**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №6(часть 2)

«Работа с рефлексией»

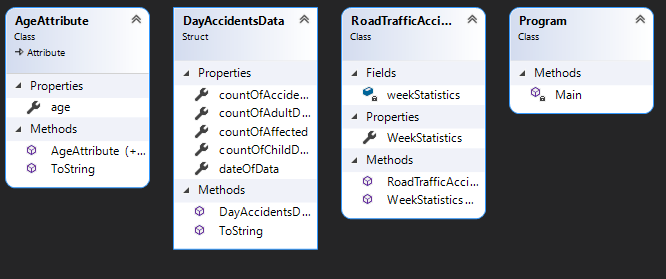
|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32 |  | преподаватель каф. ИУ5 |
| Барышников Михаил |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

**Задача**

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

**Диаграмма классов**



**Текст программы**

* Program.cs

using System;

using System.Runtime.InteropServices.WindowsRuntime;

using System.Threading;

using System.Reflection;

namespace RealizedReflection

{

[AttributeUsage(AttributeTargets.Property)]

public class AgeAttribute : System.Attribute

{

public int age { get; private set; }

public AgeAttribute(int \_age)

{

age = \_age;

}

public AgeAttribute() { }

public override string ToString()

{

return age.ToString();

}

}

public struct DayAccidentsData

{

public DateTime dateOfData { get; private set; }

public int countOfAccidents { get; private set; }

public int countOfAffected { get; private set; }

[AgeAttribute(0)]

public int countOfChildDead { get; private set; }

[AgeAttribute(18)]

public int countOfAdultDead { get; private set; }

public DayAccidentsData(DateTime \_dateOfData, int \_countOfAccidents, int \_countOfAffected, int \_countOfChildDead, int \_countOfAdultDead)

{

dateOfData = \_dateOfData;

countOfAccidents = \_countOfAccidents;

countOfAffected = \_countOfAffected;

countOfChildDead = \_countOfChildDead;

countOfAdultDead = \_countOfAdultDead;

}

public override string ToString()

{

return "Дата: " + dateOfData.ToString()

+ "\nКоличество происшествий: " + countOfAccidents.ToString()

+ "\nКоличество пострадавших: " + countOfAffected.ToString()

+ "\nКоличество погибших детей: " + countOfChildDead.ToString()

+ "\nКоличество погибших взрослых: " + countOfAdultDead.ToString();

}

}

class RoadTrafficAccidentsData

{

private DayAccidentsData[] weekStatistics = new DayAccidentsData[7];

public DayAccidentsData[] WeekStatistics

{

get

{

return weekStatistics;

}

private set

{

weekStatistics = value;

}

}

public RoadTrafficAccidentsData ()

{

Random x = new Random();

DateTime date = new DateTime(2020, x.Next(1, 12), x.Next(1, 26));

for (int i = 0; i != 7; ++i)

{

weekStatistics[i] = new DayAccidentsData(date, x.Next(20, 100), x.Next(25, 150), x.Next(0, 10), x.Next(10, 20));

date = date.AddDays(1);

}

}

public RoadTrafficAccidentsData(DateTime date)

{

Random x = new Random();

for (int i = 0; i != 7; ++i)

{

weekStatistics[i] = new DayAccidentsData(date, x.Next(20, 100), x.Next(25, 150), x.Next(0, 10), x.Next(10, 20));

date = date.AddDays(1);

}

}

public void WeekStatisticsOutput()

{

Console.WriteLine("\nСтатистика дорожно-траспортных происшествий за неделю:\n{0}", String.Join("\n\n", weekStatistics));

}

}

class Program

{

static void Main(string[] args)

{

RoadTrafficAccidentsData a = new RoadTrafficAccidentsData();

Type myType = Type.GetType("RealizedReflection.RoadTrafficAccidentsData", false, true);

object obj = Activator.CreateInstance(myType);

MethodInfo method = myType.GetMethod("WeekStatisticsOutput");

foreach (MemberInfo i in myType.GetMembers())

{

Console.WriteLine($"{i.DeclaringType} {i.MemberType} {i.Name}\n");

}

myType = Type.GetType("RealizedReflection.DayAccidentsData", false, true);

foreach (PropertyInfo prop in myType.GetProperties())

{

var isAttribute = prop.GetCustomAttribute(typeof(AgeAttribute), false);

if (isAttribute != null)

Console.WriteLine("{0} {1} {2} CountIs: {3}\n", prop.PropertyType, prop.Name, isAttribute.GetType(), isAttribute.ToString());

}

Console.BackgroundColor = ConsoleColor.DarkBlue;

Console.WriteLine("Call method of class:");

Console.BackgroundColor = ConsoleColor.Black;

method.Invoke(obj,new object[] { });

Console.ReadKey();

}

}

}

**Результаты выполнения программы**

