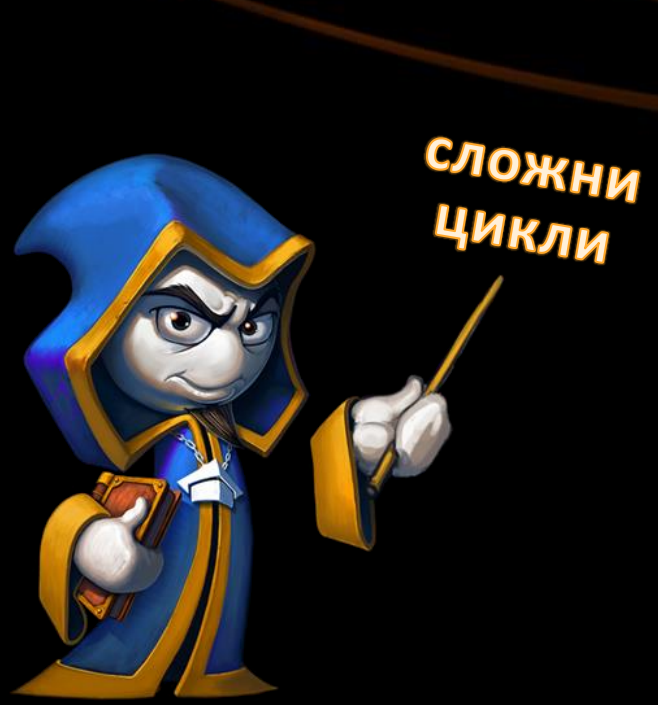
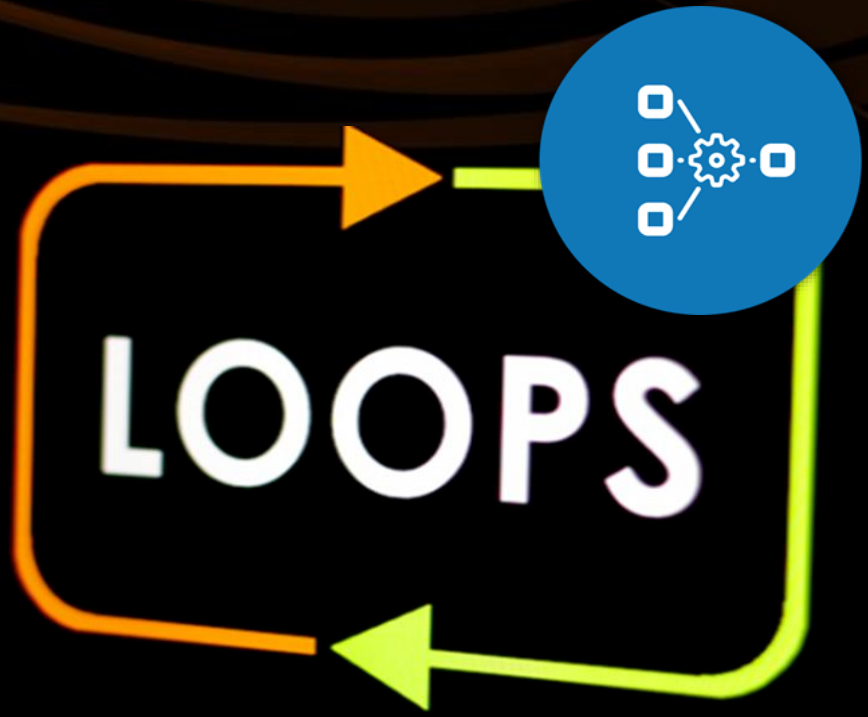
Работа с по-сложни цикли

Цикли със стъпка, While, Do…While





СофтУни

трейнърски екип

Софтуерен университет

http://softuni.bg

Have a Question?

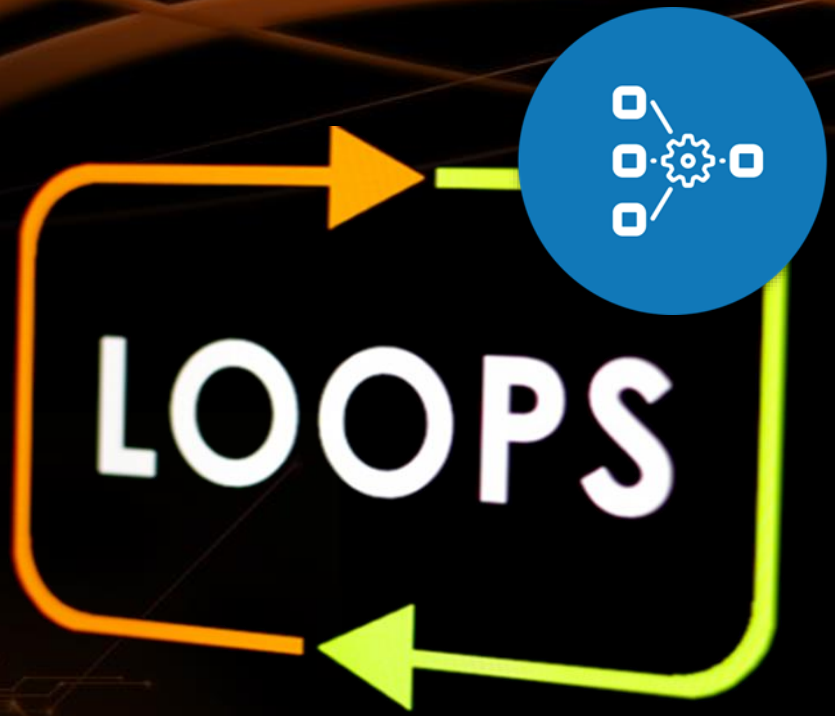
SOFTWARE UNIVERSITY FOUNDATION

sli.do

#TODO

Съдържание

- По-сложни конструкции за цикъл:
  - цикъл със стъпка
  - цикъл с намаляваща стъпка
  - `while` цикъл
  - `do-while` цикъл
  - безкраен цикъл и оператор `break`
- По-сложни задачи с вложени цикли
- Създаване на уеб игра с ASP.NET MVC



Цикли със стъпка

Работа с по-сложни `for`-цикли

Числата от 1 до N през 3

- Да се отпечатат числата от **1** до **n** със стъпка **3**
  - При **n** = 100: 1, 4, 7, 10, …, 94, 97, 100

```
int n = int.Parse(Console.ReadLine());
for (var i = 1; i <= n; i+=3)
{
    Console.WriteLine(i);
}
```

Задаване на стъпка

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#0

Числата от N до 1 в обратен ред

- Да се отпечатат числата от `n` до `1` в обратен ред (стъпка -1)
  - При `n` = 100: 100, 99, 98, …, 3, 2, 1

```
int n = int.Parse(Console.ReadLine());
for (var i = n; i >= 1; i-=1)
{
  Console.WriteLine(i);
}
```

Обърнато условие: `i >= 1`

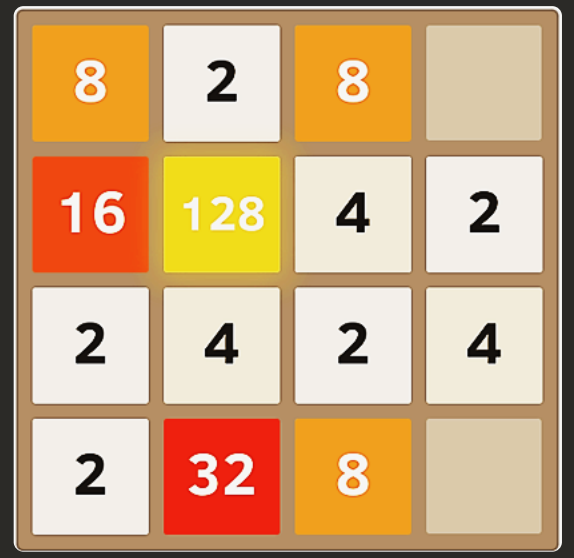
Отрицателна стъпка: `-1`

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#1

Числата от 1 до 2^n с for-цикъл

- Да се отпечатат числата от **1** до **2^n**
  - При **n** = 10: 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024

```
int num = 1;
for (var i = 0; i <= n; i++)
{
    Console.WriteLine(num);
    num = num * 2;
}
```



Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#2

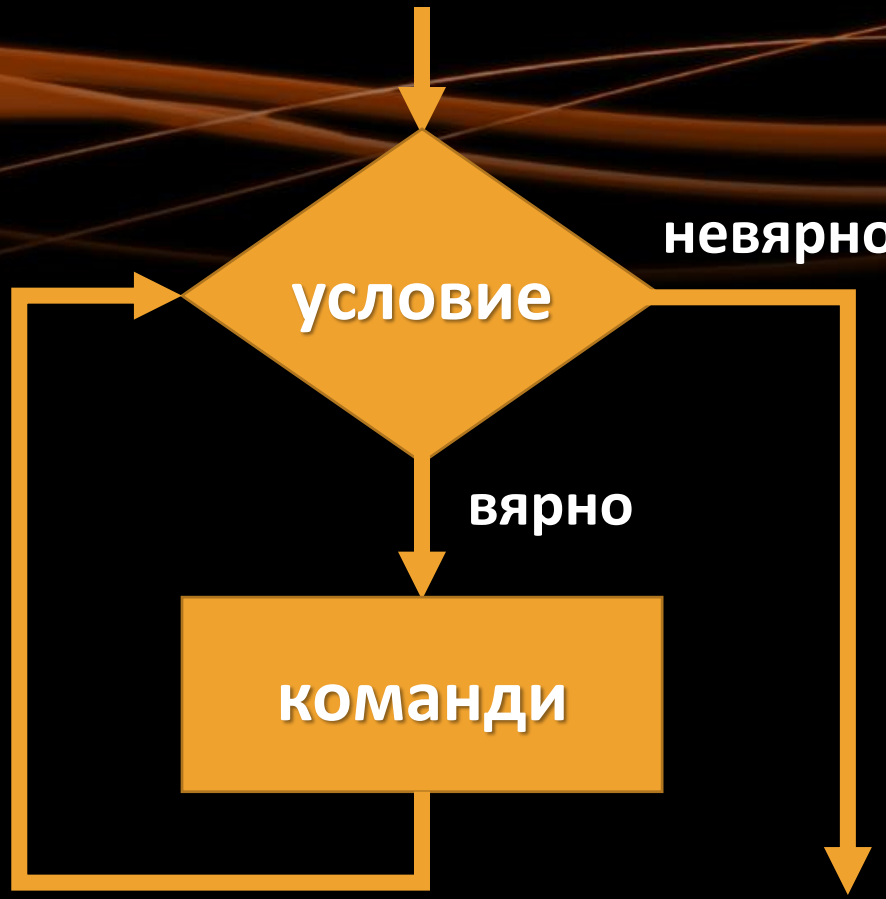
Четни степени на 2

- Да се отпечатат четните степени на 2 до 2^n: 2^0, 2^2, 2^4, 2^8, …, 2^n
  - При **n** = 10: 1, 4, 16, 64, 256, 1024

```
int num = 1;
for (var i = 0; i <= n; i+=2)
{
    Console.WriteLine(num);
    num = num * 2 * 2;
}
```

Ползваме стъпка 2

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#3



While цикъл

Повторение докато е в сила дадено условие

Редица числа 2k+1

- Да се отпечатат всички числа ≤ n от редицата: 1, 3, 7, 15, 31, …
  - Всяко следващо число = предишно число * 2 + 1

```
int num = 1;
while (num <= n)
{
    Console.WriteLine(num);
    num = 2 * num + 1;
}
```

