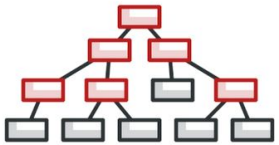
Composite

Pedro Pinto - 115304

April 24th, 2024

When should we use this pattern?

- When you want to:
  - implement a **tree-like** object structure.
  - treat both simple and complex **elements uniformly**.
  - apply the same **uniform operation in all hierarchical objects**



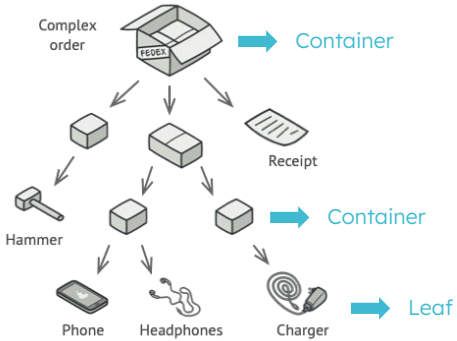
Composite

Lets you compose objects into tree structures and then work with these structures as if they were individual objects.

How to implement this pattern?

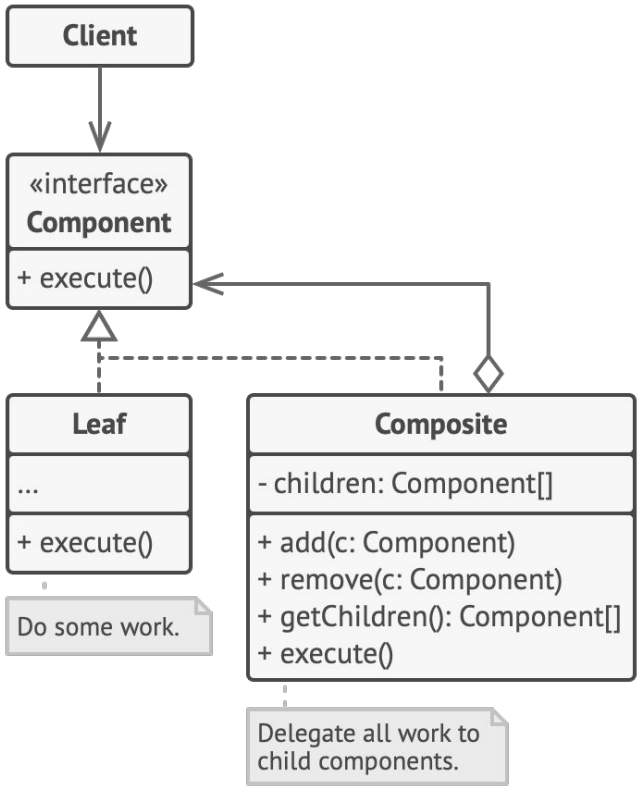
- Break the tree structure into **simple elements and containers**.
- Define a **uniform interface**. With a list of methods that make sense for both simple and complex components. (e.g., *Component*)

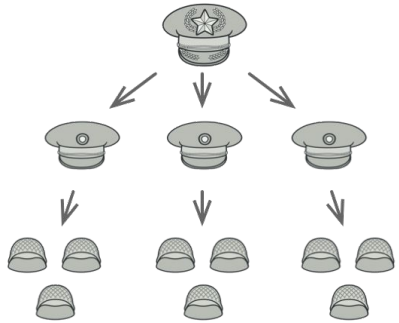
> ***Note:***
> - With an uniform interface you can now **iterate through** all hierarchical objects and apply an uniform method in each iteration.
> - When it applies to simple elements they return the result, otherwise it iterates within all objects in the container



- Define a **leaf class to represent simple elements**. (one or more…)
- Define a container class to represent complex elements. In this class, provide an **array field for storing sub-elements (leaves and containers)**.

(Base) Class Structure





```
@Override
    public void execute() {
        for (Component Leaf : children) {
            Leaf.execute();
        }
    }
```

Code Example(s)

Common interface:

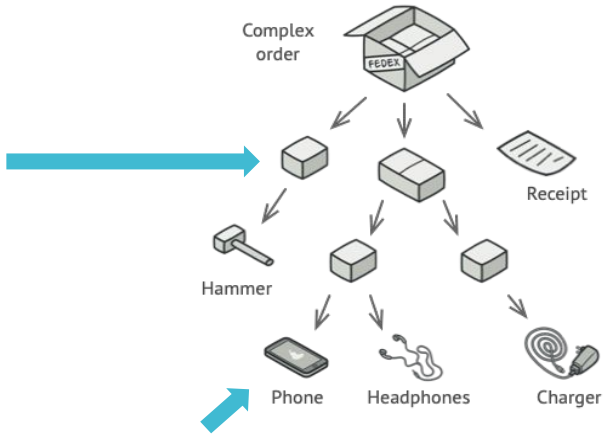
```
interface Component {
    void operation();
}
```

```
// Composite class represents complex components that may have children
class Composite implements Component {
    private List<Component> children = new ArrayList<>();

    public void add(Component component) {
        children.add(component);
    }

    public void remove(Component component) {
        children.remove(component);
    }

    @Override
    public void operation() {
        for (Component child : children) {
            child.operation();
        }
    }
}
```




```
// Leaf class represents end objects of a composition
class Leaf implements Component {
    private String name;

    public Leaf(String name) {
        this.name = name;
    }

    @Override
    public void operation() {
        System.out.println("Leaf " + name);
    }
}
```

Composite

Pedro Pinto - 115304

April 24th, 2024