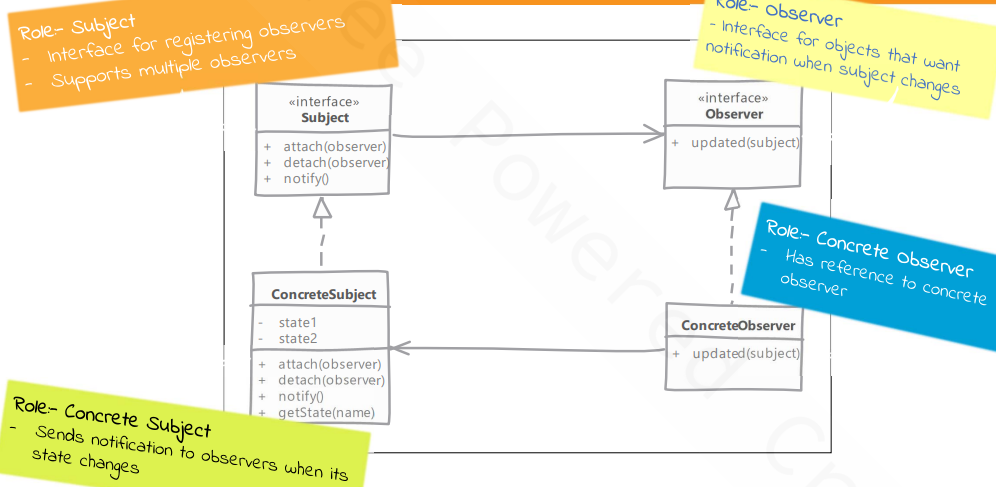
Observer

Behavioral Design Pattern

**What is an Observer:**

1. Notify multiple objects whenever an object changes state.
2. Publisher-Subscriber
3. One-to-many dependency between objects, where many objects are listening for state change of a single object, without tightly coupling all of them together.
4. This pattern is often implemented where listener only gets notification that something has changed in the object’s state. Listeners query back to find out more information if needed. This makes it more generic as different listeners may be interested in different states.

**UML:**



**Implementation Steps:**

1. Define an interface for observer. Observer is usually a very simple interface and defines a method used by subject to notify about state change.
2. Subject can be an interface if we are expecting our observers to listen to multiple objects or else subject can be any concrete class.
3. Implementing subject means taking care of handling attach, detach of observers, notifying all registered observers & providing methods to provide state information requested by observers.
4. Concrete observers use a reference passed to them called subject for getting more information about the state. If we are passing changed state in notify method then this is not required.

**Structure:**