Повтаряй докато е в сила условието num ≤ n

1, 3, 7, 15, 31, 63, …

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#4

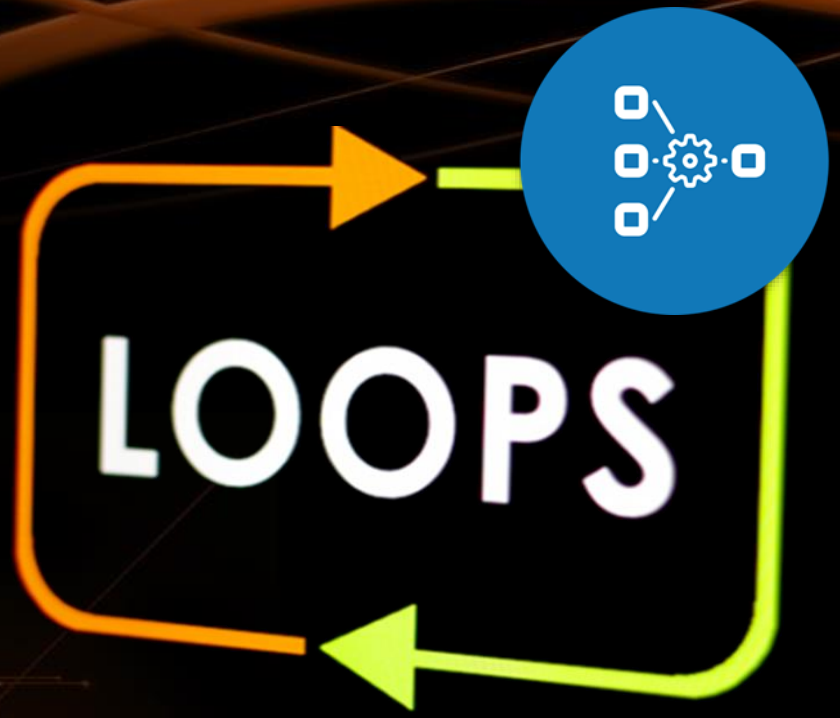
Число в диапазона [1…100]

- Да се въведе число в диапазона [1…100]
  - При невалидно число да се въведе отново

```
var num = int.Parse(Console.ReadLine());
while (num < 1 || num > 100)
{
    Console.WriteLine("Invalid number!");
    num = int.Parse(Console.ReadLine());
}
Console.WriteLine("The number is: {0}", num);
```

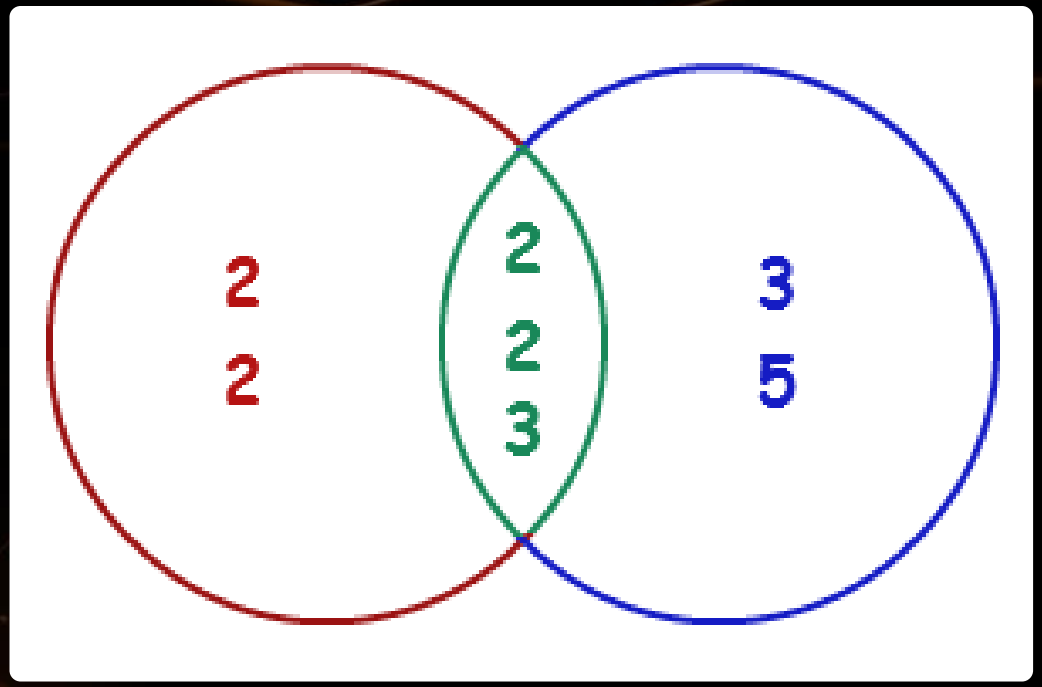
Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#5

SOFTWARE UNIVERSITY FOUNDATION



Цикли със стъпка и while цикъл

Работа на живо в клас (лаб)



Най-голям общ делител (НОД)

Алгоритъм на Евклид

Най-голям общ делител (НОД)

- Най-голям общ делител (НОД) на две естествени числа **a** и **b** е най-голямото число, което дели едновременно **a** и **b** без остатък
  - НОД(24, 16) = 8
  - НОД(12, 24) = 12
  - НОД(10, 10) = 10
  - НОД(67, 18) = 1
  - НОД(15, 9) = 3
  - НОД(100, 88) = 4
- Алгоритъм на Евклид за намиране на НОД:
  - Докато не достигнем остатък 0
    - Делим по-голямото число на по-малкото
    - Взимаме остатъка от делението

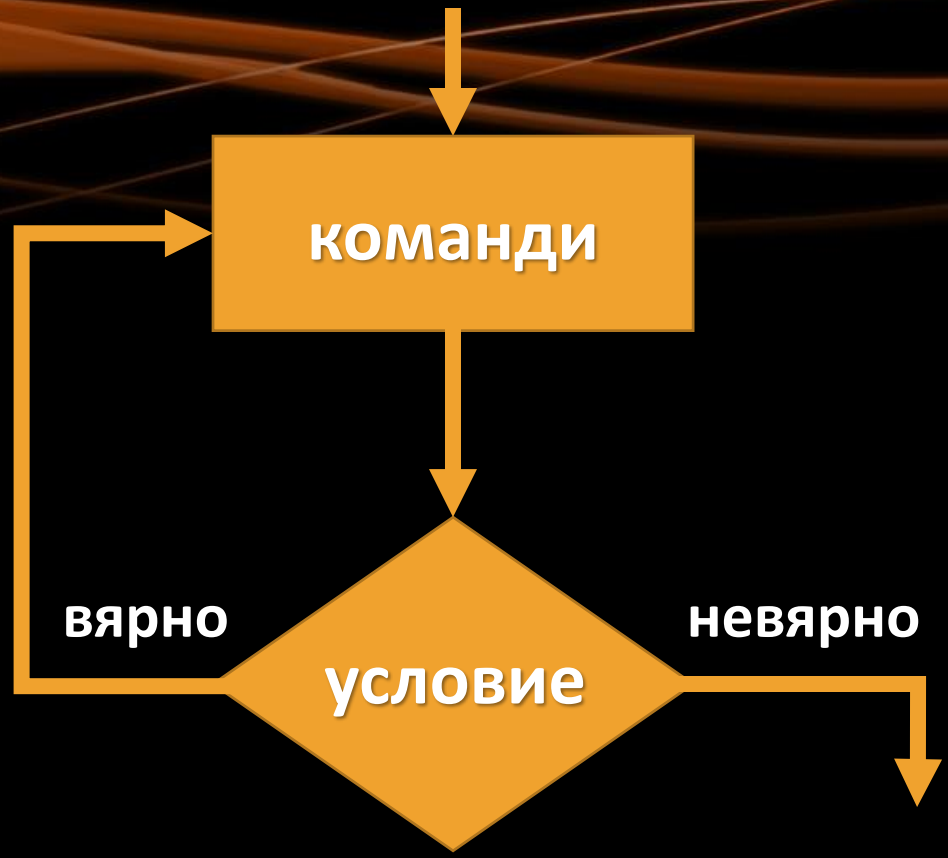
```
while b ≠ 0
  var oldB = b;
  b = a % b;
  a = oldB;
print a;
```

Алгоритъм на Евклид за НОД

- Да се въведат цели числа **a** и **b** и да се намери **НОД**(**a**, **b**)

```
var a = int.Parse(Console.ReadLine());
var b = int.Parse(Console.ReadLine());
while (b != 0)
{
    var oldB = b;
    b = a % b;
    a = oldB;
}
Console.WriteLine("GCD = {0}", a);
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#6



Do...While цикъл

Повторение докато е изпълнено условието

Изчисляване на факториел

- За естествено число n да се изчисли n! = 1 * 2 * 3 * … * n
  - Пример: 5! = 1 * 2 * 3 * 4 * 5 = 120

```
var n = int.Parse(Console.ReadLine());
var fact = 1;
do
{
    fact = fact * n;
    n--;
} while (n > 1);
Console.WriteLine(fact);
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#7

Сумиране на цифрите на число

- Да се сумират цифрите на цяло положително число **n**
  - При **n** = 5634: 5 + 6 + 3 + 4 = 18

```
var n = int.Parse(Console.ReadLine());
var sum = 0;
do
{
    sum = sum + (n % 10);
    n = n / 10;
} while (n > 0);
Console.WriteLine("Sum of digits: {0}", sum);
```

n % 10 връща последната цифра на числото n

n / 10 изтрива последната цифра на n

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#8

Безкрайни цикли и оператор break

Безкраен цикъл

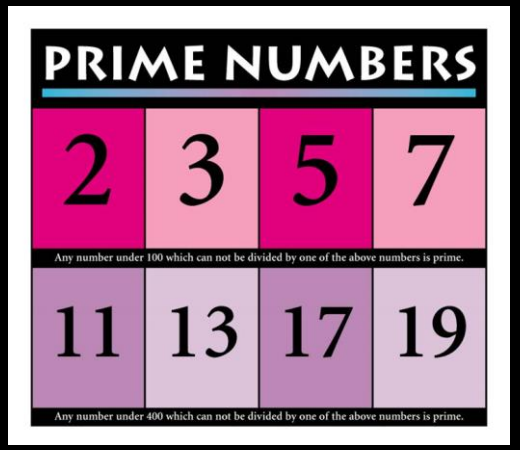
- Безкраен цикъл е когато повтаряме нещо до безкрайност:

```
while(true)
{
    Console.WriteLine("Infinite loop");
}
```

```
for (;;)
{
    Console.WriteLine("Infinite loop");
}
```

Прости числа

- Едно число **n** е просто, ако се дели единствено на **1** и **n**
  - Прости числа: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, …
  - Непрости (композитни) числа: 10 = 2 * 5, 21 = 3 * 7, 143 = 13 * 11
- Едно число **n** е просто, ако се дели на число между **2** и **n-1**
- Алгоритъм за проверка дали число е просто:
  - Проверяваме дали **n** се дели на 2, 3, …, n-1
  - Ако се раздели, значи е композитно
  - Ако не се раздели, значи е просто



- Оптимизация: вместо до n-1 да се проверяват делители до $\sqrt{n}$

Проверка за просто число. Оператор break


```
var n = int.Parse(Console.ReadLine());
var prime = true;
for (var i = 2; i <= Math.Sqrt(n); i++)
    if (n % i == 0) {
        prime = false;
        break;
    }
if (prime) Console.WriteLine("Prime");
else Console.WriteLine("Not prime");
```

break излиза от цикъла

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#9

Оператор break в безкраен цикъл

- Да се напише програма, която въвежда четно число
  - При невалидно число да връща към повторно въвеждане

