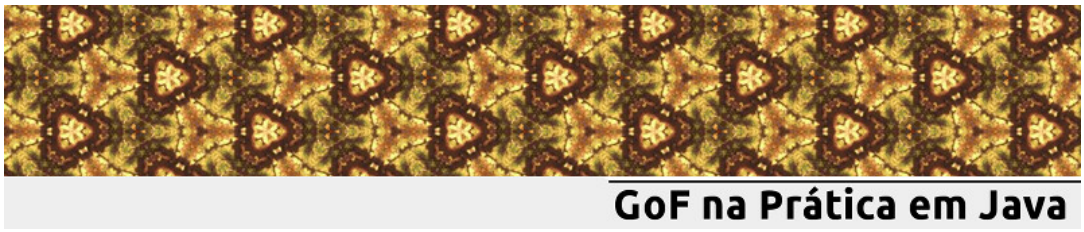
Command

Fernando Anselmo
GoF na Prática em Java

Função deste Padrão

Encapsular uma requisição como um objeto para permitir que os clientes realizem diferentes requisições, filas e suportem operações reversíveis.



1 Ficha do Padrão

Tipo : Comportamental, especificamente voltados para a comunicação entre objetos.

Conhecimentos : Interface e Classes Concretas.

Consequências : Desacopla o objeto que invoca a operação daquele que sabe como executá-la. Podem ser manipulados e extendidos como qualquer outro objeto, pois são objetos de primeira classe. Facilita adicionar novos comandos, pois não existe a necessidade de modificar as classes existentes.

É usado quando : Parametrizar objetos por uma ação a ser executada, ou para especificar, enfileirar ou executar as requisições em diferentes momentos.

2 Problema

Um usuário da classe TV necessita acessar seus comandos a distância, sem ter que tratar diretamente com a classe.

3 Prévia Estrutura de Classes

Classe concreta de implementação da TV:

Listagem 1: *Classe TV*

```
1  class TV {
2    private boolean ligada = false;
3    private byte volume = 0;
4
5    public void ligarDesligar() {
6      if (ligada = !ligada) {
7        System.out.println("TV Ligada.");
8      } else {
9        System.out.println("TV Desligada.");
10     }
11   }
12   public void aumentarVolume() {
```

```
    if (ligada) {
      if (volume < 10) volume++;
      System.out.println("Volume da TV: " + volume);
    } else {
      System.out.println("Ligue a TV.");
    }
  }
  public void abaixarVolume() {
    if (ligada) {
      if (volume > 0) volume--;
      System.out.println("Volume da TV: " + volume);
    } else {
      System.out.println("Ligue a TV.");
    }
  }
}
```

4 Aplicação do Padrão

Interface com a estrutura para a montagem do padrão:

Listagem 2: *Interface Command*

```
interface Command {
  void executar();
}
```

Classe dos comandos de ligar ou desligar a TV:

Listagem 3: *Classe TVLigDesCommand*

```
class TVLigDesCommand implements Command {
  private TV tv;

  public TVLigDesCommand(TV tv) {
    this.tv = tv;
  }
  public void executar() {
    tv.ligarDesligar();
  }
}
```

Classe do comando de aumentar o volume da TV:

Listagem 4: *Classe TVAumVolCommand*

```
class TVAumVolCommand implements Command {
  private TV tv;

  public TVAumVolCommand(TV tv) {
    this.tv = tv;
  }
  public void executar() {
    tv.aumentarVolume();
  }
}
```

Classe do comando de abaixar o volume da TV:

Listagem 5: *Classe TVAbxVolCommand*

```
class TVAbxVolCommand implements Command {
  private TV tv;

  public TVAbxVolCommand(TV tv) {
    this.tv = tv;
  }
  public void executar() {
    tv.abaixarVolume();
  }
}
```

Classe com a execução dos comandos previstos:

Listagem 6: *Classe ControleRemoto*

```
class ControleRemoto {
  private Command btLigDes, btAumVol, btAbxVol;

  public ControleRemoto(TV tv) {
    btLigDes = new TVLigDesCommand(tv);
    btAumVol = new TVAumVolCommand(tv);
    btAbxVol = new TVAbxVolCommand(tv);
  }
  public void pressLigDes() {
    btLigDes.executar();
  }
  public void pressAumVol() {
    btAumVol.executar();
  }
  public void pressAbxVol() {
    btAbxVol.executar();
  }
}
```

Classe com um exemplo de uso por um usuário da classe TV:

Listagem 7: *Classe Espectador*

```
public class Espectador {
  public static void main(String[] args) {
    new Espectador().assistir();
  }
  public void assistir() {
    ControleRemoto remote = new ControleRemoto(new TV());
    java.util.Scanner sc = new java.util.Scanner(System.in);
    byte opcao = 0;
    do {
      System.out.println("Selecione o comando (0 para sair)");
      System.out.println("1 - Ligar/Desligar");
      System.out.println("2 - Aumentar Volume");
      System.out.println("3 - Abaixar Volume");
      opcao = sc.nextByte();
      switch (opcao) {
        case 1 : remote.pressLigDes(); break;
        case 2 : remote.pressAumVol(); break;
        case 3 : remote.pressAbxVol(); break;
      }
    } while (opcao != 0);
  }
}
```

Referências

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides *Design Patterns. Elements of Reusable Object-Oriented Software 1 ed.* Estados Unidos, Addison-Wesley, 1995, ISBN 0-201-63361-2.