

Документација за проектот по визуелно програмирање “Spaceship”

Вовед:

Овој документ е напишан со цел да се покаже како изгледа и како се игра играта која е направена со проектот по предметот Визуелно програмирање.

Во нашата игра “Spaceship” играчот го контролира главниот вселенски брод со стрелките на тастатурата. И истиот може да се движи лево и десно. Пукањето на бродот го контролира со копчето space. После испукување на проектил, истиот лета по вертикална насока од леталото. Целта на играта е играчот да добие што поголем Score, а тоа го постигнува со уништување на непријатели. Секој срушен непријател носи 1 поен. Играта завршува кога некој од непријателите стигнува до координатите по Y-оска од леталото со кое игра играчот.

Инспирација за нашата игра црпемме од слични игри како “Chicken Invaders” и “Space Invaders”.

Опис на решението на проблемот:

Проблемот на играта се сведува на испукување на проектил од страна на бродот кон непријателите кои што паѓаат од горниот дел на екранот.

На самиот почеток, во кодот се креира листа на непријатели кои што се од специјална класа наречена “Enemy” за која опис има во делот: *Опис на класи и функции*. Секој објект од оваа класа е подреден во опаѓачки редослед според големината по Y оската. Со тоа се добива подредена листа во која првиот објект од класата “Enemy” е исто така најдоле на самиот екран. Секој објект кој што содржи сликичка е поставен на самата форма на случајна позиција по X и Y оската. Во самата форма има еден тајмер, кој што на секој негов тик, ја зголемува вредноста на Y координатата на секоја непријателска фигура од класата “Enemy” па така фигурите изгледаат како да паѓаат од горниот дел на екранот.

Вселенскиот брод со кој што играчот управува, може да се движи лево и десно и да испукува проектили во вертикална насока кон непријателите кои што паѓаат од горниот дел. Тоа се прави со манипулирање на вредностите на X оската после притискање на копчињата за лево и десно.

На секое испукување на проектил со притискање на копчето space, во кодот се генерира нов објект од класата “PictureBox” и се става во листа наречена “Bullets”. После неговото креирање се сетираат неговите атрибути како позиција, големина и сликичка со облик на проектил, и се повикува метода која пресметува дали овој проектил се поклопува по X оската со некој од непријателите, наречена isHit за која опис има во делот: *Опис на класи и функции*.

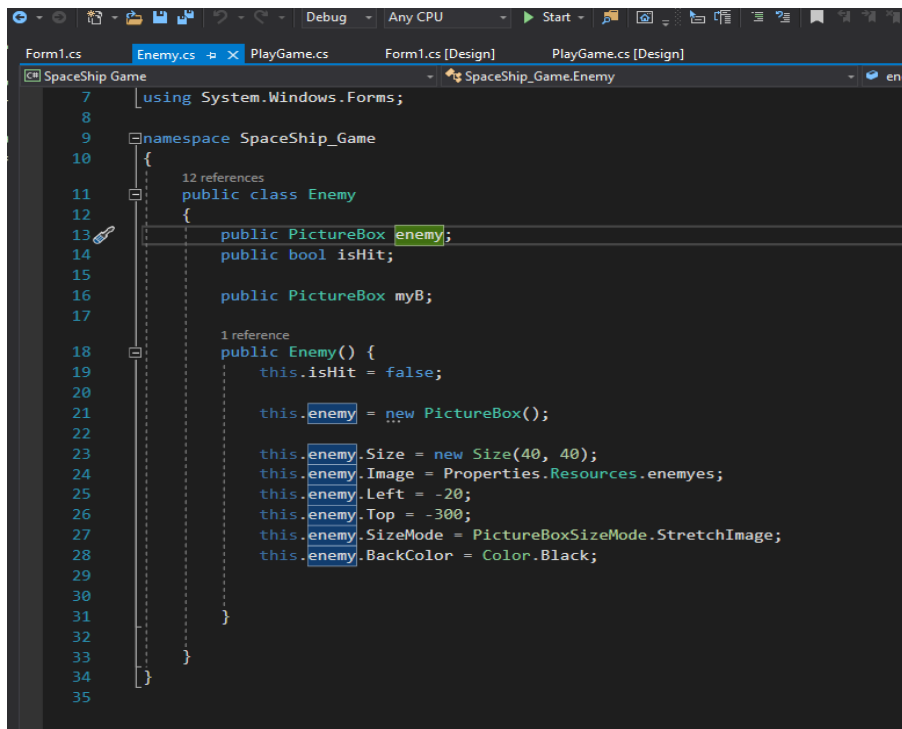
Поради подредената листа во која се чуваат непријателските фигури, функцијата ја наоѓа првата, и најдолната непријателска фигура, иако се подредени една позади друга. Ако неговата позиција се поклопува тоа значи дека проектилот лета кон некој од непријателите. Во тој случај кај објектот од класата "Enemy" кој што стои на патот на самиот проектил, се сетира еден белеан isHit на вредност True за да се знае дека тој во блиска иднина ќе треба да се избрише штом проектилот стигне и го погоди него. Исто така во самиот објект Enemy, се става референца кон објектот PictureBox што го репрезентира на проектилот кој лета према него.

После тоа се сетира уште еден белеан Fire на True во рамките на самата форма со цел, во наредниот тик на тајмерот, да се измине целата листа на проектили, и на секој објект од класата PictureBox што го репрезентира на секој проектил, се манипулира со вредноста на Y оската па така самиот проектил изгледа како да лета нагоре.

После тоа се изминува целата листа на непријатели и се баарат тие кои што имаат вредност на променливата isHit еднаква на True. Тогаш дополнително се проверува од референцата на проектилот кој што самиот објект од класата "Enemy" ја чува, дали истиот стигнал до објектот. Ако да, тогаш се повикува функција deleteHit која го брише непријателот од формата како и соодветниот проектил кој го погодил него. Проверката се врши со калкулирање по Y оската на проектилот и на непријателската фигура.

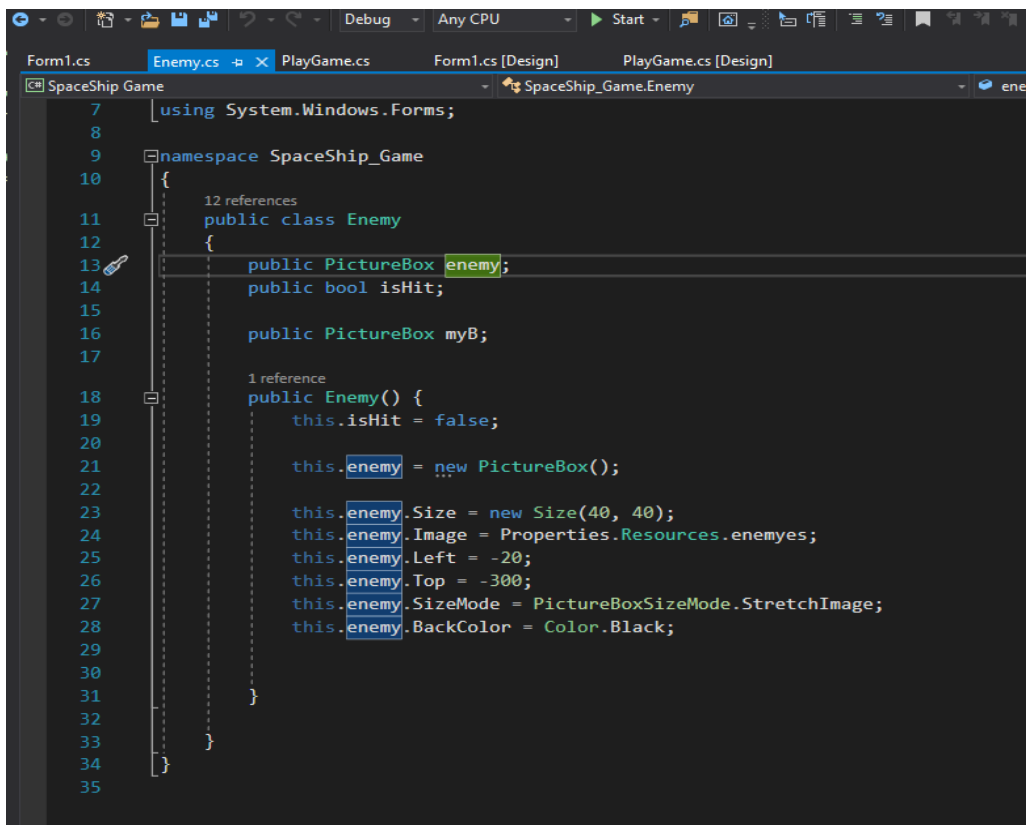
Дополнително, на секој тик на тајмерот, се проверива дали првиот елемент во подредената листа на непријатели, тој што е најдоле на екранот, стигнал до дното на екранот, при што играта завршува и се печатат освоените поени на екранот.

Опис на класи и функции:



```
7  [using System.Windows.Forms;
8
9  namespace SpaceShip_Game
10 {
11     12 references
12     public class Enemy
13     {
14         public PictureBox enemy;
15         public bool isHit;
16
17         public PictureBox myB;
18
19     1 reference
20     public Enemy() {
21         this.isHit = false;
22
23         this.enemy = new PictureBox();
24
25         this.enemy.Size = new Size(40, 40);
26         this.enemy.Image = Properties.Resources.enemies;
27         this.enemy.Left = -20;
28         this.enemy.Top = -300;
29         this.enemy.SizeMode = PictureBoxSizeMode.StretchImage;
30         this.enemy.BackColor = Color.Black;
31     }
32 }
33
34
35
```

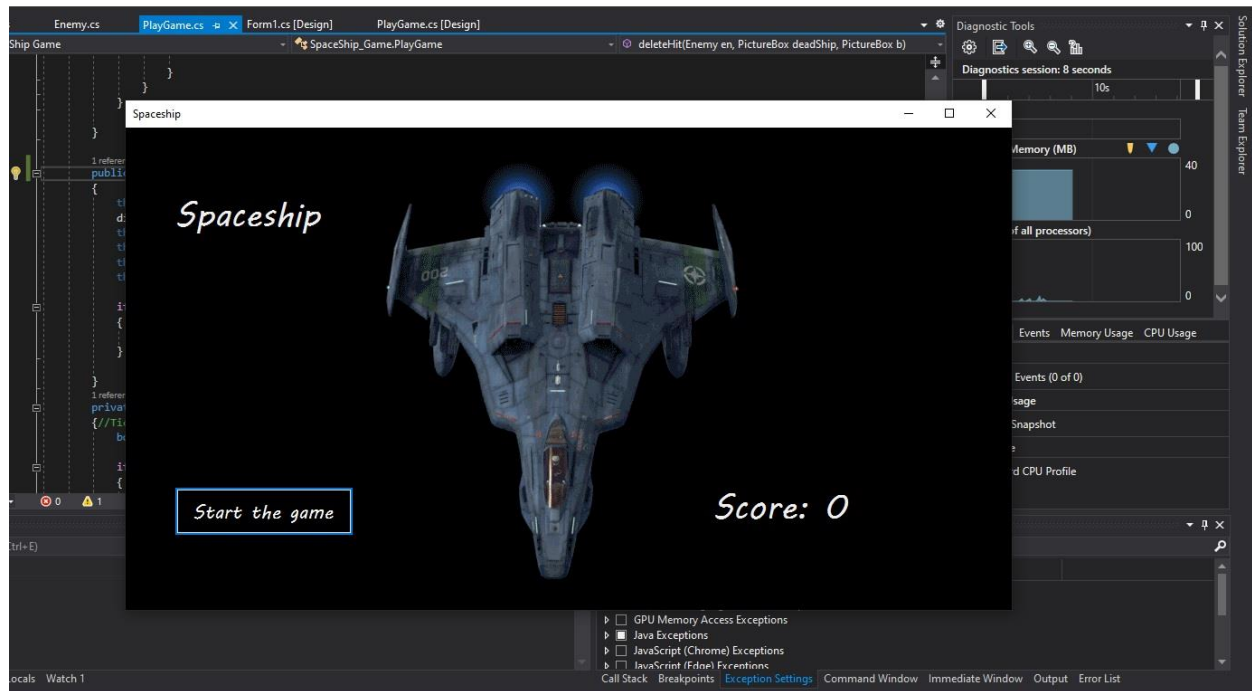
Класата Enemy има атрибути PictureBox enemy и bool isHit. Enemy() е дифолтен конструктор на класата. Кога креираме нов enemy, isHit(означува дали објектот enemy ќе биде погоден или не неговата почетна вредност е false). За објектот се креира PictureBox за кое се наведени properties Size, Image, Left, Top, SizeMode, BackColor. Во променливата myB се чува покажувач кон објектот од класата PictureBox кој што е проектилот кој го гаѓа на самиот објект од класата Enemy.