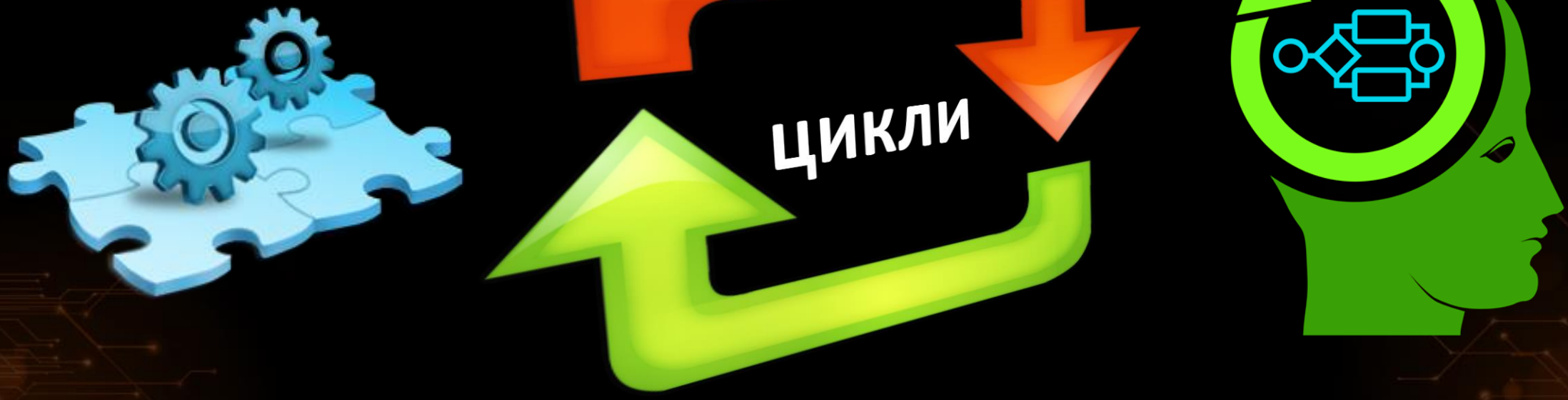
```
var n = 0;
while (true)
{
    Console.Write("Enter even number: ");
    n = int.Parse(Console.ReadLine());
    if (n % 2 == 0)
        break; // even number -> exit from the loop
    Console.WriteLine("The number is not even.");
}
Console.WriteLine("Even number entered: {0}", n);
```

Справяне с грешни числа: try … catch

```
try
{
    Console.Write("Enter even number: ");
    n = int.Parse(Console.ReadLine());
    if (n % 2 == 0)
        break;
    Console.WriteLine("The number is not even.");
}
catch
{
    Console.WriteLine("Invalid number.");
}
```

Ако `int.Parse(…)` гръмне, ще се изпълни `catch { … }` блокът

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#10



Задачи с цикли

Числа на Фибоначи

- Числата на Фибоначи са следните: 1, 1, 2, 3, 5, 8, 13, 21, 34, …
  - $F_0 = 1$
  - $F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}$
- Пример: F(15) = 987
- Да се въведе **n** и да се пресметна **n**-тото число на Фибоначи

```
var n = int.Parse(Console.ReadLine());
var f0 = 1;
var f1 = 1;
for (var i = 0; i < n-1; i++)
{
    var fNext = f0 + f1;
    f0 = f1;
    f1 = fNext;
}
Console.WriteLine(f1);
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#11

Пирамида от числа

- Да се отпечатат числата 1…**n** в пирамида като в примерите:

```
n = 7
```

```
1
2 3
4 5 6
7
```

```
n = 10
```

```
1
2 3
4 5 6
7 8 9 10
```

```
n = 12
```

```
1
2 3
4 5 6
7 8 9 10
11 12
```

```
n = 15
```

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#12

Пирамида от числа – решение

```
var n = int.Parse(Console.ReadLine());
var num = 1;
for (var row = 1; row <= n; row++)
{
    for (var col = 1; col <= row; col++)
    {
        if (col > 1) Console.Write(" ");
        Console.Write(num);
        num++;
        if (num > n) break;
    }
    Console.WriteLine();
    if (num > n) break;
}
```

