Patrones GoF

Maria Fernanda Gomez Guzman

Juan Esteban Garzon Yanquen

Juan Felipe Acosta Lopez

Introduccion al Desarrollo de Software

Daniel David Leal Lara

Bogota D.C.

2023

Los patrones GoF (Gang of Four) son un conjunto de patrones de diseño de software propuestos por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides en su libro "Design Patterns: Elements of Reusable Object-Oriented Software" (1994). Estos patrones ofrecen soluciones probadas y eficientes a problemas comunes de diseño en el desarrollo de software.

Los patrones GoF se dividen en tres categorías principales:

1. Patrones de creación: se ocupan de la creación de objetos, proporcionando diferentes mecanismos para la instanciación de clases. Ejemplos de patrones de creación incluyen el patrón de fábrica (Factory), el patrón de constructor (Builder) y el patrón de prototipo (Prototype).

2. Patrones estructurales: se centran en la composición de clases y objetos para formar estructuras más grandes y flexibles. Estos patrones ayudan a definir cómo las clases y objetos se relacionan entre sí. Ejemplos de patrones estructurales incluyen el patrón de adaptador (Adapter), el patrón de puente (Bridge) y el patrón de decorador (Decorator).

3. Patrones de comportamiento: se refieren a la interacción entre objetos y cómo se distribuye la responsabilidad entre ellos. Estos patrones definen algoritmos y asignan responsabilidades entre objetos. Ejemplos de patrones de comportamiento incluyen el patrón de observador (Observer), el patrón de estrategia (Strategy) y el patrón de visitante (Visitor).

Cada patrón GoF proporciona una solución a un problema específico y promueve el diseño de software modular, flexible y reutilizable. Al utilizar estos patrones, los desarrolladores pueden aprovechar las mejores prácticas establecidas para abordar problemas comunes de diseño y mejorar la calidad de su código. (Patrones Gof) o patrones Gang Of Four

El desarrollo de los patrones GoF fue iniciado en la década de 1990 por cuatro expertos en diseño de software: Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Estos autores colaboraron para crear y documentar los patrones en el libro «Design Patterns: Elements of Reusable Object-Oriented Software» , también conocido como el libro de los patrones GoF.

Publicado en 1994, el libro se convirtió en una referencia fundamental en el campo del diseño de software. Los patrones presentados en él fueron recopilados a partir de la experiencia práctica de los autores y la observación de patrones comunes en el diseño de software orientado a objetos. Identificaron patrones que surgían repetidamente en diferentes contextos y los describieron detalladamente, ofreciendo soluciones probadas y prácticas.

El libro presenta un total de 23 patrones de diseño diferentes, organizados en tres categorías principales: patrones de creación, patrones estructurales y patrones de comportamiento. Cada patrón viene acompañado de una descripción que incluye su propósito, la estructura de las clases involucradas, las responsabilidades de cada clase y ejemplos de código.

Tras su publicación, los patrones GoF se volvieron ampliamente reconocidos y utilizados en la comunidad de desarrollo de software. Estos patrones proporcionan un vocabulario común y una base sólida para el diseño de sistemas orientados a objetos. Muchos lenguajes de programación y marcos de desarrollo han integrado estos patrones en sus prácticas recomendadas.

Desde entonces, los patrones GoF han sido estudiados, aplicados y adaptados por numerosos desarrolladores y equipos de desarrollo en todo el mundo. Aunque los patrones originales se

centran principalmente en el diseño orientado a objetos, los conceptos y principios subyacentes también se aplican a otros paradigmas de programación.

Es importante tener en cuenta que, si bien los patrones GoF son herramientas valiosas para el diseño de software, no son una solución universal para todos los problemas. Cada patrón debe ser considerado cuidadosamente en relación con el contexto y los requisitos del proyecto, y se deben tomar decisiones de diseño informadas y equilibradas.

Pruebas de Desarrollo de Software:

1. Pruebas unitarias: Se centran en probar unidades individuales de código, como funciones o métodos, para asegurarse de que funcionan correctamente.

2. Pruebas de integración: Verifican que las diferentes partes del software se integren y funcionen correctamente juntas.

3. Pruebas de regresión: Se realizan para asegurarse de que las modificaciones o actualizaciones en el software no hayan introducido nuevos errores y que las funcionalidades existentes sigan funcionando correctamente.

4. Pruebas de aceptación: Estas pruebas se llevan a cabo para validar que el software cumple con los requisitos y expectativas del cliente.

5. Pruebas de rendimiento: Evalúan el rendimiento y la capacidad de respuesta del software bajo diferentes condiciones de carga y estrés.

6. Pruebas de seguridad: Se realizan para identificar vulnerabilidades y asegurar que el software sea resistente a ataques maliciosos.

7. Pruebas de usabilidad: Evalúan la facilidad de uso y la experiencia del usuario, asegurándose de que el software sea intuitivo y cumpla con los requisitos de usabilidad.

8. Pruebas de compatibilidad: Se realizan para asegurar que el software funcione correctamente en diferentes sistemas operativos, navegadores web y dispositivos.