***Chicken Run***

***Documentatie***

Introducere

Jocul a fost realizat in Unity, in limbaj de programare C#. Astfel este bazat pe obiecte care au scripturi asociate ce realizeaza functionalitatile jocului.

Structura

Jocul e alcatuit din 3 scene. Cea principal este Main menu de unde pot fi accesate restul scenelor. Scena TutorialCR unde sunt display-uite controalele si regulile jocului si scena Chicken\_run in care se afla modulul de endless runner al jocului.

1. Main menu

In scena de main menu avem urmatoarele obiecte:

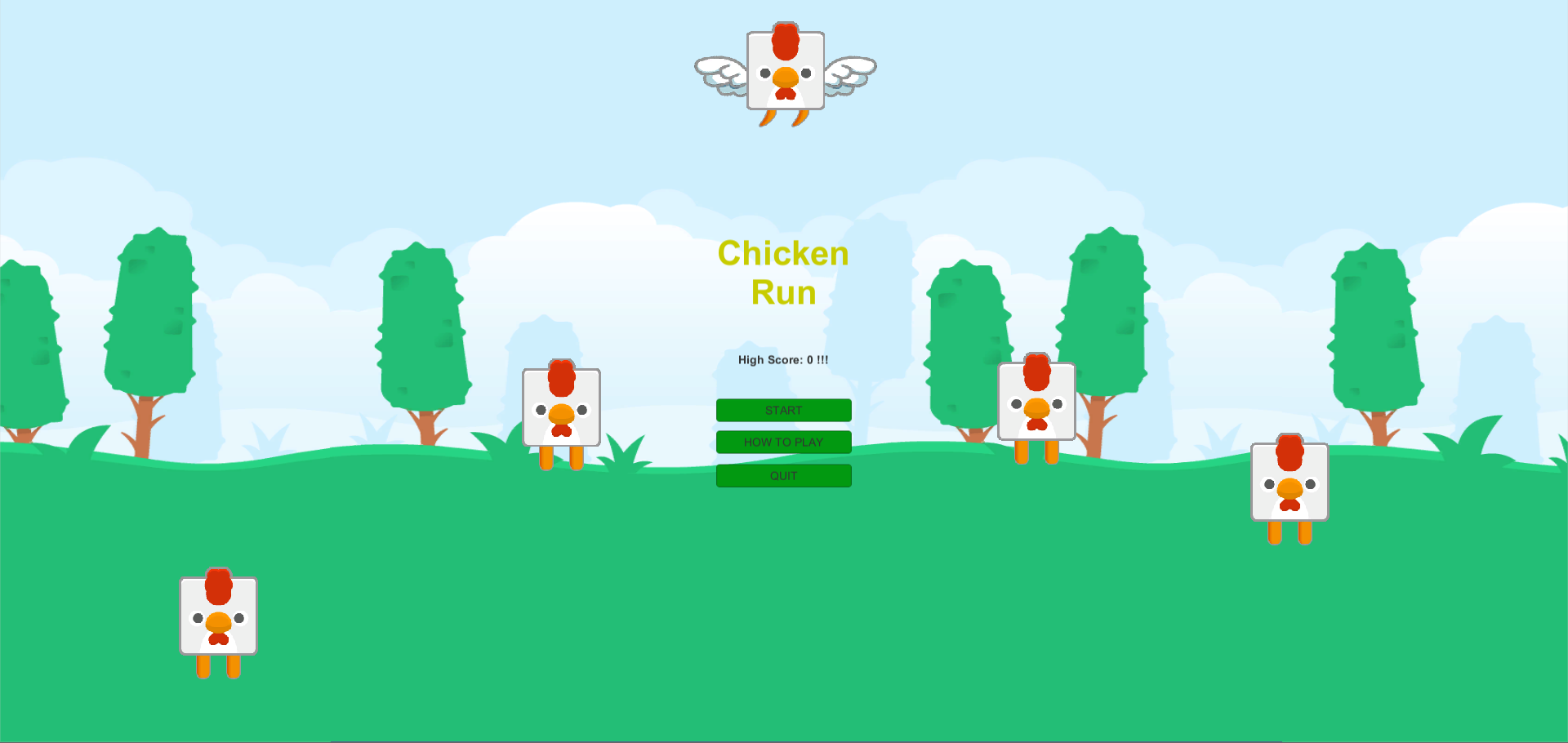
* Main Camera – obiect unity care seteaza pozitia camerei
* Canvas – obiect unity ce contine elemente de UI
* EventSystem – obiect unity autogenerat pentru a handle-ui script events
* Background – Imagine de background
* Characters – Niste modele de character pt esteticitate
* Barroom Ballet(Audio File) – Obiect pentru a reference-ui sound file-ul care autoplay-uie la inceput si loop-uie

Canvas-ul contine inauntrul lui mai multe obiecte de UI :

* Play Button – un obiect unity de tip button care odata apasat load-uie scena cu jocul propriu zis
* QuitButton – un obiect unity de tip button care odata apasat inchide aplicatia
* Title – un obiect unity de tip text pentru a scrie titlul jocului
* High score – un obiect unity de tip text pentru a afisa ultimul high score
* TutorialButton -– un obiect unity de tip button care odata apasat load-uie scena cu explicatiile key-urilor si a regulilor jocului.