```
1
2 3
4 5 6
7 8 9 10
11 12
```

Таблица с числа

- Да се отпечатат числата 1…**n** в таблица като в примерите:

n = 2	n = 3	n = 4	n = 5
1 2 2 1	1 2 3 2 3 2 3 2 1	1 2 3 4 2 3 4 3 3 4 3 2 4 3 2 1	1 2 3 4 5 2 3 4 5 4 3 4 5 4 3 4 5 4 3 2 5 4 3 2 1

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#13

Таблица с числа – решение

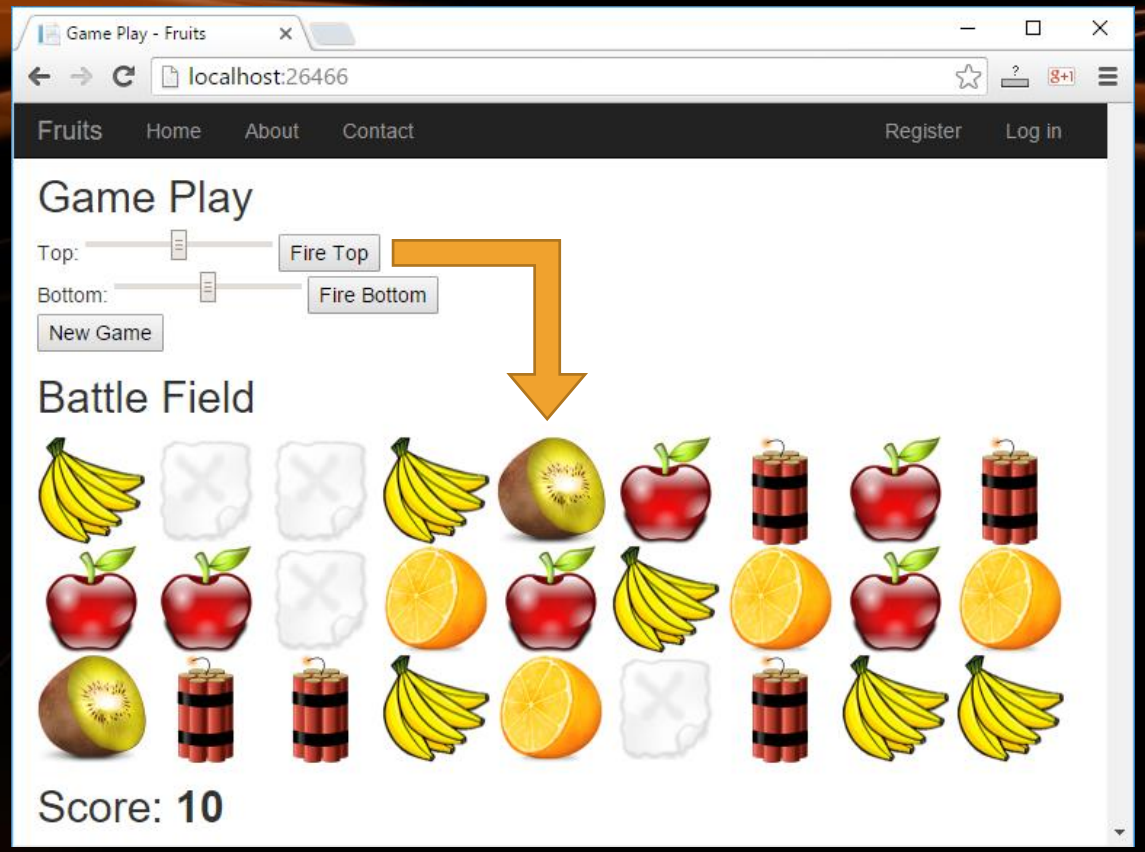
```
var n = int.Parse(Console.ReadLine());
for (int row = 0; row < n; row++)
{
    for (int col = 0; col < n; col++)
    {
        var num = row + col + 1;
        if (num > n) num = 2 * n - num;
        Console.Write(num + " ");
    }
    Console.WriteLine();
}
```

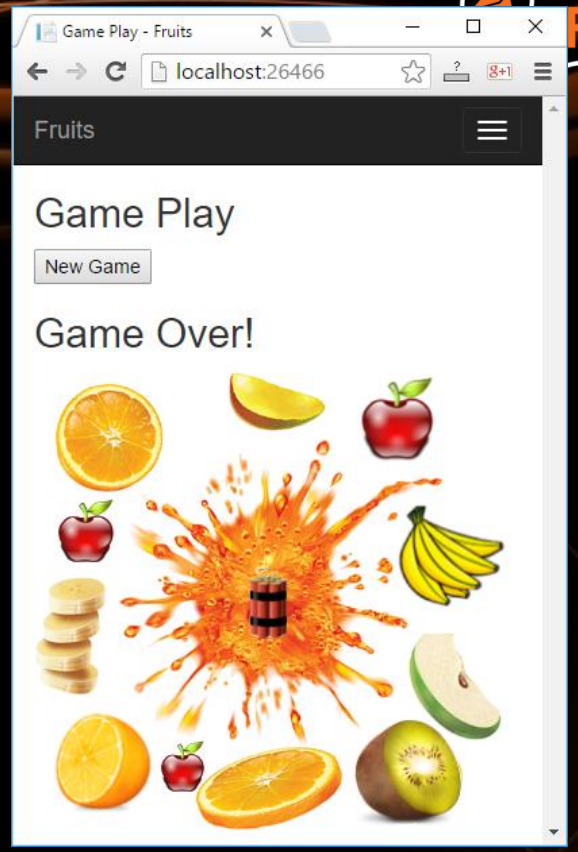
```
1 2 3 4 5
2 3 4 5 4
3 4 5 4 3
4 5 4 3 2
5 4 3 2 1
```

Тестване на решението: https://judge.softuni.bg/Contests/Practice/Index/156#13

По-сложни задачи с цикли

Работа на живо в клас (лаб)



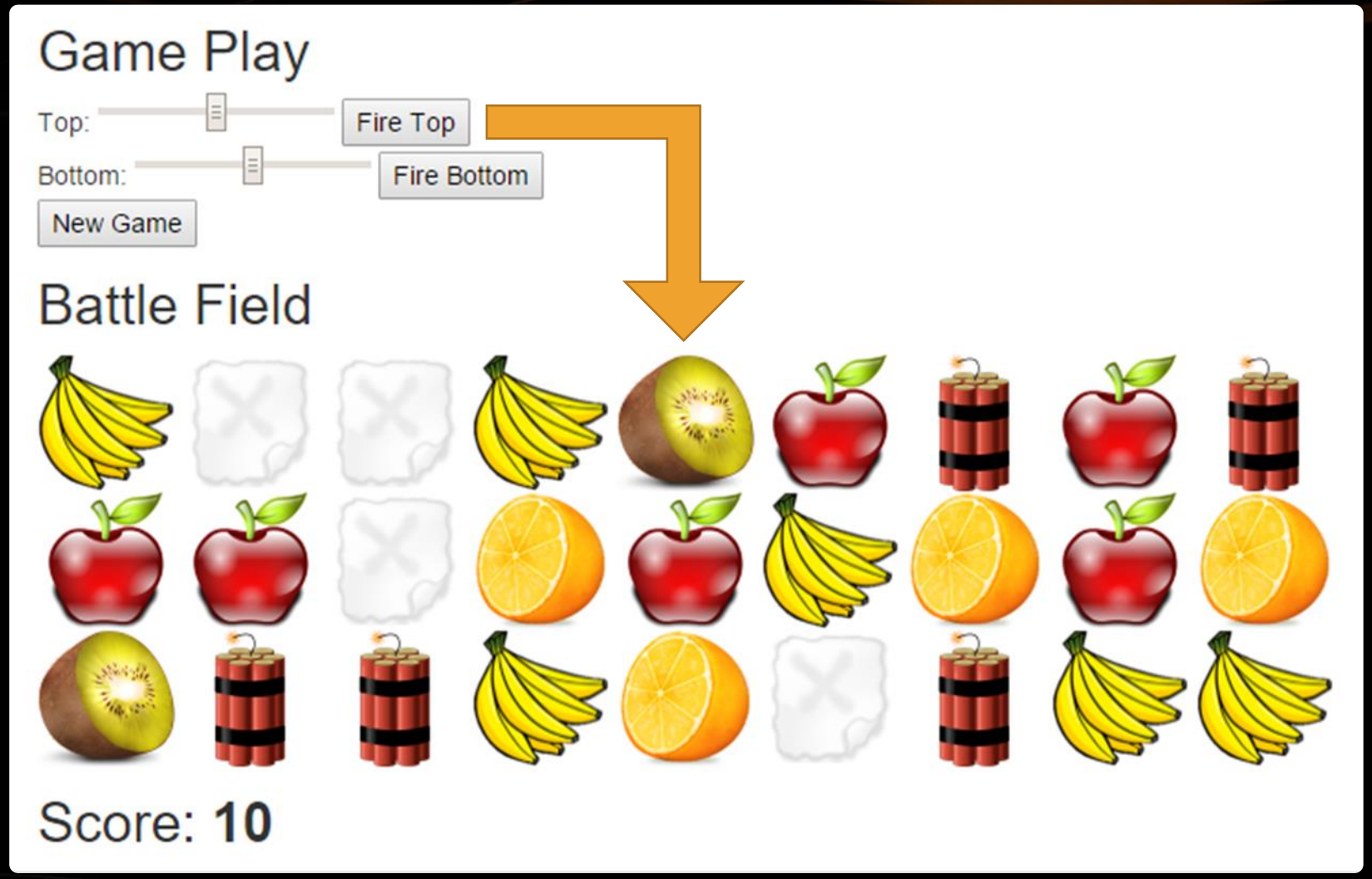


Уеб игра с ASP.NET MVC

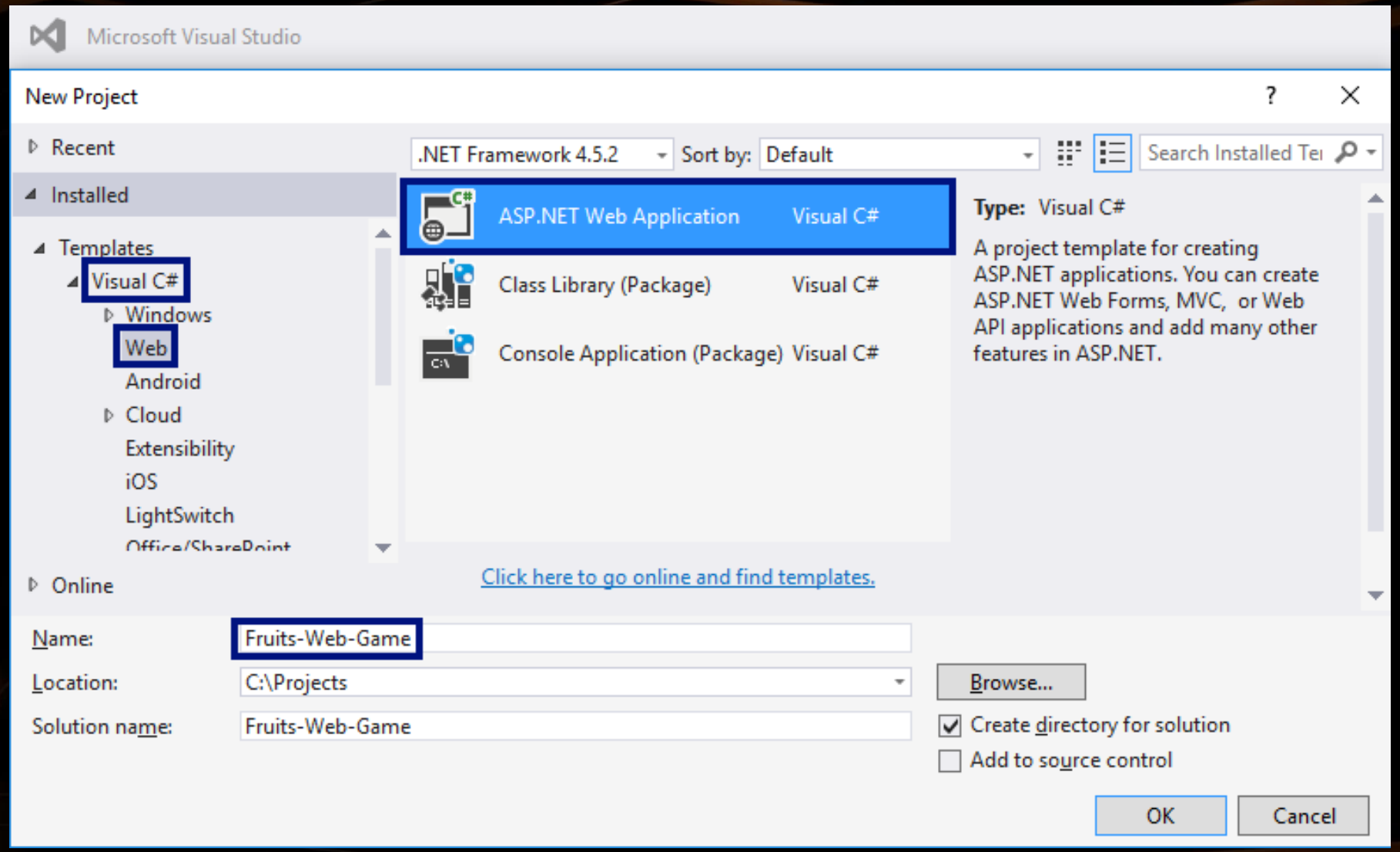
Обстреляй плодовете!

SOFTWARE UNIVERSITY FOUNDATION

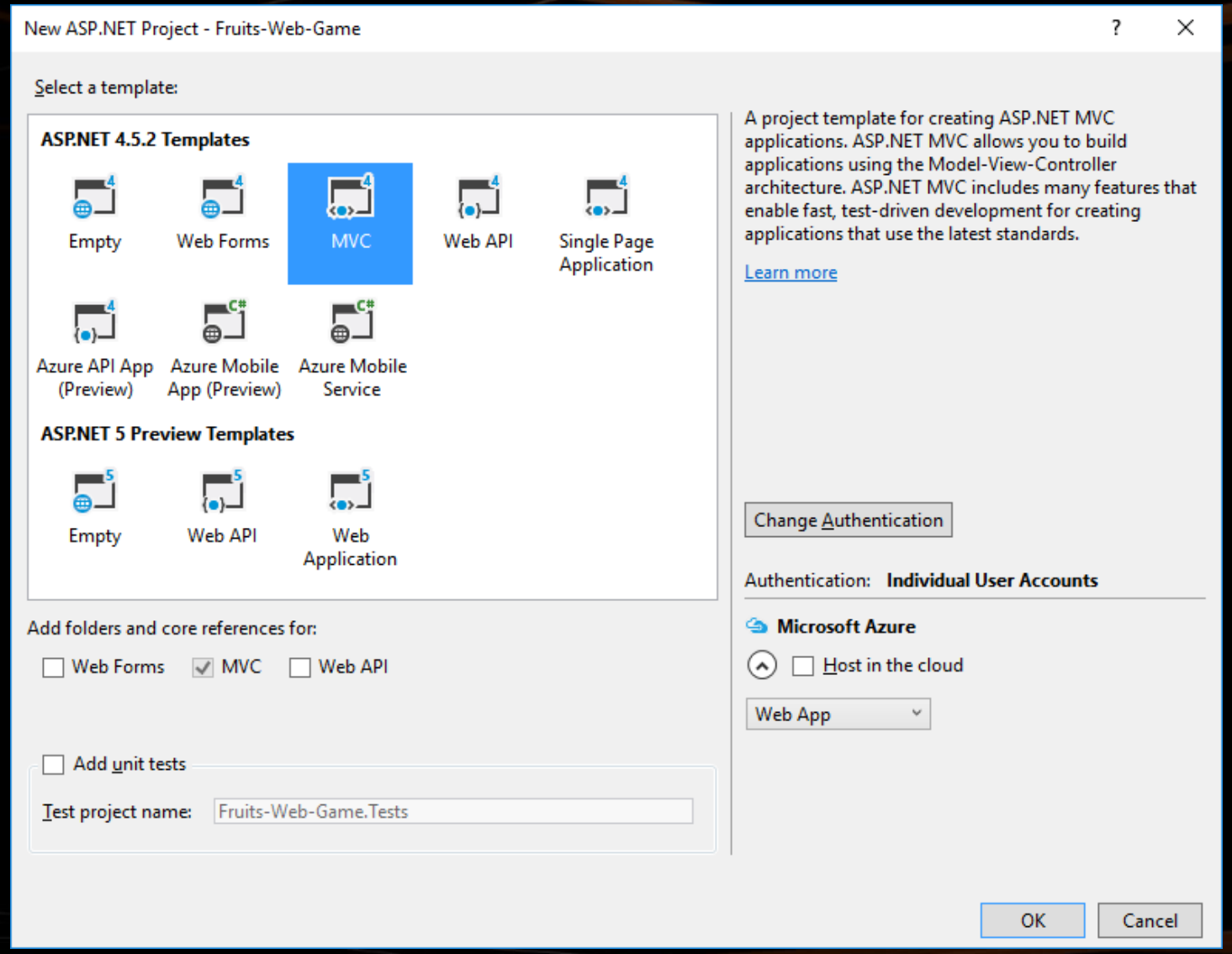
Уеб игра "Обстреляй плодовете!"



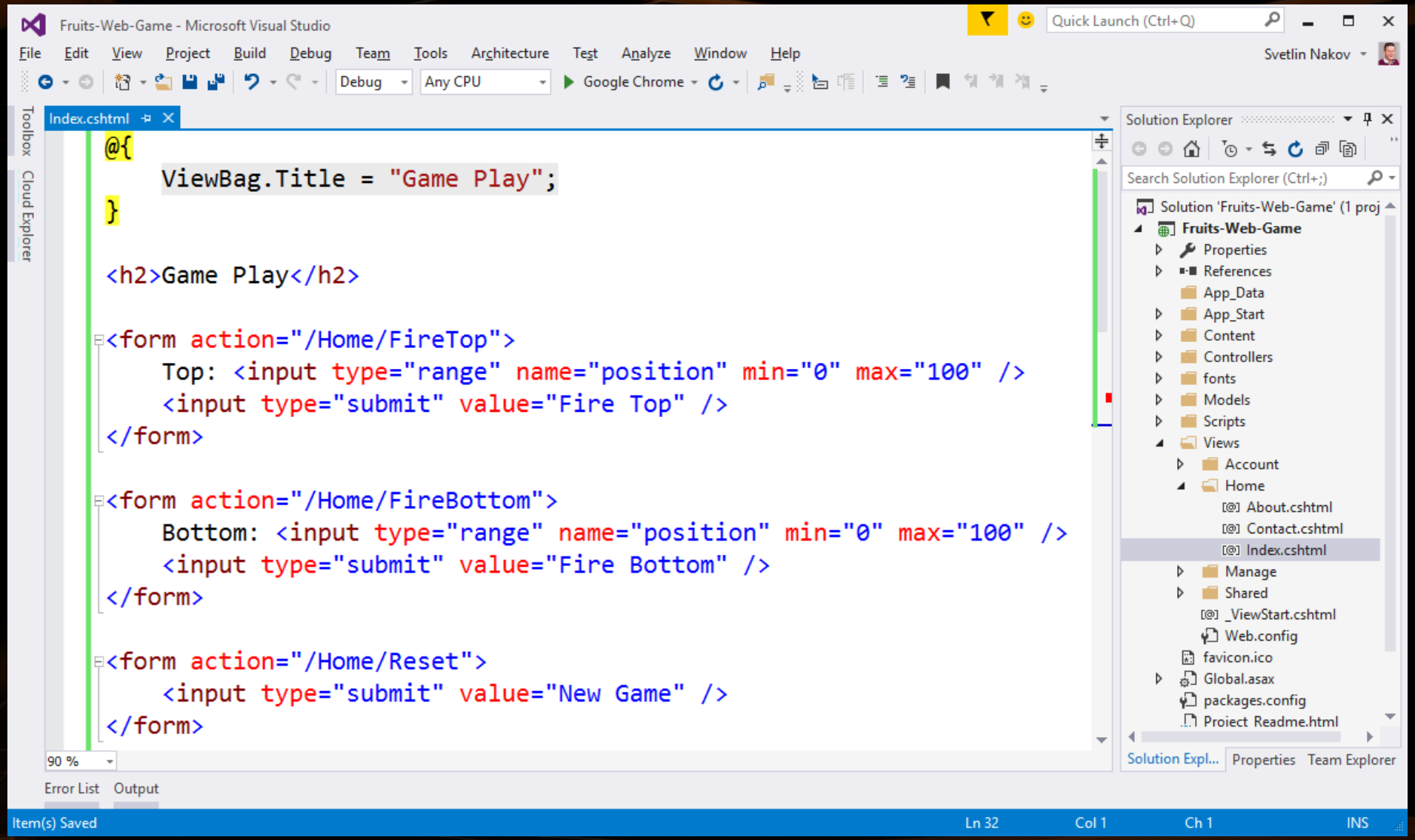
Създаваме уеб приложение във VS



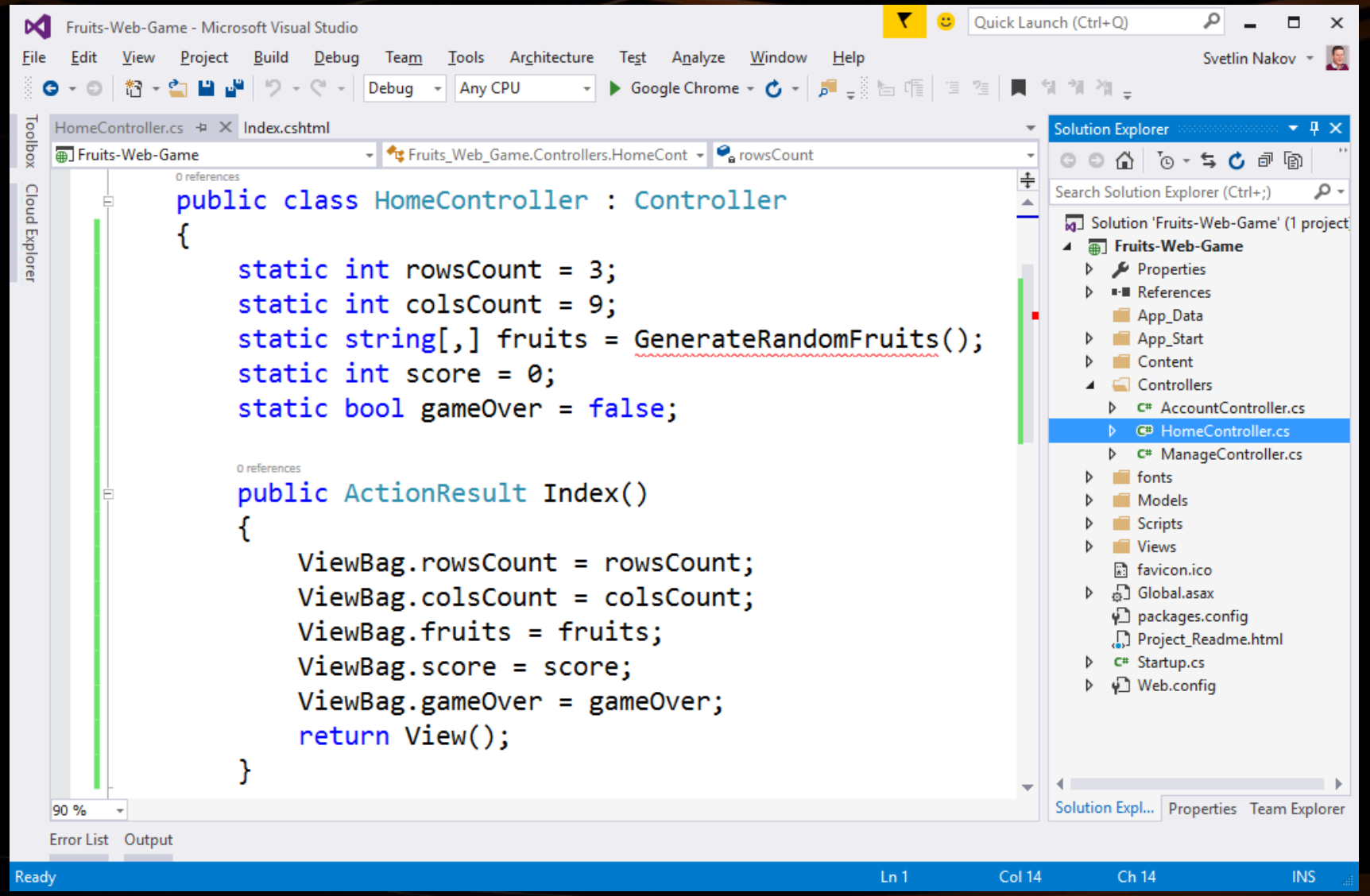
