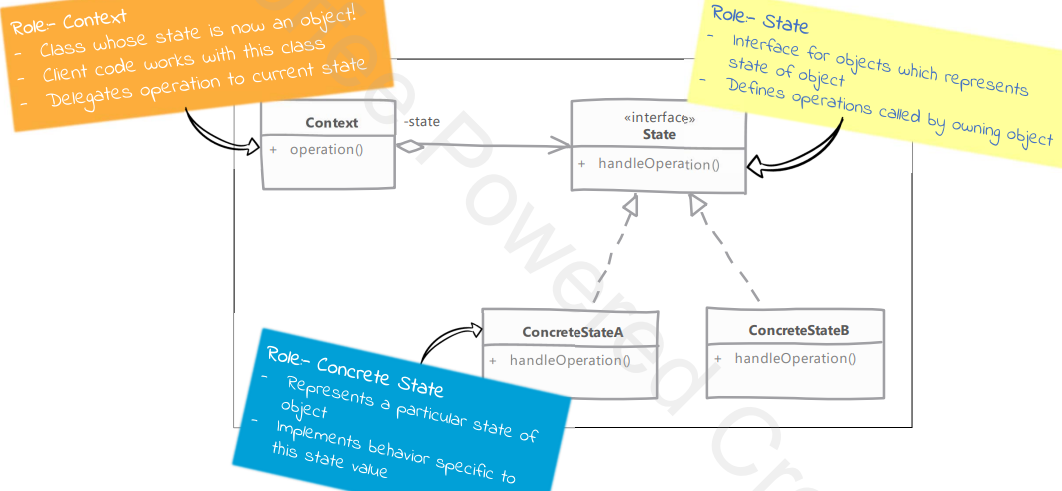
State

Behavioral Design Pattern

**What is a State:**

1. Allows our objects to behave differently based on its current internal state.
2. Allows to define the state specific behaviors in separate classes.
3. Operations defined in the class delegate to the current state object’s implementation of that behavior.
4. State transitions can be triggered by states themselves in which case each state knows about at least one other state’s existence.
5. New states and thus new behaviors can be added without changing our main class.
6. Let’s an object alter its behavior when its internal state changed. It appears as if the object changed its class.

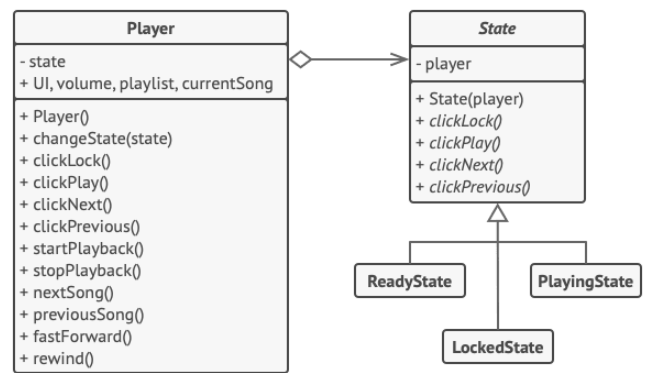
**UML:**



**Implementation steps:**

1. Identify distinct values for state of our object (context). Each state value will be a separate class in our implementation. These classes will provide behavior specific to the state value they represent.
2. Context class method implementations we will delegate the operation to current state object.
3. Decide how our state transition is going to happen. States can themselves transition to next state based on input received in a method.
4. Client interacts with our main class or context and is unaware of existence of state.

**Structure:**

****