Padrões de Projeto de Software Orientados a Objetos

Tecnologia em Análise e Desenvolvimento de Sistemas

Paulo Mauricio Gonçalves Júnior

Instituto Federal de Educação, Ciência e Tecnologia de Pernambuco

23 de abril de 2018

Parte I

Template Method

Introdução I

- Este padrão serve para encapsular partes de um algoritmo de forma que subclasses podem executar partes da computação.
- Por exemplo, para fazermos café e chá os passos são os seguintes:
  - Café:
    1. Ferver água
    2. Preparar café em água fervente
    3. Despeje o café no copo
    4. Adicione açucar e leite
  - Chá:
    1. Ferver água
    2. Colocar chá em água fervente
    3. Despeje o chá no copo
    4. Adicione limão

Introdução II

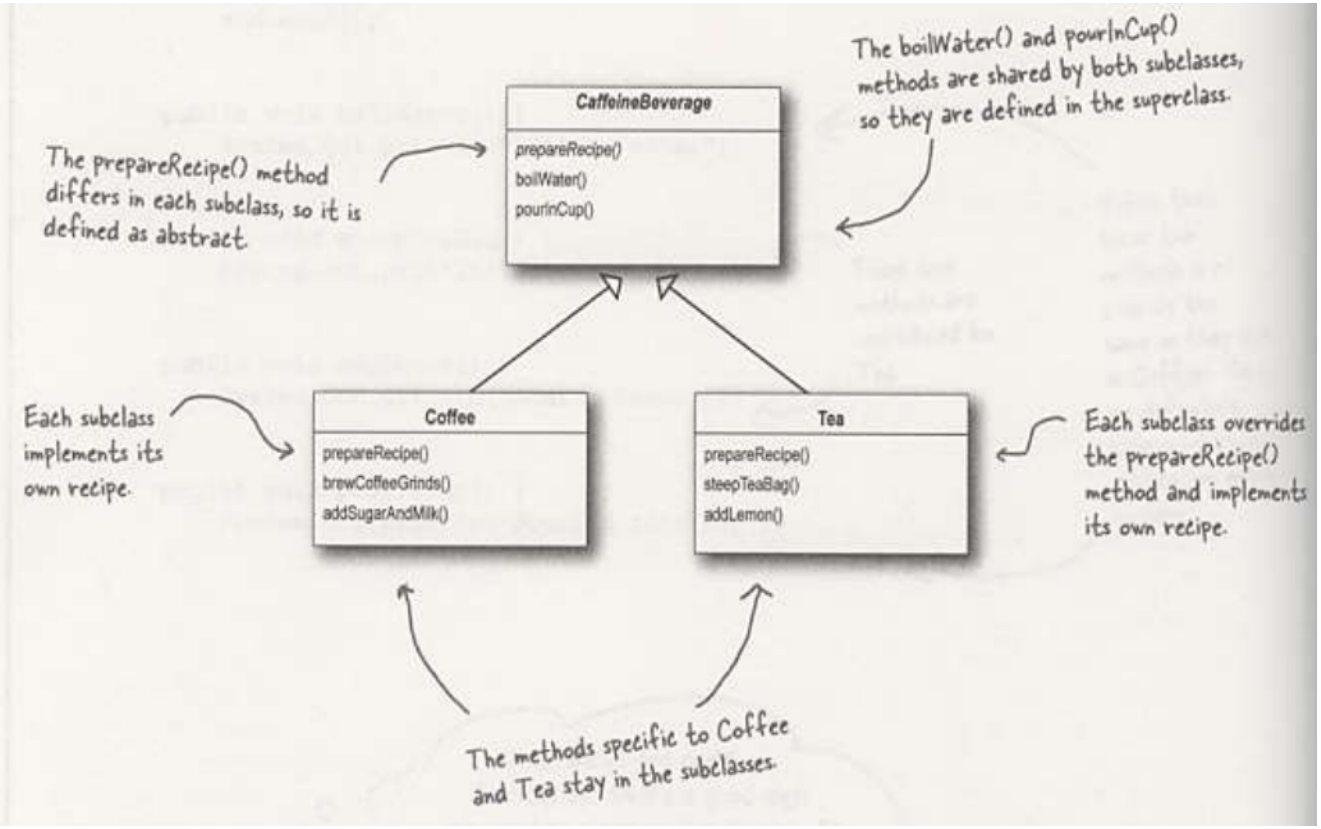
```
public class Coffee {
  void prepareRecipe() {
    boilWater();
    brewCoffeeGrinds();
    pourInCup();
    addSugarAndMilk();
  }
  public void boilWater() {
    System.out.println("Boiling water");
  }
  public void brewCoffeeGrinds() {
    System.out.println("Dripping Coffee through filter");
  }
  public void pourInCup() {
    System.out.println("Pouring into cup");
  }
  public void addSugarAndMilk() {
    System.out.println("Adding Sugar and Milk");
  }
}
```

