Universitatea Tehnica din

Cluj-Napoca

**Proiect Masuratori și Traductoare**

**Facultatea de Automatica și Calculatoare**

**Anul II**

**Pătrățel**

Autor:

**Dologa Răzvan**

**Cuprins**

[1. Motivația alegerii temei. 3](#_Toc513042098)

2. Noțiuni teoretice de bază ..................................................................................................................3

[3. Descrierea aplicației. 6](#_Toc513042103)

[4. Aplicația 7](#_Toc513042104)

[5. Assets 8](#_Toc513042107)

[6. Scripts 9](#_Toc513042108)

[7. Dezvoltare și înbunătățire ulterioară 19](#_Toc513042114)

[8. Utilizari 19](#_Toc513042115)

[9.Bibleografie 19](#_Toc513042115)

# 

# 1. Motivația alegerii temei.

Am ales ca temă jocurile video datorită pasiunii mele pentru jocuri și datorita accesibilitatii acestui subiect.Totodată am vazut o oportunitate de a-mi testa cunoștințele acumulate pana in prezent si de a iesi din zona de comfort acceptând noi provocari.

# 2. Noțiuni teoretice de bază.

## Unity este un game engine cu ajutorul caruia putem realiza jocuri 2D cat si 3D pentru PC,console, telefoane mobile si website. Acesta a fost lansat in 2005 si este scris in C,C++ si C#.Initial acest game engine a fost lansat doar pentru Sistemele de operare Mac, iar in versiunea lansata initial continea shadere orientate OpenGL. In urmatoarele versiuni de Unity 1.x s-a inclus suport pentru rularea jocurilor pe sistemul de operare Windows. Unity este un game engine care a evoluat mult de-a lungul anilor, reusind sa isi extinda ariaca dispozitive target.

## A devenit mult mai usor de utilizat, oferind programatorului cat si amatorilor posibilitatea de a creea jocuri 2D cat si 3D cu un bagaj mediu de cunostinte. Cu timpul a devenit tot mai cunoscut prin jocurile create si s-a format o intreaga comunitate de developeri amatori si profesionisti care pot pune la dispozitie raspunsuri la intrebarile utilizatorilor.

## De asemena pe site-ul official (www.unity3d.com) se gasesc o multime de tutoriale inclusive live, cu scopul de a demonstra cat de practic si versatil poate fi acest game engine, iar ca un plus, noua versiune lansata in acest an Unity 5, estegratuita. Unity suporta 3 limbaje diferite ( C#, JavaScript si Boo), oferind programatorilor posibilitatea sa lucreze in limbajul preferat. In acest moment C# este limbajul cel mai utilizat,Boo insa este un limbaj de programare aflat inca in umbra folosit de o parte mica a programatorilor in Unity.

## In Unity se lucreaza in Monodevelop, ceea ce este o versiune customizata pentru Unity.Monodevelop este un mediu de lucru open source pentru Linux si Windows. Contine functii similare cu Microsoft Visual Studio si NetBeans precum completarea automata a codului sau crearea interfetelor grafice (GUI), suporta Boo, C#, JavaScript etc.Unity game engine are o interfata simpla si intuitiva, ferestrele cu care interactionezi cel mai des sunt urmatoarele:

1. Scena - reprezinta toate componentele pe care le ai in joc (fiecare scena contine

o camera);

2. Game - reprezinta fereastra cu jocul si este ceea ce camera ta vede in scena;

3. Ierarhie – contine structurate obiectele din scena;

4. Proiect - reprezinta proiectul utilizatorului si contine scripturile, elemente prefabricate,

materiale, texture, assets etc.

5. Consola – pentru a obtine informatii legate de:proiectul curent,anumite erori de compilare, de sintaxa in cod precum si zona in care se va putea urmari comportamentul variabilelor , adica un debug.

6.Inspectorul - aici apare fiecare componenta a obiectului tau (fiecare obiect este considerat un GameObject), se poate modifica fiecare parametru dorit.

Ca editor pentru Scripturi(programele care contin codul efectiv și care pun totul in mișcare) unity foloseste Visual Studio 2019.

La momentul actual între 60-80% din jocuri sunt realizate cu ajutorul lui Unity.

### 2. 1. 2 Evenimente generate de controale

Atunci când utilizatorul acţionează asupra unui control (apasă un buton, selectează o opţiune radio, scrie într-o listă, etc.) aplicaţia trebuie să fie capabilă să sesizeze faptul ca s-a acţionat asupra controlului respectiv, să identifice tipul acţiunii şi să execute o secvenţă de cod care să rezolve problema în concordanţă cu tipul acţiunii. Pentru a sesiza acţiunea asupra lor, controalele generează evenimente. Clasa ***Control*** implementează un set de evenimente comune tuturor controalelor, iar acestea, la rândul lor, vor genera şi un set de evenimente specifice. Cele mai uzuale evenimente sunt prezentate mai jos, lista completă a acestora fiind disponibilă în MSDN.

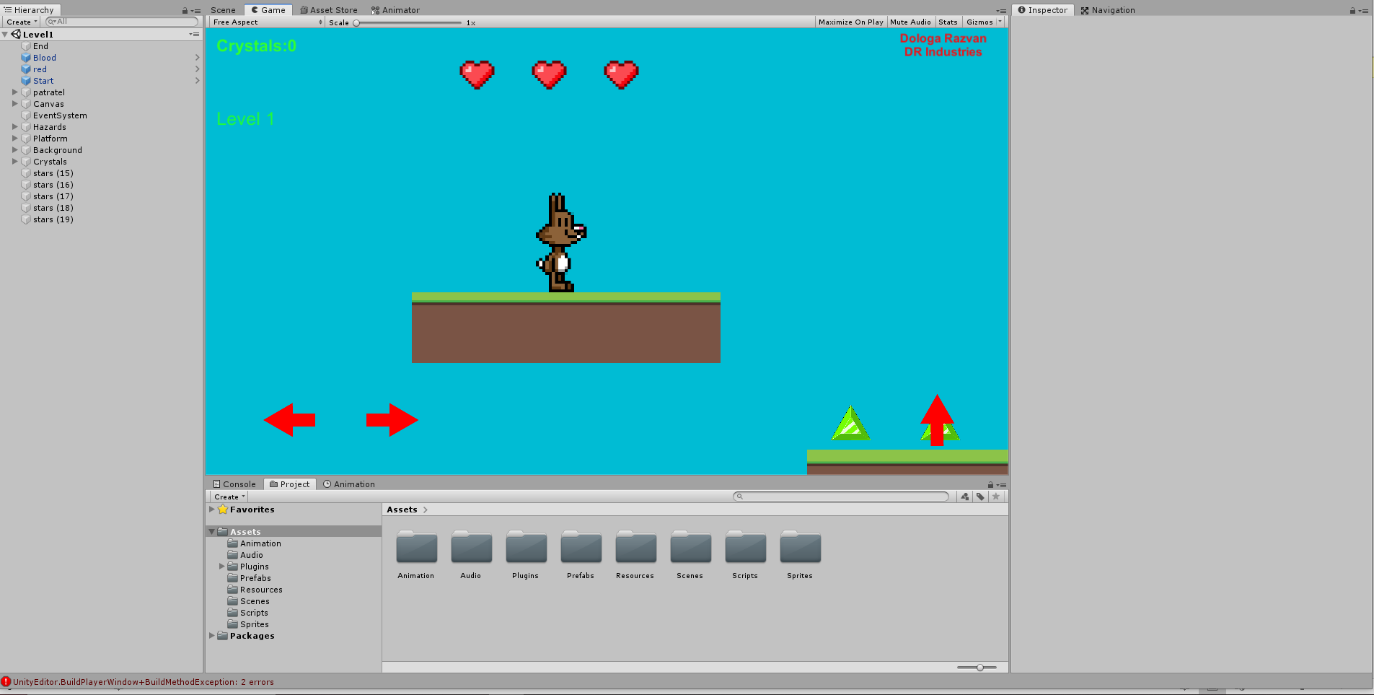
* **Click** – este generat când se apasă click cu mouse-ul asupra unui control. Poate fi uneori generat şi de apăsarea tastei Enter.
* **DoubleClick** – este generat cînd se apasă dublu click asupra unui control. Dacă controlul este buton, acest eveniment nu va putea fi generat, pentru ca la prima apăsare se generează automat Click.
* **DragDrop** – este generat când se finalizează (este eliberat butonul mouse-lui) o operaţie drag-and-drop în care este implicat controlul.
* **DragEnter** – este generat atunci când obiectul implicat într-o operaţie drag-and-drop ajunge în interiorul controlului.
* **DragLeave** - este generat atunci când obiectul implicat într-o operaţie drag-and-drop părăseşte suprafaţa controlului.
* **DragOver** – este generat când un obiect implicat într-o operaţie drag-and-drop ajunge deasupra controlului.
* **KeyDown** – este generat când o tastă devine apăsată în timp ce controlul deţine input focusul. Se generează întotdeauna înainte de KeyPress şi KeyUp.
* **KeyPress** - este generat când o tastă devine apăsată în timp ce controlul deţine input focusul. Se generează întotdeauna după KeyDown şi înainte de KeyUp. Spre deosebire de KeyDown care furnizează codul de scanare al tastei apăsate, KeyPress furnizează codul ASCII al tastei.
* **KeyUp** - este generat când o tastă este eliberată în timp ce controlul deţine input focusul. Se generează întotdeauna după KeyDown şi KeyPress.
* **GotFocus** – este generat când controlul primeşte input focusul.
* **LostFocus** – este generat când controlul pierde input focusul.
* **MouseDown** – este generat când prompterul mouse-lui este deasupra controlului şi se apasă o tastă a mouse-lui.
* **MouseMove** – este generat continuu atâta timp cât prompterul mouse-lui traversează controlul.
* **MouseUp** - este generat când prompterul mouse-lui este deasupra controlului şi se eliberează o tastă a mouse-lui.
* **Paint** – se generează la desenarea controlului.
* **Validated** – este generat când un control este în curs de a primi focusul. Se generează după ce evenimentul Validating se termină şi indică faptul că validarea controlului este completă.
* **Validating** - este generat când un control este în curs de a primi focusul.