Obiectele din aceasta scena se folosesc toate de MainMenu script in care avem mai multe functii:

* Start – functie care se apeleaza la inceputul run-ului unei scene; in ea verificam daca s-a salvat un highscore pana acum si daca da il load-uim si il punem pe ecran
* PlayGame – functie asociata butonului PlayButton pentru a load-ui scena de joc
* HowToPlay – functie asociata butonului TutorialButton pentru a load-ui scena de tutorial
* QuitGame – functie asociata butonului QuitButton pentru a iesi din aplicatie.



1. TutorialCR

In scena de tutorial avem urmatoarele obiecte:

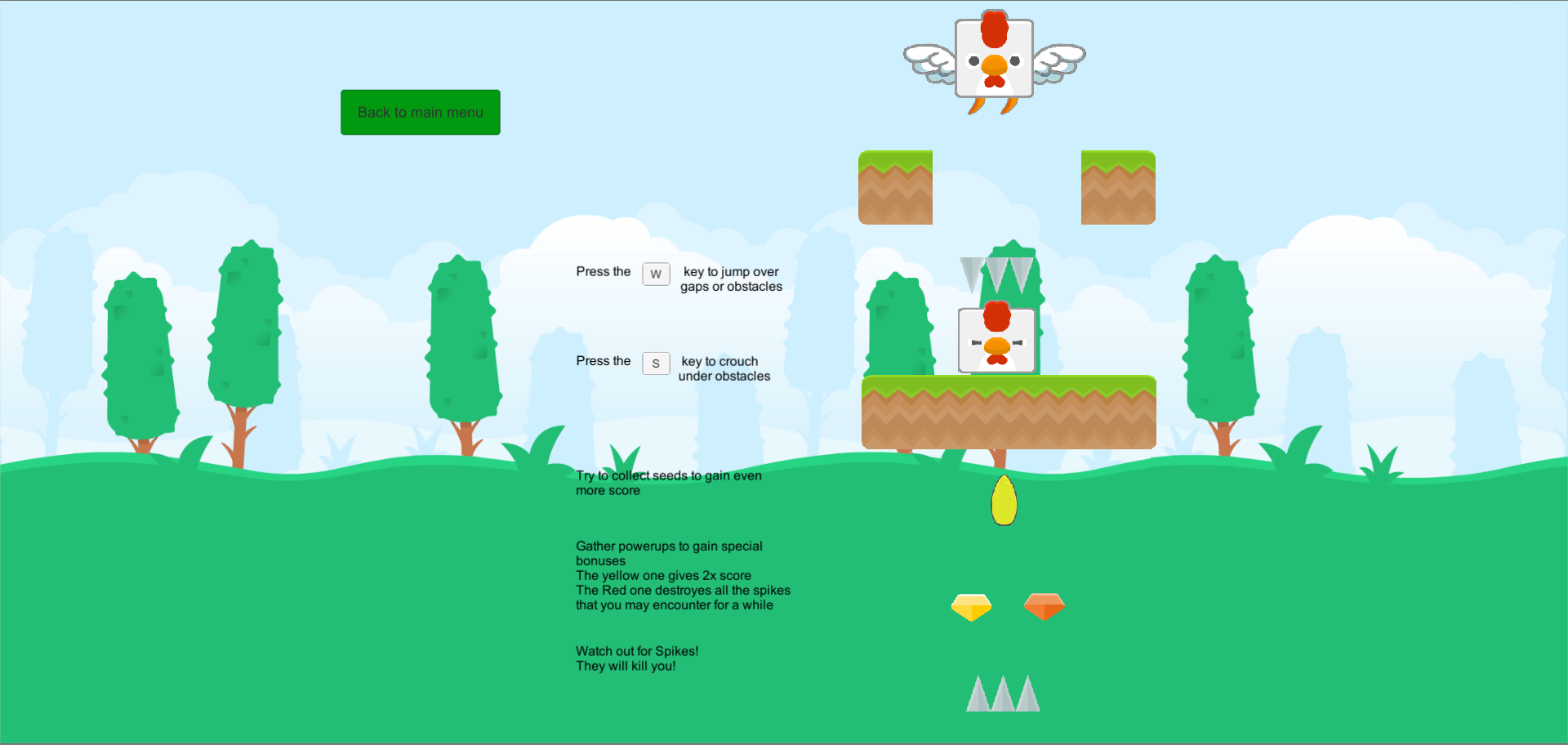
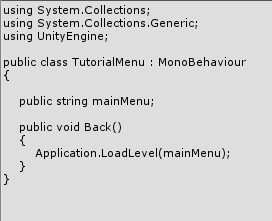
* Main Camera – obiect unity care seteaza pozitia camerei
* Canvas – obiect unity ce contine elemente de UI
* EventSystem – obiect unity autogenerat pentru a handle-ui script events
* Sprite-uri de ground – Modele pentru crearea blocurilor de pamant pe care fuge character-ul
* Background – Imagine de background
* Characters – Niste modele de character pentru a reprezenta situatii de joc
* Seed – Modelul folosit pentru semintele ce dau extra scor
* Powerups – Modele folosite pentru powerup-uri ce dau puteri speciale
* Spikes – Modelul folosit pentru tile-seturile ce omoara caracterul la atingerea acestora

