una sola línea.

Comentarios de Bloque

Los comentarios de bloque se utilizan para explicar secciones más grandes de código.



Beneficios de Seguir PEP8

1

Legibilidad

El código legible es más fácil de entender y mantener.

2

Colaboración

PEP8 facilita la colaboración entre desarrolladores al garantizar la consistencia del código.

3

Mantenimiento

El código consistente es más fácil de modificar y depurar.

4

Eficiencia

Seguir PEP8 puede ayudar a evitar errores comunes y mejorar la eficiencia del código.