# 3. Descrierea aplicației.

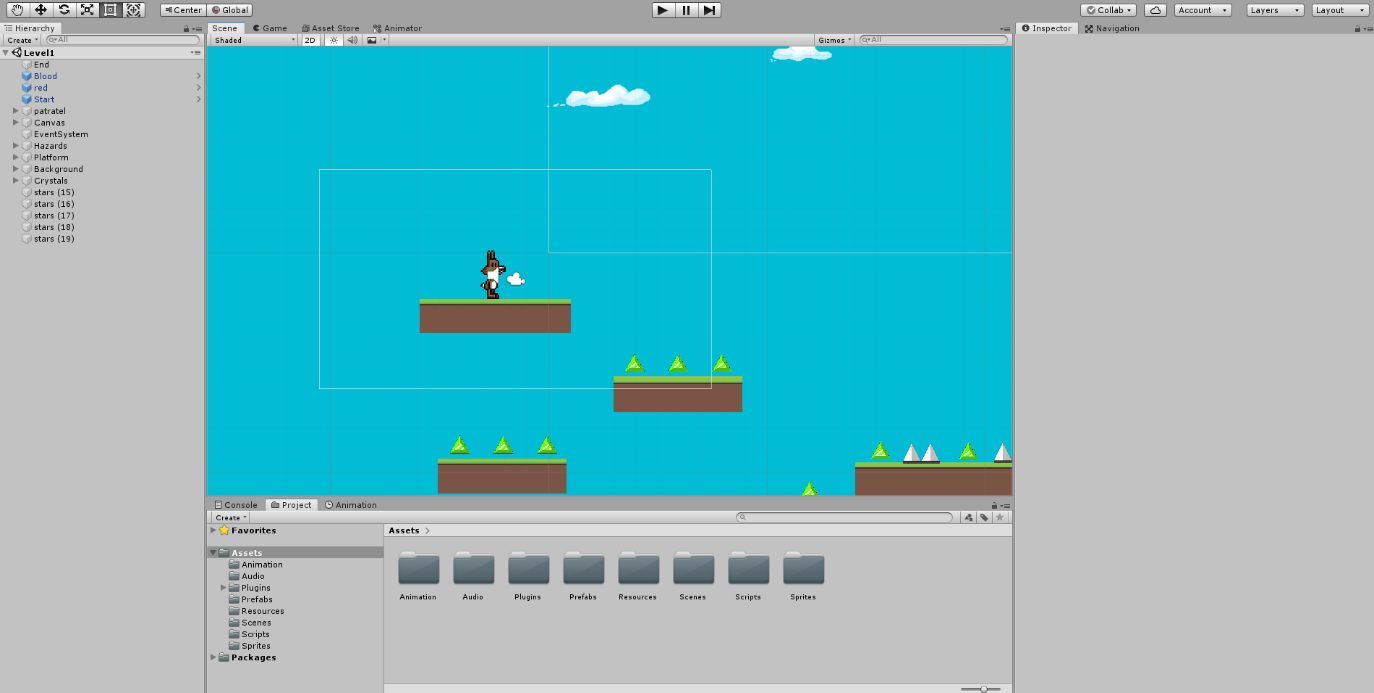
Aplicația se deschide direct cu primul nivel al jocului iar odata cu aceasta incepe si coloana sonoră a jocului.În versiunea actuală a jocului sunt prezente 3 nivele cu o dificultate care crește gradual odata cu nivelul,trecerea de la un nivel la altul se realizeaza în momentul în care jucatorul colectează un anumit număr de cristale(afișate în timp real în partea stangă sus a ecranului).Jocul prezintă de asemenea un sistem de vieți(vizibile în partea de sus a ecranului) menit să faca jocul mai palpitant și mai realist.Totuși daca jucatorul moare de 3 ori(prin interacțiunea cu inamicii,cu obstacolele sau prin căderea în gol) riscă sa piardă vieți,iar în momentul cand a pierdut 3 vieți(a murit de 3 ori) nivelul este reluat de la 0 fiind nevoit sa colecteza iar cristalele.

“Pătrațel ” este un joc de tip arcade inspirat din jocuri clasice precum Mario,Donkey Kong sau Legend of Zelda jocuri foarte populare in urma cu 10-20 de ani.

# 4. Aplicația



## 



Se observă cele doua moduri de vizualizare ale aplicației din programul Unity:

**1.Scene view**

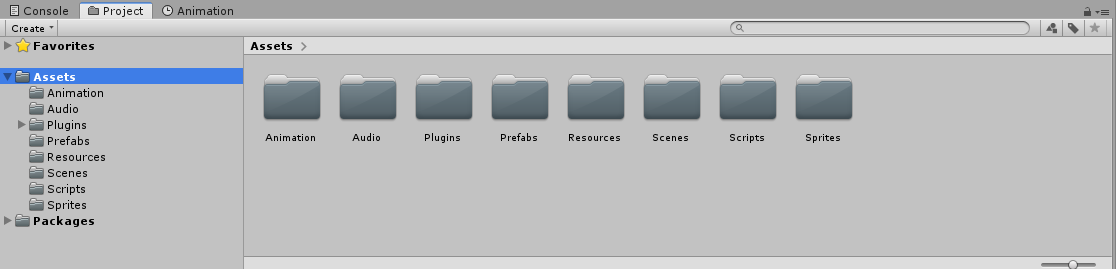
Aici putem vedea si edita nivelul,acesta in sine fiind o scena,care permite editarea si manipularea diferitelor obiecte din joc(inamici,platforme,personal principal,start,delimitari ale suprafetei,) asezarea lor fiind posibila cu ajutorul unui set de coordonate vizibile din modul inspector.Vizibil doar in interiorul programului si doar de catre programator.