Canvas-ul contine inauntrul lui mai multe obiecte de UI :

* WButton – un obiect unity de tip button care reprezinta butonul W de pe tastatura
* SButton – un obiect unity de tip button care reprezinta butonul S de pe tastatura
* Obiecte Text – mai multe obiecte unity de tip text care descriu regulile si tastele jocului
* MainMenuButton – un obiect unity de tip button care odata apasat load-uie scena de main menu.

Obiectul Canvas din aceasta scena se foloseste de script-ul TutorialMenu:

* Back – functie asociata butonului MainMenuButton pentru a facilita rentoarcera la MainMenu prin load-uirea scenei de MainMenu



1. Chicken\_run

In scena de joc endless runner avem urmatoarele obiecte:

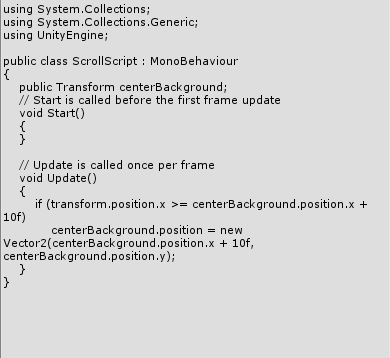
* Main Camera – obiect unity care seteaza pozitia camerei
* Player – obiect ce reprezinta avatarul jucatorului in lumea jocului
* BackgroundCenter – obiect de referinta pentru a genera background-ul
* PlatformGenerator – obiect de referinta pentru a genera noi platforme
* Platforms – obiect de stocare a diferitelor tipuri de platforme
* ObjectPools – obiect de stocare a platformelor create pentru a fi aduse sau scoase din modul activ
* GameManager – obiect ce contine scriptul de tratat incheierea si restartarea jocului
* Canvas – obiect unity ce contine elemente de UI
* EventSystem – obiect unity autogenerat pentru a handle-ui script events
* SeedGenerator – obiect ce contine scriptul de genereare a semintelor si ObjectPool-ul de seminte numit SeedPool
* SoundFX – obiect ce contine toate sunetele de care se foloseste jocul
* SpikePool – obiect de tip ObjectPool pentru generarea spike-urilor
* PowerupManager – obiect ce contine scriptu-ul de handle-uit powerup-urile
* PowerupPool – obiect de tip ObjectPool pentru generarea powerup-urilor

**Main Camera** mai contine inauntrul ei punctele de platform destruction si platform generation care sunt pozitile de care se foloseste platform generator-ul si destroyer-ul pentru a crea sau distruge platforme. De asemenea mai contine un obiect Catcher ce detecteaza daca character-ul “a cazut de pe ecran” si il omoara astfel sfarsind jocul. Main Camera se foloseste de scriptul CameraController si ScrollScript.

* CameraController – scriptul care se asigura ca player-ul este mereu in viziunea camerei.



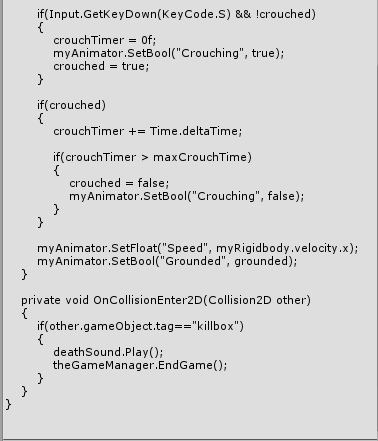
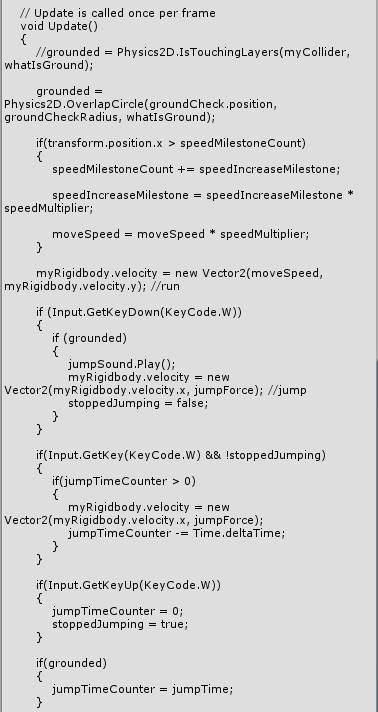
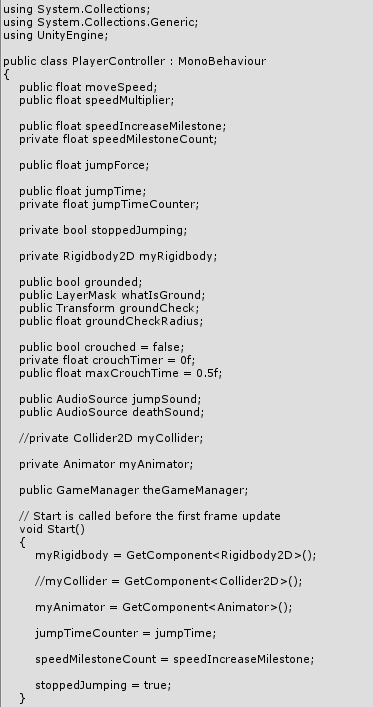
* + Start – functie ce se apeleaza la inceput si gaseste obiectul de tip player si ii da pozitia acestuia camerei pentru a putea sa o proceseze
  + Update – functie ce se apeleaza la fiecare frame al jocului si se asigura ca atunci cand player-ul se muta camera se va muta cu el; se foloseste de pozitia luata din Start()
* ScrollScript – scriptul care misca imaginile de backround impreuna cu camera pentru a da iluzia da deplasare pe un fundal infinit



