Structural Design Patterns

Composite: Represents object hierarchies where individual objects and compositions of objects are treated the same way.

Adapter: Allows converting the interface of a class into another interface that clients expect.

Decorator: Adds additional behavior to an object dynamically.

Facade: Provides a simplified, higher-level interface to a subsystem. Clients can talk to the facade rather than individual classes in the subsystem.
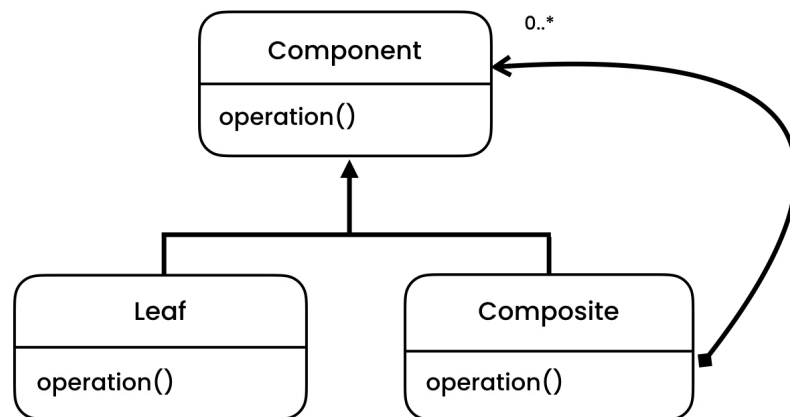
Flyweight: Allows sharing common state between multiple objects.

Bridge: Allows representing hierarchies that grow in two different dimensions independently.

Proxy: Allows providing a substitute for another object. The proxy object delegates all the work to the target object and contains some additional behavior.
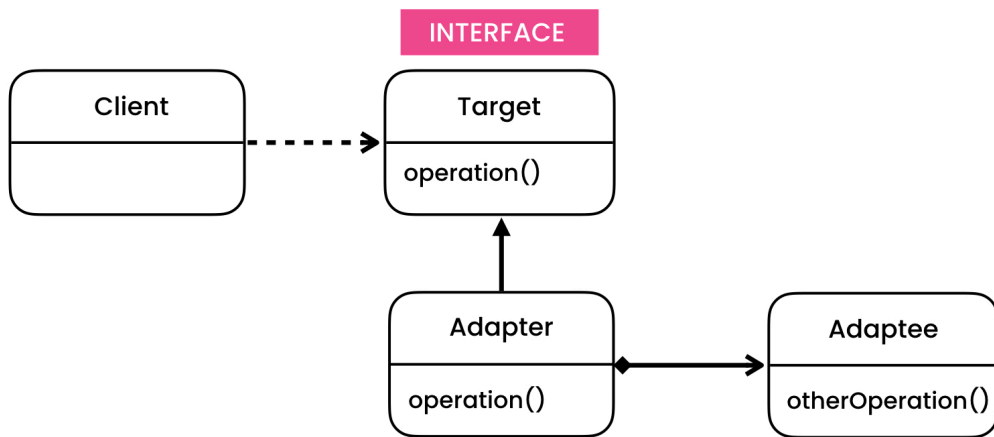
Composite Pattern

Represents object hierarchies where individual objects and compositions of objects are treated the same way.

```
                              Component                0..*
                          ┌─────────────────┐
                          │   Component      │◄─────────┐
                          ├─────────────────┤          │
                          │  operation()    │          │
                          └─────────────────┘          │
                                   △                    │
                          ┌────────┴────────┐           │
                  ┌───────────────┐  ┌───────────────┐  │
                  │     Leaf      │  │   Composite   │  │
                  ├───────────────┤  ├───────────────┤◄─┘
                  │  operation()  │  │  operation()  │
                  └───────────────┘  └───────────────┘
```
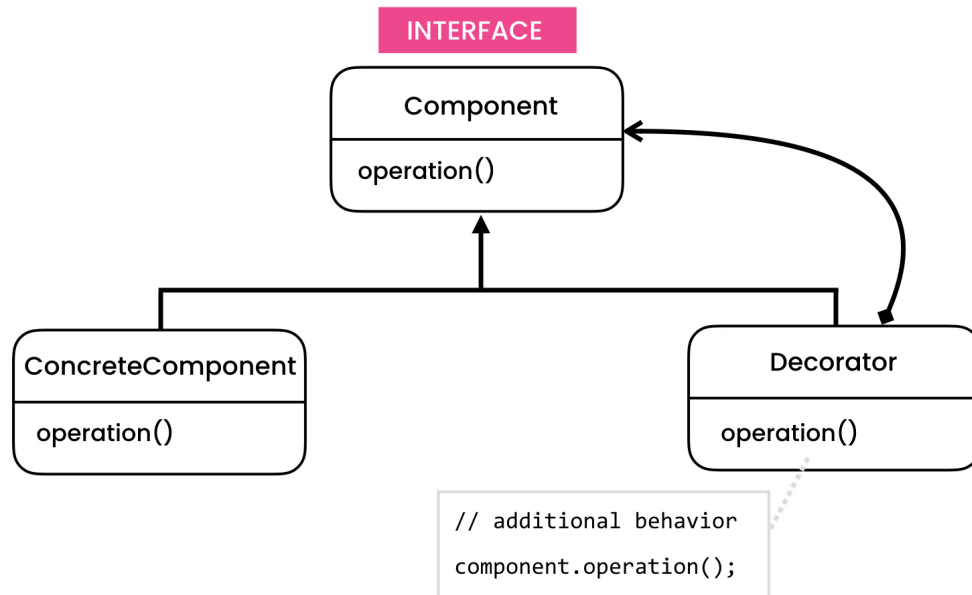
Adapter Pattern

Allows converting the interface of a class into another interface that clients expect.