**2.Game view**

Acest mod nu permite manipularea sau editarea datelor sau diferitelor obiecte,reprezendând vederea din puctul de vedere al jucatorului(Aplicatia).

Se observa butoanele virtuale(pentru ecranul tactil).

În partea de sus se observa un set de butoane,acestea faciliteaza testarea programului in orice moment(cu conditia ca acesta sa nu aiba erori de compilare sau de sintaxa),la apăsarea butonului Play jocul intra automat in modul de vizualizare game,hotkeyurile(tastele utilizate pentru controlul personajului) devin active.Butonul Pause permite oprirea aplicatiei in orice moment,pentru a observa anumite detalii precum pozitia anumitor caractere,distanta etc.



În partea de jos se observa un “meniu” care permite navigarea intre consola(Locul in care sunt evidentiate erorile din program)Project(Fisierele proiectului nostru) și Animation sau animator unde putem realiza animatii(cu ajutorul unor sprite-uri realizate de noi).

**5.Assets**

Acest fisier reprezintă inima si sufletul aplicației noastre,fiind locul in care sunt stocate toate elementele componente:

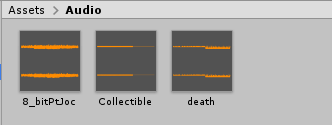
1.Animation



Fișierul în care sunt stocate animațiile personajului nostru,în cazul de fata animatia pentru deplasare si una pentru stat,alternanta între ele fiind realizata în fisierul numit patratel,in functie de apasarea anumitor taste sau butoane virtuale(in cazul aplicatiei de telefon).

2.Audio

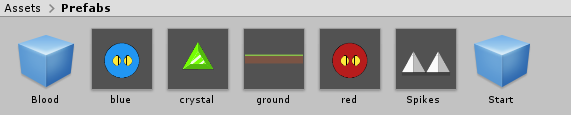
Aici sunt prezente sunetele utilizate in cadrul jocului,pentru colectarea cirstalelor,moartea personajului,si coloana sonoră de fundal.Acestea alternând in functie de evenimente.



3.Plugins

Contine fisierele necesare conversiei aplicatiei,aceasta putând fii convertita la diferite tipuri de sisteme de operare (ios,android,windows) sau chiar console.(Xbox de exemplu),în urma conversiei aplicatia devine de sine statatoare.

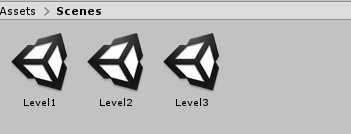
4.Prefabs



Acest fisier contine elemente predefinite de programator,pentru a usura generarea de noi nivele.

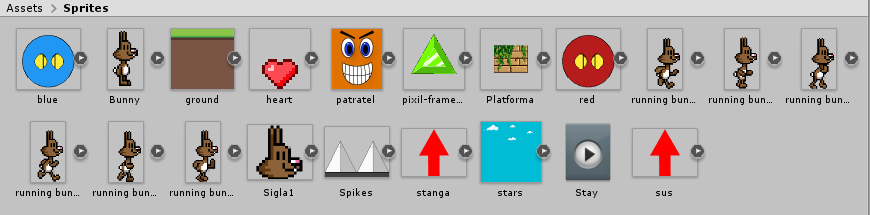
Deoarece avem nevoie de mai multe obiecte pentru a realiza un nivel,in loc sa creeam mai multe game objecturi si sa le atribuim fiecaruia in parte proprietatile necesare functionarii,realizam elemente prefabricate,deci daca dorim sa adaugam o platforma sau un inamic,este suficient sa dam drag din fisierul prefabs.

5.Scenes



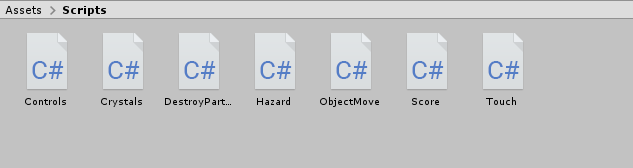
Scenele jocului reprezentand nivelele,trecerea de la un nivel la altul se realizeaza automat prin trecerea de la o scena la alta odata ce conditiile sunt indelplinite.

6.Sprites



Toate elementele de grafica necesare joculului,precum:diferitele imagini necesare animatiei de deplasare,inamicii,cristalelele,vietile,personajul,fundalul,platforma,sigla jocului etc.