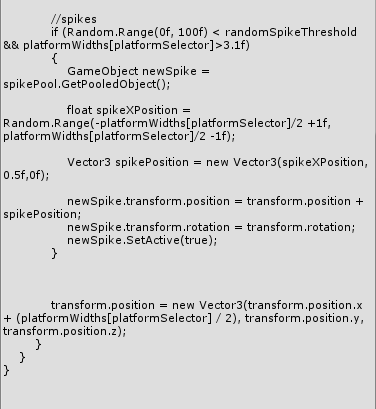
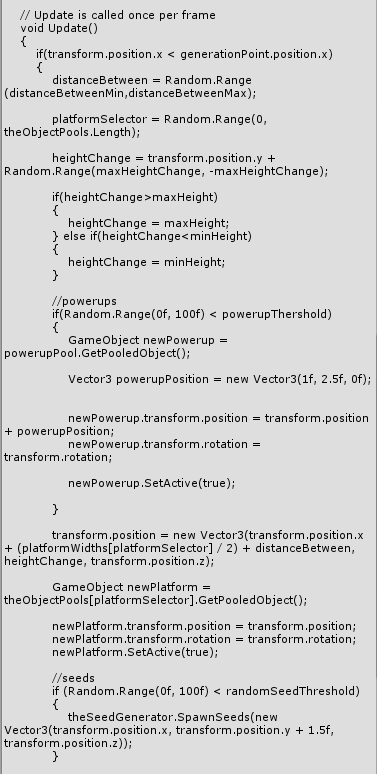
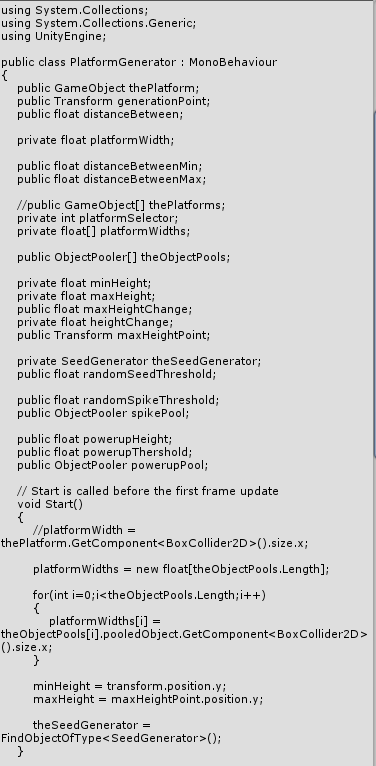
* + Update – functie ce se apeleaza la fiecare frame al jocului si verifica daca camera s-a mutat aceasta muta ultima piesa de background in fata camerei.

**Player** contine inauntrul lui un obiect GroundCheck care e folosit ca referinta pentru a verifica daca se afla pe o platforma. Player contine un element unity numit BoxCollider cu ajutorul caruia putem s ail facem sa interactioneze cu alte obiecte care au collidere. Obiectul se foloseste de scriptul PlayerController si de asemenea are un element unity Animator cu ajutorul caruia i-am animat actiunile.

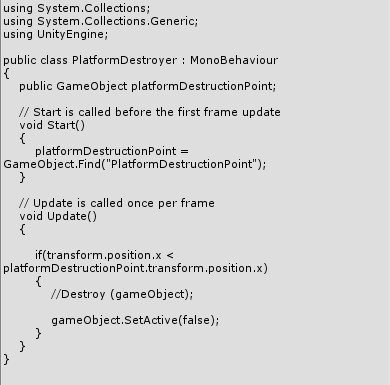
* PlayerController – scriptul ce handle-uie controalele, miscarile animatiile si moartea character-ului jucatorului.



