Decorator

Nombre:
Decorator

Clasificación del patrón:
Estructural

Intención:
Ligar dinámicamente una responsabilidad adicional a un objeto. Decorator provee una alternativa flexible a subclases para una funcionalidad adicional.

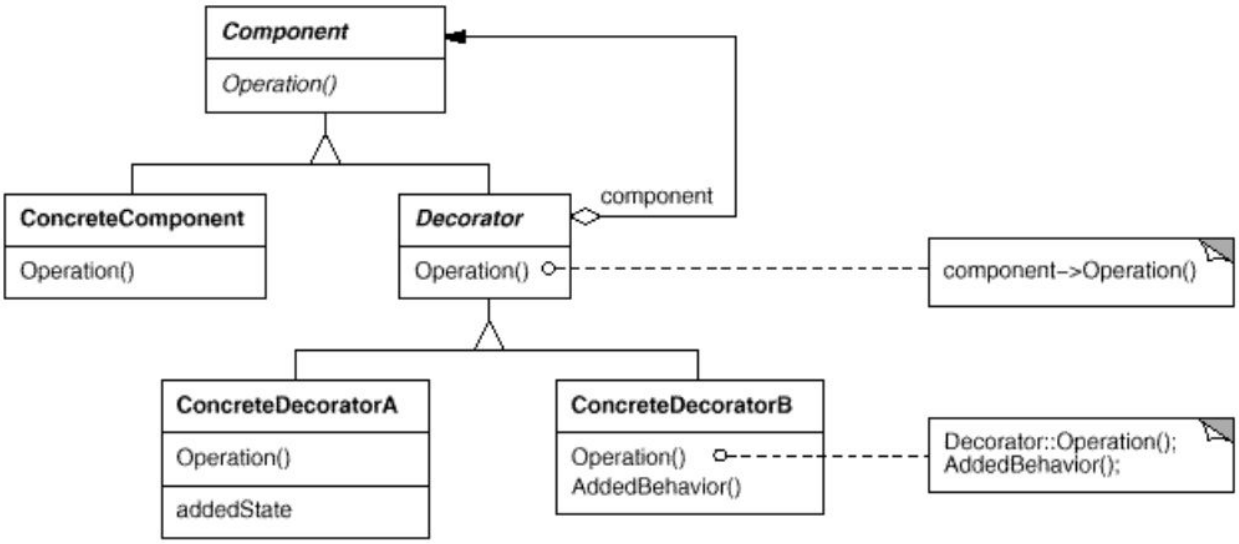
Otros nombres:
Wrapper

Motivación:
A veces se quiere añadir funcionalidad a un objeto concreto, no a una clase concreta. Solucionar la situación puede intentarse resolver por medio de la herencia entre clases, pero se pierde flexibilidad y las funcionalidades se añaden de forma estática. Decorator permite definir una clase que envuelva al componente y añada la funcionalidad deseada.

Aplicabilidad:
Se usa cuando se quiere:

- Añadir responsabilidades dinámicamente.
- Añadir responsabilidades que deben ser retiradas después.
- Subclases, pero no prácticas.
- Subclases, pero la clase no está disponible para cada característica y se quiere añadir nuevas características a cada clase.

Estructura:



Participantes:

- Componente: Define la interfaz de los objetos a los que se puede añadir responsabilidades de manera dinámica.
- ComponenteConcreto: Define el objeto al que añadir responsabilidades de manera dinámica.
- Decorador: Mantiene una referencia al objeto componente y define una interfaz conforme a la del componente.
- DecoradorConcreto: Añade responsabilidades al componente al que referencia.

Colaboraciones:

Decorador adelanta posiciones al objeto Componente. En ocasiones se realizan algunas operaciones antes de enviar la petición.

Ventajas:

- Usar decoradores es una manera de reemplazar funcionalidades sin usar herencia.
- Características son añadidas incrementalmente a medida que progresa el código.
    - No es necesario poner todas las características en una sola clase.
- Se paga el precio de usar características sólo cuando se están usando.
- Es fácil combinar varias clases para obtener una nueva funcionalidad.

Desventajas:

- Se crean muchos objetos pequeños.
- Los clientes necesitan conocer cómo los objetos interactúan con otros.
- Los sistemas se convierten en difíciles de entender y de depurar.

Implementación:

- Un componente y su decorador deben compartir la misma interfaz.
- Se puede omitir la clase abstracta decorator cuando se va a definir una única responsabilidad.
- La diferencia con Strategy es que en Decorator el componente no cambia.

Código de ejemplo:

- Documento:

```
class Documento {
    public void operacion() {
        System.out.println("Creando documento");
    }
    public void close() {
        System.out.println("Cerrando documento");
    }
}

abstract class DocumentoCodificado extends Documento {
    private Documento documento;
    public DocumentoCodificado(Documento componente) {
        this.documento = componente;
    }
    public void operacion() {
        documento.operacion();
    }
}

class DecoradorConcretoA extends DocumentoCodificado {
    public DecoradorConcretoA(Documento componente) {
        super(componente);
    }
    public void operacion() {
        super.operacion();
        System.out.println("Separador al documento");
    }
}

class DecoradorConcretoB extends DocumentoCodificado {
    public DecoradorConcretoB(Documento componente) {
        super(componente);
    }
    public void operacion() {
        super.operacion();
        System.out.println("Contando lineas del documento");
    }
}
```

- Instanciar clases:

```
public class Prueba {
    public static void main(String[] args) {
        Documento c = new Documento();
        DecoradorConcretoA d1 = new DecoradorConcretoA(c);
        DecoradorConcretoB d2;
        d2 = new DecoradorConcretoB(d1);
        d2.operacion();
        d2.close();
    }
}
```

Usos conocidos:

- Frecuentemente es usado en Toolkits

Bibliografía:

No específico. (No específico). GoF Design Patterns (Versión 2.1.0) [Aplicación móvil]. Descargado de: https://drive.google.com/file/d/0BywiVyFlIabXcVhGZlJBcnhWTkU/view.

Gil Gala, F, J. Qué es el patrón de diseño Decorator. (26 de enero de 2015). Qué es el patrón de diseño Decorator. Consultado en: https://rootear.com/desarrollo/patron-decorator.

Junta de Andalucía. (s.f). Decorador. Marco de Desarrollo de la Junta de Andalucía. http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/186.