**10** **Паттерны проектирования**

Задание 1. Разработать UISettings, который управляет темами и настройками

интерфейса.

Листинг задачи:

using System;

public class UISettings

{

private static UISettings \_instance;

private static readonly object \_lock = new object();

private string \_currentTheme = "светлая"; // по умолчанию

private UISettings() { ApplyThemeToConsole(); }

public static UISettings Instance

{

get

{

if (\_instance == null)

{

lock (\_lock)

{

if (\_instance == null)

{

\_instance = new UISettings();

}

}

}

return \_instance;

}

}

public void SetTheme(string theme)

{

if (string.IsNullOrEmpty(theme))

{

throw new ArgumentException("Название темы не может быть пустым.");

}

\_currentTheme = theme;

ApplyThemeToConsole();

Console.WriteLine($"Тема изменена на: {\_currentTheme}");

}

public string GetTheme()

{

return \_currentTheme;

}

private void ApplyThemeToConsole()

{

if (\_currentTheme == "темная")

{

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.Gray;

}

else if (\_currentTheme == "светлая")

{

Console.BackgroundColor = ConsoleColor.White;

Console.ForegroundColor = ConsoleColor.Black;

}

else

{

Console.ResetColor();

}

Console.Clear();

}

}

namespace Task1

{

class Program

{

static void Main(string[] args)

{

UISettings settings = UISettings.Instance;

settings.SetTheme("темная");

string currentTheme = settings.GetTheme();

Console.WriteLine($"Текущая тема: {currentTheme}");

Console.ReadKey();

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Темная тема | Светлая тема |

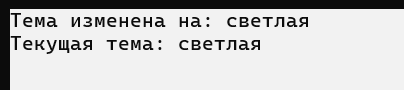
Анализ результатов:

Рисунок 1.1 – Результат работы программы