Padrão de Projeto: STRATEGY

Descrição:

O padrão Strategy permite definir uma família de algoritmos, encapsular cada um deles e torná-los intercambiáveis.

Strategy permite que o algoritmo varie independentemente dos clientes que o utilizam.

Exemplo Implementado: Calculadora de Descontos

Contexto:

Criamos uma interface chamada Desconto que define o método aplicarDesconto.

A partir dela, criamos três estratégias concretas de desconto:

- DescontoClienteComum: aplica 5%

- DescontoVip: aplica 10%

- DescontoPromocional: aplica 20%

A classe CalculadoraDesconto utiliza uma dessas estratégias dinamicamente.

```
// Interface Desconto
public interface Desconto {
       double aplicarDesconto(double valorTotal); // Método que cada estratégia irá
implementar
}

// Estratégia para cliente comum
public class DescontoClienteComum implements Desconto {
    @Override
    public double aplicarDesconto(double valorTotal) {
        return valorTotal * 0.95; // Aplica 5% de desconto
    }
}

// Estratégia para cliente VIP
public class DescontoVip implements Desconto {
    @Override
    public double aplicarDesconto(double valorTotal) {
        return valorTotal * 0.90; // Aplica 10% de desconto
    }
}

// Estratégia para promoção especial
public class DescontoPromocional implements Desconto {
    @Override
    public double aplicarDesconto(double valorTotal) {
        return valorTotal * 0.80; // Aplica 20% de desconto
    }
}

// Classe que utiliza uma estratégia de desconto
public class CalculadoraDesconto {
    private Desconto desconto; // Interface, pode ser qualquer estratégia

    public void setDesconto(Desconto desconto) {
        this.desconto = desconto; // Define a estratégia dinamicamente
    }

    public double calcular(double valor) {
        if (desconto == null) {
            throw new IllegalStateException("Estratégia de desconto não definida!");
        }
        return desconto.aplicarDesconto(valor); // Aplica a estratégia
    }
}

// Classe principal de teste
public class Main {
    public static void main(String[] args) {
        CalculadoraDesconto calculadora = new CalculadoraDesconto();
        double valorCompra = 100.0;

        // Cliente comum
        calculadora.setDesconto(new DescontoClienteComum());
```

```
        System.out.println("Cliente comum: " + calculadora.calcular(valorCompra));

        // Cliente VIP
        calculadora.setDesconto(new DescontoVip());
        System.out.println("Cliente VIP: " + calculadora.calcular(valorCompra));

        // Promoção
        calculadora.setDesconto(new DescontoPromocional());
        System.out.println("Promoção especial: " + calculadora.calcular(valorCompra));
    }
}
```