Zen de Python

¿Que es?

Es una colección de 19 aforismos que resumen la filosofía de diseño del lenguaje de programación Python. Escritos por Tim Peters, estos principios destacan las cualidades que el código Python debe tener para ser considerado "pythónico", es decir, alineado con los valores y prácticas que los desarrolladores de Python promueven. Los principios no solo guían el diseño del lenguaje en sí, sino que también sirven como una guía para escribir código que sea claro, legible y mantenible.

Principio de Explicito mejor que Implicito

El principio de "Explícito es mejor que implícito" es uno de los aforismos del Zen de Python y destaca la importancia de la claridad y la transparencia en el código. En términos prácticos, este principio sugiere que el código debe ser escrito de manera que los comportamientos y las intenciones sean claramente visibles y entendibles para cualquier persona que lo lea, sin necesidad de hacer suposiciones o deducciones.

Desglose del Principio Claridad en la intención:

Cuando algo es explícito, sus efectos y resultados son claros y evidentes. No hay necesidad de adivinar o interpretar lo que el código intenta hacer. Esto ayuda a otros desarrolladores (y al propio autor en el futuro) a entender rápidamente el propósito y funcionamiento del código.

Facilita el mantenimiento:

El código explícito es más fácil de mantener porque los futuros desarrolladores pueden entender su intención y funcionamiento sin necesidad de una documentación extensa o de un conocimiento profundo del contexto original.

Reducción de errores:

La claridad reduce la posibilidad de errores. Cuando los comportamientos son explícitos, es menos probable que ocurran malentendidos o suposiciones incorrectas que puedan llevar a errores.

Ejemplo en Python:

1. Uso de argumentos con nombre en lugar de argumentos posicionales:

Implícito:

```python
def create_user(name, age, active=True):
    # código para crear un usuario
    pass

create_user("Alice", 30)
create_user("Bob", 25, False)
```

Figure 1:

Explicito:

```python
def create_user(name, age, active=True):
    # código para crear un usuario
    pass

create_user(name="Alice", age=30)
create_user(name="Bob", age=25, active=False)
```

Figure 2: