Liceul Teoretic „Emil Racoviță” Vaslui

PACMAN

Lucrare de atestat profesional la informatică

Candidat Îndrumător

Pintilie Bogan-Ioan prof. Dan Pracsiu

2018

Introducere

În cadrul proiectului de atestat la disciplina informatică, am optat pentru adaptarea jocului care a cucerit mapamondul, PacMan, la o idee originală. Clonarea jocului reprezenta din punctul meu de vedere un lucru lipsit de orice interes, și pe deasupra inutil, de aici și dorința de a crea un joc demo în care să-mi pot transpune ideea dezvoltată pe marginea unui joc clasic, îndrăgit de toți iubitorii jocurilor arcade încă din anii ”80. Ideea transpusă proiectului constă în inversarea rolului pe care îl îndeplinea PacMan-ul în jocul original, adică, PacMan-ul nu mai este vânat, ci vânează fantome continuu, astfel jocul căpătând o nouă valență, amintind de un alt joc cunoscut de aproape întreaga populație a lumii, și anume jocul Snake. În fond, acest proiect capătă o formă hibridă, fiind o combinație nemaiîntâlnită între două jocuri cu rezonanțe puternice în sfera jocurilor arcade, o îmbinare a jocului PacMan cu jocul Snake.

Totuși, chiar dacă ideea de atestat pleacă de la jocul PacMan, acest proiect de atestat și jocul arhicunoscut au un număr restrâns de elemente în comun. Joculețul care constituie proiectul beneficiează de același „maze”(labirint). În centru se află tot PacMan-ul, dar de data aceasta el este nevoit să „alerge” fantomele pentru un scor cât mai mare. Numărul de fantome este tot egal cu patru, iar lucrul care diferă este poziția de început a celor patru fantome, dar și faptul că pleacă în același timp cu PacMan-ul. Un alt lucru care îndepărtează acest proiect de jocul de la care s-a pornit este faptul că în componența sa nu există puncte bonus, nici punctele care trebuiau „mâncate” de PacMan pentru a câștiga. Faptul că fantomele vor fi vânate pe tot parcursul jocului face ca prezența acelor puncte care permit PacMan-ului să mănânce fantome să fie inutilă. Un alt lucru interesant despre acest joculeț este faptul că fantomele sunt „oarbe”. Ele nu pot decide în mod corect în ce direcție trebuie să meargă pentru a nu fi mâncate de PacMan. Acest lucru se realizează prin alegerea aleatorie a traseului pe care îl urmează o fantomă. Când una dintre cele patru fantome ajunge la o intersecție, ea alege la întâmplare direcția următoare. Pentru a crea acest efect, este eliminată animația fantomelor care făcea ca ochii acestora să fie orientați în funcție de cum mergea fantoma. Toate fantomele din această aplicație au ochii ficși.

Jucătorul care controlează PacMan-ul prin intermediul celor patru săgeți directoare(sus, jos, stânga, dreapta) are la dispoziție 90 de secunde pentru a prinde cât mai multe fantome. Odată ce o fantomă a fost mâncată, ea dispare din „peisaj” timp de 5 secunde, timp în care și scorul este în creștere. Doar cât timp cel puțin o fantomă nu este în labirint, adică a fost mâncată și nu au trecut cele cinci secunde în care aceasta nu este prezentă, scorul crește. Ceea ce trebuie menționat este faptul că, în ciuda faptului că ideea care a dus la crearea acestui joculeț s-a născut și din experiența jocului Snake, PacMan-ul care este în ipostaza „vânătorului” nu crește odată cu capturarea unei fantome. Jocul se termină când cele 90 de secunde s-au scurs.

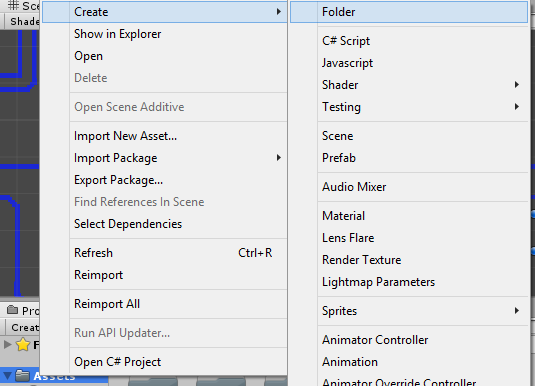
Aplicția transpune și „drama trăită” de PacMan în raport cu existența fantomelor. Povestea jocului poate fi interpretată și astfel. PacMan-ul, terorizat de cele patru fantome, care l-au tot vânat, poate în sfârșit să doboare cele patru fantome fără a se teme că are un timp limitat, după care fantomele se vor întoarce către el. Totuși, fericirea acestuia nu este una eternă deoarece fantomele continuă să apară din nou, oricât de mult le-ar mânca. În concluzie, PacMan-ul, prin toate lui încercările de a elimina fantomele, constată că îi este imposibil să le elimine definitiv, trăind acut, captiv în universul jocului, și fiind condamnat la teroarea fantomelor.

