Exemplo Flyweight pattern

Certified Developer
The Ultimate Tech Degree

DigitalHouse >
Coding School

Exemplo Flyweight pattern

Suponha que temos um **livro de receitas** que contém **receitas** que estão em coleções diferentes, como: carne; sopas; saladas; ou em diferentes **categorias**, como comida italiana, mexicana, argentina e fast food.

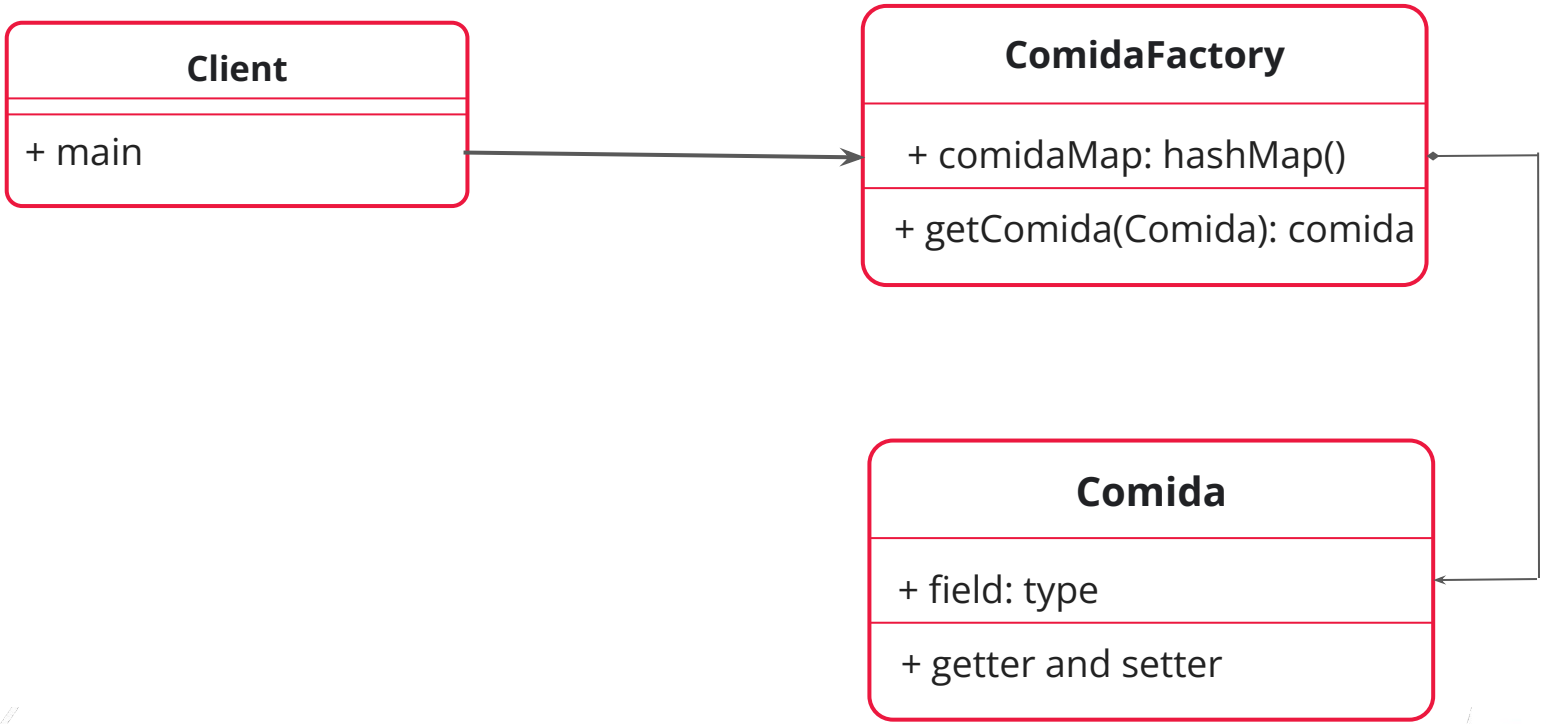
Exemplo Flyweight pattern

A receita de um hambúrguer poderia estar em várias **seções**: americana, carnes, fast food, etc. Se precisássemos ter um objeto receita de hambúrguer em cada uma das coleções, teríamos um desempenho muito baixo e consumiríamos muita memória.

Então, o cliente solicita um objeto da FlyweightFactory. Esta verifica se ele existe no cache e, em caso contrário, cria um novo. Flyweight compartilha o estado dos objetos.

Vejamos como representamos esta solução em um diagrama UML.

Solução


Client
+ main
ComidaFactory
+ comidaMap: hashMap()
+ getComida(Comida): comida
Comida
+ field: type
+ getter and setter

Implementação do diagrama UML

Código da **classe** que define o método antes de criar o objeto. Valida se já existe um objeto idêntico ao que está sendo solicitado. Em caso afirmativo, retorna o objeto existente; se não existir, cria o novo objeto e o armazena em cache para ser reutilizado posteriormente.

```
import java.util.HashMap;
public class ComidaFactory {
   private static final HashMap<String, Comida> comidaMap = new HashMap();

   public static Comida getComida(String tipoComida) {
       Comida comida = (Comida) comidaMap.get(tipoComida);
```

Implementação do diagrama UML

```
        if(comida == null) {
            comida = new Comida(tipoComida);
            comidaMap.put(tipoComida, comida);
            System.out.println("Criando objeto comida de tipo: "
+ tipoComida);
        }
        return comida;
    }
}
```

Código da classe

Implementação do diagrama UML

```
public class Comida {
    private String tipoComida;
    private int preco;
    private boolean temAlface;

    public Comida(String tipoComida) {
        this.tipoComida = tipoComida;
    }

    public String getTipoComida() {
        return tipoComida;
    }
```

Objeto comida, com atributos **preco**, **tipoComida** e **temAlface**.

Implementação do diagrama UML

```
public int getPreco() {
    return preco;
}

public void setPreco(int preco) {
    this.preco = preco;
}
public boolean isTemAlface() {
    return temAlface;
}

public void setTemAlface(boolean temAlface) {
    this.temAlface = temAlface;
}
```

Objeto comida, com atributos **preco**, **tipoComida** e **temAlface**.

Implementação do diagrama UML

```
    public void descricaoDaComida ()
{System.out.println("É um/uma " + getTipoComida() +
" que custa: " + getPreco());
    }}
```

Objeto comida, com atributos **preco**, **tipoComida** e **temAlface**.


DigitalHouse>
Coding School