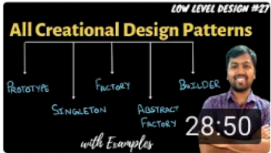



# All Behavioral Patterns (Concept and Coding)



27. All Creational Design Patterns  
Chapters: 00:00 - Introduction 00:50  
Pattern 09:05 - Singleton Design Pat

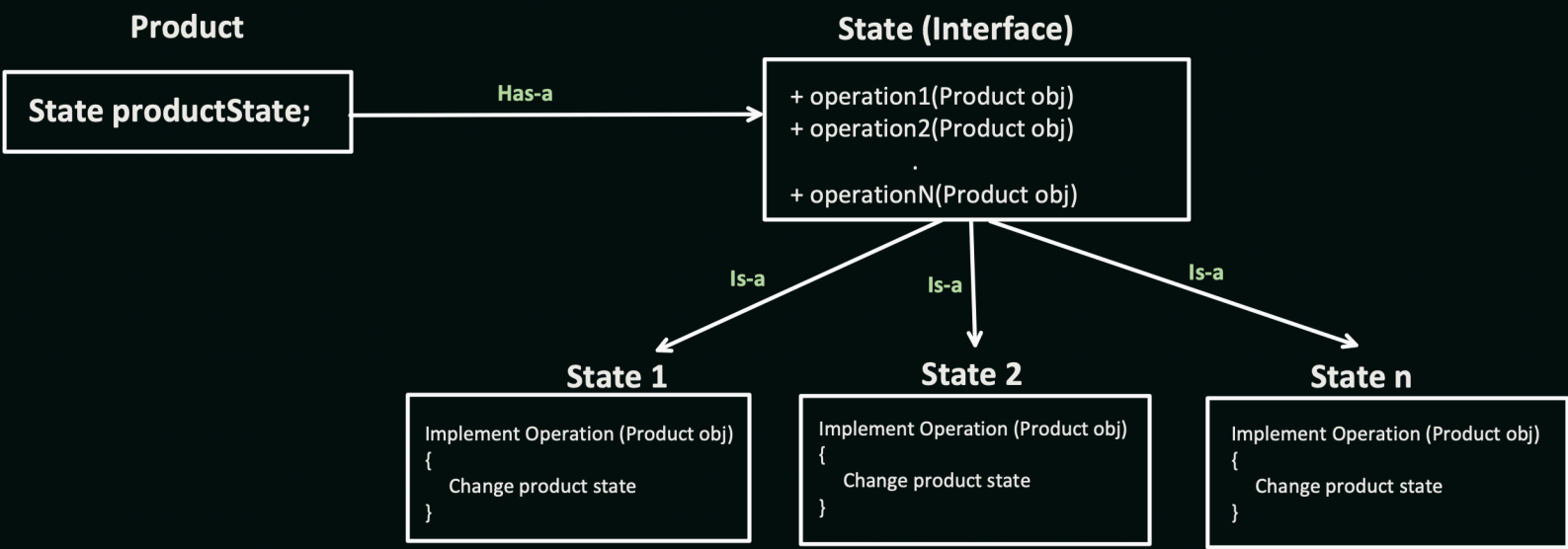


32. All Structural Design Patterns  
➡ Notes: Shared in the Member Co  
are Member of this channel, then pl

**Behavioral Design Patterns:**  
Guides how different objects communicate with each other effectively and Distribute tasks efficiently, making software system flexible and easy to maintain.

- 1. State Pattern
- 2. Observer Pattern
- 3. Strategy Pattern
- 4. Chain of Responsibility Pattern
- 5. Template Pattern
- 6. Interpreter Pattern
- 7. Command Pattern
- 8. Iterator Pattern
- 9. Visitor Pattern
- 10. Mediator Pattern
- 11. Memento Pattern

1. **State Pattern:** allows an object to alter its behaviour when its internal state changes.



```
public class VendingMachine {  
  
    VendingState machineState;  
  
    public VendingState getMachineState() {  
        return machineState;  
    }  
  
    public void setMachineState(VendingState machineState) {  
        this.machineState = machineState;  
    }  
}
```

```
public interface VendingState {  
  
    void insertCoin(VendingMachine product);  
  
    void dispenseItem(VendingMachine product);  
}
```

```
public class IdleState implements VendingState{  
  
    @Override  
    public void insertCoin(VendingMachine product) {  
        //insert coin logic  
        System.out.println("Coin Inserted");  
        product.setMachineState(new WorkingState());  
    }  
  
    @Override  
    public void dispenseItem(VendingMachine product) {
```

```
public class WorkingState implements VendingState{  
  
    @Override  
    public void insertCoin(VendingMachine product) {  
        //not doing anything here  
    }  
  
    @Override  
    public void dispenseItem(VendingMachine product) {  
        System.out.println("Product dispensed");  
    }  
}
```