În finalul intoducerii documentației atestatului, țin să menționez că prin acest proiect am urmărit dezvoltarea abilității de „game development” prin crearea unui joc simplu, plecând de la un joc cunoscut, prezentând note de originalitate, jocul fiind dezvoltat în Unity și CSharp, în Unity realizându-se tot ce ține de partea grafică, iar prin intermediul limbajului de programare CSharp, se realizează tot ce ține de partea teoretică a jocului, adică „creierul” aplicației.

Conținutul lucrării

1. Setările inițiale

În ceea ce privește crearea jocului cu aplicația Unity este nevoie în primul rând de crearea unor altor foldere în cel în care se află jocul pentru o mai bună organizare a conținutului. Astfel, sunt create foldere în „Assets”, și anume: folderul „Graphics”, în care punem resursele grafice ale jocului, „Scenes”, folder în care sunt situate „scenele” jocului, și folderul „Scripts” în care sunt situate „scenariile” CSharp.



2.Crearea labirintului

Fig. 1

[Cite your source here.]

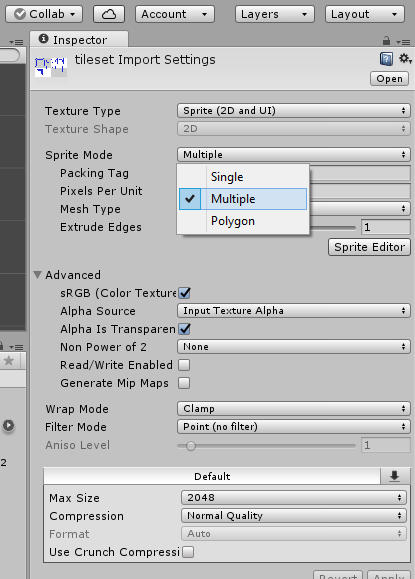
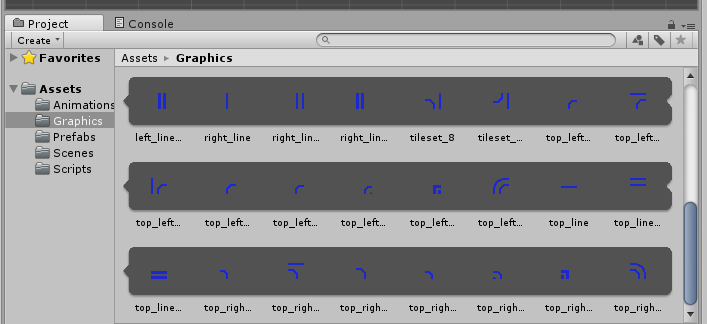
 Pentru a crea labirintul, avem nevoie de o resursă grafică, numită „tileset” care conține toate liniile de care avem nevoie, resursă preluată de pe siteul [1] . Odată ce se descarcă această resursă este adăugată la folderul „Graphics” din Unity. După ce acesta este adăugat la folderul „Graphics”, selectăm din „Inspector”, SpriteMode Multiple, ca să se separe fiecare bucată și sunt luate fiecare și așezate în scenă pentru a modela labirintul, lucru întreprins urmărind tutorialul [2].

Fig. 2

[Cite your source here.]

Fig. 3

[Cite your source here.]

3.Mișcarea PacMan-ului

Pentru mișcarea PacMan-ului nu vom folosi „collusion detection”, o funcție predefinită din Unity, ci vom folosi noduri situate la fiecare „intersecție”, idee observată în tutorialul [3]. Pentru aceste noduri, vom determina vecinii, și direcțiille valide, adică partea în care se poate deplasa PacMan-ul. Pentru aceste noduri avem nevoie de un „C# script” pe care îl numim „Node.cs”(figura 4).

