

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 Системы обработки информации и управления

## **Лабораторная работа №6**

Студент Абдрашитова Анжела Наилевна

Группа ИУ5-31Б

Название дисциплины Базовые компоненты интернет-технологий

Преподаватель

Гапанюк Ю.Е  
Фамилия И.О.

\_\_\_\_\_  
подпись

Москва 2020

## Описание задания:

### Часть 1. Разработать программу, использующую делегаты.

1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Определите делегат, принимающий несколько параметров различных типов и возвращающий значение произвольного типа.
3. Напишите метод, соответствующий данному делегату.
4. Напишите метод, принимающий разработанный Вами делегат, в качестве одного из входным параметров. Осуществите вызов метода, передавая в качестве параметра-делегата:
  - метод, разработанный в пункте 3;
  - лямбда-выражение.
5. Повторите пункт 4, используя вместо разработанного Вами делегата, обобщенный делегат Func<> или Action<>, соответствующий сигнатуре разработанного Вами делегата.

### Часть 2. 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.

## Текст программы:

### Animal.cs

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Lab6_2
{
    class Dog
```

```

{
    public Dog() {}
    public Dog(int age = 1)
    {
        Age = age;
    }
    public Dog(int age = 1, string name="Dog")
    {
        Age = age;
        Name = name;
    }
    public int Age { get; private set; }
    [AttributeClass("Имя")]
    public string Name { get; set; }

    public void Gav()
    {
        Console.WriteLine("Gav Gav");
    }
}
}

```

#### AttributeClass.cs

```

using System;
using System.Collections.Generic;
using System.Text;

namespace Lab6_2
{
    [AttributeUsage(AttributeTargets.Property, AllowMultiple = false, Inherited = false)]
    class AttributeClass : Attribute
    {
        public AttributeClass() {}
        public AttributeClass(string description)
        {
            Description = description;
        }
        public string Description { get; set; }
    }
}

```

#### Program.cs

```

using System;
using System.Reflection;

namespace Lab6_2
{
    class Program
    {
        public static bool hasPropertyAttribute(PropertyInfo checkType, Type
attributeType, out object attribute)
        {
            attribute = null;
            var isAttribute = checkType.GetCustomAttributes(attributeType, false);
            if (isAttribute.Length > 0)
            {
                attribute = isAttribute[0];
                return true;
            }
            return false;
        }

        static void Main(string[] args)
        {
            Type t = typeof(Dog);
            Console.WriteLine("Тип " + t.FullName);
            Console.WriteLine("\nКонструкторы:");
            foreach (var x in t.GetConstructors())
            {
                Console.WriteLine(x);
            }
            Console.WriteLine("\nМетоды:");
            foreach (var x in t.GetMethods())
            {
                Console.WriteLine(x);
            }
            Console.WriteLine("\nСвойства:");
            foreach (var x in t.GetProperties())
            {
                Console.WriteLine(x);
            }
            Console.WriteLine("\nСвойства, помеченные атрибутом:");
            foreach (var x in t.GetProperties())
            {
                object attrObj;
                if (hasPropertyAttribute(x, typeof(AttributeClass), out attrObj))
                {

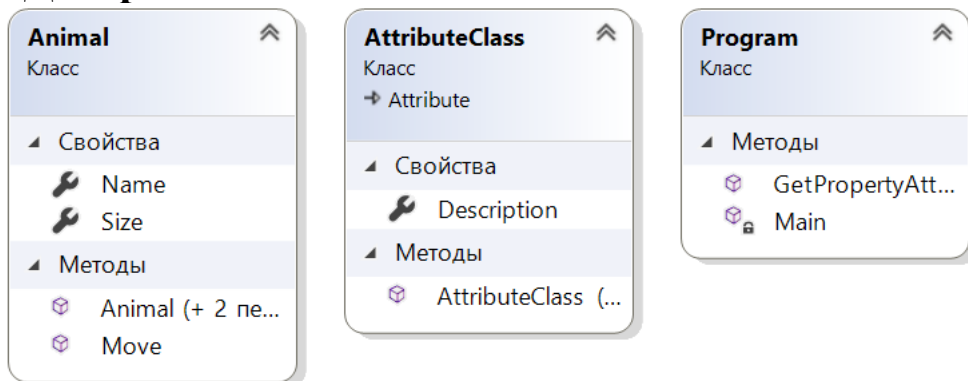
```

```

        var attr = attrObj as AttributeClass;
        Console.WriteLine(x.Name + " - " + attr.Description);
    }
}
Console.WriteLine("\nВызов метода:");
var fi = (Dog)t.InvokeMember(null, BindingFlags.CreateInstance, null,
null, null);
t.InvokeMember("Gav", BindingFlags.InvokeMethod, null, fi, null);
Console.ReadLine();
    }
}
}

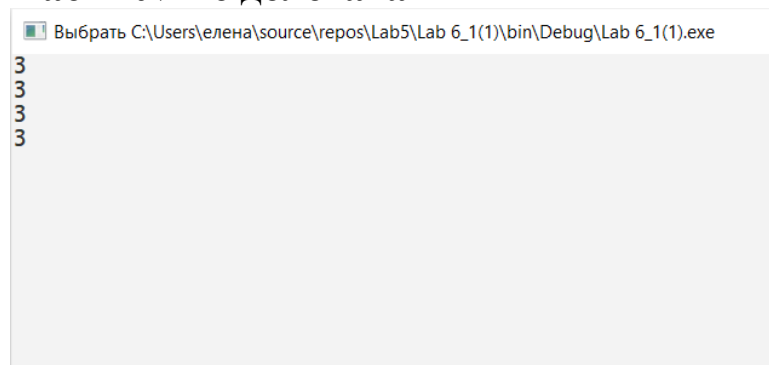
```

### Диаграммы классов:




### Результаты программы:

#### Часть №1 с делегатами



#### Часть №2 с рефлексией

 Выбрать C:\Users\елена\source\repos\Lab5\Lab6\_2\bin\Debug\Lab6\_2.exe

### Тип Lab6\_2.Dog

#### Конструкторы:

```
Void .ctor()  
Void .ctor(Int32)  
Void .ctor(Int32, System.String)
```

#### Методы:

```
Int32 get_Age()  
System.String get_Name()  
Void set_Name(System.String)  
Void Gav()  
Boolean Equals(System.Object)  
Int32 GetHashCode()  
System.Type GetType()  
System.String ToString()
```

#### Свойства:

```
Int32 Age  
System.String Name
```

#### Свойства, помеченные атрибутом:

```
Name - Имя
```