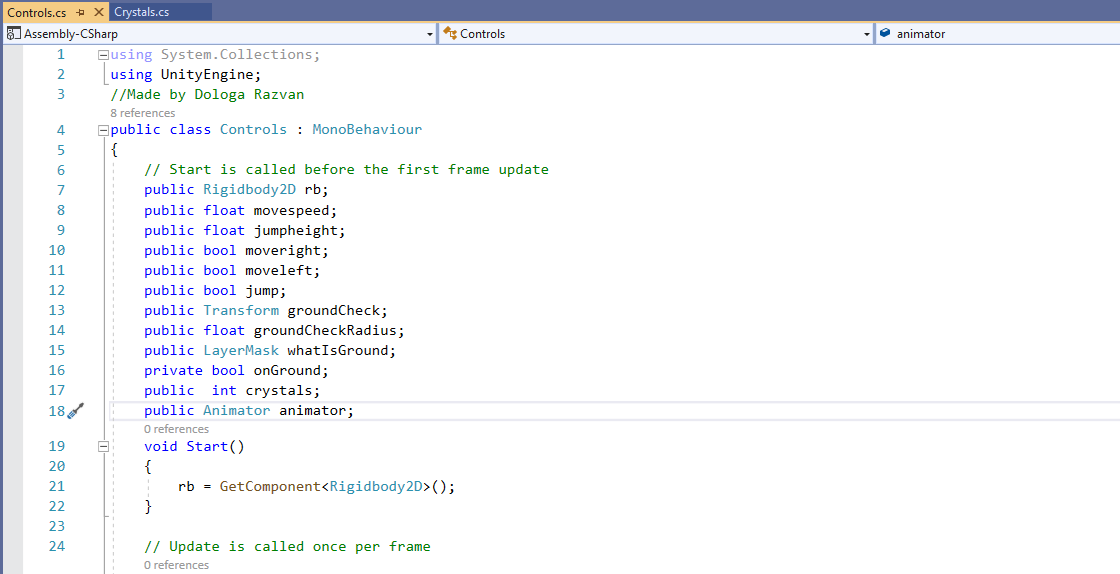
7.Scipts



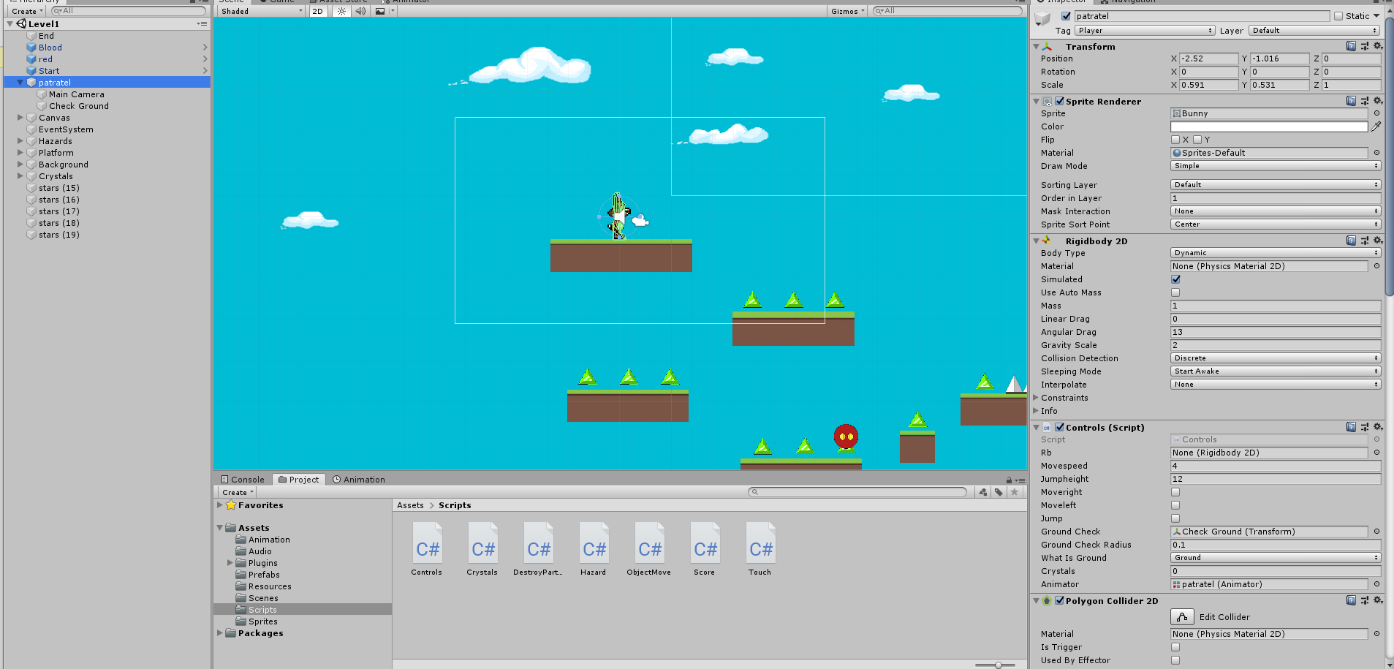
Acestea reprezinta toata mecanica jocului,miscarile,evenimentele,interactiunile etc,ele fiind atribuite sprite-urilor mentionate anterior ca proprietati.

Un script in general consta in 3 parti,partea de sus in care se regasesc librariile si eventualele variabile globale,partea de start in care se seteaza anumiti parametri inainte de rularea efectiva a aplicatiei,si partea de update in cadrul careia se intample evenimente in timp real.

**7.1.Controls**

****

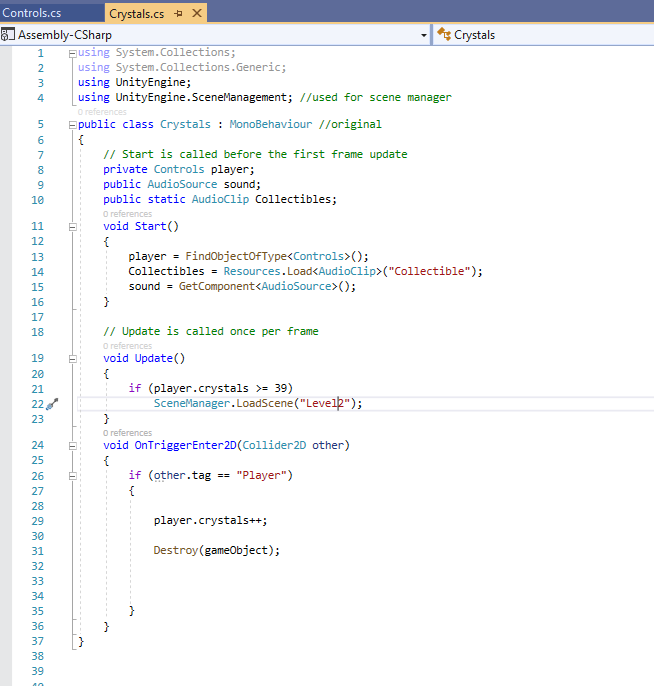


Aici se gasesc toate codurile de control ale personajului nostru pentru a facilita funciitle de deplasare si saritura,se observa ca acestea sunt realizate atat pentru tastatura(KeyCode) cat si pentru ecranul tactil(Moveright,Moveleft) precum si subprogramul onGround care verifica daca personajul nostru se afla pe pamnt inainte de a efectua o saritura,pentru acesta se utilizeaza un game object numit ground check care se afla mereu sup personaj si verifica daca exista o platforam pe o raza de 1 pixel.

Se observa ca personajul nostru are atribuit scriptul pentru control pentru a efectua miscarile mentionate anterior,de asemenea in stanga sus este vizibila structura ierarhica in care se observa ca personajului îi sunt atribuite camera și check ground pentru ca acestea sa se deplaseze odata cu el.

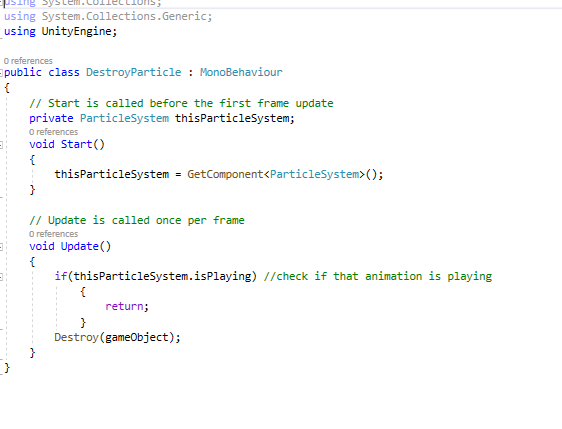
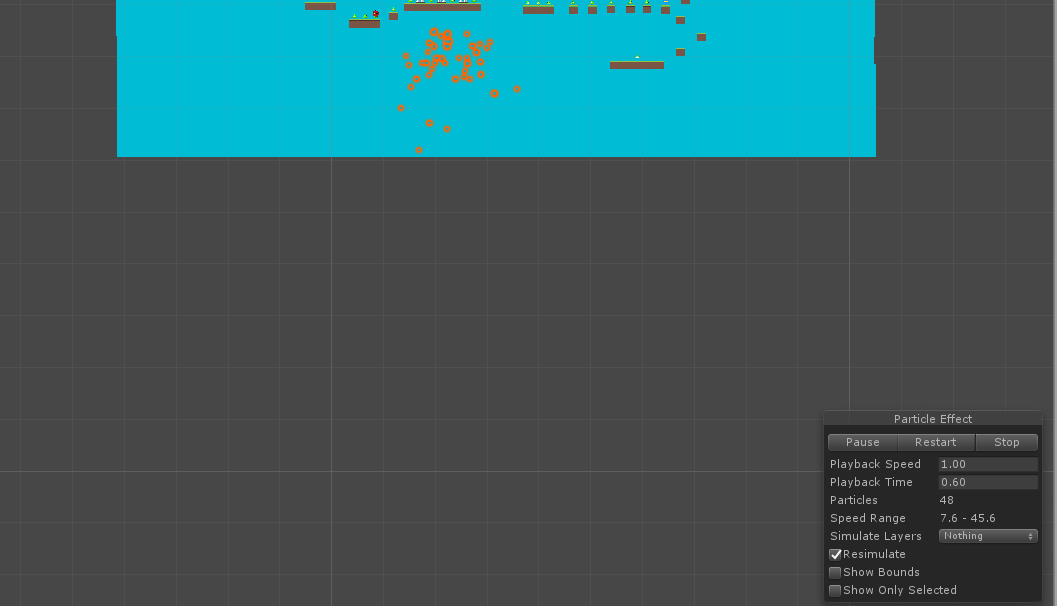
Liniile verzi de pe personaj reprezinta un Polygon colider de tip 2D,un hitbox care faciliteaza interactiunile acestuia cu alte elemente,pe baza unui anumit tag(eticheta).