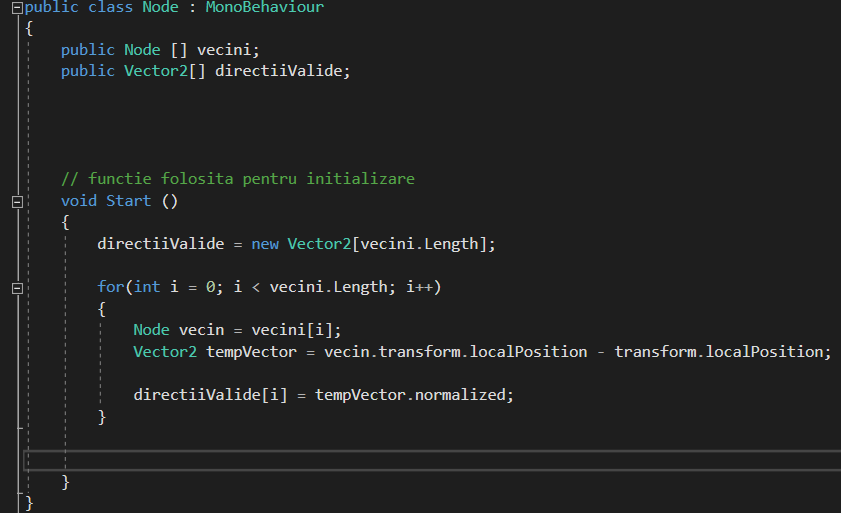
Nodul este reprezentat de o parte grafică, numită pellet. Se importează această resursă și se pune câte una la fiecare intersecție. La toate aceste noduri se adaugă componenta „Node.cs” din Unity prin selectarea tuturor nodurilor, după care click pe „Add Component” de la „Inspector”. În continuare, vom asocia fiecărui nod numărul de noduri vecine, dar și care sunt aceștia, ca în figura 5.

Fig. 4

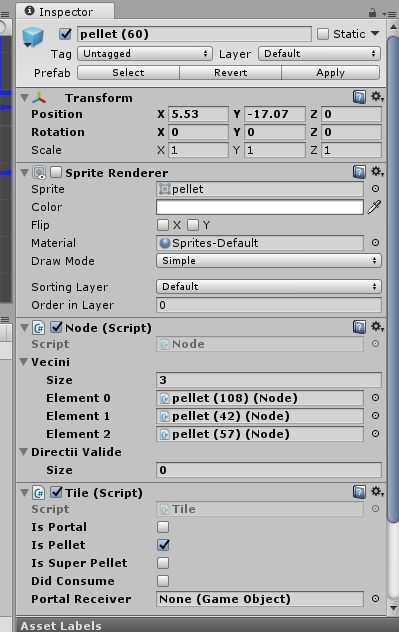
 Vom crea clasa „PacMan.cs” dedicată PacMan-ului. În interiourul clasei vom defini funcțiile care vor ajuta la mișcarea acestuia, lucru realizat prin vizionarea [4], dar și alte funcții care vor contribui la crearea jocului. În ceea ce privește mișcarea PacMan-ului, vom folosi patru funcții. Prima funcție, numită CheckInput(), verifică ce tastă este apăsată. Dacă se apasă una din cele patru săgeți, se apelează funcția ChangePosition(Vector2 d), figura 6, în care se verifică dacă PacMan-ul își poate schimba poziția spre direcția transmisă de variabila d. De exmplu, dacă se apasă tasta care reprezintă săgeata indicată către stânga se apelează funcția ChangePostion(Vector2.left) ca în figura 7. O altă funcție care contribuie la mișcarea PacMan-ului este UpdateOrientation() care actualizează orientarea acestuia în funcție de direcție(sus, jos, stânga, dreapta).

