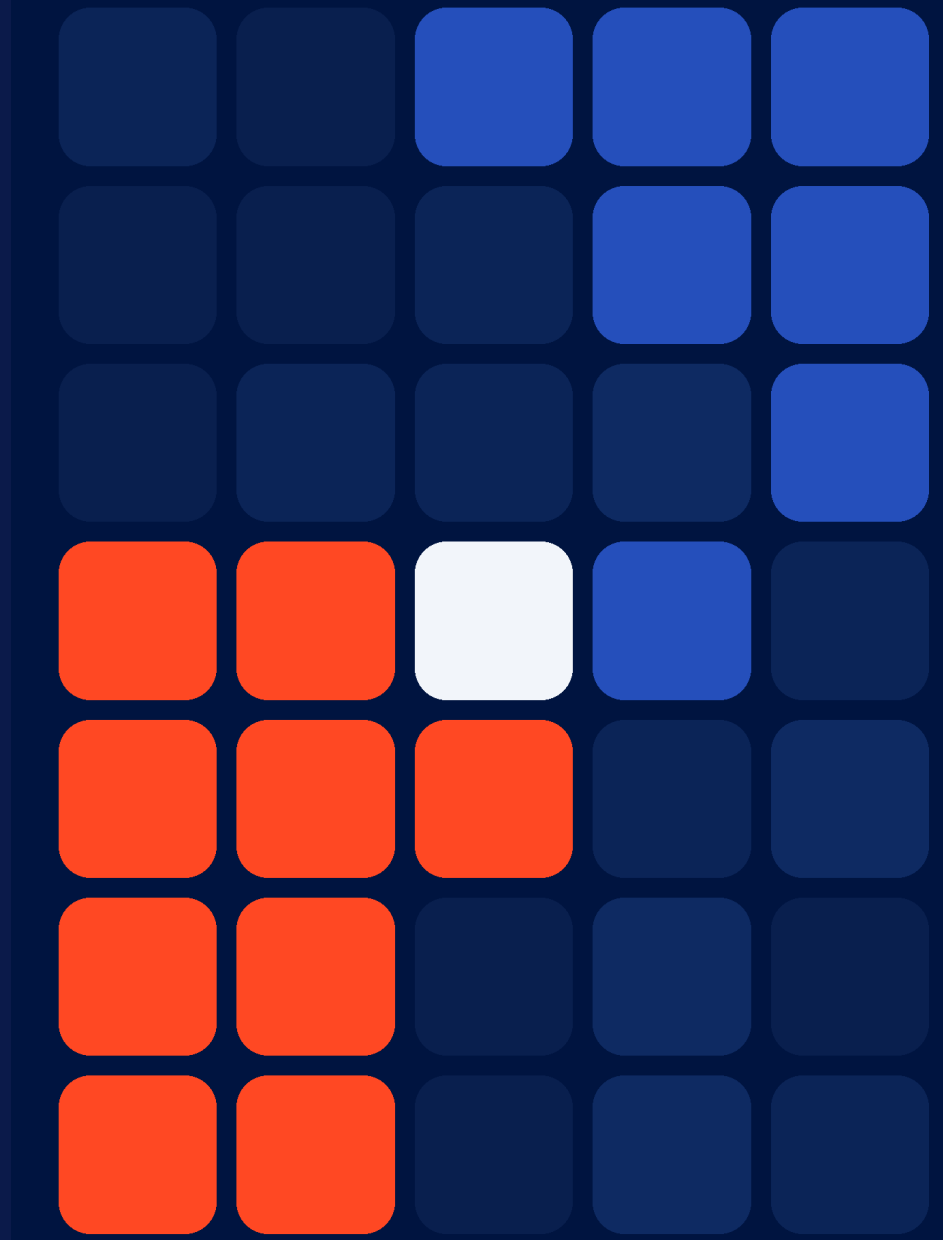
Soltein

ACADEMIA SOLTEIN · INTRODUCCIÓN A LA AE, AS Y LPS

Guía de estudio: patrones GoF

Los 23 patrones de diseño de la Banda de los Cuatro.



Lección: Guía de estudio sobre patrones GoF · Elaborado por MsC René Lazo (2024-10-10)

Odoo ERP · Suite Soltein

Contenido de la guía

01 ¿Qué son los patrones GoF?

02 Patrones creacionales (5)

03 Patrones estructurales (7)

04 Patrones de comportamiento (11)

05 Beneficios

06 Conclusiones

¿Qué son los patrones GoF?

“Soluciones probadas y reutilizables a problemas comunes de diseño de software, que mejoran flexibilidad, extensibilidad y mantenibilidad.”

Popularizados por Gamma, Helm, Johnson y Vlissides en «Design Patterns: Elements of Reusable Object-Oriented Software»

Tres categorías, 23 patrones

Soltein

1

Creacionales

Crean objetos de forma flexible, desacoplando el código del tipo concreto.

2

Estructurales

Componen clases y objetos en estructuras mayores y flexibles.

3

De comportamiento

Definen algoritmos y asignan responsabilidades entre objetos.

Cada patrón describe un problema recurrente, su solución abstracta y sus consecuencias. No son código fijo.

