



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

О Т Ч Е Т

по лабораторной работе № 7

Название: Рефлексия

Дисциплина: Разработка приложений на языке C#

Студент

ИУ6-73Б

(Группа)

(Подпись, дата)

К.А. Логачев

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

А.М. Минитаева

(И.О. Фамилия)

Москва, 2024

Задание:

Разработать программу, реализующую работу с рефлексией.

1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Создайте класс, содержащий конструкторы, свойства, методы.
3. С использованием рефлексии выведите информацию о конструкторах, свойствах, методах.
4. Создайте класс атрибута (унаследован от класса System.Attribute).
5. Назначьте атрибут некоторым свойствам классам. Выведите только те свойства, которым назначен атрибут.
6. Вызовите один из методов класса с использованием рефлексии

Выполнение

Было разработано приложение по работе с делегатами. Код программы представлен на листинге 1.

Листинг 1 –Код основной программы

```
using System;
using System.Reflection;

class Program {
    // Класс с конструкторами, свойствами и методами
    public class SampleClass {
        public int Number { get; set; }

        [CustomAttribute]
        public string Text { get; set; }

        public SampleClass() {}

        public SampleClass(int number) {
            this.Number = number;
        }

        public void Display() {
            Console.WriteLine($"Number: {this.Number}, Text: {this.Text}");
        }

        public string GetFormattedText() {
            return $"Formatted: {Text}";
        }
    }

    // Пользовательский атрибут
    [AttributeUsage(AttributeTargets.Property)]
    public class CustomAttribute : Attribute {}

    static void Main(string[] args) {
        // Получение информации о типе
        Type type = typeof(SampleClass);
```

```

// Вывод информации о конструкторах
Console.WriteLine("Constructors:");
foreach (ConstructorInfo constructor in type.GetConstructors()) {
    Console.WriteLine(constructor.ToString());
}

// Вывод информации о свойствах
Console.WriteLine("\nProperties:");
foreach (PropertyInfo property in type.GetProperties()) {
    Console.WriteLine(property.ToString());

    // Вывод свойств с пользовательским атрибутом
    if (Attribute.IsDefined(property, typeof(CustomAttribute))) {
        Console.WriteLine($"Property with CustomAttribute: {property.Name}");
    }
}

// Вывод информации о методах
Console.WriteLine("\nMethods:");
foreach (MethodInfo method in type.GetMethods()) {
    Console.WriteLine(method.ToString());
}

// Вывод информации о свойствах с пользовательским атрибутом
Console.WriteLine("\nProperties with CustomAttribute:");
foreach (PropertyInfo property in type.GetProperties()) {
    if (Attribute.IsDefined(property, typeof(CustomAttribute))) {
        Console.WriteLine(property.ToString());
    }
}

// Вызов метода с использованием рефлексии
Console.WriteLine("\nInvoking Display method:");
SampleClass instance = (SampleClass)Activator.CreateInstance(type);
MethodInfo displayMethod = type.GetMethod("Display");
displayMethod.Invoke(instance, null);

// Вызов метода GetFormattedText
Console.WriteLine("\nInvoking GetFormattedText method:");
MethodInfo getFormattedTextMethod = type.GetMethod("GetFormattedText");
string result = (string)getFormattedTextMethod.Invoke(instance, null);
Console.WriteLine(result);
}
}

```

На рисунке 1 представлен результат работы программы.

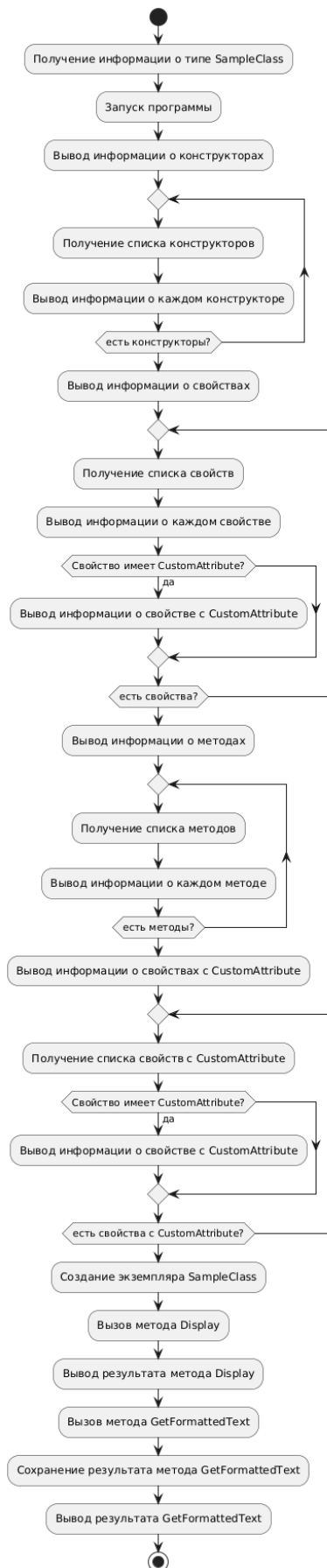


Рисунок 1 – Схема алгоритма

```
dotnet run
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(14,12): warning CS8618: свойство "Text", не допускающий значения NULL, должен содержать значение, отличное от NULL, при выходе из конструктора. Возможно, стоит объявить свойство как допускающий значения NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(16,12): warning CS8618: свойство "Text", не допускающий значения NULL, должен содержать значение, отличное от NULL, при выходе из конструктора. Возможно, стоит объявить свойство как допускающий значения NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(82,28): warning CS8600: Преобразование литерала, допускающего значение NULL или возможного значения NULL в тип, не допускающий значение NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(83,32): warning CS8600: Преобразование литерала, допускающего значение NULL или возможного значения NULL в тип, не допускающий значение NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(84,5): warning CS8602: Разыменование вероятной пустой ссылки. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(88,41): warning CS8600: Преобразование литерала, допускающего значение NULL или возможного значения NULL в тип, не допускающий значение NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(89,29): warning CS8602: Разыменование вероятной пустой ссылки. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
/Users/lt/Documents/bmstu/dotnet/lab7/lab7/Program.cs(89,21): warning CS8600: Преобразование литерала, допускающего значение NULL или возможного значения NULL в тип, не допускающий значение NULL. [/Users/lt/Documents/bmstu/dotnet/lab7/lab7/lab7.csproj]
Constructors:
Void .ctor()
Void .ctor(Int32)

Properties:
Int32 Number
System.String Text
Property with CustomAttribute: Text

Methods:
Int32 get_Number()
Void set_Number(Int32)
System.String get_Text()
Void set_Text(System.String)
Void Display()
System.String GetFormattedText()
System.Type GetType()
System.String ToString()
Boolean Equals(System.Object)
Int32 GetHashCode()

Properties with CustomAttribute:
System.String Text

Invoking Display method:
Number: 0, Text:

Invoking GetFormattedText method:
Formatted:
```

Рисунок 2 – Результат работы программы

Вывод: в ходе выполнения лабораторной работы было разработано приложение по работе с рефлексией. Приложение работает корректно. Изучена работа с рефлексией в приложениях на языке C#.