Fig. 5

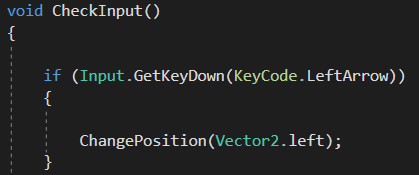
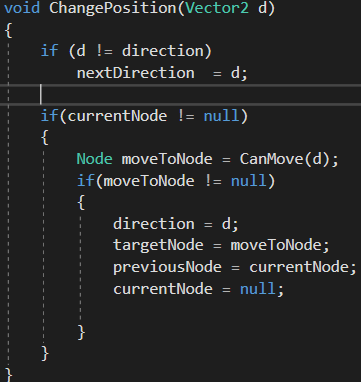
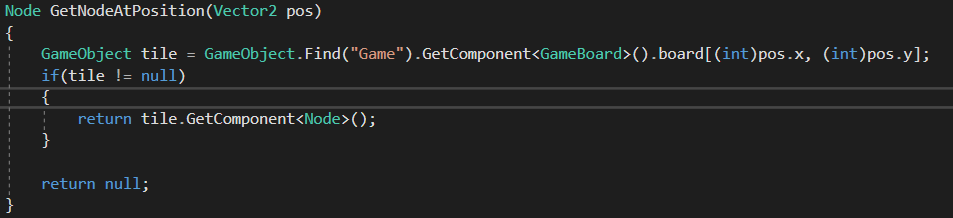
 Funcția Start() din „PacMan.cs” este folosită în scopul inițializării de la începutul jocului. Variabila „direction” care este de tip Vector2 este inițializată cu Vector2.zero. Funcția de tip Node, GetNodeAtPositon(Vector2 pos), găsită în [4], returnează nodul de la poziția transmisă prin variabila „pos”.(fig. 8)

Fig. 6

Fig. 8



Tot în cadrul acestei funcții, Start(), reprezentată în figura 9, verificăm dacă nodul căutat la poziția locală este diferit de null, inițializăm variabila „currentNode” care reține la fiecare pas al jocului nodul current la care se raportează PacMan-ul, și variabila „initialNode” care se inițializează doar în cadrul acestei funcții, cu variabila „node” căruia i se atribuie rezultatul funcției, GetNodeAtPosition(transform.localPosition). În finalul funcției Start() actualizăm valoarea variabilei „direction” cu Vector2.left, variabila „orientation” tot cu Vector2.left și apelăm funcția ChangePosition(direction).

Fig. 7

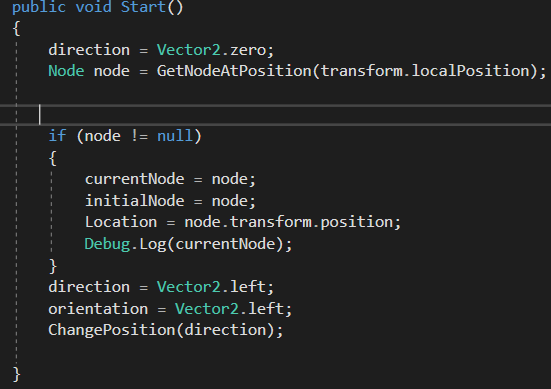


Fig. 9

4.Animația PacMan-ului și controlarea acesteia

În primul rând, vom crea un folder în „Assets„ cu numele de Animations. Pentru animația PacMan-ului avem nevoie de câteva elemente grafice, și anume PacMan-ul cu gura închisă complet, cu gura întredeschisă și cu gura deschisă complet, elemente descărcate de pe [1]. În continuare vom deschide fereastra „Animation”. Acolo vom crea o animație nouă, lungă de 15 secunde. La poziția de zero secunde vom așeza PacMan-ul cu gura complet închisă. La poziția de 5 secunde se va pune elementul grafic în care PacMan-ul este cu doar jumătate de gură deschisă, iar la poziția unde animația se află la 10 secunde se va afla PacMan-ul cu gura larg deschisă. Pentru a nu oferi o animație în care PacMan-ul își închide brusc gura, se va așeza la poziția de 15 secunde PacMan-ul cu gura pe jumătate deschisă, procedura fiind reprezentată în figura 10. Apoi, după ce animația a fost creată, aceasta este atribuită obiectului de joc, PacMan.

În funcție de starea PacMan-ului, animația creată pentru acesta se oprește sau pornește. Animația este oprită în doar două cazuri. Primul caz este reprezentat de momentul în care PacMan-ul ajunge într-o intersecție în care continuarea pe aceeași direcție este imposibilă și nu este schimbată direcția, PacMan-ul implicit oprindu-se așteptând următoarea comandă validă.

