Centro Universitário UniBH
Ciência da Computação
Práticas de Programação
Professor: Lucas Schmidt

Aula Prática - 22 Padrões de Comportamento - Template Method

Intenção

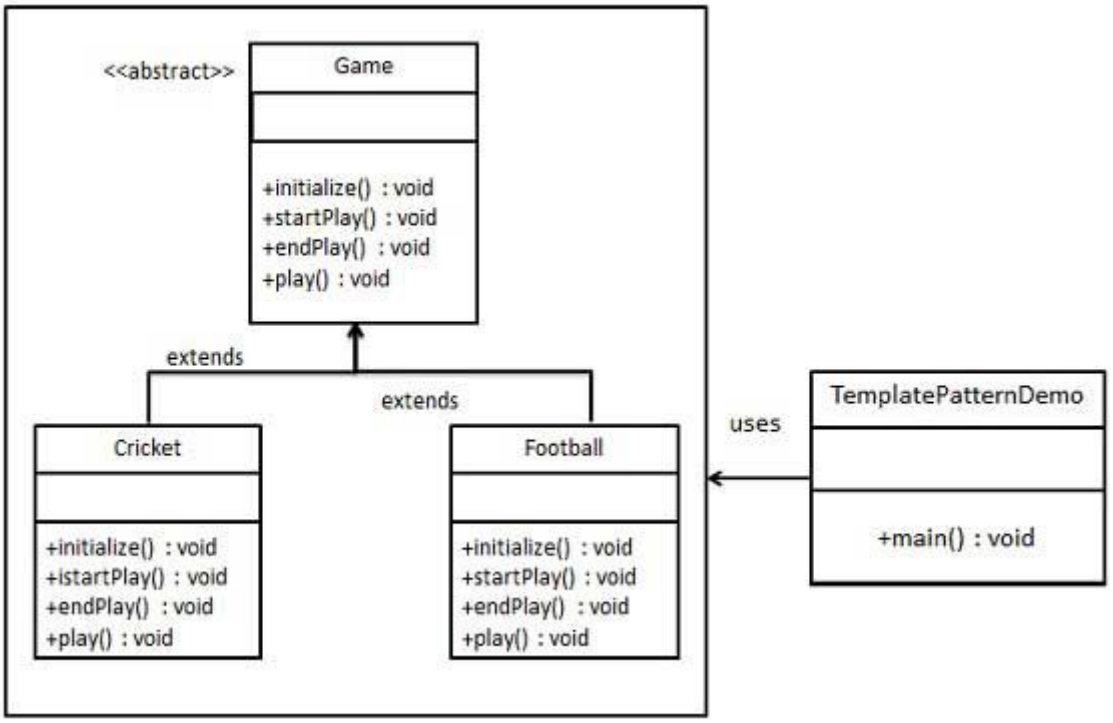
Definir o esqueleto de um algorítmo numa classe, delegando alguns passos às subclasses. Permite que as subclasses alterem partes do algorítmo, sem mudar sua estrutura geral.

Usar este padrão quando…

- Quiser implementar partes invariantes de um algorítmo na superclasse e deixar o restante para as subclasses;
- Comportamento comum de subclasses deve ser generalizado para evitar duplicidade de código;
- Quiser controlar o que as subclasses podem estender (métodos finais).

Vantagens e desvantagens

- Reuso de código:
  - Partes de um algorítmo são reutilizadas por todas as subclasses.
- Controle:
  - É possível permitir o que as subclasses podem estender (métodos finais).
- Comportamento padrão extensível:
  - Superclasse pode definir o comportamento padrão e permitir sobrescrita.



Passo 1
Crie uma classe abstrata com um método de modelo sendo final.

Game.java

```
public abstract class Game {
   abstract void initialize();
   abstract void startPlay();
   abstract void endPlay();

   //template method
   public final void play(){

      //initialize the game
      initialize();

      //start game
      startPlay();

      //end game
      endPlay();
   }
}
```

Passo 2
Crie classes concretas estendendo a classe acima.

Cricket.java

```
public class Cricket extends Game {

   @Override
   void endPlay() {
      System.out.println("Cricket Game Finished!");
   }

   @Override
   void initialize() {
      System.out.println("Cricket Game Initialized! Start playing.");
   }

   @Override
   void startPlay() {
      System.out.println("Cricket Game Started. Enjoy the game!");
   }
}
```

Football.java

```
public class Football extends Game {

   @Override
   void endPlay() {
      System.out.println("Football Game Finished!");
   }

   @Override
   void initialize() {
      System.out.println("Football Game Initialized! Start playing.");
   }

   @Override
   void startPlay() {
      System.out.println("Football Game Started. Enjoy the game!");
   }
}
```

Passo 3
Use o método play() da classe Game.java para demonstrar uma forma definida de jogar o jogo.

TemplatePatternDemo.java

```
public class TemplatePatternDemo {
   public static void main(String[] args) {

      Game game = new Cricket();
      game.play();
      System.out.println();
      game = new Football();
      game.play();
   }
}
```

Passo 4

Teste sua implementação!