Тип на уеб проекта → MVC



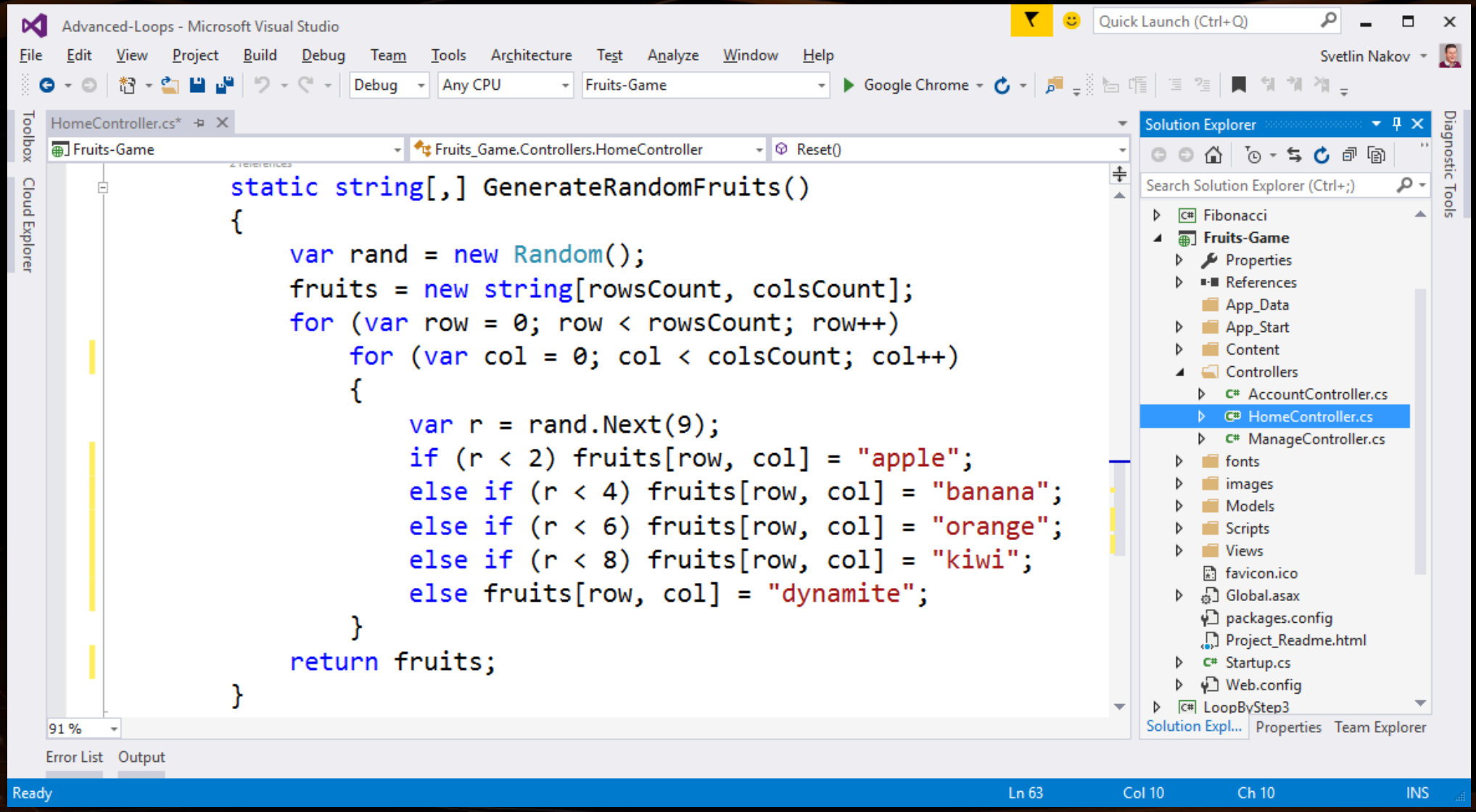
Създаване на контролите за играта



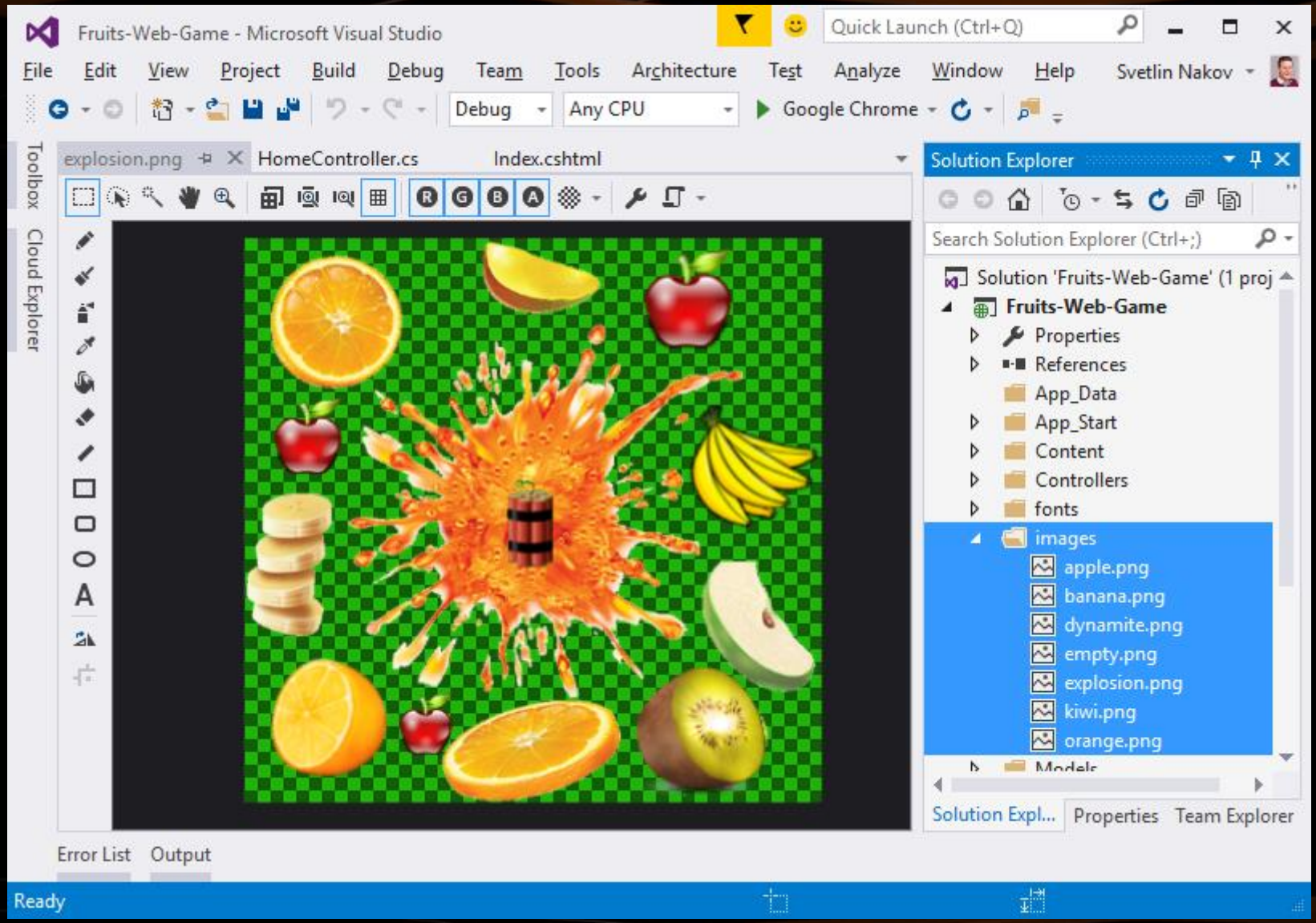
Подготовка на плодовете в HomeController.cs



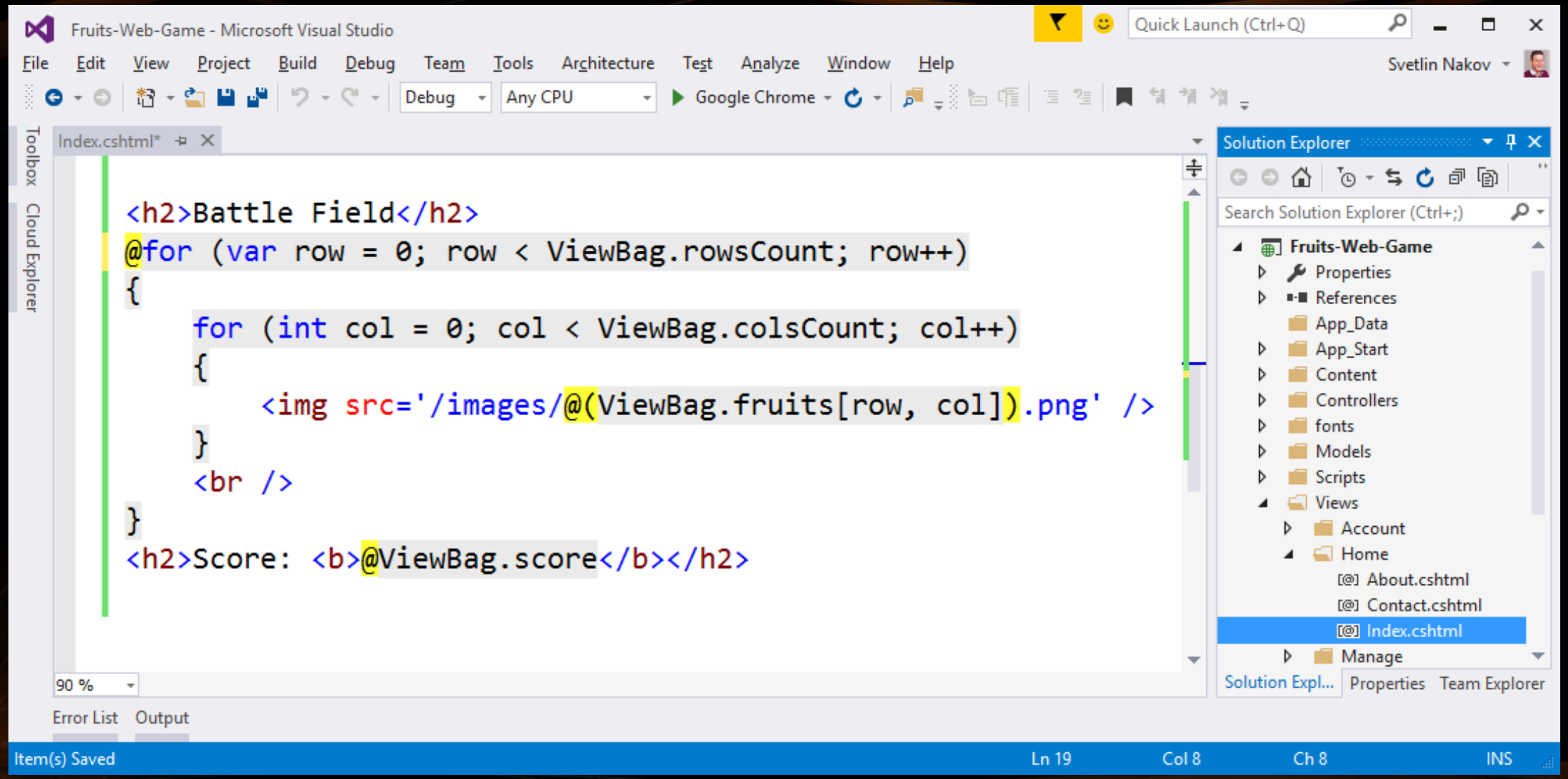
Генериране на случайни плодове



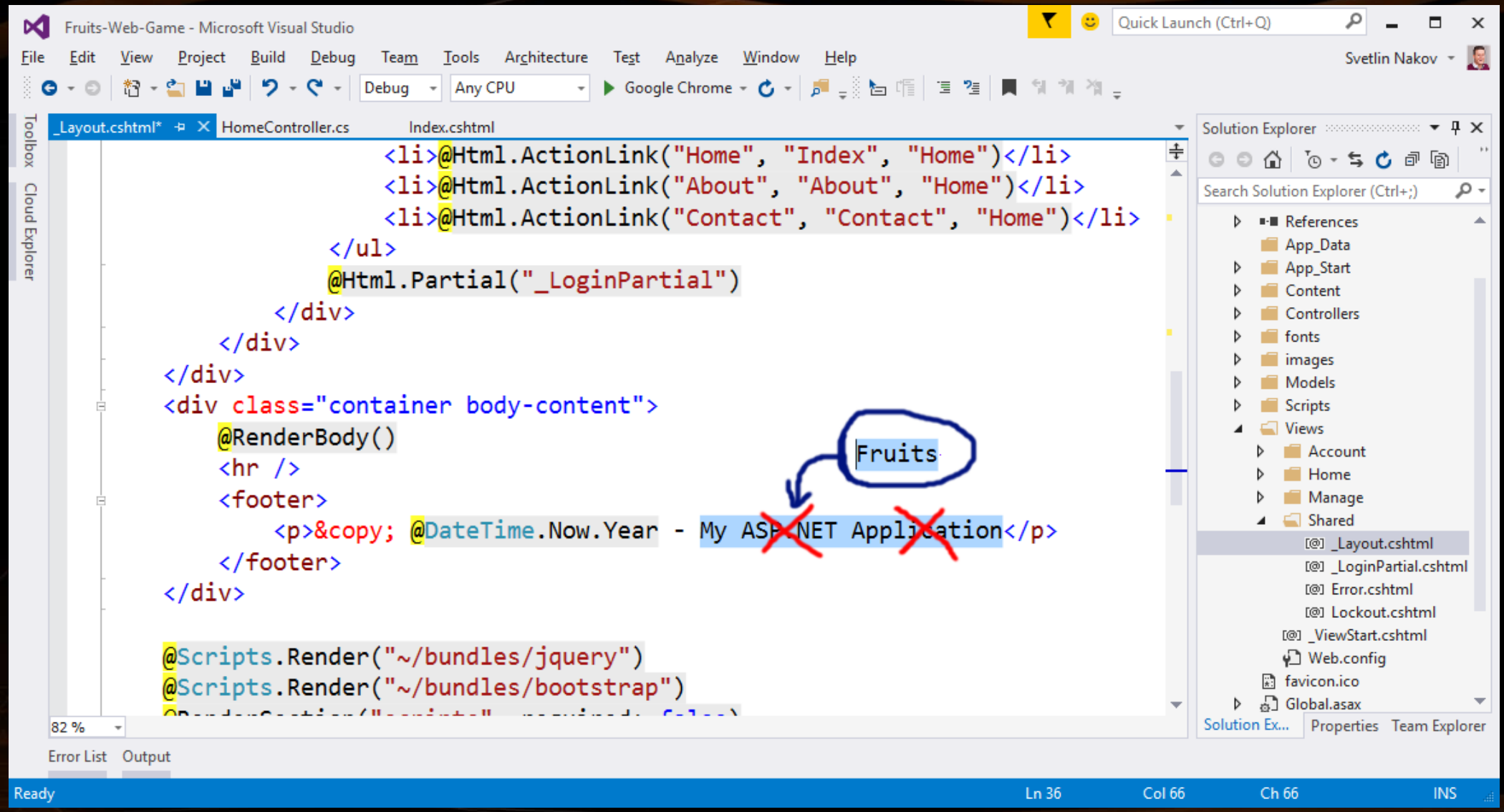
Добавяне на картинките в проекта



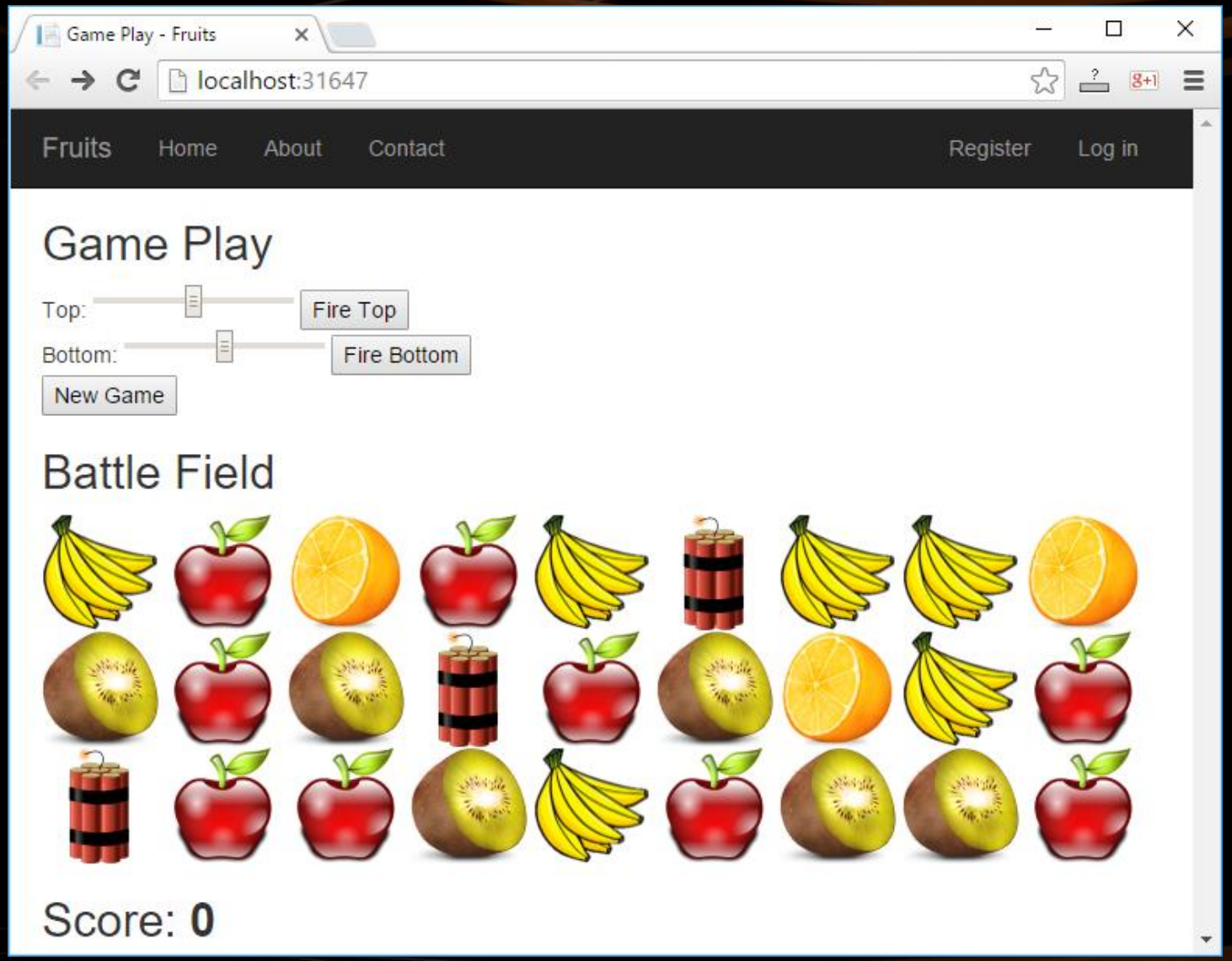
Чертане на плодовете в Index.cshtml



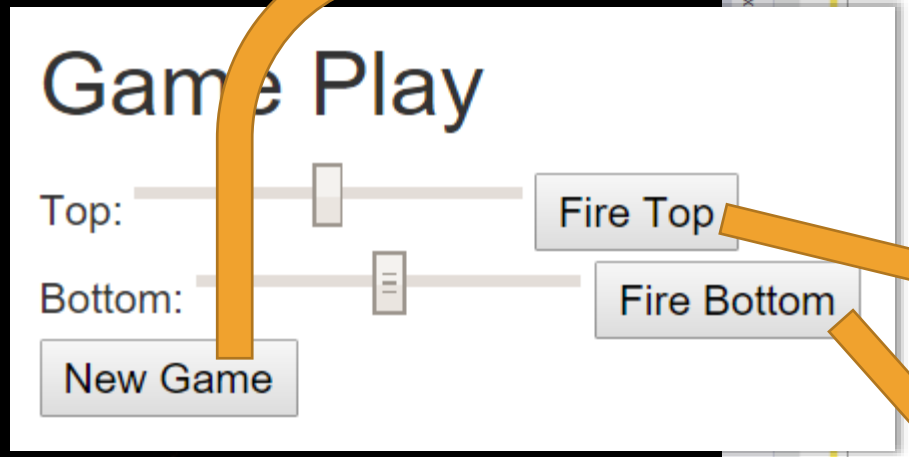
Нагласяме текстовете в _Layout.cshtml



Стартираме с [Ctrl+F5] и тестваме



Добавяме действията "Reset" и "Fire"

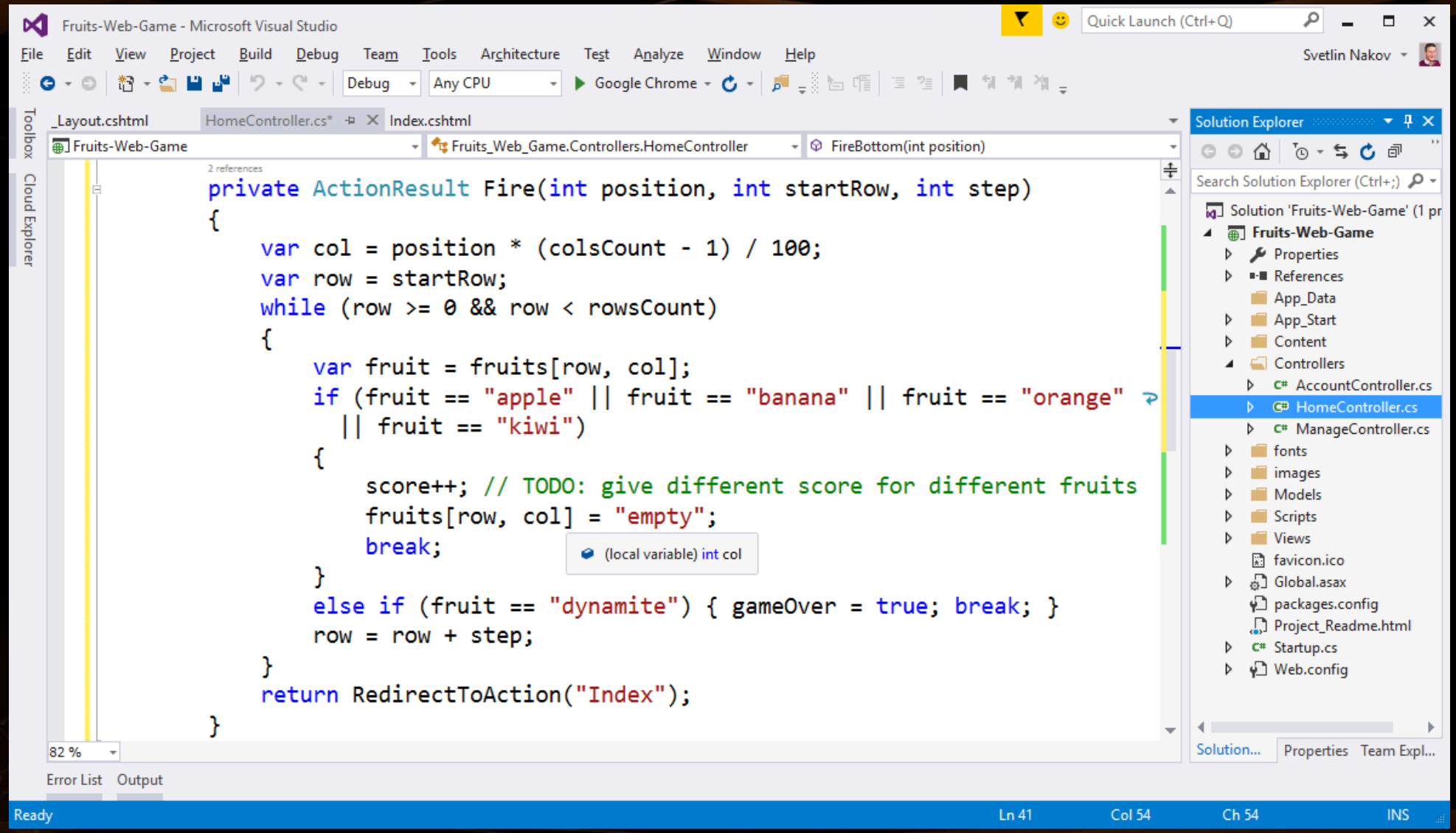