Al doilea caz în care animația este oprită este atunci când cele 90 de secunde s-au scurs, adică jocul a luat sfârșit. Pornirea animației este făcută implicit, adică ea nu trebuie pornită la începutul jocului. Trebuie menționat tot aici că, spre deosebire de jocul original, cele patru fantome nu vor fi „animate”. Lucrul acesta i se datorează faptului că ele vor reprezenta într-un fel sau altul o fantomă oarbă care nu poate decide eficient și rațional drumul pe care va merge, ceea ce va face imposibilă fuga lor de PacMan.

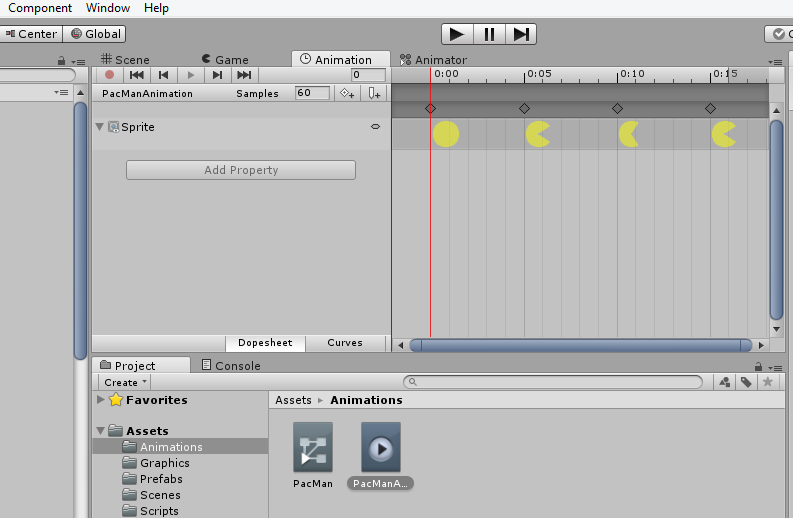


Fig. 10

Comanda folosită pentru oprirea animației este ilustrată în figura 11, iar cea în care se repornește este asemănătoare celeilalte și este exemplificată în figura 12, comenzi preluate de pe [5].

Fig. 11



Fig. 12



Un alt caz, nemenționat mai sus, în care animația este oprită este cel în care se pune pauză jocului. Atunci, prin comanda din figura 11 este oprită animația pentru a creea efectul de „freezing” al aplicației care este în modul de pauză. Iar prin ieșirea din modul pauză, se activează din nou animația prin intermediul comandei echivalente cu cea din figura 12.

5. Introducerea fantomelor în joc

Vom importa cele patru fantome în folderul Graphics, după ce au fost descărcate de pe [6], din proiectul Unity și le vom așeza în scenă diferit față de jocul inițial, în care toate pleacă din așa-numita locație „Ghost House”. Doar una dintre cele patru fantome va pleca din „casa fantomelor”, fantoma cea roz, numită Pinky. Fantoma roșie, cunoscută sub numele de Blinky va pleca de la poziția exprimată pe axele OX și OY, 15 și 28. Fantoma albastră, poreclită Inky, pleacă de la poziția 1 pe axa OX și 28 pe axa OY. Cea de-a patra fantomă, cea portocalie, având și ea o poreclă, și anume Clyde, pornește de la pozițiile 9 și 22, pe axa OX, respectiv OY. Locul de început al tuturor celor patru fantome este prezentat în figura 13.