Decorator Pattern

Adds additional behavior to an object dynamically.

INTERFACE

Component

operation()

ConcreteComponent

operation()

Decorator

operation()

```
// additional behavior
component.operation();
```
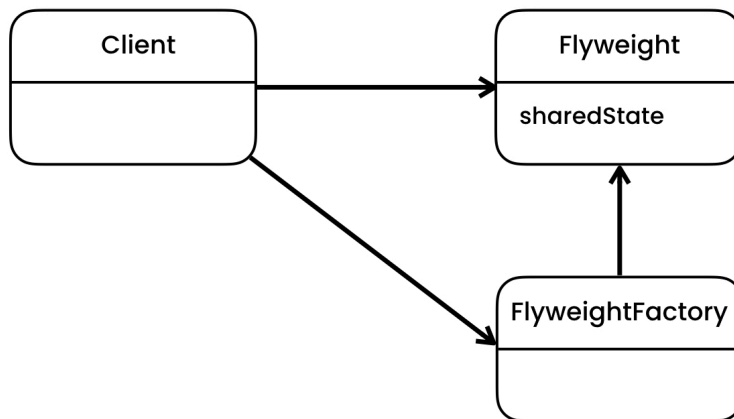
Facade Pattern

Provides a simplified, higher-level interface to a subsystem. Clients can talk to the facade rather than individual classes in the subsystem.
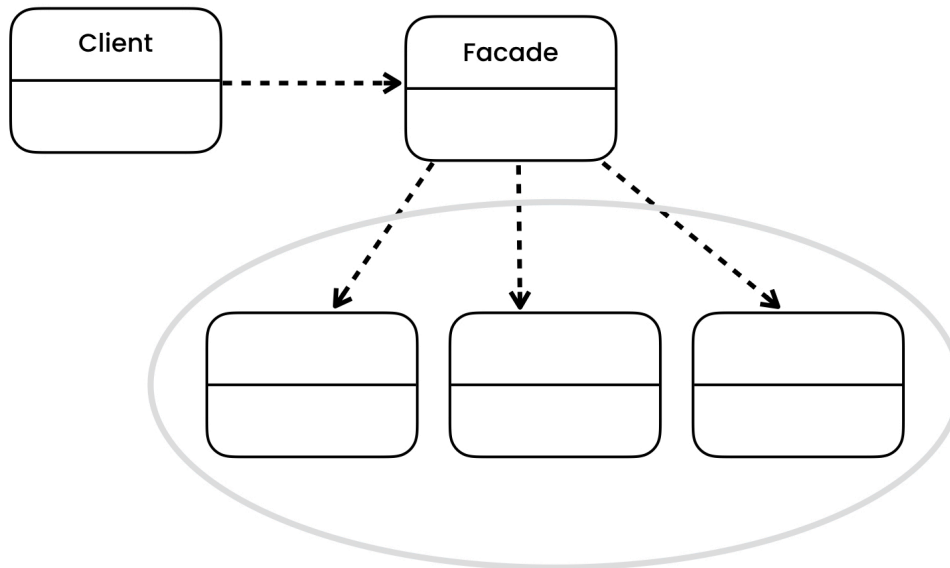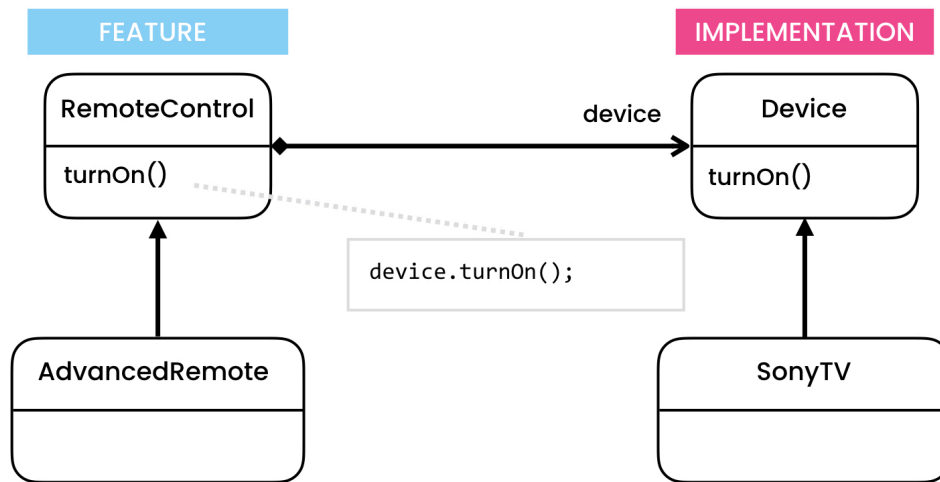
Flyweight Pattern

Provides a simplified, higher-level interface to a subsystem. Clients can talk to the facade rather than individual classes in the subsystem.

Bridge Pattern

Allows representing hierarchies that grow in two different dimensions
independently.

Proxy Pattern

Allows providing a substitute for another object. The proxy object delegates all the work to the target object and contains some additional behavior.