ESTÁCIO

GILBERTO GIL

Padrão de Projeto Composite

Gustavo Lima Martin – 202109309587
Gabriel Nascimento Batista – 202102086541
Shelton do Nascimento Goes – 202203771851
Nilton Luiz dos Santos Brito – 202204142279
Calvin Felipe Medeiros Brito – 202202388432
Andrei Azevedo da Paz – 202204230411

Helder Guimarães Aragão

2023
SALVADOR/BAHIA

COMPOSITE

O que é o Composite?

O Composite é um padrão de projeto estrutural que permite tratar de maneira uniforme os objetos individuais e composições de objetos, formando uma estrutura hierárquica.

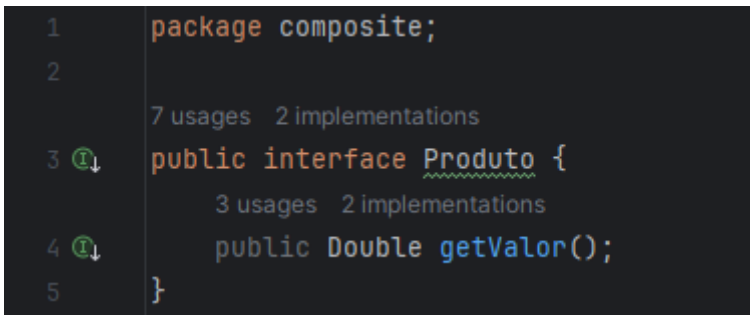
A intenção oficial deste padrão é compor objetos em estruturas de árvore para representar hierarquias. Com o Composite, um objeto individual e um grupo de objetos são tratados da mesma maneira, pois ambos implementam uma interface comum.

Para entender melhor a implementação do Composite, vamos analisar o exemplo a seguir:

Em um supermercado, você possui duas formas de comprar um refrigerante:

- É possível comprar a unidade.
- Ou comprar um fardo com 12 unidades.

Aqui, nós temos um Composite: a noção de que o fardo (estrutura) se comporta como um produto, e que ele pode dizer seu preço delegando o seu valor, somando assim os custos das unidades filhas e utilizando como seu próprio.



No exemplo acima, definimos uma interface Produto, que será implementada tanto pela unidade de “refrigerante”, quanto pelo fardo. O método em comum que será implementado por eles, é o getValor().

```java
package composite;

2 usages
public class Refrigerante implements Produto {
    2 usages
    private String nome;
    2 usages
    private Double valor;
    1 usage
    public Refrigerante(String nome, Double valor) {
        this.nome = nome;
        this.valor = valor;
    }
    1 usage
    public String getNome() {
        return nome;
    }
    3 usages
    @Override
    public Double getValor() {
        return valor;
    }
}
```

O refrigerante por sua vez, possui o valor e um nome. A implementação do getValor() por ser uma única unidade, apenas retorna o seu próprio valor.

```java
package composite;

import ...
2 usages
public class FardoRefrigerante implements Produto {
    4 usages
    private ArrayList<Produto> produtos = new ArrayList<>();
    no usages
    public void add(Produto ...produtos) { this.produtos.addAll(Arrays.asList(produtos)); }
    1 usage
    public void add(Produto produto, int quantidade) {
        for (int i = 0; i < quantidade; i++) {
            this.produtos.add(produto);
        }
    }
    no usages
    public void remove(Produto produto) { produtos.remove(produto); }
    3 usages
    @Override
    public Double getValor() {
        Double soma = 0d;
        for (Produto produto : produtos) {
            soma += produto.getValor();
        }
        return soma;
    }
}
```

Acima temos o fardo de refrigerantes, atuando como Composite, seu “valor” é delegado para as unidades que há dentro dele. Se houver 5 unidades de refrigerante e cada uma com um valor de “5”, o valor do fardo será 25.

```java
package composite;

public class Main{

    public static void main(String[] args) {

        Refrigerante coca_cola = new Refrigerante( nome: "Coca Cola", valor: 8.5);
        FardoRefrigerante fardoCocaCola = new FardoRefrigerante();
        fardoCocaCola.add(coca_cola, quantidade: 6);

        System.out.println(
                            "Produto: " + coca_cola.getNome() +
                            "\n" + "Valor: " + coca_cola.getValor() +
                            "\n" + "Fardo: " + fardoCocaCola.getValor()
                            );
    }

}
```

```
Run  Main ×
"C:\Program Files\Java\zulu17.28.13-ca-jdk17.0.0-win_x64\bin\java.exe" "-javaagent:
Produto: Coca Cola
Valor: 8.5
Fardo: 51.0

Process finished with exit code 0
```

O interessante neste design pattern é que a complexidade desse tipo de estrutura não acaba por aí. Como a classe “FardoRefrigerante” possui uma lista dentro de si do tipo Produto, ela é capaz de suportar até mesmo outros fardos dentro dela mesma.

Referencias:

https://www.softplan.com.br/tech-writers/tech-writers-composite/#:~:text=O%20padr%C3%A3o%20Composite%20%C3%A9%20ideal,%C3%A1rvores%20e%20outras%20estruturas%20hier%C3%A1rquicas.