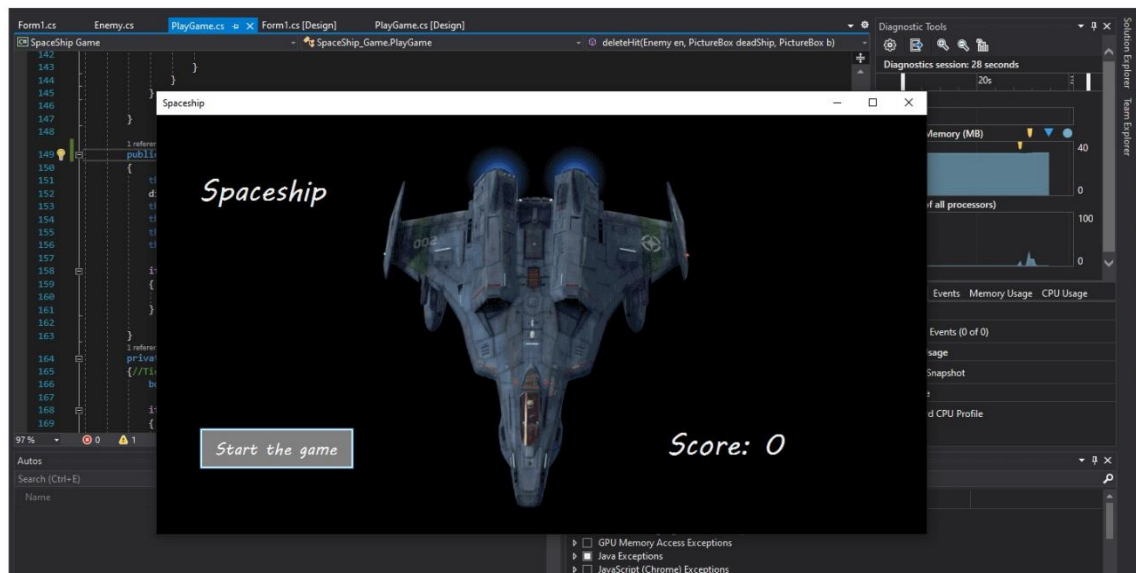
```
7 using System.Windows.Forms;
8
9 namespace SpaceShip_Game
10 {
11     12 references
12     public class Enemy
13     {
14         public PictureBox enemy;
15         public bool isHit;
16
17         public PictureBox myB;
18
19         1 reference
20         public Enemy() {
21             this.isHit = false;
22
23             this.enemy = new PictureBox();
24
25             this.enemy.Size = new Size(40, 40);
26             this.enemy.Image = Properties.Resources.enemeyes;
27             this.enemy.Left = -20;
28             this.enemy.Top = -300;
29             this.enemy.SizeMode = PictureBoxSizeMode.StretchImage;
30             this.enemy.BackColor = Color.Black;
31         }
32     }
33 }
34
35
```

Функцијата isHit прима еден аргумент од тип PictureBox. Потоа за секој објект во листата Enemy правиме проверка дали се поклопува во координатите со некој овјект од оваа листа, доколку се поклопува правиме наредна проверка дали има некој претходен куршум кој исто така се поклопува со координатите на истиот објект по y-оска. Со тоа што ако нема претходен куршум знаеме дека овој куршум ќе го погоди објектот со кој се поклопува па во објектот го менуваме атрибутот isHit= true, и соодветно за објектот зачувуваме дека куршумот ќе го погоди е.myB=bullet. Ако пак вториот if-услов не е точен односно објектот кој се поклопува со сегашниот куршум веќе има друг куршум со кој ќе биде погоден, сегашниот куршум ќе го прескокни тој објект и ќе проба дали неговите координати се поклопуваат со некој од другите членови во листа Enemy. Доколку првиот if-услов не е точен, преминуваме на нареден елемент за споредба од класата Enemy.