Introdução III

```
public class Tea {
  void prepareRecipe() {
    boilWater();
    steepTeaBag();
    pourInCup();
    addLemon();
  }
  public void boilWater() {
    System.out.println("Boiling water");
  }
  public void steepTeaBag() {
    System.out.println("Steeping the tea");
  }
  public void addLemon() {
    System.out.println("Adding Lemon");
  }
  public void pourInCup() {
    System.out.println("Pouring into cup");
  }
}
```

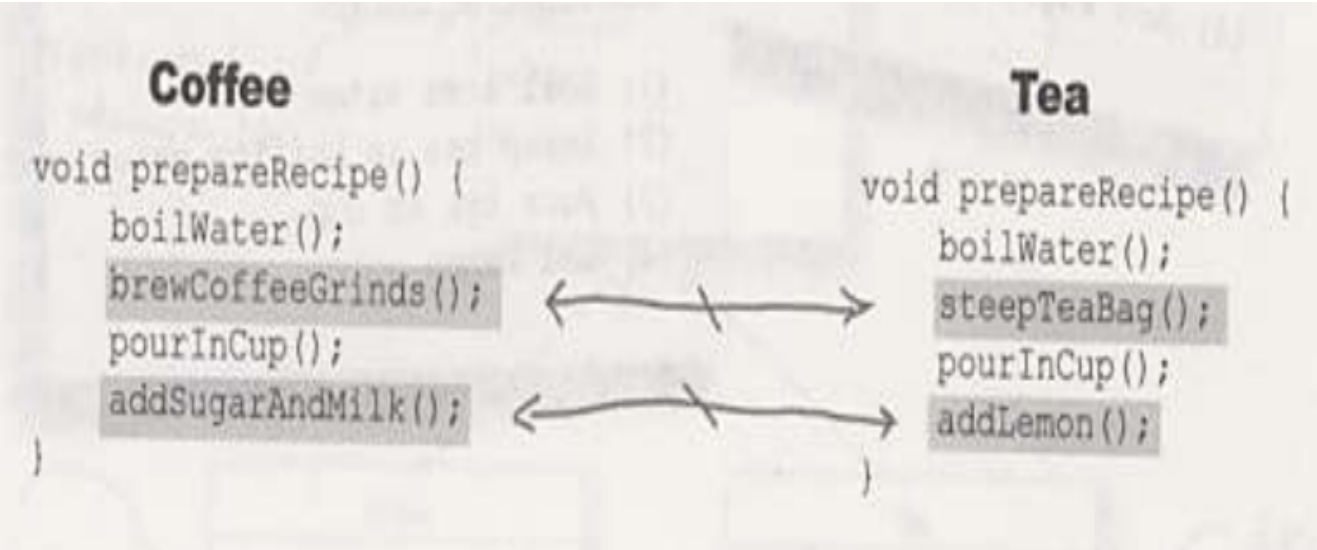
Introdução IV

- Existe uma duplicação de código. Podemos então abstrair as partes comuns em uma classe base já que preparar café e chá são semelhantes.



Introdução V

- Podemos abstrair o método de preparar receitas.



- Vamos abstrair a colocação do café ou chá e a finalização.

Introdução VI

```
public abstract class CaffeineBeverage {

  final void prepareRecipe() {
    boilWater();
    brew();
    pourInCup();
    addCondiments();
  }

  abstract void brew();
  abstract void addCondiments();

  void boilWater() {
    System.out.println("Boiling water");
  }

  void pourInCup() {
    System.out.println("Pouring into cup");
  }
}
```

