Text

Description automatically generated with medium confidence

wzorce projektowe

Sprawozdanie

Ćwiczenie 3 builder

Adam Zajler

Informatyka, programowanie

34\_Inf\_P\_NW\_6

Rok akademicki 2024

# Cel

Sprawozdanie powstało w ramach ćwiczenia na temat wzorca projektowego builder. Ma to służyć udokumentowaniu wiedzy wykorzystanej w zadaniu. Program służy do tworzenie pizzy na podstawie podanych przez użytkownika składników.

# Opis rozwiązania

Wzorzec projektowy Builder zastosowano w języku C# do skonfigurowania i zamówienia pizzy. Użytkownik wybiera składniki krok po kroku, a interfejs IPizzaBuilder definiuje metody ich dodawania. Klasa PizzaBuilder implementuje te metody i buduje obiekt Pizza. Klasa Director zarządza procesem budowy, wywołując odpowiednie metody w odpowiedniej kolejności. Klasa Program(Klient/Client) demonstruje użycie tych klas do skonfigurowania i zamówienia pizzy.

**Propozycja diagramu klas**

**Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie**

**Pseudokod:**

1. **Zdefiniuj klasę Pizza z polami:**
   1. Ciasto
   2. Ser
   3. Mieso
   4. Warzywa
   5. Przyprawy
2. **Zdefiniuj interfejs IPizzaBuilder z metodami do dodawania każdego składnika:**
   1. DodajCiasto(składnikCiasta)
   2. DodajMięso(składnikMięsa)
   3. DodajSer(składnikSera)
   4. DodajWarzywa(składnikWarzyw)
   5. DodajPrzyprawy(składnikPrzypraw)
   6. PobierzPizzę()
3. **Zdefiniuj klasę PizzaBuilder implementującą IPizzaBuilder:**
   1. Przechowuj obiekt Pizza wewnętrznie
   2. Zaimplementuj metody ustawiające każdy składnik na wewnętrznym obiekcie Pizza
4. **Zdefiniuj klasę Director:**
   1. Przyjmij obiekt IPizzaBuilder jako argument
   2. Zdefiniuj metody do konstruowania różnych pizz (np. KonstruujHawajskąPizzę)
   3. Wywołaj metody na builderze, aby ustawić składniki dla danej pizzy
   4. Zwróć zbudowany obiekt Pizza
5. **W Main:**
   1. Utwórz obiekt PizzaBuilder
   2. Utwórz obiekt Director
   3. Wywołaj metodę Directora (np. KonstruujHawajskąPizzę) z builderem
   4. Pobierz zbudowany obiekt Pizza
   5. Wydrukuj szczegóły pizzy

# Implementacja

using Builder;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Policy;

using System.Text;

using System.Threading.Tasks;

namespace Builder

{

// Tworzymy klase która zawiera wszystkie dane jakie powinen lub MOŻE zawierać produkt.

// Dodatkowo powinny być tu funkcje do ustawiania / pobierania danych

public class Pizza

{

private String ciasto;

private String ser;

private String mieso;

private String warzywa;

private String spice;

public String getCiasto()

{

return ciasto;

}

public void setCiasto(String ciasto)

{

this.ciasto = ciasto;

}

public String getSer()

{

return ser;

}

public void setSer(String ser)

{

this.ser = ser;

}

public String getMieso()

{

return mieso;

}

public void setMieso(String mieso)

{

this.mieso = mieso;

}

public String getWarzywa()

{

return warzywa;

}

public void setWarzywa(String warzywa)

{

this.warzywa = warzywa;

}

public String getSpices()

{

return spice;

}

public void setSpices(String spice)

{

this.spice = spice;

}

}

// Interfejs który definiuje metody buildera

public interface IPizzaBuilder

{

void AddCrust(string crust);

void AddMeats(string meats);

void AddCheeses(string cheeses);

void AddVegetables(string vegetables);

void AddSpices(string spices);

Pizza GetPizza();

}

// Builder pizzy

public class PizzaBuilder: IPizzaBuilder

{

private Pizza pizza = new Pizza();

public void AddCrust(string crust)

{

pizza.setCiasto(crust);

}

public void AddMeats(string meats)

{

pizza.setMieso(meats);

}

public void AddCheeses(string cheeses)

{

pizza.setSer(cheeses);

}

public void AddVegetables(string vegetables)

{

pizza.setWarzywa(vegetables);

}

public void AddSpices(string spices)

{

pizza.setSpices(spices);

}

public Pizza GetPizza()

{

return pizza;

}

}

// Director zbiera różne metody budowania pizzy

public class Director

{

public Pizza ConstructHawaiiPizza(IPizzaBuilder builder)

{

builder.AddCrust("Cienkie ciasto");

builder.AddMeats("Szynka");

builder.AddCheeses("Ser x2");

builder.AddVegetables("Ananas");

builder.AddSpices("Oregano, Bazylia");

return builder.GetPizza();

}

}

}

internal class Program

{

static void Main(string[] args)

{

IPizzaBuilder pizza = new PizzaBuilder();

var director = new Director();

Pizza hawaiianPizza = director.ConstructHawaiiPizza(pizza);

Console.WriteLine($"Hawajska: {hawaiianPizza.getCiasto()}, {hawaiianPizza.getSer()}, {hawaiianPizza.getWarzywa()}, {hawaiianPizza.getSpices()}");

Console.ReadKey();

}

}

# Podsumowanie

W tej części proszę krótko podsumować swoją pracę:

* Uzasadnić, dlaczego do rozwiązania zadania został użyty określony wzorzec projektowy
* Czy implementacja wzorca w celu optymalnego rozwiązania zadania powiodła się i dlaczego
* Zaproponować inny wzorzec projektowy, jeśli wydaje się to lepszym sposobem na rozwiązanie zadania lub uzasadnić, dlaczego ten jest najlepszy

# Lista załączników

<https://github.com/AdamZajler/wzorce-projektowe>