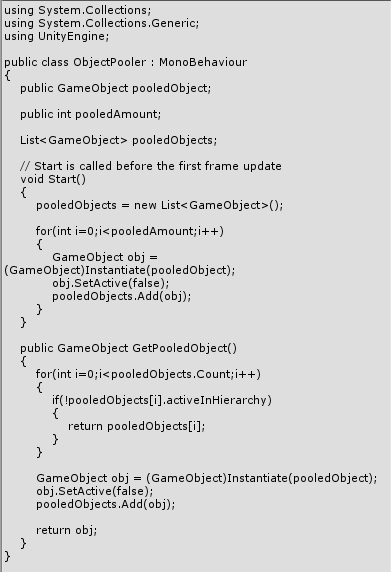
**PlatformGenerator** se foloseste de scriptul PlatformGenerator pentru a creea noi platform cu spike-uri, seed-uri sau powerup-uri. Se foloseste de tehinca object pooling in care la inceput sunt create un anumit nr de obiecte de care este nevoie iar acestea devin active si inactive cand jocul are sau nu nevoie de ele. Odata ce am ajuns la generation point acesta se va uita in pool-ul de obiecte sa vada daca are obiecte inactive pe care le poate folosi si le pune la locul lor si le activeaza,daca nu are obiecte isi creeaza.



**Platforms** store-uie diferite tipuri de platform care se folosesc de scriptul PlatformDestroyer pentru ale “sterge”. In cazul nostrum numa le setam ca fiind inactive si gata de a fi refolosite.



**ObjectPools** contine call-urile la object pooler pentru a gasi obiectele din pool.

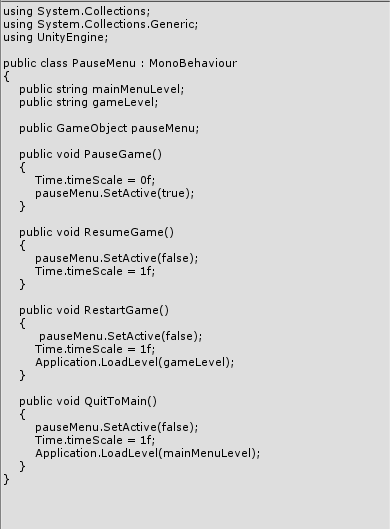


**GameManager** e obiectul de referinta pentru script-ul GameManager ce controleaza starea de end game si reseteaza anumite valori pentru restartarea acestuia.

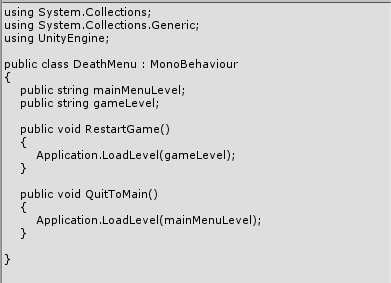


**Canvas** contine elemente de UI precum Textele de High Score si Score. Contine butonul PauseButton ce se foloseste de script-ul PauseMenu si contine si obiectul de UI PauseMenu care se foloseste de acelasi script. Contine si obiectul DeathMenu, de asmenea setat original pe inactive.

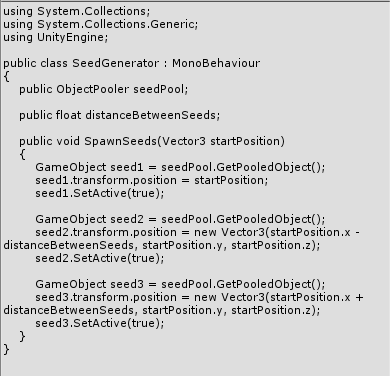
Pause menu contine o imagine complete neagra dar cu valoarea de alpha setat la 30% pentru a arata ca jocul este in modul de pausa si de asemenea contine un element text ce contine “Paused” si 3 butoane. ResumeButton ce reia continuarea jocului, RestartButton ce reseteaza scena Chicken\_run, si QuitMMButton ce load-uie scena de Main\_menu. PauseMenu este setat pe inactive pana cand apasam butonul de pauza pentru ca jocul sa nu il arate. Odata apasat butonul acesta devine active si setam timescale-ul jocului la 0, oprind-ul efectiv. Cand revenim la joc sau iesim in main menu resetam valoare de timescale si dezactivam meniul de pauza.



Death menu se foloseste de scriptul DeathMenu si contine aceasi imagine neagra cu alpha la 30% pentru a arata ca jocul este inactiv, butonul de restart al jocului RestartButton ce va load-ui din nou scena de Chicken\_run si butonul de iesire la main menu QuitMMButton.