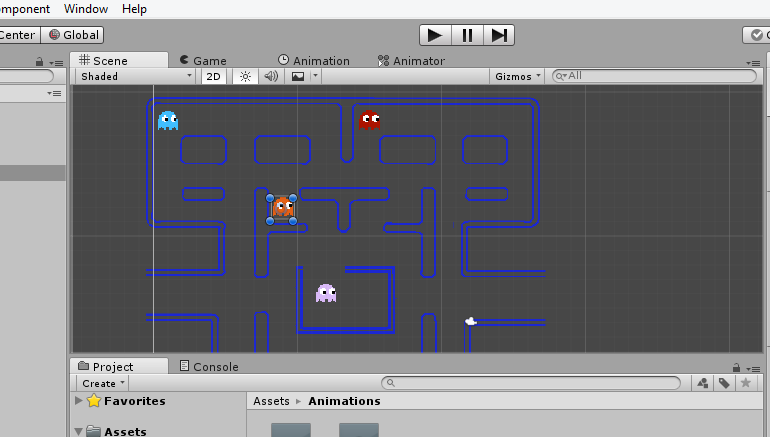


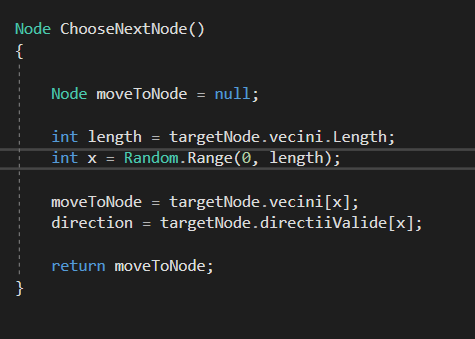
Fig. 13

Spre deosebire de jocul de la care s-a pornit, cele patru fantome, nici măcar una dintre ele nu mai trebuie să aștepte un interval de timp până când să plece. Ele încep să se miște odată cu PacMan-ul căci dacă ar fi fost ca PacMan-ul să plece înaintea acestor fantome, jucătorul s-ar fi dus direct către fantome, capturându-le cel puțin o dată fără a depune foarte mult efort, acesta fiind factorul care a dus la pornirea simultană a celor patru fantome cu PacMan-ul. Astfel, prin lipsa timpului în care fantomele sunt inactive, vânătoarea începe direct.

6. Mișcarea fantomelor

Pentru a putea ca aceste fantome să se deplaseze corect prin labirint, avem nevoie de un nou „scenariu” CSharp care se creeaza în folderul „Scripts” situat în proiect. Acesta va avea numele „Ghost.cs”. Se selectează apoi cele patru fantome și li se adaugă componenta „Ghost.cs”.

Funcțiile care determină mișcarea fantomelor sunt la fel ca cele ale PacMan-ului. Pe același principiu se mișcă și fantomele, folosindu-se de nodurile situate la fiecare intersecție din labirint. Dar, spre deosebire de mișcarea PacMan-ului care este făcută de player prin apăsarea uneia dintre cele patru săgeți directoare, fantomele se vor mișca singure, automat, fiind într-o formă sau altul autonome. Diferența apare la alegerea următorului nod. Dacă pentru PacMan, următorul nod la care se deplasează este determinat de jucător, pentru fantome alegerea următorului nod este determinată de funcția ChooseNextNode() care este de tip Node, prezentată în figura 14. Funcția aceasta caută printre vecinii nodului țintă, adică nodul spre care se deplasează fantoma. Funcția alege la întamplare unul dintre vecinii nodului țintă. Apoi, în funcția Move() nodul țintă devine nodul ales de funcția ChoosenextNode().



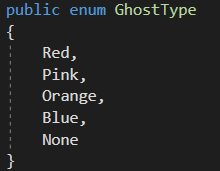


Fig. 15

Fig. 14

Pentru pasul următor, trebuie aplicat fiecărei fantome o anumită etichetă, de exemplu, pentru fantoma roșie va fi „ghost\_red” sau ceva asemănător. Vom crea apoi o enumerație în „Ghost.cs” având numele „GhostType”, surprinsă și în [7], în care vom scrie fiecare culoare a celor patru fantome. Acest lucru va facilita operațiile prin care fantomele vor porni, dispărea sau când sunt mâncate ori când o să apară din nou.

7. Capturarea fantomelor

Odată ce PacMan-ul a ajuns langă fantomă, trebuie ca ea să dispară pentru a putea spune că a fost mâncată. Pentru acest lucru, avem nevoie de o nouă funcție pentru fiecare fantomă, intitulată CatchBlueGhost() pentru fantoma albastră de exemplu. În această funcție se va verifica în primul rând dacă PacMan-ul este lângă fantoma respectivă. Mai întai se caută cele două obiecte care vor fi supuse analizei, PacMan-ul și fantoma la care se referă funcția(figura 16) [8]. După ce cele două obiecte au fost găsite, se obține poziția lor prin apelarea unei proprietăți a clasei „Vector2”, „transform.Position”(figura 17), toate aceste comenzi fiind preluate de pe [9].

