**Лабораторная работа №72-73**

**«Использование порождающих шаблонов»**

**Цель работы:** Научиться использовать порождающие шаблоны.

**Задания для практического занятия:**

5. Клонирование сложных объектов. В системе есть объекты с множеством полей (например, настройки графики в игре: разрешение, качество текстур, уровень детализации). Пользователь должен иметь возможность создавать копии этих объектов, чтобы быстро применять сохраненные настройки. Клонирование должно быть эффективным и не требовать ручного копирования каждого поля. Должна поддерживаться глубокая копия (вложенные объекты тоже должны клонироваться).

using System;

public class GraphicsSettings : ICloneable

{

public Resolution Resolution { get; set; }

public TextureQuality TextureQuality { get; set; }

public DetailLevel DetailLevel { get; set; }

public GraphicsSettings(Resolution resolution, TextureQuality textureQuality, DetailLevel detailLevel)

{

Resolution = resolution;

TextureQuality = textureQuality;

DetailLevel = detailLevel;

}

public object Clone()

{

return new GraphicsSettings(

(Resolution)Resolution.Clone(),

(TextureQuality)TextureQuality.Clone(),

(DetailLevel)DetailLevel.Clone());

}

}

public class Resolution : ICloneable

{

public int Width { get; set; }

public int Height { get; set; }

public Resolution(int width, int height)

{

Width = width;

Height = height;

}

public object Clone()

{

return new Resolution(Width, Height);

}

}

public class TextureQuality : ICloneable

{

public string Quality { get; set; }

public TextureQuality(string quality)

{

Quality = quality;

}

public object Clone()

{

return new TextureQuality(Quality);

}

}

public class DetailLevel : ICloneable

{

public int Level { get; set; }

public DetailLevel(int level)

{

Level = level;

}

public object Clone()

{

return new DetailLevel(Level);

}

}

class Program

{

static void Main()

{

GraphicsSettings originalSettings = new GraphicsSettings(

new Resolution(1920, 1080),

new TextureQuality("High"),

new DetailLevel(3));

GraphicsSettings clonedSettings = (GraphicsSettings)originalSettings.Clone();

clonedSettings.Resolution.Width = 1280;

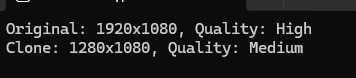
clonedSettings.TextureQuality.Quality = "Medium";

Console.WriteLine($"Original: {originalSettings.Resolution.Width}x{originalSettings.Resolution.Height}, Quality: {originalSettings.TextureQuality.Quality}");

Console.WriteLine($"Clone: {clonedSettings.Resolution.Width}x{clonedSettings.Resolution.Height}, Quality: {clonedSettings.TextureQuality.Quality}");

}

}



**Контрольные вопросы**

* Какой паттерн гарантирует, что объект будет создан только один раз, даже при многократных запросах?

Singleton (Одиночка)

* Какой паттерн позволит конструировать объекты поэтапно, особенно если у них много параметров?

Builder (Строитель)

* Какой паттерн позволяет создавать семейства связанных объектов, не указывая их конкретные классы?

Abstract Factory (Абстрактная фабрика)

* Какой паттерн поможет централизовать создание объектов, чтобы избежать жесткой привязки к конкретным классам документов?

Factory Method (Фабричный метод)

* Какой паттерн позволяет создавать точные копии объектов, избегая сложной логики инициализации?

Prototype (Прототип)

**Вывод:** Научился использовать порождающие шаблоны.