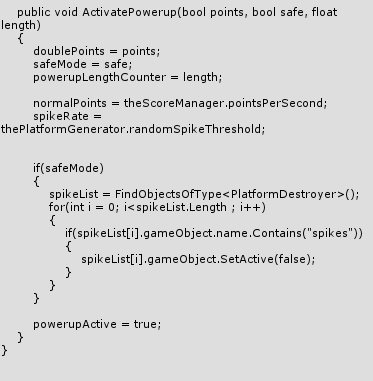
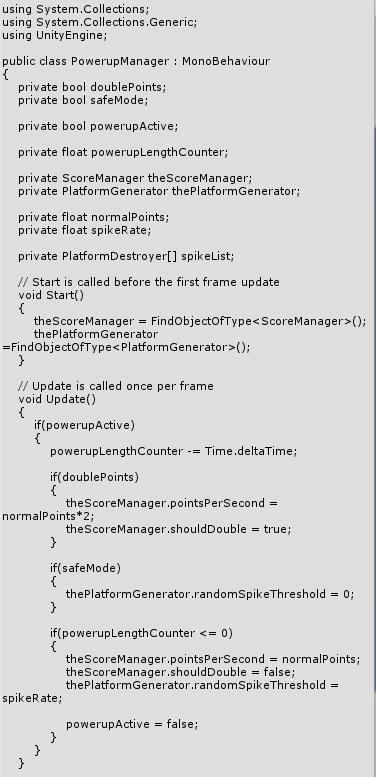


**SeedGenerator** este obiectul de referinta pentru scriptul de generare al semintelor si contine un SeedPool ce se foloseste din nou de ObjectPooler pentru a genera semintele

