Servicio Nacional de Aprendizaje Sena

Patrón de diseño

01 ¿Qué es?

un patrón de diseño no se copia y pega como una función o biblioteca, sino que sirve como guía para resolver un problema específico; se adapta a cada proyecto, permitiendo crear soluciones personalizadas.
Mientras un algoritmo define una secuencia exacta de pasos, un patrón es una idea más general sobre cómo estructurar una solución

02 Estructura

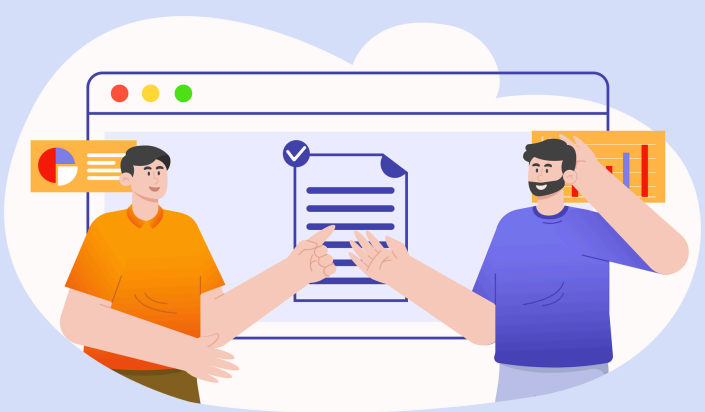
- Propósito: Explica de manera breve el problema que resuelve el patrón y la solución que ofrece.
- Motivación: Detalla el problema y la solución, brindando un contexto más amplio.
- Estructura: Muestra cómo están organizadas las clases y componentes del patrón, y cómo se relacionan entre sí.
- Ejemplo de código: Presenta una implementación del patrón en un lenguaje de programación común, facilitando su comprensión.
- Detalles adicionales: Pueden incluir aplicabilidad, pasos de implementación y relaciones con otros patrones.

03 Tipos

- Patrones de arquitectura: Son los más generales y se pueden aplicar a cualquier lenguaje. Se utilizan para diseñar la estructura completa de una aplicación.
- Creacionales: Se encargan de crear objetos de manera flexible y reutilizable.
- Estructurales: Ayudan a combinar objetos y clases en estructuras más grandes, manteniendo la flexibilidad y eficiencia.
- Comportamentales: Se ocupan de la comunicación y asignación de responsabilidades entre objetos.

04 ¿Por qué aprender patrones de diseño?

- Soluciones probadas: Los patrones de diseño son como soluciones predefinidas a problemas comunes en programación. Conocerlos te permite resolver problemas de manera más eficiente y efectiva.
- Lenguaje común: Los patrones establecen un vocabulario compartido entre programadores. Esto facilita la comunicación y colaboración en equipos de desarrollo.



05 Tipos

- Patrones de arquitectura: Son los más generales y se pueden aplicar a cualquier lenguaje. Se utilizan para diseñar la estructura completa de una aplicación.
- Creacionales: Se encargan de crear objetos de manera flexible y reutilizable.
- Estructurales: Ayudan a combinar objetos y clases en estructuras más grandes, manteniendo la flexibilidad y eficiencia.
- Comportamentales: Se ocupan de la comunicación y asignación de responsabilidades entre objetos.