**7.2.Crystals**

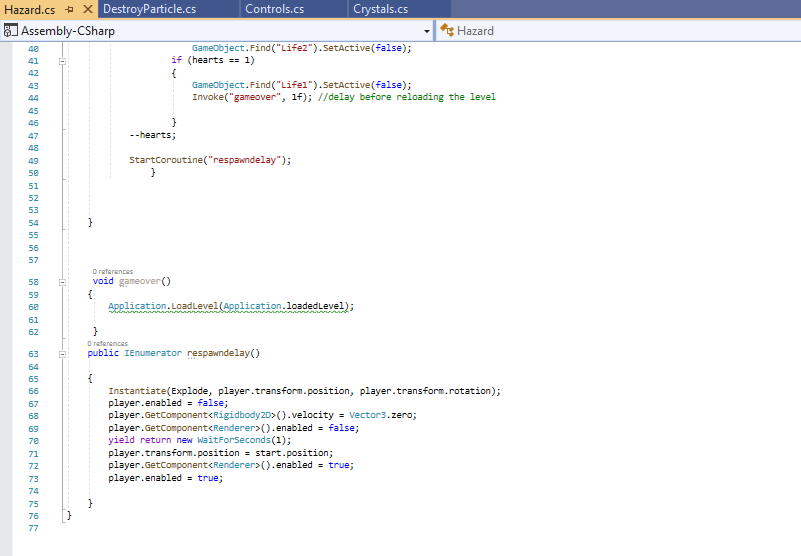
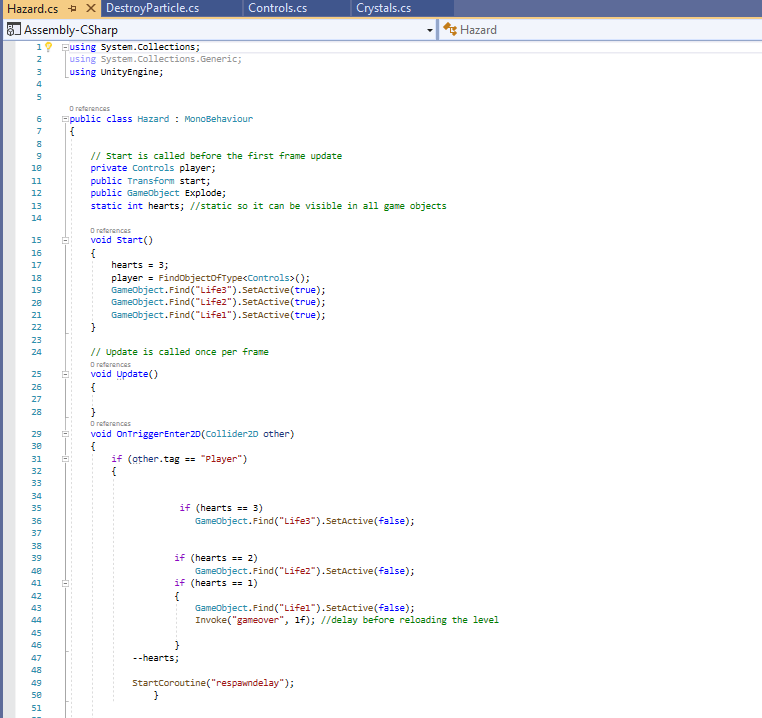


Codul pentru cristale,in momentul in care caracterul nostru atinge cristalele incrementam numarul de cristale colectate pana atunci.Se observa ca la colectarea unui anumit numar de cristale se realizeaza trecerea la urmatorul nivel prin procedura LoadScene(numele nivelului).

**7.3.Animatia pentru moartea personajului**

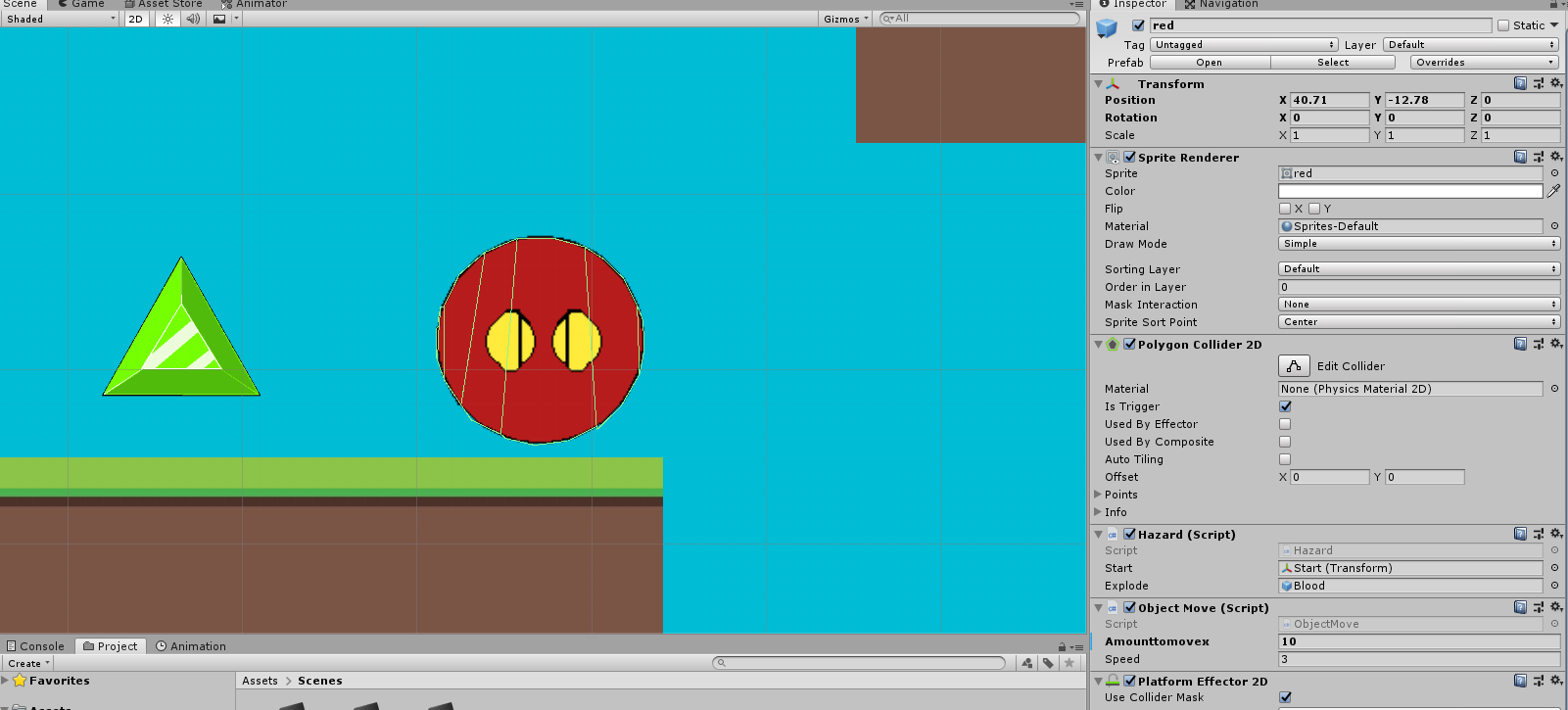
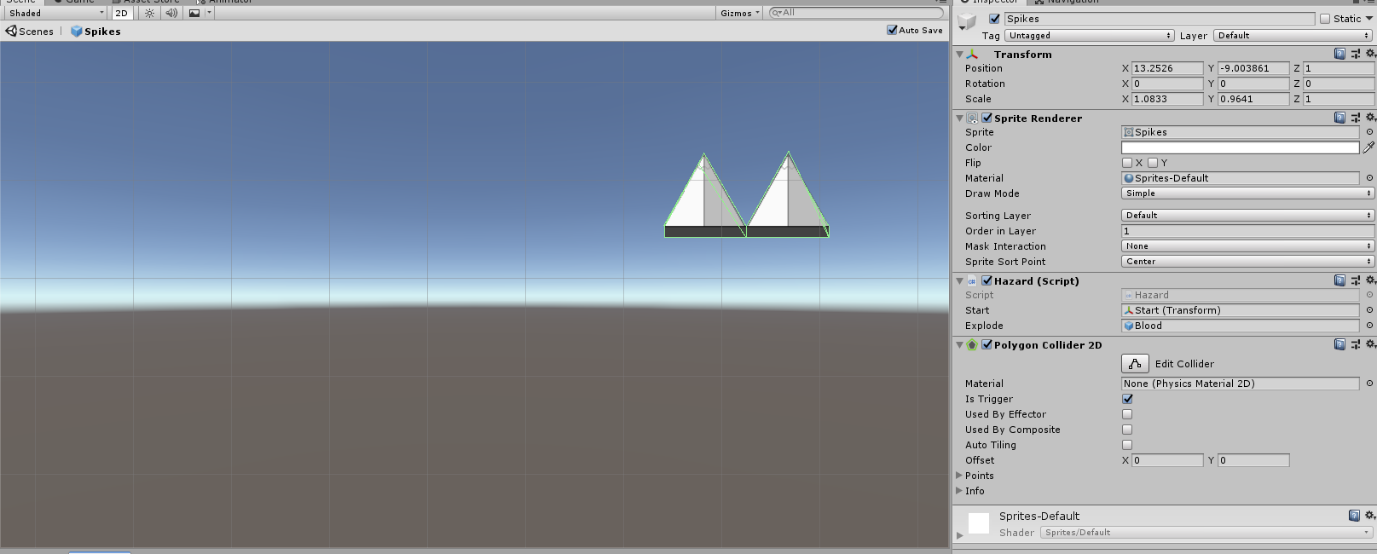