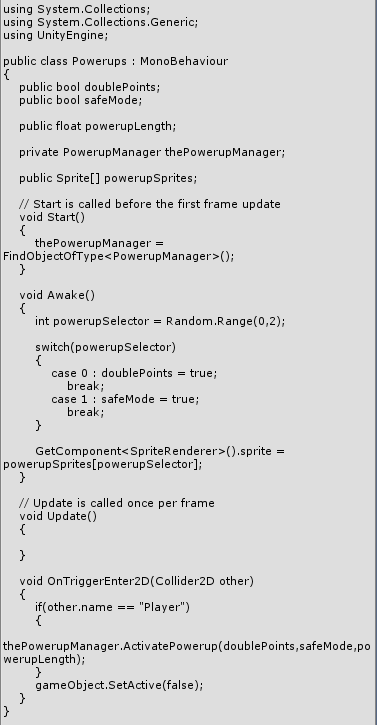


**SpikePool** foloseste ObjectPool pentru a genera spike-uri.

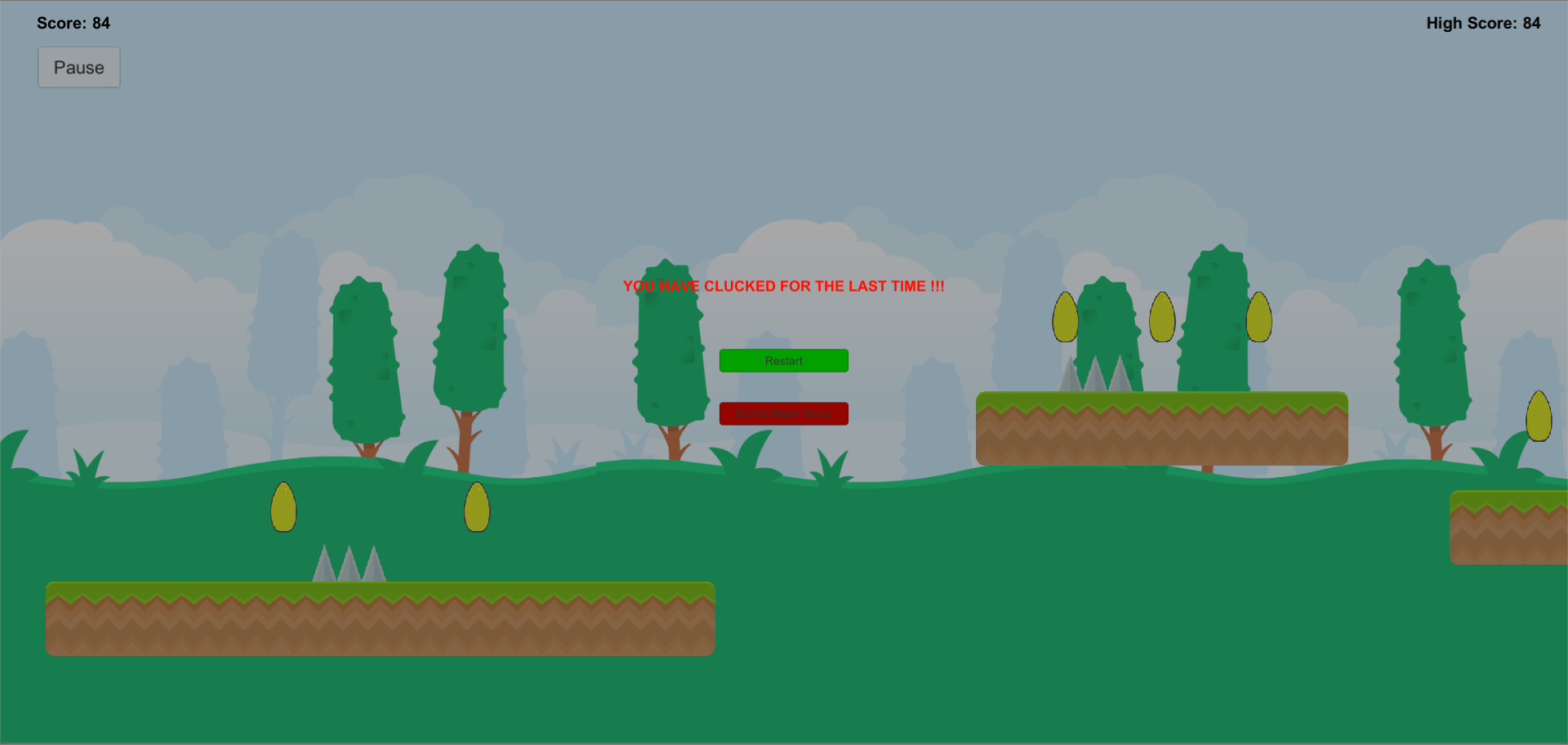
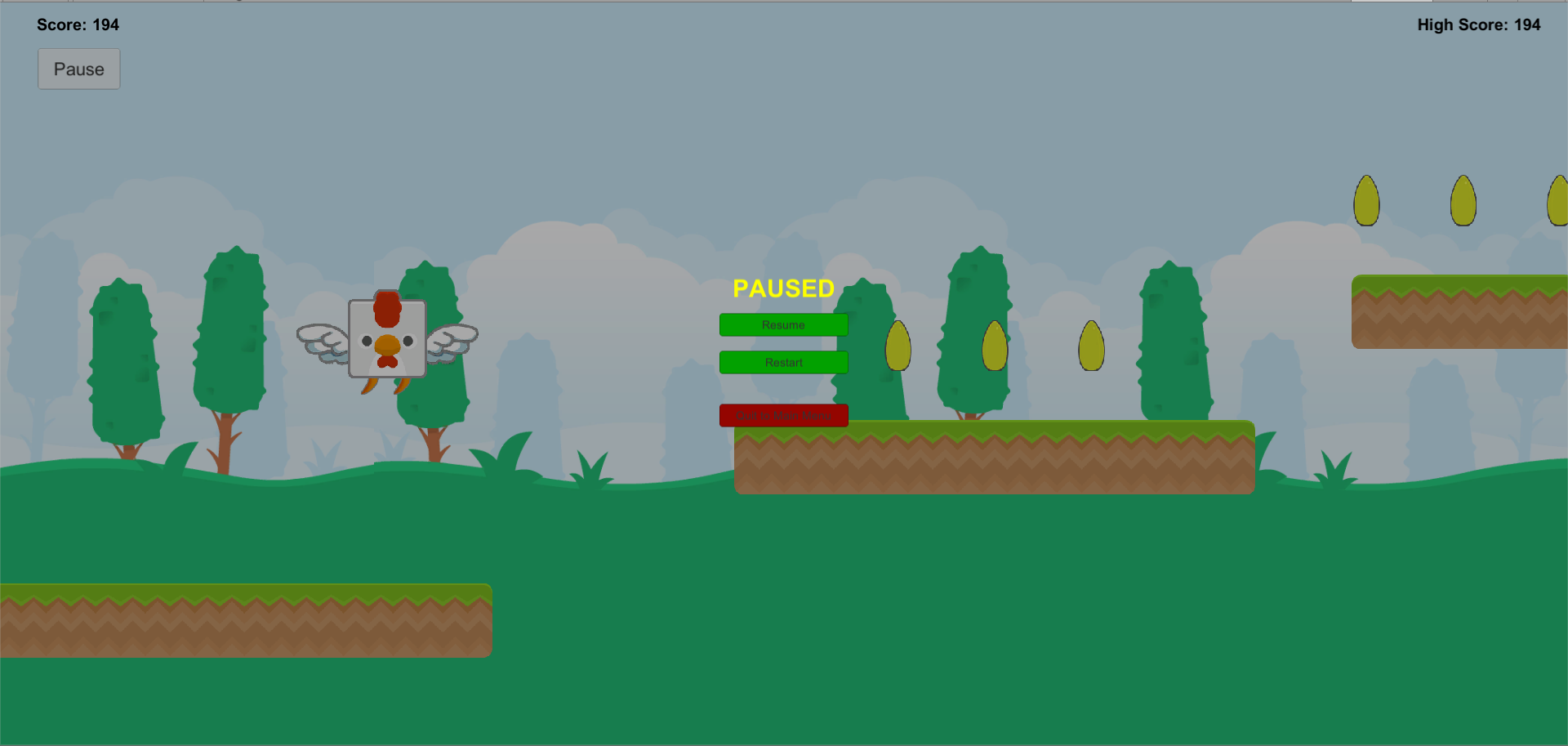
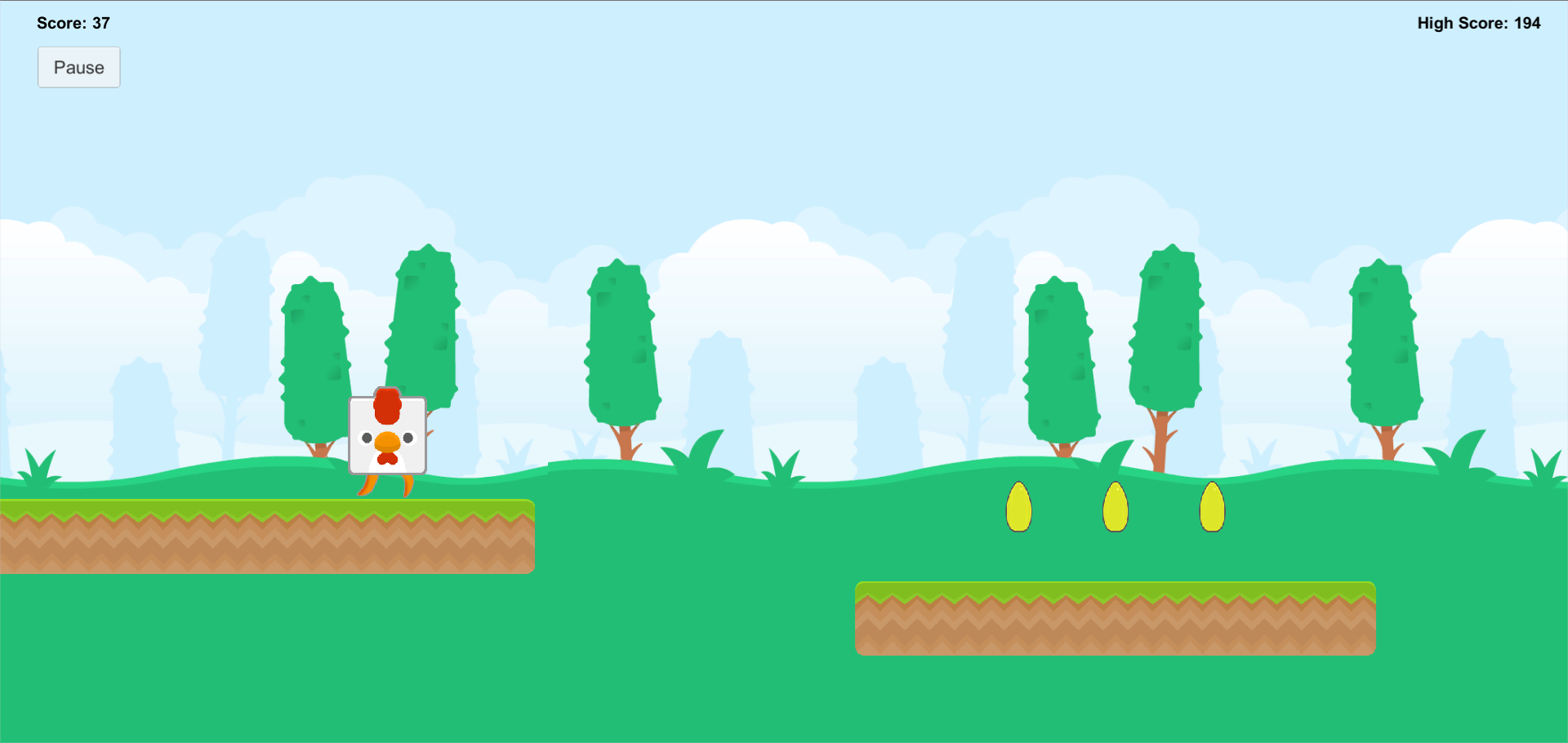
**PowerupManager** este obiectul de referinta pentru scriptul de handle ale powerup-urilor PowerupManager



De asemenea powerup-urile sunt definite in scriptul Powerups



**PowerupPool** foloseste ObjectPool pentru a genera Powerup-uri.



**Testare**

Pentru a testa functionalitatiile am folosit modul de runtime si world editor al unity-ului pentru fiecare modul adaugat.

* Player inputs – am testat apasarea tastelor specifice si am asteptat raspunsul dorit de la personajul jucatorului ( W - jump , S- crouch, buton pause- pauza)
* Generare si distrugere platforme - am testat generarea si distrugerea platformelor din pool-ul de obiecte in editorul unity. Prin setarea punctului de generare si distrugere mai aproape sau mai departe de camera am testat faptul ca nu este depasita aceasta limita
* Testare Scor – Am run-uit jocul pentru a verifica faptul ca scorul creste continuu si linear si am verificat si pick-up-ul de seed setand gametime-ul din unity mai jos pentru a observa modificarile.
* Testare moarte si restart – Am testat repetitiv lasand characterul sap ice/ sa loveasca spike-uri ca acesta este setat pe inactive iar la restartare readus la pozitia de start si setat pe active.
* Am testat power-up-urile crescand rata de aparitie a acestora si schimband valorile de timp active sau scorul bonus generat.