Text

Description automatically generated with medium confidence

wzorce projektowe

Sprawozdanie

Ćwiczenie 9 - Observer

Adam Zajler

Informatyka, programowanie

34\_Inf\_P\_NW\_6

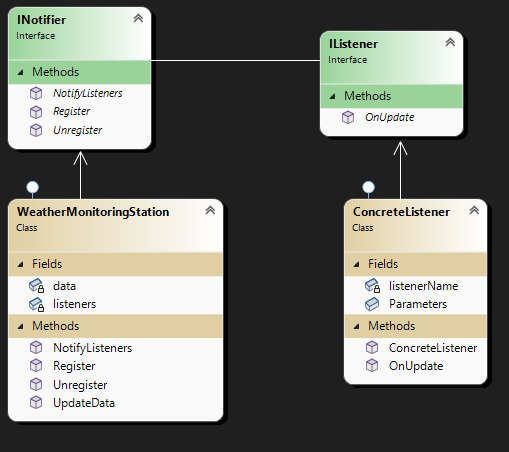
Rok akademicki 2023/24

# Cel

Poniższy dokument został stworzony w celu zapisania przebiegu ćwiczenia. Podczas zajęć został opracowany program w języku C#, który ma za zadanie monitorować dane stacji pogodowej. Program ten został zbudowany na podstawie wzorca projektowego Obserwator.

# Opis rozwiązania

Rozwiązaniem zadania jest program napisany w języku C#, demonstrujący zastosowanie wzorca projektowego Obserwator. W tej propozycji użytkownik może dostarczyć konkretną prognozę pogody, która jest następnie przekazywana do odpowiednich obserwatorów.



# Implementacja

using System;

using System.Collections.Generic;

using System.Linq;

namespace Zadanie\_9

{

interface IListener

{

void OnUpdate(MeteorologicalData data);

}

interface INotifier

{

void Register(IListener listener);

void Unregister(IListener listener);

void NotifyListeners(MeteorologicalData data);

}

class MeteorologicalData

{

public double Temperature;

public string Cloudiness;

public int PrecipitationProbability;

public MeteorologicalData(double temperature, string cloudiness, int precipitationProbability)

{

this.Temperature = temperature;

this.Cloudiness = cloudiness;

this.PrecipitationProbability = precipitationProbability;

}

}

class ConcreteListener : IListener

{

private string listenerName;

public string[] Parameters;

public ConcreteListener(string listenerName, string[] parameters)

{

this.listenerName = listenerName;

this.Parameters = parameters;

}

public void OnUpdate(MeteorologicalData data)

{

foreach (string parameter in Parameters)

{

var singleData = typeof(MeteorologicalData).GetField(parameter).GetValue(data);

if (singleData != null)

{

Console.WriteLine(listenerName + ": " + singleData);

}

}

}

}

class WeatherMonitoringStation : INotifier

{

private List<IListener> listeners = new List<IListener>();

private MeteorologicalData data;

public void Register(IListener listener)

{

listeners.Add(listener);

}

public void Unregister(IListener listener)

{

listeners.Remove(listener);

}

public void NotifyListeners(MeteorologicalData data)

{

foreach (IListener listener in listeners)

{

listener.OnUpdate(data);

}

}

public void UpdateData(MeteorologicalData data)

{

this.data = data;

NotifyListeners(this.data);

}

}

class Program

{

static void Main(string[] args)

{

WeatherMonitoringStation weatherStation = new WeatherMonitoringStation();

ConcreteListener tempListener = new ConcreteListener("Temperature Listener", new string[] { "Temperature" });

ConcreteListener cloudinessListener = new ConcreteListener("Cloudiness Listener", new string[] { "Cloudiness" });

ConcreteListener rainChanceListener = new ConcreteListener("Rain Chance Listener", new string[] { "PrecipitationProbability" });

weatherStation.Register(tempListener);

weatherStation.Register(cloudinessListener);

weatherStation.Register(rainChanceListener);

weatherStation.UpdateData(new MeteorologicalData(25.00, "Cloudy", 70));

Console.ReadKey();

}

}

}

# Podsumowanie

# Do wykonania zadania został użyty wzorzec projektowy Observer. Ten wzorzec jest idealny do tego typu zastosowań, ponieważ umożliwia aktualizację i wyświetlanie danych natychmiast po ich zmianie.

# Lista załączników

[AdamZajler/wzorce-projektowe (github.com)](https://github.com/AdamZajler/wzorce-projektowe/tree/main)