**PATRON OBSERVER**

**Propósito**

Este es un patrón de diseño de comportamiento que te permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto que están observando.

**Dominios de aplicación**

El patrón se utiliza en los siguientes casos:

* Una modificación en el estado de un objeto genera modificaciones en otros objetos que se determinan dinámicamente.
* Un objeto quiere avisar a otros objetos sin tener que conocer su tipo, es decir sin estar fuertemente acoplado a ellos.

**Participantes**

Son los siguientes:

* Subject es la clase abstracta que incluye la asociación con los observadores, así como los métodos para agregar o suprimir observadores.
* Observer es la interfaz que es necesario implementar para recibir las notificaciones (método actualiza).
* ConcreteSubject es una clase correspondiente a la implementación de un sujeto. Un sujeto envía una notificación cuando su estado se ha modificado.
* ConcreteObserver es una clase de implementación de un observador. Mantiene una referencia hacia el sujeto e implementa el método actualiza. Solicita a su sujeto información que forma parte de su estado durante las actualizaciones invocando al método getState.

**Estructura**

