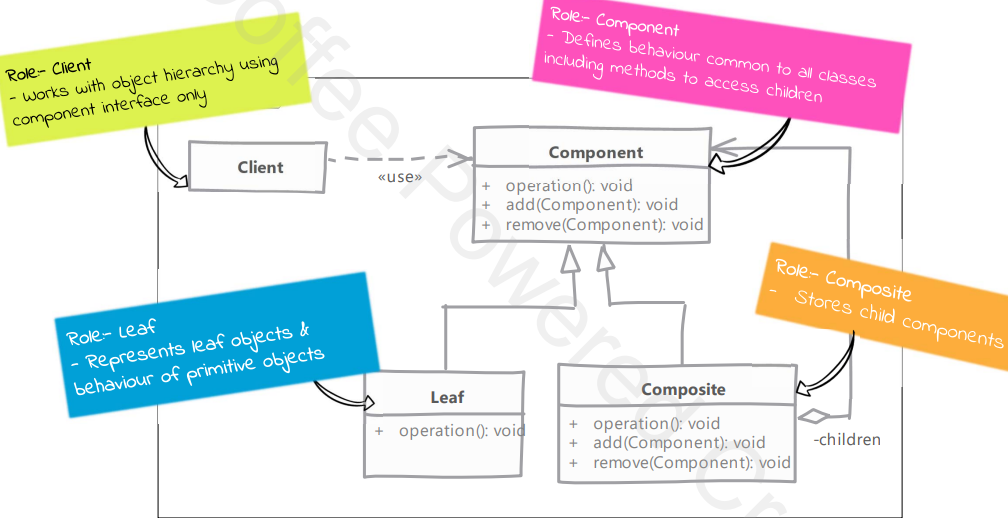
Composite Pattern

Structural Design Pattern

**What is Composite:**

1. When we have a part-hole relationship or hierarchy of objects and we want to be able to treat all objects in this hierarcht uniformly.
2. Enhancedment of the composition concept.
3. Think of composite pattern when dealing with tree structure of objects.

**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.

**Implementation Considerations:**

1. You can provide a method to access parent of a node
2. You can define the collection field to maintain children in base components instead of composite.
3. If leaf objects can be repeated in the hierachy, nodes can be used to save memor and initialization costs.

**Comparison with Decorator:**

1. Composite: deals with tree strcutre of objects.
2. Composite: leaf nodes and composites have same interface and composites simply delegate the operation to children.
3. Decorator: contains another object.
4. Decorator: adds or modifies the behaviour of contained objects and do not have notion of children.

**Examples:**