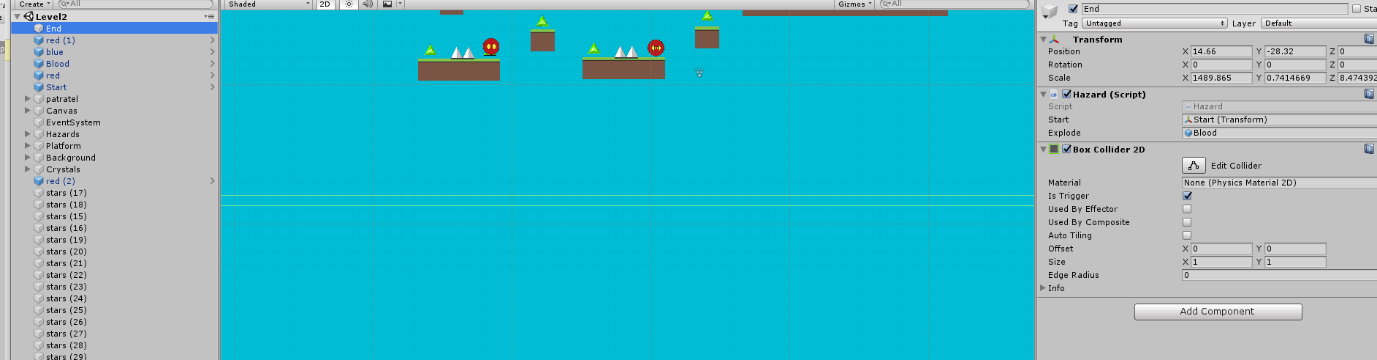
**7.4.Hazards**

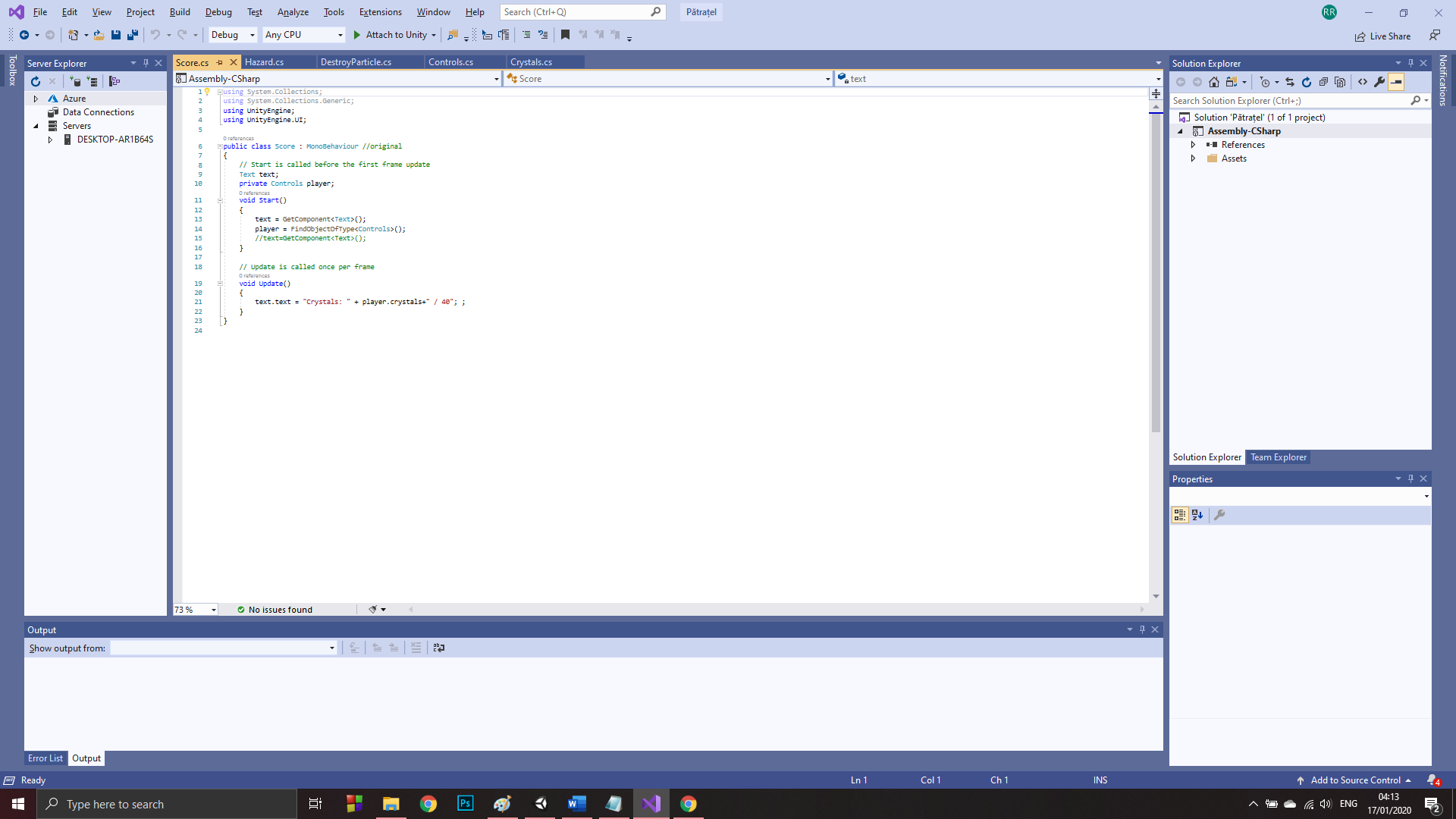
****

Aici se regasesc codurile pentru inamici personajului dar si sistemul de vieti,in cazul in care jucatorul ramane fara vieti,nivelul este reluat de la inceput.