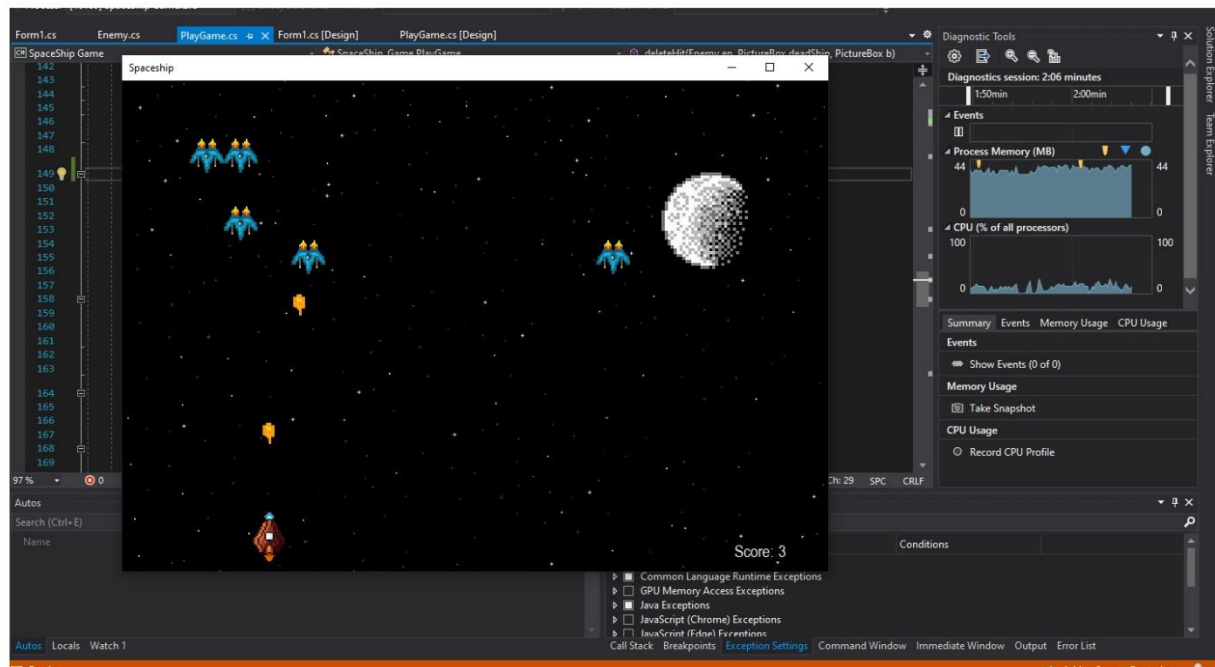
Изглед на играта:



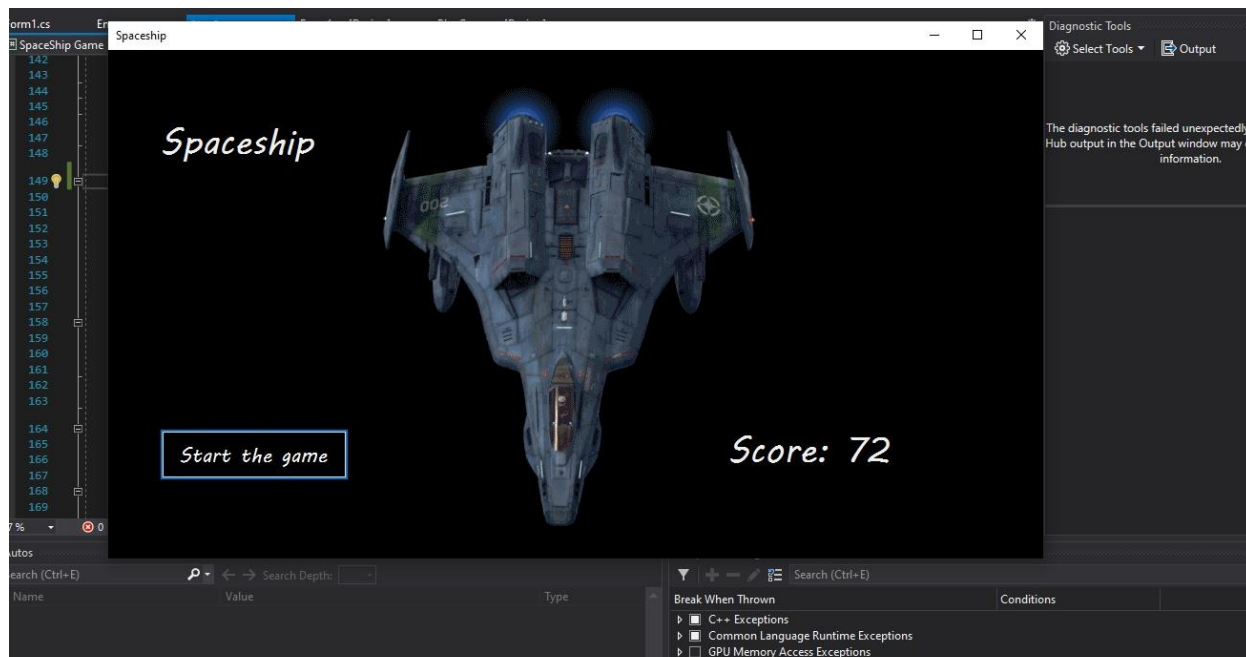
Фиг1. Почетен екран.



Фиг2. Кликнување на копчето за стартување на играта.



Фиг3. Околната за играње, непријателите летаат кон леталото, додека играчот испукува проектили кон ниб.



Фиг5. Испитување на максималните поени освоени во текој на играта после нејзино завршување.