```
public ActionResult Reset()
{
    fruits = GenerateRandomFruits();
    gameOver = false;
    return RedirectToAction("Index");
}
```

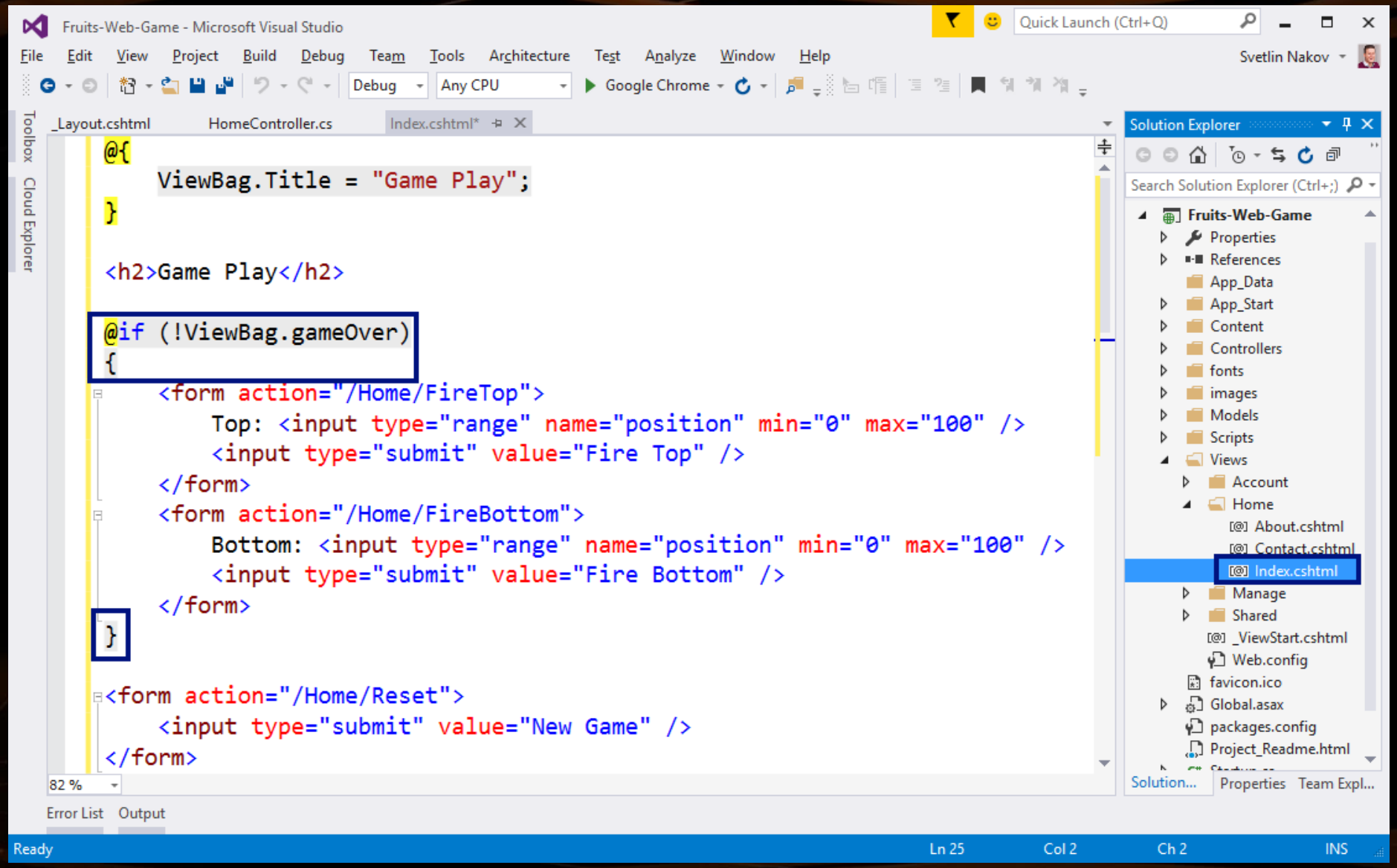
```
public ActionResult FireTop(int position)
{
    return Fire(position, 0, 1);
}
```