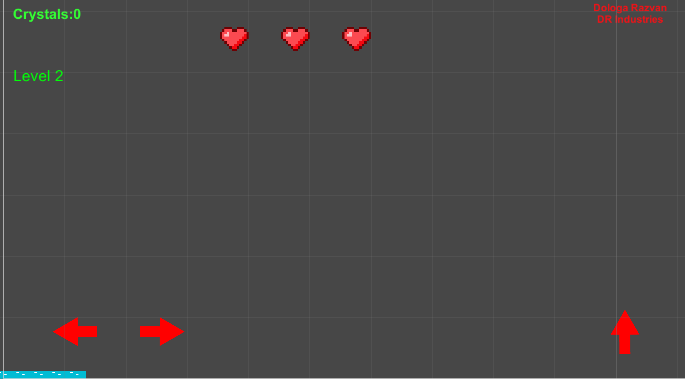
Orice game object care are atasat codul hazard si un hitbox (polygon collider) devine un inamic pentru personaj,deoarece orice interactiune cu acesta va “costa” o viata.

Delimitarea nivelului folosind acelasi cod,fara prezenta acestuia,personajul ar cadea la infinit. 

**7.5.Score**  

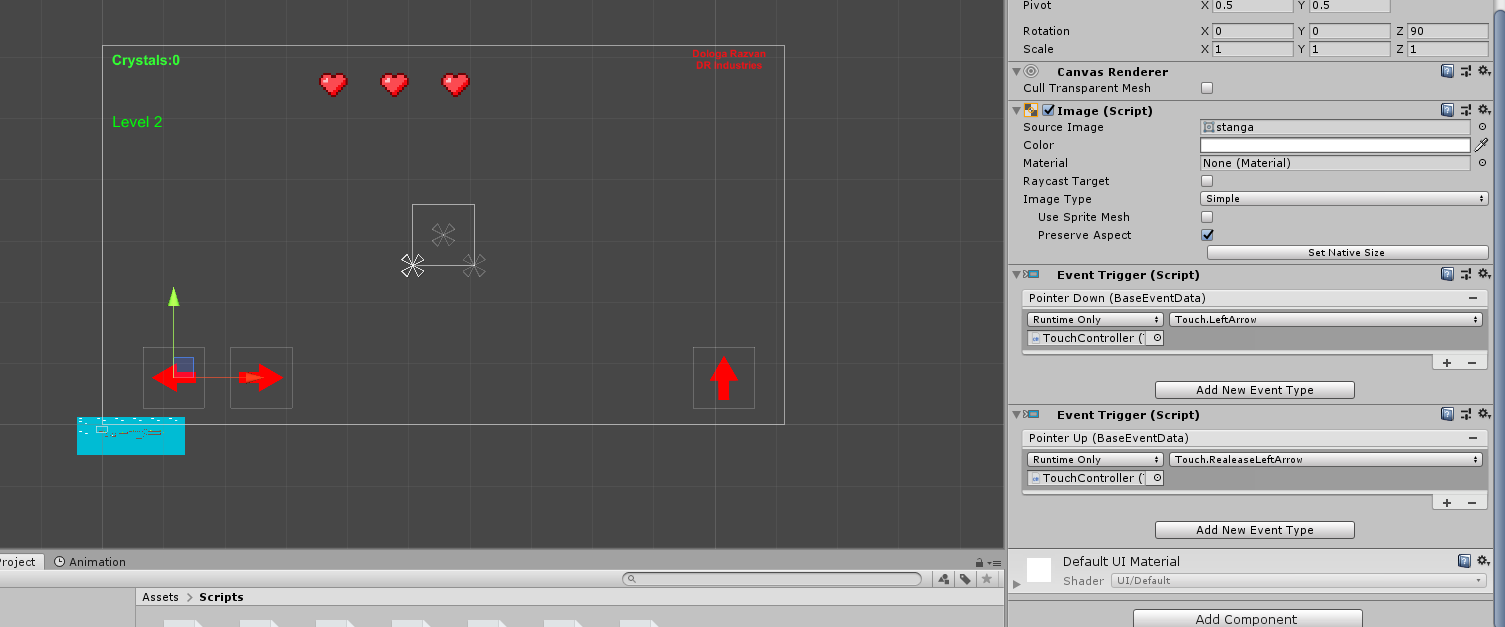
Codul pentru afisarea numarului de cristale in timp real utilizand UI objects (precum butoanele pentru ecranul tactil) vizibile doar in modul Game.



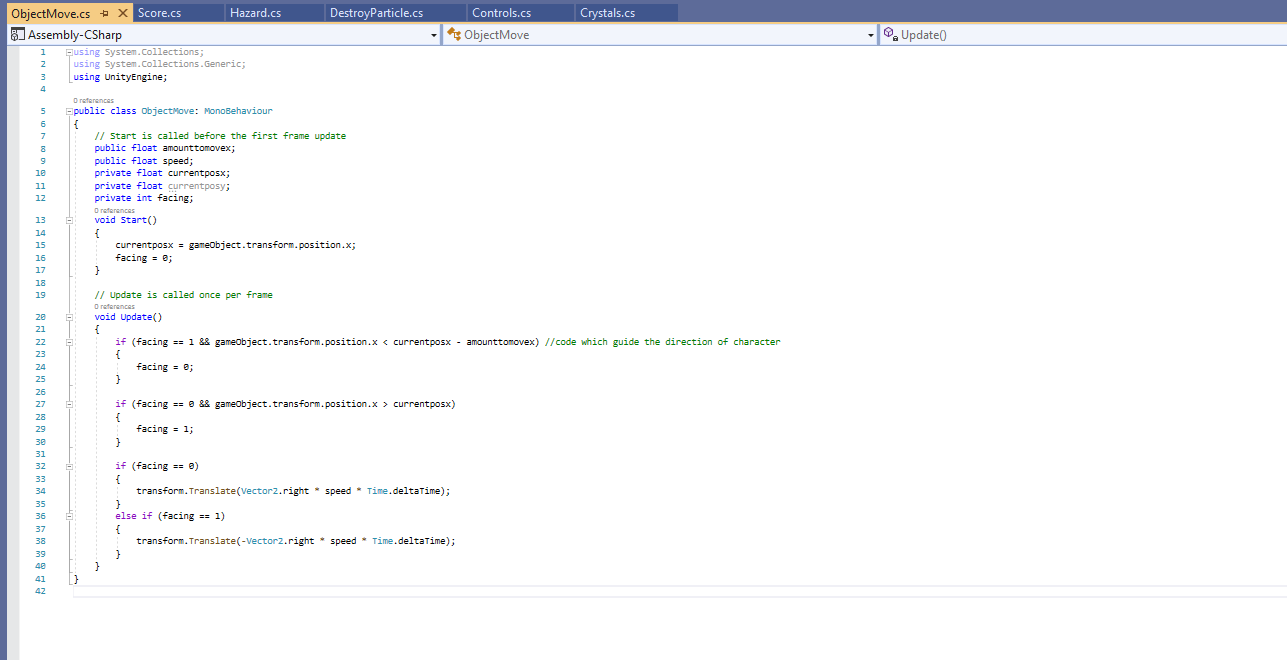
**7.6.Touch**



Aici se regasesc apelurile functiilor definite in scriptul controls pentru utilizarea controalelor ecranului tactil.