Introdução VII

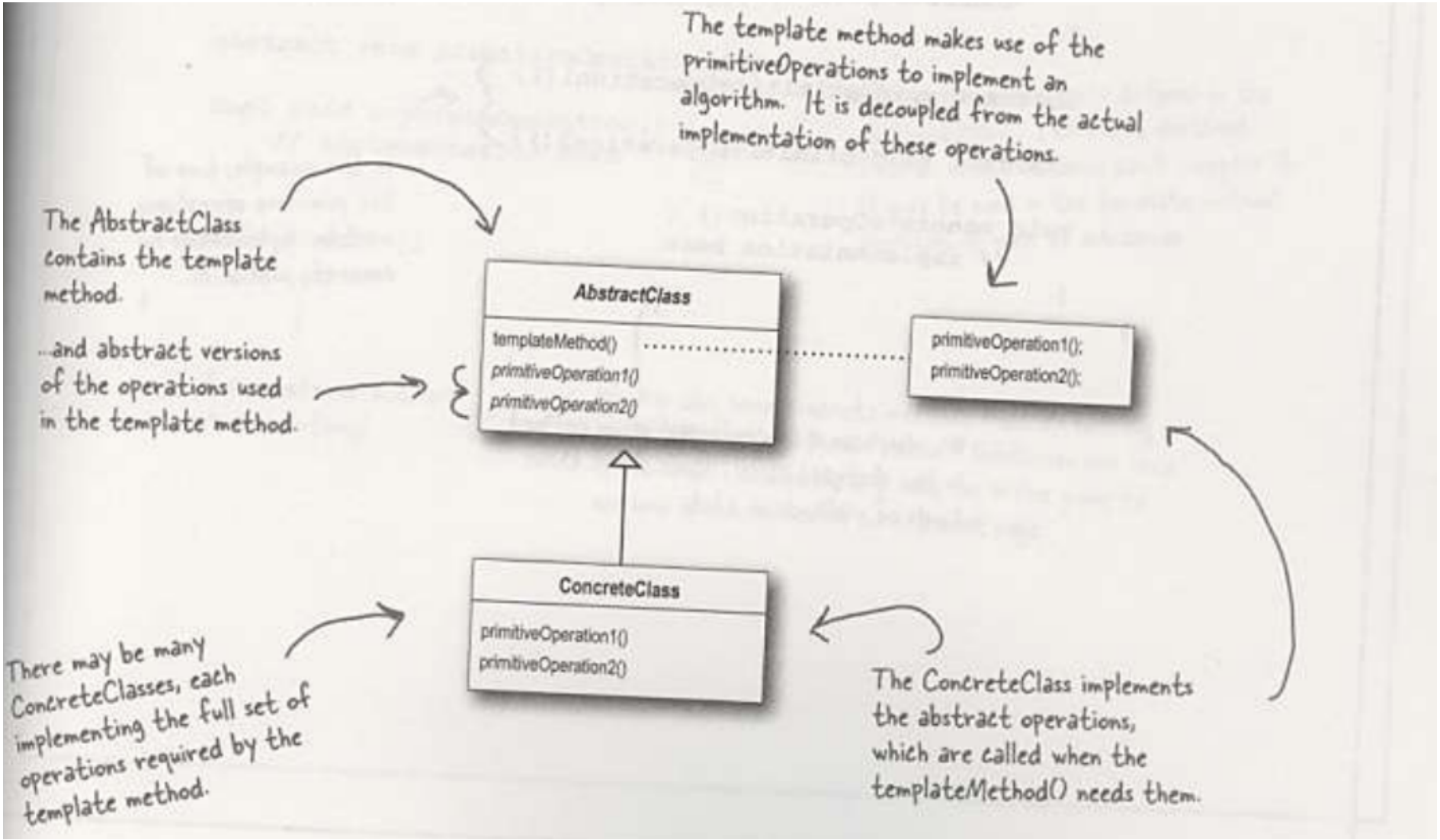
```
public class Coffee extends CaffeineBeverage {
  public void brew() {
    System.out.println("Dripping Coffee through filter");
  }
  public void addCondiments() {
    System.out.println("Adding Sugar and Milk");
  }
}

public class Tea extends CaffeineBeverage {
  public void brew() {
    System.out.println("Steeping the tea");
  }
  public void addCondiments() {
    System.out.println("Adding Lemon");
  }
}
```

Introdução VIII

Definição

O padrão Método Template define o esqueleto de um algoritmo em um método, com alguns passos sendo executados por subclasses. Método Template permite a subclasses redefinir alguns passos de um algoritmo sem mudar a estrutura do algoritmo.

Introdução IX



Gancho I

- Um gancho é um método declarado na classe abstrata, mas é apenas dada uma implementação vazia ou padrão.
- Existem de diversos tipos de ganchos.

```
public abstract class CaffeineBeverageWithHook {
  final void prepareRecipe() {
    boilWater();
    brew();
    pourInCup();
    if (customerWantsCondiments()) {
      addCondiments();
    }
  }
  abstract void brew();
  abstract void addCondiments();
  void boilWater() {
    System.out.println("Boiling water");
  }
  void pourInCup() {
```

Gancho II

```
    System.out.println("Pouring into cup");
  }
  boolean customerWantsCondiments() {
    return true;
  }
}

public class CoffeeWithHook extends CaffeineBeverageWithHook {
  public void brew() {
    System.out.println("Dripping Coffee through filter");
  }
  public void addCondiments() {
    System.out.println("Adding Sugar and Milk");
  }
  public boolean customerWantsCondiments() {

    String answer = getUserInput();

    if (answer.toLowerCase().startsWith("y")) {
      return true;
    } else {
```

Gancho III

```
      return false;
    }
  }
  private String getUserInput() {
    String answer = null;
    System.out.print("Would you like milk and sugar with your
        coffee (y/n)? ");
    BufferedReader in = new BufferedReader(new InputStreamReader(
        System.in));
    try {
      answer = in.readLine();
    } catch (IOException ioe) {
      System.err.println("IO error trying to read your answer");
    }
    if (answer == null) {
      return "no";
    }
    return answer;
  }
}
```

Gancho IV

Princípio de Projeto

Princípio Hollywood Não me chame, eu chamo você.