Creacionales: propósito

Controlan la instanciación y evitan dependencias rígidas

- Factory Method: un método crea objetos; las subclases deciden la clase.
- Abstract Factory: interfaz para crear familias de objetos relacionados.
- Builder: separa la construcción compleja de la representación final.
- Prototype: crea objetos clonando instancias existentes.
- Singleton: una única instancia con un punto de acceso global.

Creacionales: ejemplos

- Factory Method: figuras Círculo y Cuadrado con CreadorCirculo y CreadorCuadrado.
- Abstract Factory: FabricaGUI con FabricaWindows y FabricaMac.
- Builder: una Casa (madera o ladrillo) armada por un Director.
- Prototype: editor gráfico que duplica figuras con clonar().
- Singleton: clase Configuración con un solo objeto.

Estructurales: propósito (1/2)

Buscan eficiencia, reutilización y facilidad de modificación

- Adapter: convierte una interfaz en otra que el cliente espera.
- Bridge: desacopla una abstracción de su implementación.
- Composite: estructuras tipo árbol tratadas de forma uniforme.
- Decorator: agrega responsabilidades dinámicamente, alternativa a la herencia.

Estructurales: propósito (2/2)

- Facade: interfaz unificada y simple a un subsistema complejo.
- Flyweight: objetos compartidos para muchos objetos de grano fino.
- Proxy: sustituto de otro objeto para controlar su acceso.

Estructurales: ejemplos (1/2)

- Adapter: usar una biblioteca de pagos antigua en una interfaz moderna.
- Bridge: jerarquías Figura y Color que varían de forma independiente.
- Composite: archivos y directorios con operaciones comunes.
- Decorator: Bebida con ConLeche y ConAzucar que suman comportamiento y precio.

Estructurales: ejemplos (2/2)

- Facade: SistemaDeVideo que enciende TV, DVD y ajusta el sonido.
- Flyweight: reutilizar instancias de caracteres en un editor de texto.
- Proxy: carga diferida que crea el objeto real solo cuando se necesita.

Comportamiento: propósito (1/2)

Flexibilidad en el flujo de ejecución y la comunicación

- Chain of Responsibility: la petición pasa por una cadena de objetos.
- Command: encapsula una solicitud como objeto (undo, registro).
- Interpreter: gramática de un lenguaje y su intérprete.
- Iterator: acceso secuencial a una colección sin exponer su representación.
- Mediator: encapsula cómo interactúan varios objetos.
- Memento: captura y restaura el estado interno sin violar la encapsulación.

Comportamiento: propósito (2/2)

- Observer: dependencia uno-a-muchos; notifica al cambiar el estado.
- State: el objeto cambia de comportamiento al cambiar su estado.
- Strategy: familia de algoritmos intercambiables en tiempo de ejecución.
- Template Method: esqueleto de un algoritmo; las subclases completan pasos.
- Visitor: nuevas operaciones sin modificar las clases de los elementos.

Comportamiento: ejemplos (1/2)

- Chain: soporte técnico, de operador a supervisor.
- Command: botón que invoca Copiar o Pegar en un editor.
- Interpreter: evaluar expresiones matemáticas desde un árbol sintáctico.
- Iterator: recorrer una lista de objetos de una colección.
- Mediator: una interfaz gráfica coordina botones y campos de texto.
- Memento: deshacer cambios en un editor de texto.

Comportamiento: ejemplos (2/2)

- Observer: publicar-suscribir; Sujeto notifica a varios Observadores.
- State: máquina expendedora (SinMoneda, ConMoneda, Vendiendo).
- Strategy: ordenamiento con QuickSort o MergeSort.
- Template Method: ProcesadorDocumento con pasos abstractos en subclases.
- Visitor: chequeo de tipos o conteo de nodos sobre un árbol de sintaxis.

Beneficios de los patrones GoF

- Reutilización de soluciones: evitar reinventar la rueda con métodos probados.
- Mejor comunicación: un vocabulario compartido para discutir el diseño.
- Flexibilidad y mantenibilidad: código más fácil de modificar, entender y extender.

Herramientas conceptuales

No son recetas rígidas: su correcta selección da robustez y solidez a largo plazo.

Lo que debes recordar

23 patrones

Creacionales (5), estructurales (7) y de comportamiento (11).

Soluciones probadas

Reutilizables ante problemas comunes de diseño.

Vocabulario común

Mejoran la comunicación entre desarrolladores.

Guías flexibles

Se adaptan al contexto; no son código fijo.

Soltein · Una empresa que evoluciona · soltein.mx · cursosodoo.soltein.net

Fuentes y pendientes por validar

FUENTES

· Guía de estudio doc-guia-estudio-patrones-GOF-2024-12-v1.0 (Drive), MsC René Lazo.
· Libro «Design Patterns: Elements of Reusable Object-Oriented Software» (citado en el original).

REQUIERE VALIDACIÓN

· CONTRADICCIÓN DE ENLACE: las lecciones 83 (GoF) y 33 (Arquitectura ERP basada en Odoo) apuntan al mismo archivo de Drive, cuyo contenido real es esta guía GoF; la lección 33 debe corregirse en el EVA.
· Los ejemplos de código Java del original no se incluyen.
· Fecha 2024-12 en el nombre del archivo y 2024-10-10 en la portada.