**7.8.Object Move**

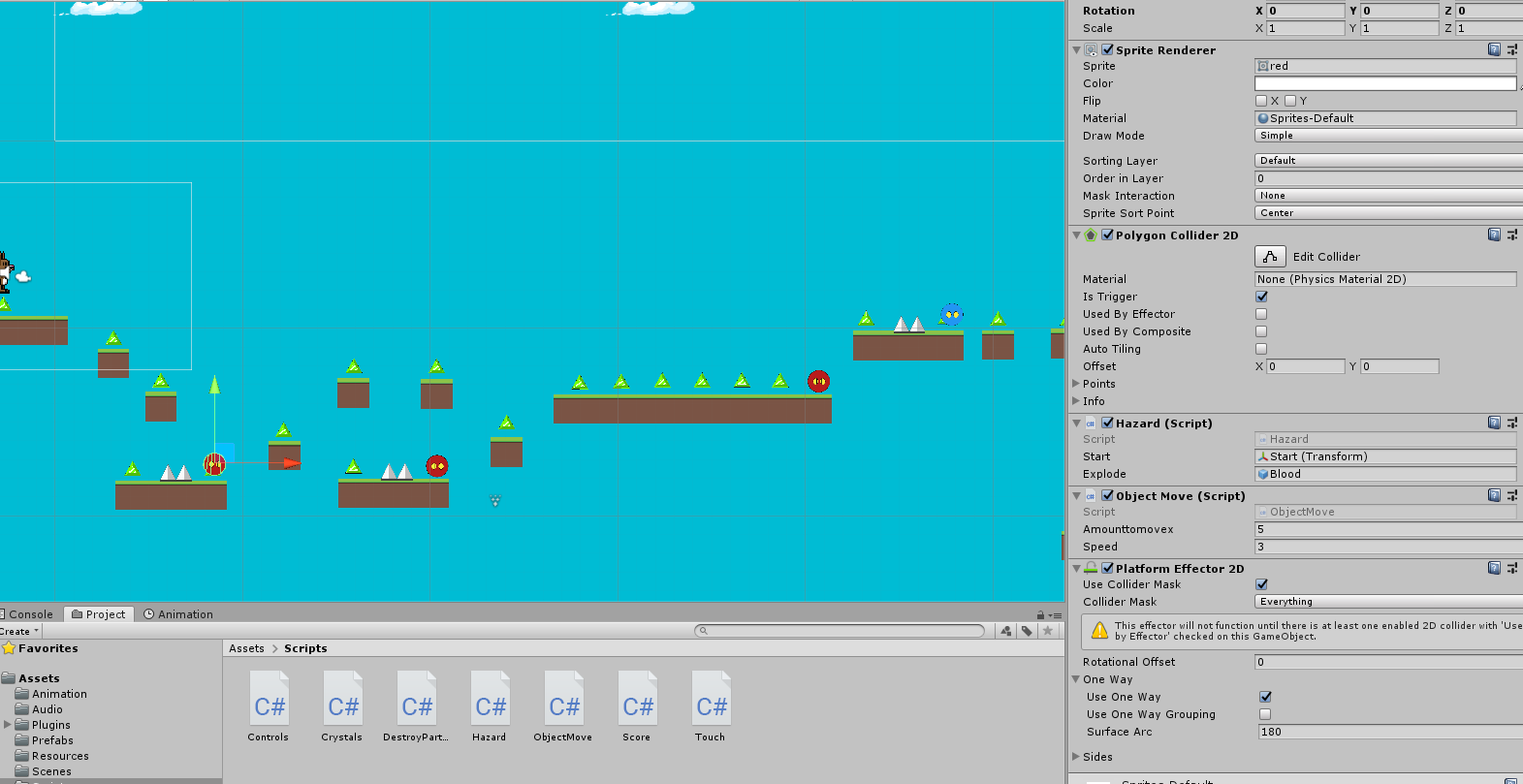
****

Codurile raspunzatoare de miscarile inamicilor,miscari care se realizeaza fara a fii controlate de noi pe baza unor limite:

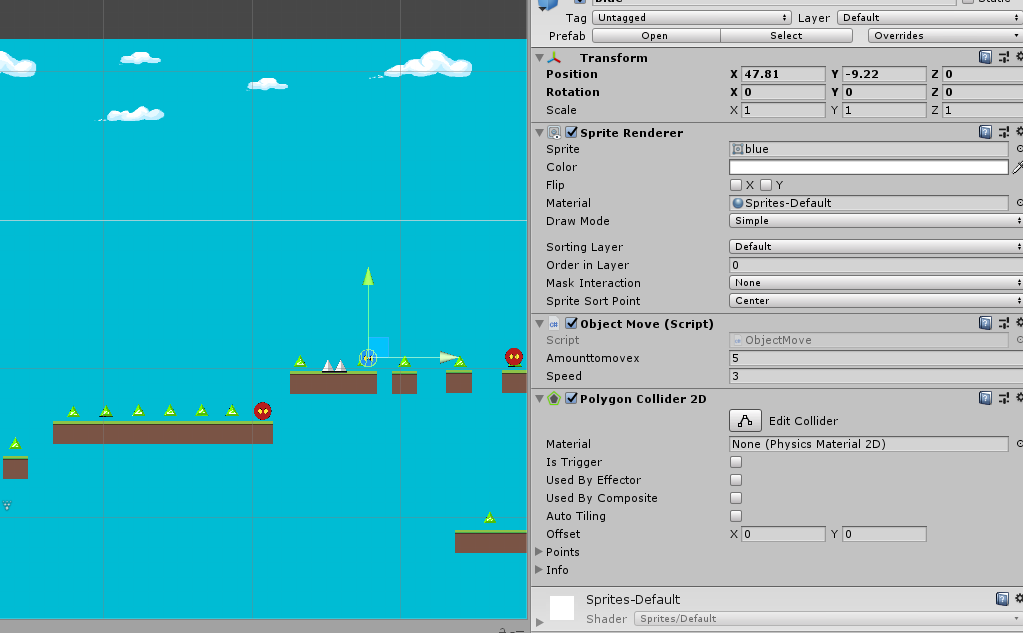
Distanta predefinita

Viteza predefinita

Miscare limitata pe axa Ox pe baza unui vector predefinit de unity.



Acestea pot sa fie alese in functie de suprafata platformei,nivel etc.



Se poate observa ca desi caracterul albastru este considerat un inamic,acesta nu dispune de scriptul de tip Hazard.

Pentru a trece nivelul jucatorul trebuie sa dea dovada de creativitate,si sa utilizeze acest avantaj pentru a colecta toate cristalele necesare trecerii la urmatorul nivel.

**8. Dezvoltare și înbunătățire ulterioară**

* Corectarea eventualelor bug-uri.
* Adaugarea unui meniu care sa prezinte :

1. Instructiunile jocului si cateva animatii din nivelele superioare pentru a motiva jucatorul sa joace.

2. O pagina de gestiune a nivelelor.

3. Butoane pentru orprire sunetului.

* Adaugarea unui sub-aplicatii de Level maker in care jucatorul sa isi poata face prorpirul nivel.
* Adaugarea de power-ups(abilitati care sa simplifice finalizarea nivelelor precum leviatie,trecere prin obiecte solide,abilitatea de a riposte impotriva inamicilor,invincibilitate etc) care sa fie deblocate la colectarea unui anumit nivel de cristale.
* Adaugarea de noi animatii precum cele da saritura si cele pentru power-ups.
* Adaugarea de noi personaje.
* Adaugarea de noi teme pentru nivele la fiecare 5-10 nivele finalizate.

**9.Utilizari**

Scopul jocului este de a distra si relaxa jucatorul dar si de a-i stimula creativitate si gandirea libera pentru a trece printer nivele.Jocul dezvolta coordonarea mana ochi,ingeniozitatea si reflexele.

**10. Bibliografie**

https://msdn.microsoft.com/library

https://images.google.com/

https://www.pixilart.com/draw#

https://stackoverflow.com

https://social.msdn.microsoft.com/Forums/en-US/home

Diverse forumuri de specialitate dar si mici comunitati online cu programatori specializati.