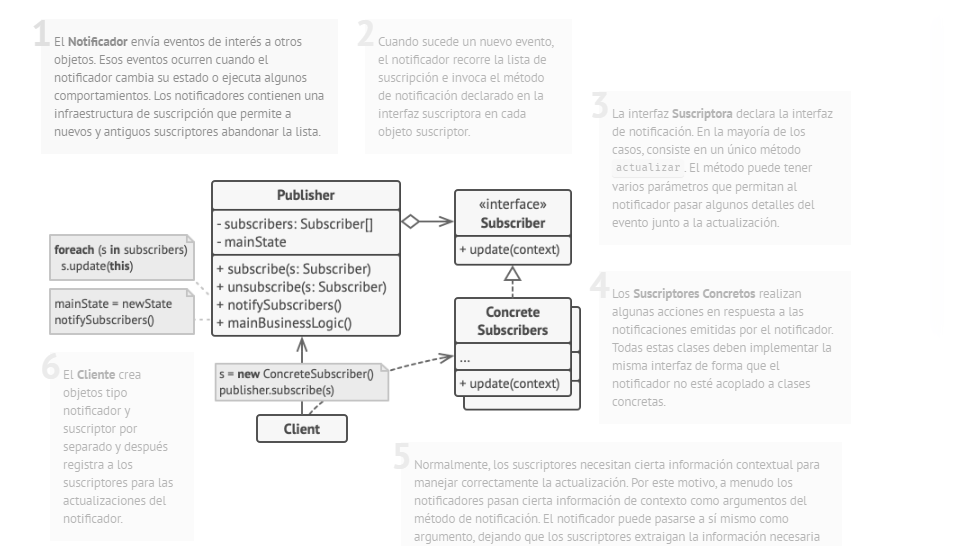
OBSERVER



**Tipo de patrón:** Comportamiento (Behavioral)

**Descripción:**  
El patrón Observer define una relación de dependencia uno-a-muchos entre objetos, de manera que cuando uno de ellos cambia su estado, todos sus dependientes son notificados y actualizados automáticamente. Es comúnmente usado en sistemas de eventos o actualizaciones automáticas.

**Características principales:**

* Desacopla el objeto emisor (sujeto) de los receptores (observadores).
* Permite a múltiples objetos escuchar los cambios de un mismo sujeto.
* La actualización de los observadores se hace de forma automática al cambiar el estado del sujeto.
* Facilita la construcción de sistemas reactivos y modulares.

**Componentes:**

1. **Subject (Sujeto):** Objeto que mantiene una lista de observadores y gestiona su registro y notificación.
2. **Observer (Observador):** Define una interfaz para recibir notificaciones del sujeto.
3. **ConcreteSubject:** Implementación del sujeto que contiene el estado de interés.
4. **ConcreteObserver:** Observadores que reaccionan a los cambios del sujeto.

**Ventajas:**

* Fuerte desacoplamiento entre sujeto y observadores.
* Permite implementar sistemas de publicación/suscripción fácilmente.
* Mejora la escalabilidad del sistema.

**Desventajas:**

* El orden en que se notifican los observadores no siempre está definido.
* Puede haber problemas de rendimiento si hay muchos observadores.
* Complejidad en la depuración cuando existen múltiples dependencias.

**Aplicaciones comunes:**

* Sistemas de notificación o suscripción a eventos.
* Interfaces gráficas donde varios elementos deben actualizarse ante cambios del modelo.
* Implementación del patrón Modelo-Vista-Controlador (MVC), donde la vista observa al modelo.