PIZZA WORLD

ADVANCED PROGRAMMING LAB-2 PROJECT

The Team

ABHAY GARG

181B007

ADITI NEGI

181B013

Project Faculty - Dr. Prateek Pandey

Introduction

- Task

  To solve real world problems using design pattern

- Idea

  To prepare different types of pizzas

The Problem Statement

We have a Pizza shop which makes two types of pizzas one is Italian and other one is Veggie. In both types of pizzas we provide three types of condiments : Oil, Vinegar, Butter. We offer 3 types of meat(Pork, Pepperoni, Chicken) and several types of veggies. Customer can order any pizza based on their preference.

Task

Our project will make the pizza according to the choice of customer.

Design Pattern Used

Template Method Pattern...

Template Method Pattern

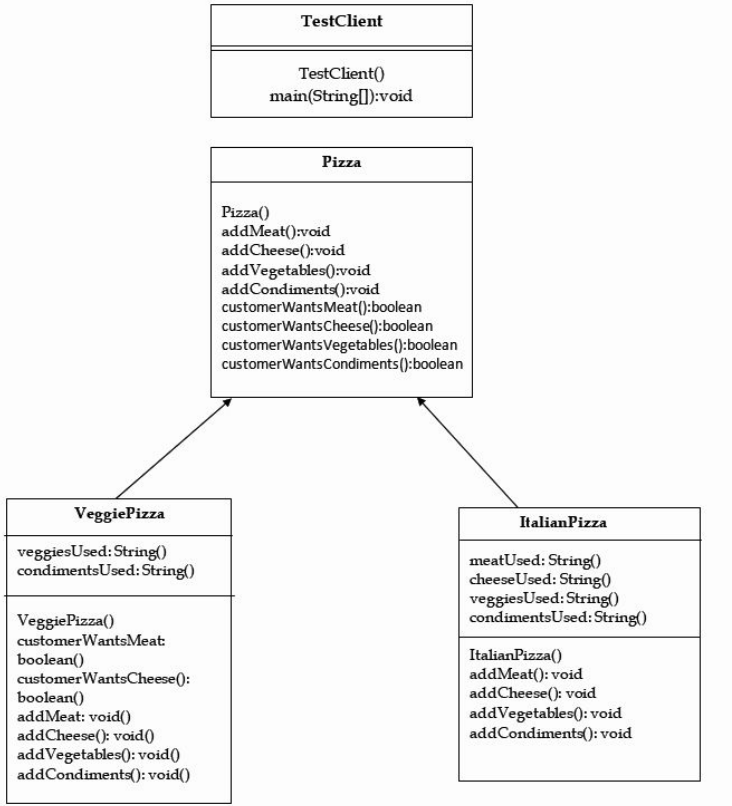
In Template pattern, an abstract class exposes defined way(s)/template(s) to execute its methods. Its subclasses can override the method implementation as per need but the invocation is to be in the same way as defined by an abstract class. This pattern comes under **behavior pattern category**.

Importance

The template method is used for the following reasons :

- Let subclasses implement varying behavior (through **method overriding)**
- Avoid duplication in the code, the general workflow structure is implemented once in the abstract class's algorithm,
- necessary variations are implemented in the subclasses.

SOLUTION DESIGN



Screenshots...

MAIN CLASS

TestClient.java | ItalianPizza.java | Pizza.java | VeggiePizza.java

```java
public class TestClient {

    public static void main(String[] args) {

        System.out.println("Start: TemplateMethod\n");

        //
        // Create ItalianPizza
        //

        Pizza customer1Pizza = new ItalianPizza();
        customer1Pizza.templateMethod();

        System.out.println("\n");

        //
        // Create VeggiePizza
        //

        Pizza customer2Pizza = new VeggiePizza();
        customer2Pizza.templateMethod();

    }

}
```

PIZZA CLASS


TestClient.java
ItalianPizza.java
Pizza.java
VeggiePizza.java
public abstract class Pizza {
//
// TemplateMethod
final void templateMethod() {
cutBase();
if (customerWantsMeat()) {
addMeat();
}
if (customerWantsCheese()) {
addCheese();
}
if (customerWantsVegetables()) {
addVegetables();
}
if (customerWantsCondiments()) {
addCondiments();
}
packThePizza();

CONTINUED....

TestClient.java | ItalianPizza.java | Pizza.java | VeggiePizza.java

```
        addCheese();
    }

    if (customerWantsVegetables()) {
        addVegetables();
    }

    if (customerWantsCondiments()) {
        addCondiments();
    }

        packThePizza();

}

public void cutBase() {
    System.out.println("The Pizza is cut into slices");
}

public void packThePizza() {
    System.out.println("Pack the Pizza");
}

abstract void addMeat();

abstract void addCheese();

abstract void addVegetables();
```

