**13 СОБЫТИЯ**

Задание №1. Создать 3 метода для вывода трех различных сообщений. Создать делегат. В программе вызывать все три метода при помощи делегата.

Листинг программы:

Message message = MessageOne;

message += MessageTwo;

message += MessageThree;

PrintMessage(message);

void MessageOne() => Console.WriteLine("Hello world!");

void MessageTwo() => Console.WriteLine("Bye!");

void MessageThree() => Console.WriteLine("Good morning!");

void PrintMessage(Message someMessage) => someMessage();

delegate void Message();

Таблица 13.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| MessageOne, MessageTwo, MessageThree | SomeMessage |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.1.



Рисунок 13.1 – Результат работы программы

Источник: собственная разработка

Задание №2. На основе своей программы создать метод, в котором делегат будет параметром.

Листинг программы:

void PrintMessage(Message someMessage) => someMessage();

Таблица 13.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Delegate | someMessage |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.2.



Рисунок 13.2 – Результат работы программы

Источник: собственная разработка

Задание №3. Для класса MyInfo содержащего поле name (Ваше имя) создать событие выводящее оповещение в случае изменения значения поля name.

Листинг программы:

class Program

{

class MyInfo

{

public delegate void MyEventHandler(string message);

public event MyEventHandler NameEvent = delegate { };

string name;

public string MyProp

{

get

{

return name;

}

set

{

name = value;

NameEvent($"Установлено значение имени: {name}");

}

}

}

static void Main(string[] args)

{

MyInfo obj = new MyInfo();

obj.NameEvent += x => Console.WriteLine(x);

obj.MyProp = "Леша";

obj.MyProp = "Никита";

obj.MyProp = "Артём";

Console.Read();

}

}

Таблица 13.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| String | NameEvent |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.3.



Рисунок 13.3 – Результат работы программы

Источник: собственная разработка

Задание №4. Создать класс, в нем делегат и соответствующее ему событие. Создать два класса-наблюдателя, в методах которых будет описана их реакция на событие. Добавить 2 обработчика к событию из первого класса и один из второго. Вывести результат на экран. Удалить один обработчик события и вывести результат на экран.

Листинг программы:

try

{

Console.Write("Введите имя: ");

string name = Console.ReadLine();

MyInfo inf = new MyInfo();

X xObj = new X();

Y yObj = new Y();

inf.Notify += inf.DisplayMessage;

inf.Notify += xObj.XHandler;

inf.Notify += yObj.YHandler;

inf.ChangeName(name);

Console.WriteLine();

inf.Notify -= yObj.YHandler;

inf.ChangeName(name);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

delegate void Info(string message);

class MyInfo

{

public string name = "Tom";

public event Info? Notify;

public void ChangeName(string name)

{

this.name = name;

Notify?.Invoke($"Имя было изменено на {name}");

}

public void DisplayMessage(string message) => Console.WriteLine(message);

}

class X

{

public void XHandler(string name) => Console.WriteLine($"{name} - хорошее имя");

}

class Y

{

public void YHandler(string name) => Console.WriteLine($"{name} - прекрасное имя");

}

Таблица 13.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Алексей | XHandler, YHandler |

Источник: собственная разработка

Анализ результатов представлен на рисунке 13.4.

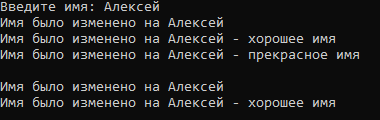


Рисунок 13.4 – Результат работы программы

Источник: собственная разработка