Decorator Pattern

Structural Design Pattern

**What is Decorator:**

1. When we want to enhance behavior of our existing object dynamically as and when required.
2. Decorator warps an object within itself and provides the same interface as the wrapped object. So the client of the original object doesn’t need to change.
3. Alternative to subclassing for extending functionality of existing classes.
4. Decorator lets you attach new behaviors to objects by placing these objects inside special wrapper objects that contain the behaviors.

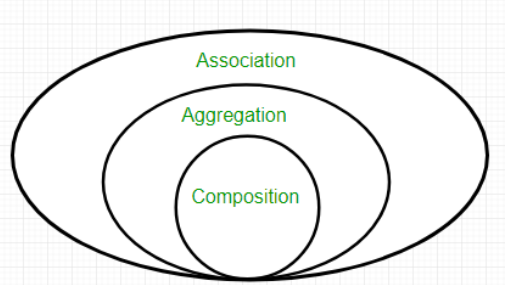
**Problem:**

1. Extending a class is the first thing that comes to mind when you need to alter an object’s behavior. However, inheritance has several serious caveats.
2. Inheritance is static. Can’t alter the behavior of an existing object at runtime.
3. Subclasses can have just one parent class.

**Solution:** Use Aggregation or Composition instead of Inheritance.

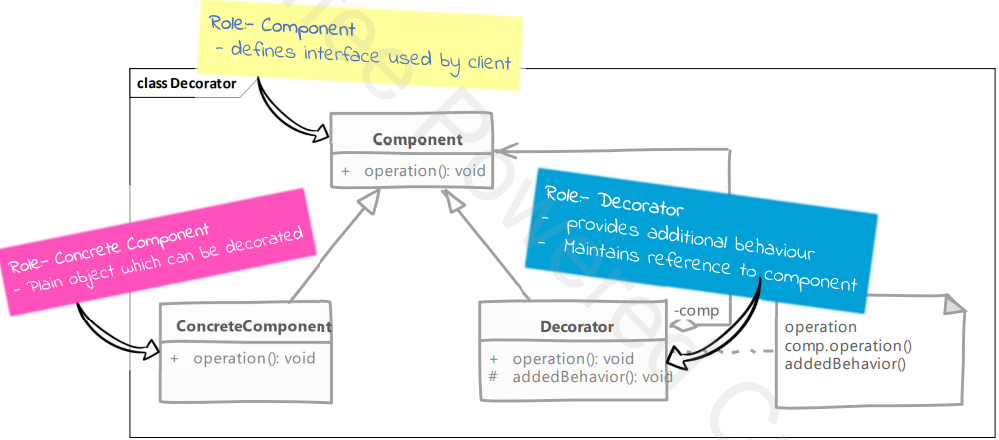
**Aggregation:** a special form of Association, where there is a unidirectional association. One-way relationship. (Department can have students but vice versa it not possible). Both entries can survive individually.

**Composition:** form of Aggregation in which two entities are highly dependent on each other. Object A is part of Object B. if B gets destroyed all objects A in B will also be destroyed.



**Real-World Analogy:** Wearing clothes is an example of using decorators. When you’re cold, you wrap yourself in a sweater. If you’re still cold with a sweater, you can wear a jacket on top. If it’s raining, you put on a raincoat. All of these garments extend your basic behavior but aren’t part of you, and you can easily take off any piece of clothing whenever you don’t need it.

**UML:**



**Implementation:**

1. Create an interface (Shape)
2. Create concrete classes (Circle, Rectangle) implementing the same interface (Shape).
3. Create an abstract (ShapeDecorator) decorator class implementing the Shape interface.
4. Create concrete decorator classes extending the ShapeDecorator class.
5. Use them to decorate the Shape objects.

**Examples:**

