**Лабораторна робота №5**

ДИНАМІЧНА ІДЕНТИФІКАЦІЯ ТИПІВ

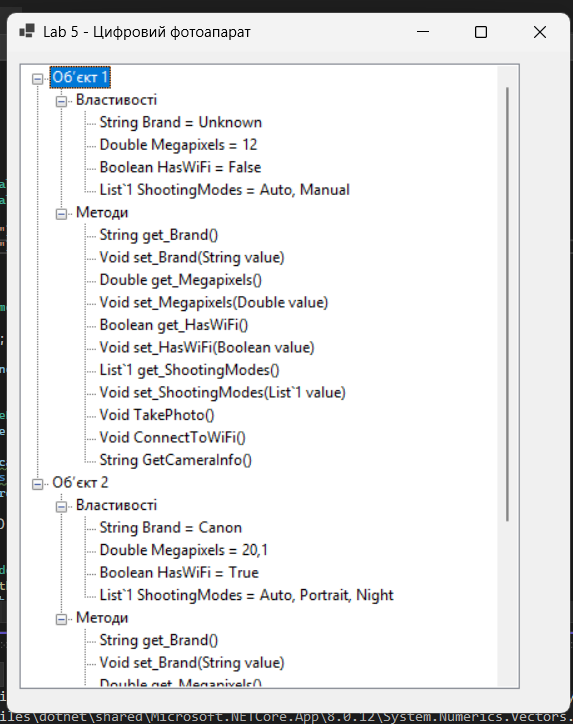
Завдання:

1. Створити новий проект за шаблоном Windows Forms Application.
2. На форму додати елемент керування TreeView.
3. Описати клас, забезпечивши наявність:

* 4-х властивостей різних типів, одна з яких мала б тип колекції;
* 3-х методів;
* 2-х конструкторів.

1. Описати метод, який би виводив перелік всіх властивостей (разом з типом та значенням) об’єкта описаного класу у TreeView.

Варіант завдання - 10 class Цифровий фотоапарат.



Код виконання завдання

using System;

using System.Collections.Generic;

using System.Reflection;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace lab5

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

PopulateTreeView();

}

private void PopulateTreeView()

{

DigitalCamera camera1 = new DigitalCamera();

DigitalCamera camera2 = new DigitalCamera("Canon", 20.1, true, new List<string> { "Auto", "Portrait", "Night" });

ShowCameraInfo(camera1, "Об’єкт 1");

ShowCameraInfo(camera2, "Об’єкт 2");

}

private void ShowCameraInfo(DigitalCamera camera, string nodeTitle)

{

Type type = typeof(DigitalCamera);

TreeNode rootNode = new TreeNode(nodeTitle); // назва об'єкта

// властивості

TreeNode propertiesNode = new TreeNode("Властивості");

foreach (PropertyInfo prop in type.GetProperties())

{

object value = prop.GetValue(camera);

string displayValue = value is List<string> list ? string.Join(", ", list) : value?.ToString();

propertiesNode.Nodes.Add($"{prop.PropertyType.Name} {prop.Name} = {displayValue}");

}

rootNode.Nodes.Add(propertiesNode);

// методи

TreeNode methodsNode = new TreeNode("Методи");

// отримуємо методи із прапорцями

MethodInfo[] methods = type.GetMethods(BindingFlags.Public | BindingFlags.Instance | BindingFlags.DeclaredOnly);

foreach (MethodInfo method in methods)

{

string methodSignature = method.ReturnType.Name + " " + method.Name + "(";

ParameterInfo[] parameters = method.GetParameters();

foreach (ParameterInfo param in parameters)

{

methodSignature += param.ParameterType.Name + " " + param.Name + ", ";

}

if (parameters.Length > 0)

methodSignature = methodSignature.Remove(methodSignature.Length - 2);

methodSignature += ")";

methodsNode.Nodes.Add(methodSignature);

}

rootNode.Nodes.Add(methodsNode);

treeView1.Nodes.Add(rootNode);

treeView1.ExpandAll();

// виклик методу через рефлексію

MethodInfo m = type.GetMethod("GetCameraInfo");

if (m != null && m.GetParameters().Length == 0)

{

object result = m.Invoke(camera, null);

if (result != null)

MessageBox.Show($"[{nodeTitle}]\n" + result.ToString());

}

}

}

public class DigitalCamera

{

public string Brand { get; set; }

public double Megapixels { get; set; }

public bool HasWiFi { get; set; }

public List<string> ShootingModes { get; set; }

public DigitalCamera()

{

Brand = "Unknown";

Megapixels = 12.0;

HasWiFi = false;

ShootingModes = new List<string> { "Auto", "Manual" };

}

public DigitalCamera(string brand, double megapixels, bool hasWiFi, List<string> modes)

{

Brand = brand;

Megapixels = megapixels;

HasWiFi = hasWiFi;

ShootingModes = modes;

}

public void TakePhoto()

{

MessageBox.Show($"{Brand} робить фото з роздільною здатністю {Megapixels} МП.");

}

public void ConnectToWiFi()

{

MessageBox.Show(HasWiFi ? $"{Brand} підключено до Wi-Fi." : $"{Brand} не підтримує Wi-Fi.");

}

public string GetCameraInfo()

{

return $"{Brand} - {Megapixels} МП, Wi-Fi: {(HasWiFi ? "є" : "немає")}";

}

}

}

У цій лабораторній роботі реалізовано приклад використання рефлексії в середовищі .NET (C#) для динамічного отримання інформації про тип даних.

Об’єктом дослідження є клас DigitalCamera, що містить властивості, методи та конструктори.

Створено Windows Forms додаток з елементом керування TreeView, у якому відображається структура класу реалізованого клас DigitalCamera

За допомогою рефлексії:

* Отримано повний список властивостей з їхніми типами та поточними значеннями;
* Виведено список методів з параметрами у форматі сигнатур;
* Викликано метод GetCameraInfo() через MethodInfo.Invoke(), і результат виведено у повідомленні.

У роботі застосовано ключові класи простору імен System.Reflection:

Type — для доступу до метаінформації про клас;

PropertyInfo — для роботи з властивостями;

MethodInfo — для аналізу та виклику методів;

ParameterInfo — для отримання інформації про параметри методів.

Ключова перевага рефлексії це можливість аналізувати структуру об’єктів і викликати методи на етапі виконання, що забезпечує гнучкість і розширюваність програмного забезпечення.

**Висновки**: На даній лабораторній роботі було розглянуто поняття рефлексії та досліджено її застосування. Були отримані практичні навички з отримання відомостей про методи та виклику методів за допомогою рефлексії.