Patrones de diseño

Definición de un patrón de diseño

Los patrones de diseño dan a los desarrolladores unas reglas para que a la hora de solucionar un problema no tengan que empezar de nuevo sino que partan de un diseño bien implementado. Hay que saber cómo utilizarlos y su objetivo es escribir software re usable y extensible. No se puede elegir un patrón y copiarlo en el programa como si fuera una función por ejemplo. El patrón no es una porción específica de código, sino un concepto general para resolver un problema particular. Esto da lugar a que puedas implementar una solución para tu propio programa. Los patrones tienen normalmente unas secciones:

- Una parte donde explica brevemente el problema y la solución
- Una parte donde muestra todas las partes del patrón y como se relacionan
- Un ejemplo del código

Ventajas de los patrones de diseño

Los patrones de diseño tienen dos partes de beneficios: los técnicos y los personales. Por un lado por la parte técnica es la posibilidad de usar diseños más estándares para que sea más ordenado; poder re usar el código como hemos comentado antes; y la capacidad de escalar rápido nuestro programa para poderse adaptar a todos los cambios que hayan.

Por la parte personal tenemos como ventajas por ejemplo el poder hablar el mismo idioma en el equipo; el poder aplicar los patrones sin importar el lenguaje de programación que uses; y la facilidad de aprender un lenguaje nuevo.

Los patrones de diseño más importantes

1. Factory method

   Poder crear objetos sin especificar la clase exacta del objeto creado.

2. Singleton

   Patrón utilizado para limitar una clase a un solo objeto.

3. Observer

   Dependencia de uno a muchos entre objetos, causando que cuando uno cambia, se notifique a sus dependientes.

4. Strategy

   Permite agrupar algoritmos relacionados bajo una abstracción.

5. Adapter

   Permite que clases incompatibles trabajen juntas gracias a una interfaz usada de una clase en otra.

6. Builder

   Patrón de construcción para construir objetos paso a paso.

7. State

   Patrón que encapsula los estados de la máquina y permite que un objeto cambie su comportamiento cuando cambia el estado de la máquina.