**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №6

«Разработка программы, использующей делегаты»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34Б |  | преподаватель каф. ИУ5 |
| Сергеев Илья |  | Гапанюк Ю. Е. |
|  |  |  |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2018 г.

**Описание задания**

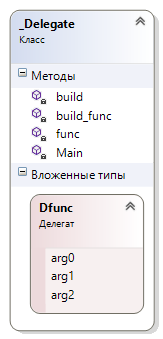
1. Программа должна быть разработана в виде консольного приложения на языке C#.
2. Определите делегат, принимающий несколько параметров различных типов и возвращающий значение произвольного типа.
3. Напишите метод, соответствующий данному делегату.
4. Напишите метод, принимающий разработанный Вами делегат, в качестве одного из входным параметров. Осуществите вызов метода, передавая в качестве параметра-делегата:

* метод, разработанный в пункте 3;
* лямбда-выражение.

1. Повторите пункт 4, используя вместо разработанного Вами делегата, обобщенный делегат Func< > или Action< >, соответствующий сигнатуре разработанного Вами делегата.

**Диаграмма классов**

Диаграмма классов генерируется автоматически в среде Visual Studio:



**Текст программы (листинг)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Delegate

{

class \_Delegate

{

/// <summary>

/// Example of a delegate with 3 params

/// </summary>

/// <param name="arg0"></param>

/// <param name="arg1"></param>

/// <param name="arg2"></param>

/// <returns></returns>

delegate double Dfunc(int arg0, double arg1, bool arg2);

/// <summary>

/// Funciton satisfy for a delegate

/// </summary>

/// <param name="a0"></param>

/// <param name="a1"></param>

/// <param name="a2"></param>

/// <returns></returns>

static double func(int a0, double a1, bool a2)

{

if (a2)

return a0 \* a1;

else

return a1 - a1 / a0;

}

/// <summary>

/// Method with delegate typed arg (with MY delegate)

/// </summary>

/// <param name="a"></param>

/// <param name="a1"></param>

/// <param name="flag"></param>

/// <param name="inp"></param>

/// <param name="f"></param>

/// <returns></returns>

static string build(int a, double a1, bool flag, string inp, Dfunc f)

{

return inp + "var equals " + f(a, a1, flag).ToString();

}

/// <summary>

/// Method with Func<> arg

/// </summary>

/// <param name="a"></param>

/// <param name="a1"></param>

/// <param name="flag"></param>

/// <param name="inp"></param>

/// <param name="f"></param>

/// <returns></returns>

static string build\_func(int a, double a1, bool flag, string inp, Func<int, double, bool, double> f)

{

return inp + "var equals " + f(a, a1, flag).ToString();

}

static void Main(string[] args)

{

//params for delegate-argumented function

string inp = "The weather is fine, and ";

int a = 3;

double a1 = 5.23;

bool flag = true;

//call with delegate-typed method

Console.WriteLine(build(a, a1, flag, inp, func));

//call with lambda-function

Console.WriteLine(build(a, a1, flag, inp,

(int b, double b1, bool b2) =>

{

if (b2)

return b1 - b;

else

return b + b1;

}));

Console.WriteLine();

//call with delegate-typed method

Console.WriteLine(build\_func(a, a1, flag, inp, func));

//call with lambda-function

Console.WriteLine(build\_func(a, a1, flag, inp,

(int b, double b1, bool b2) =>

{

if (b2)

return b1 - b;

else

return b + b1;

}));

Console.ReadKey(); //delay for a user

}

}

}

**Экранные формы с примерами выполнения программы (скриншоты)**