```
public ActionResult FireBottom(int position)
{
    return Fire(position, rowsCount - 1, -1);
}
```

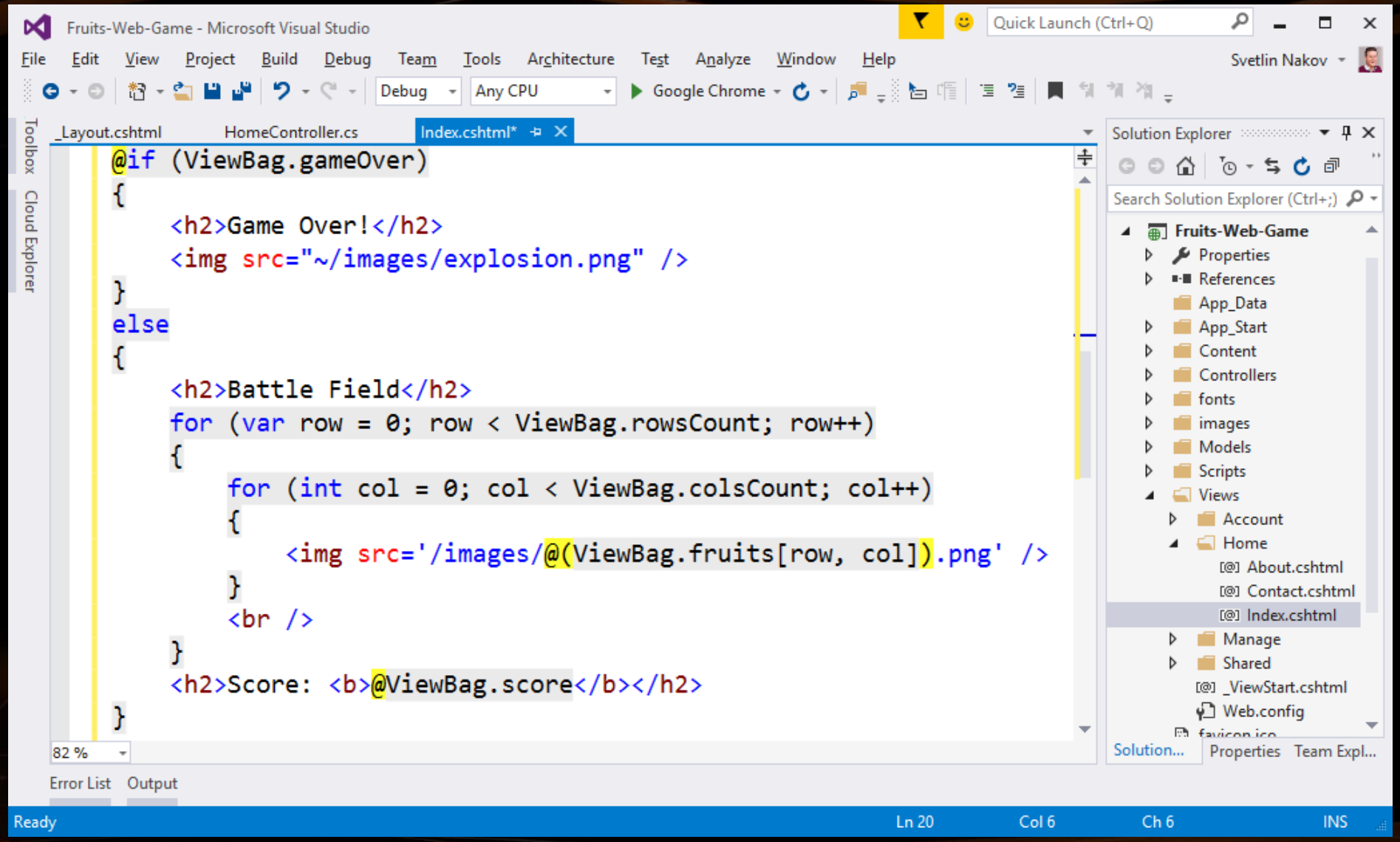
Имплементираме "стрелянето"



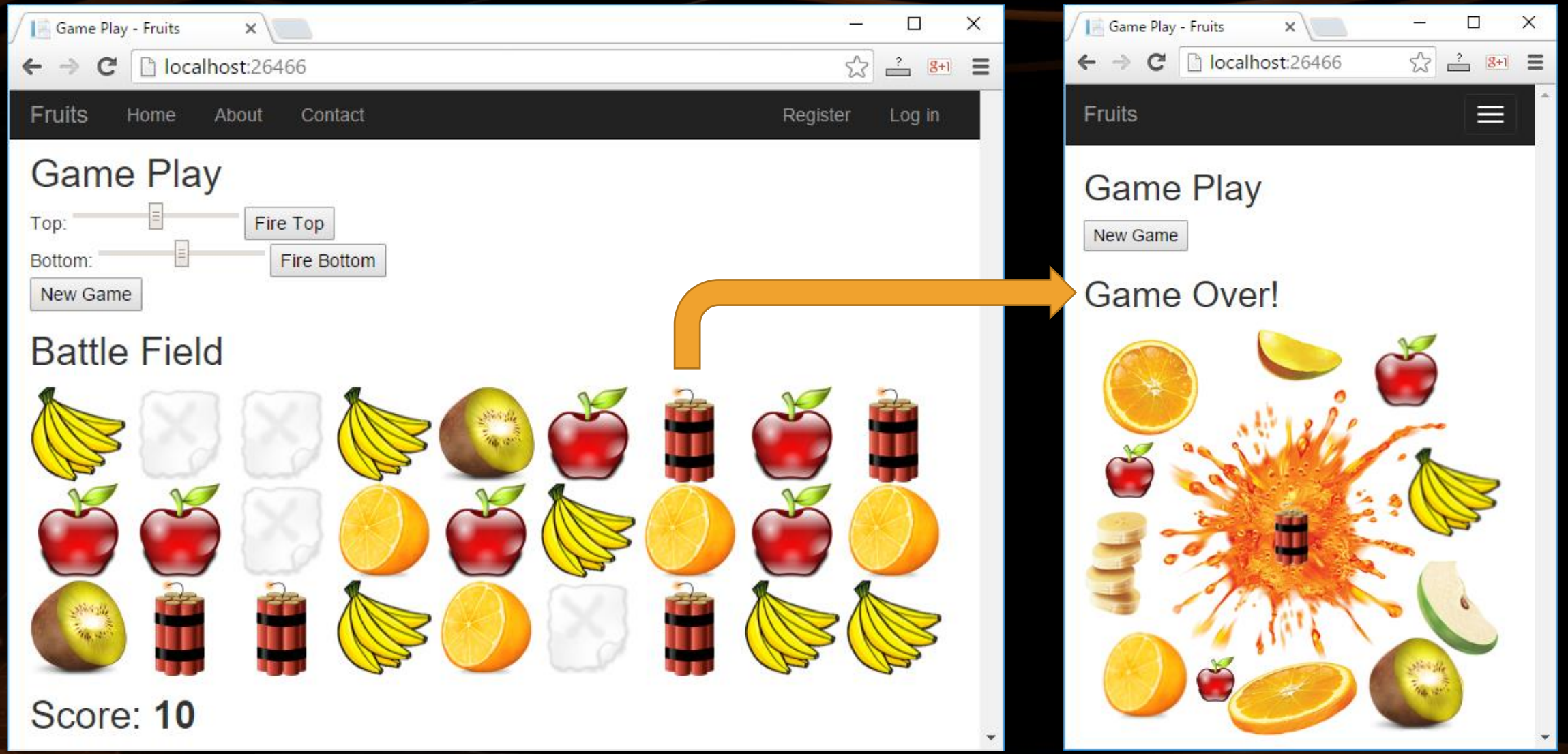
Имплементираме "Край на играта"

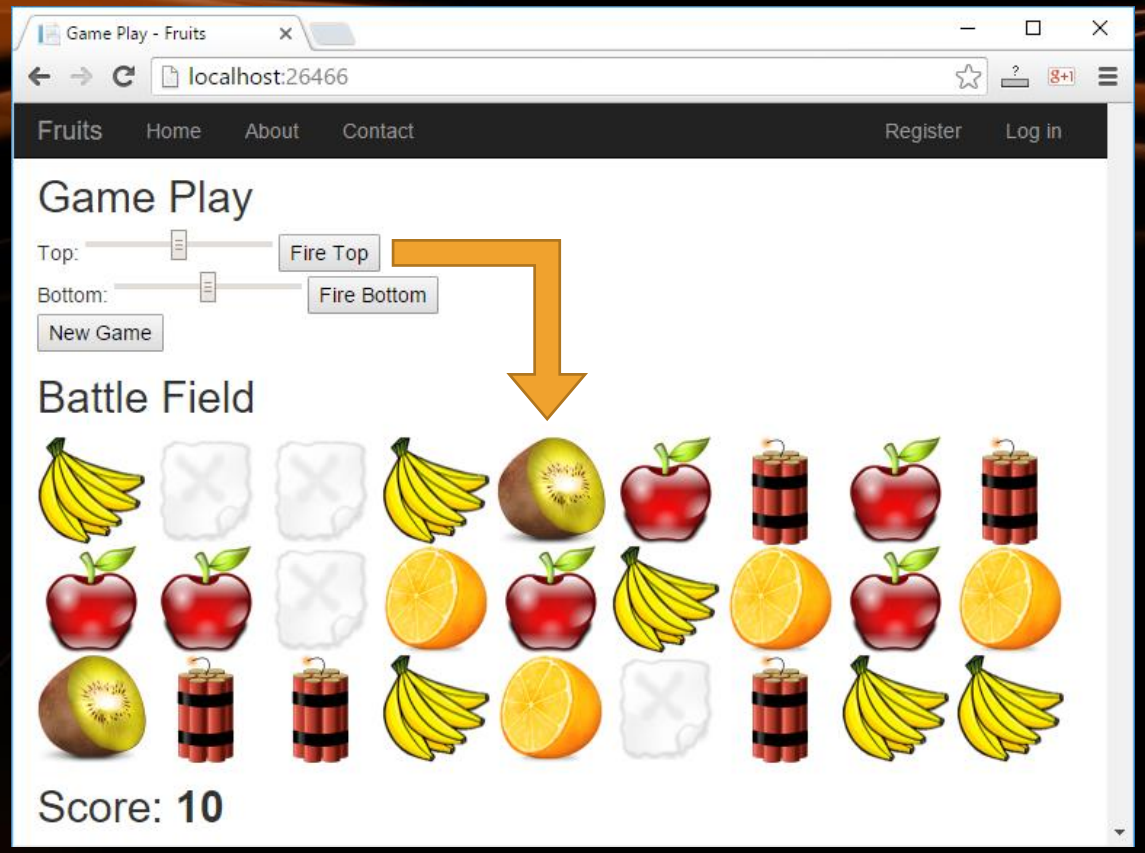


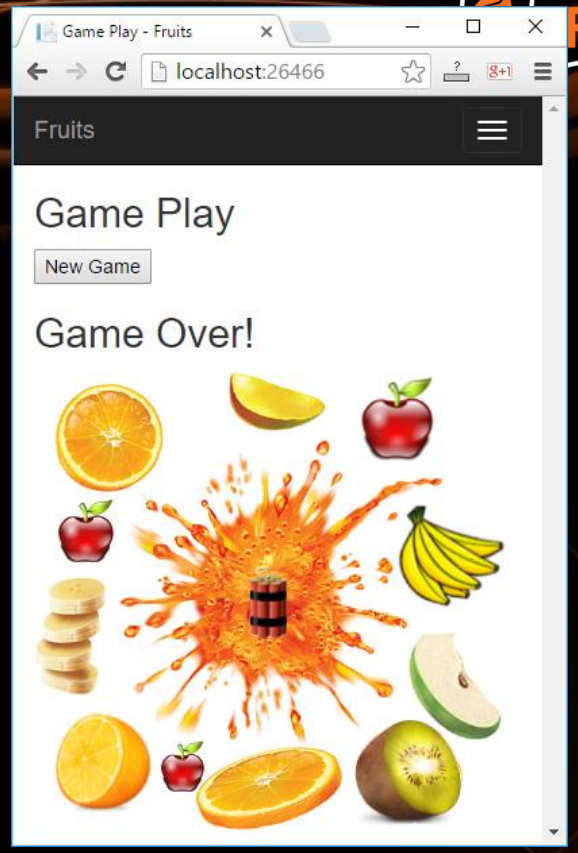
Имплементираме "Край на играта" (2)



Стартираме с [Ctrl+F5] и тестваме







Уеб игра "Обстреляй плодовете!"

Работа на живо в клас (лаб)

Какво научихме днес?

- Можем да ползваме **for**-цикли със стъпка:

```
for (var i = 1; i <= n; i+=3)
    Console.WriteLine(i);
```

- Цикли **while** / **do-while** повтаря докато е в сила дадено условие:

```
int num = 1;
while (num <= n)
    Console.WriteLine(num++);
```

Работа с по-сложни цикли

Въпроси?

https://softuni.bg/courses/programming-basics/

Лиценз

- Настоящият курс (слайдове, примери, видео, задачи и др.) се разпространяват под свободен лиценз "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International"

- Благодарности: настоящият материал може да съдържа части от следните източници
  - Книга "Основи на програмирането със C#" от Светлин Наков и колектив с лиценз CC-BY-SA

Безплатни обучения в СофтУни

- Фондация "Софтуерен университет" – softuni.org
- Софтуерен университет – качествено образование, професия и работа за софтуерни инженери
  - softuni.bg
- СофтУни @ Facebook
  - facebook.com/SoftwareUniversity
- СофтУни форуми – forum.softuni.bg