ITALIAN PIZZA CLASS

TestClient.java | ItalianPizza.java | Pizza.java | VeggiePizza.java

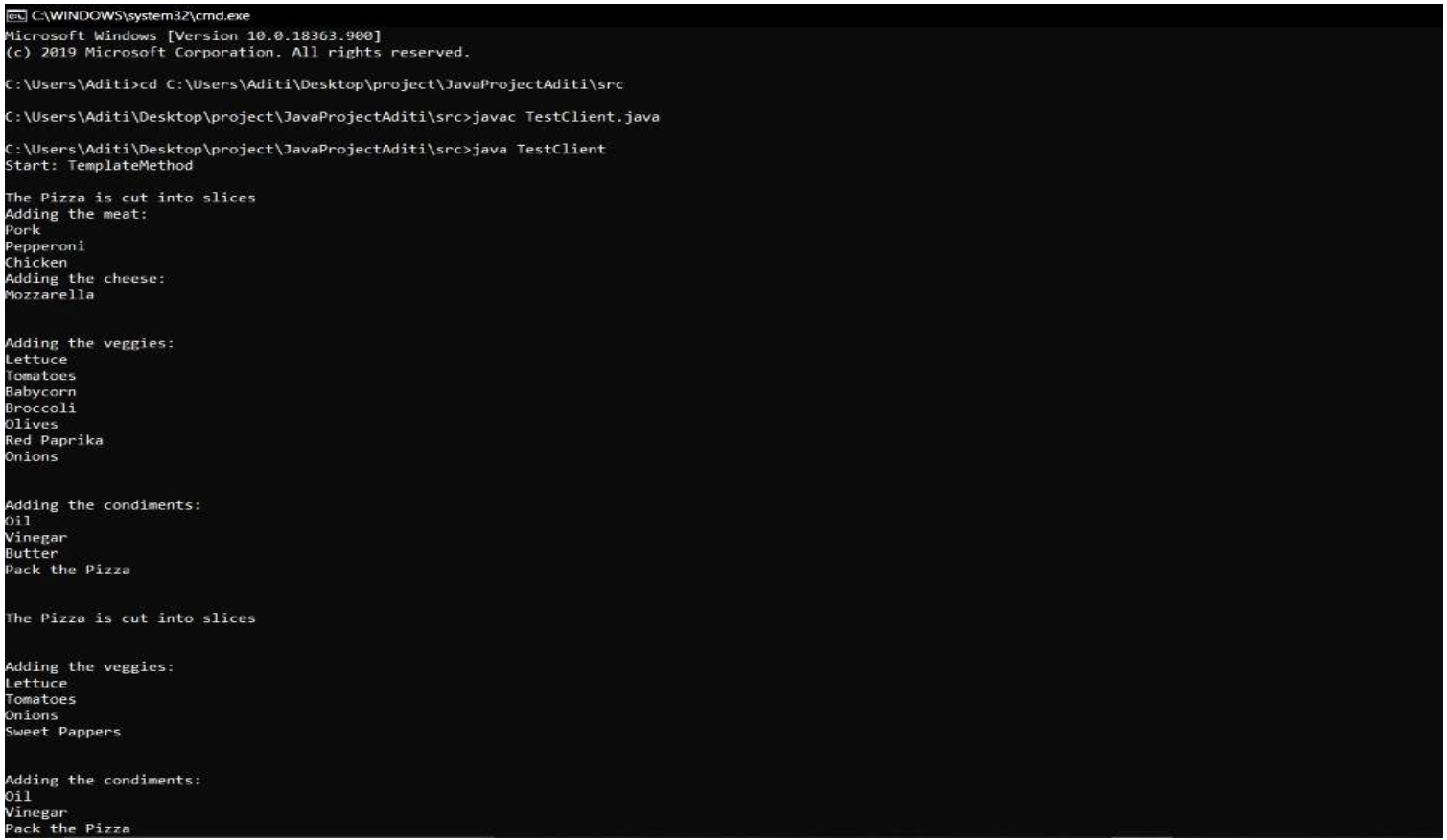
```java
public class ItalianPizza extends Pizza {

    String[] meatUsed = { "Pork", "Pepperoni", "Chicken" };
    String[] cheeseUsed = { "Mozzarella" };
    String[] veggiesUsed = { "Lettuce", "Tomatoes", "Babycorn","Broccoli", "Olives","Red Paprika", "Onions" };
    String[] condimentsUsed = { "Oil", "Vinegar","Butter"};

    @Override
    void addMeat() {

        System.out.println("Adding the meat: ");

        for (String meat : meatUsed) {
            System.out.println(meat + " ");
        }

    }

    @Override
    void addCheese() {

        System.out.println("Adding the cheese: ");

        for (String cheese : cheeseUsed) {
            System.out.println(cheese + " ");
        }

    }
```

VEGGIE PIZZA CLASS

TestClient.java | ItalianPizza.java | Pizza.java | VeggiePizza.java

```
1
2  public class VeggiePizza extends Pizza {
3
4      String[] veggiesUsed = { "Lettuce", "Tomatoes", "Onions", "Sweet Pappers" };
5      String[] condimentsUsed = { "Oil", "Vinegar" };
6
7
8      boolean customerWantsMeat() {
9          return false; //false
10     }
11
12     boolean customerWantsCheese() {
13         return false; //false
14     }
15
16     //
17     //
18     //
19
20     @Override
21     void addMeat() {
22
23
24     }
25
26     @Override
27     void addCheese() {
28
```

OUTPUT


C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18363.900]
(c) 2019 Microsoft Corporation. All rights reserved.
C:\Users\Aditi>cd C:\Users\Aditi\Desktop\project\JavaProjectAditi\src
C:\Users\Aditi\Desktop\project\JavaProjectAditi\src>javac TestClient.java
C:\Users\Aditi\Desktop\project\JavaProjectAditi\src>java TestClient
Start: TemplateMethod
The Pizza is cut into slices
Adding the meat:
Pork
Pepperoni
Chicken
Adding the cheese:
Mozzarella
Adding the veggies:
Lettuce
Tomatoes
Babycorn
Broccoli
Olives
Red Paprika
Onions
Adding the condiments:
Oil
Vinegar
Butter
Pack the Pizza
The Pizza is cut into slices
Adding the veggies:
Lettuce
Tomatoes
Onions
Sweet Pappers
Adding the condiments:
Oil
Vinegar
Pack the Pizza

THANK YOU !