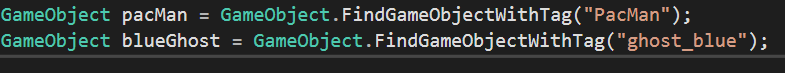


Fig. 16

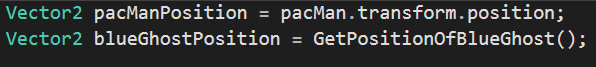
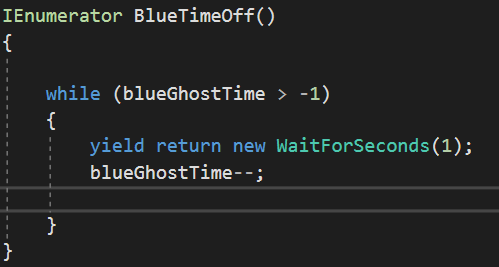


Fig. 17

După ce s-au stabilit pozițiile celor două obiecte se compară diferența dintre cele doua poziții pe axele OX și OY. Se convertește la tipul întreg al celor patru coordonate, două coordonate reprezentând pozițiile celor două obiecte pe OX, și două pe axa OY. Dacă diferența celor două coordonate care reprezintă pozițiile celor două obiecte pe axa OX este egală cu unu, iar cele două de pe axa OY sunt egale atunci: fantoma respectivă dispare din „peisaj” prin comanda din figura 18, variabila „blueGhostTime” devine 5 și reprezintă timpul în care fantoma este inactivă, această variabilă decrementându-se prin apelarea unui „IEnumerator”(prezentat pe [10]) ca cel din figura 19, care face ca variabila să scadă cu unu la fiecare secundă. Analog se face și atunci când cele două coordonate de pe OY au diferența egală cu unu și cele de pe OX sunt egale.

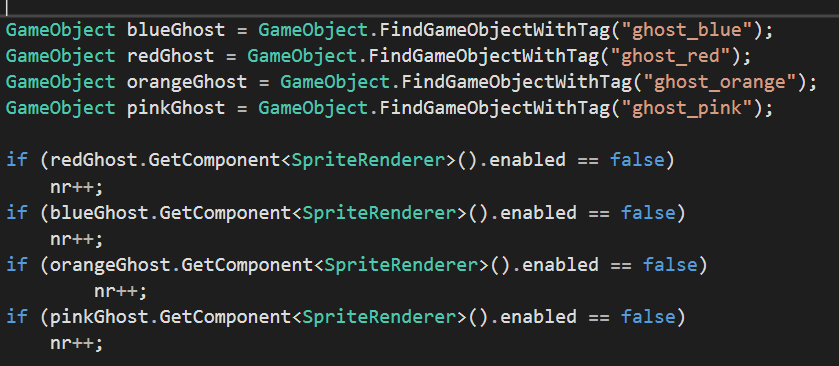
Fig. 18



Pe același principiu se realizează și celelalte trei funcții care vor determina dispariția fantomelor. Astfel, prin aceste funcții se va determina corect dispariția fantomelor.

Fig. 19

8.Scor.

 Scorul va crește doar când cel puțin o fantomă este absentă. Scorul este actualizat în funcția Update() care se apelează per cadru, aidcă, dacă sunt 30 de cadre per secundă, funcția se apelează de 30 de ori într-o secundă.

Secvența care determină scorul, „nr” fiind variabila care reține scorul

Când jocul s-a sfârșit, adică după ca s-au scurs cele 90 de secunde, în funcție de scorul obținut se va afișa un mesaj sub acesta. Dacă scorul este mai mare de 7500, atunci se va afișa mesajul: „Extraordinar! Scorul tău este imens!”, iar dacă se afla între 4000 și 7500, se va afișa: „Scorul tău denotă adevărata valoare a ta!”. Se afișează mesajul „Foarte bine, ești pe cale să devii maestru la jocul ăsta!” atunci când scorul obținut este între 2000 și 4000. Dacă scorul se situează undeva între 1000 și 2000, mesajul care se va afișa este următorul: „Ups! Poate data viitoare...”. În ultimul caz, când scorul final nu depășește valoarea 1000, se afișează mesajul: „Nu te descuraja! Încearcă din nou!”. În timpul jocului nu se va afișa niciun mesaj din cele menționate, doar scorul.

9. Momentul de evadare a fantomelor

Fantomele ajung într-un punct în care PacMan-ului îi este imposibil să ajungă. Prin aceasta, se creează o „portiță” prin care fantomele evadează. De fapt, este o falsă evadare deoarece acele fantome care ajung în punctul respectiv sunt trimise înapoi în labirint. Acest punct se află la portalul labirintului situat în partea centrală(figura 20).

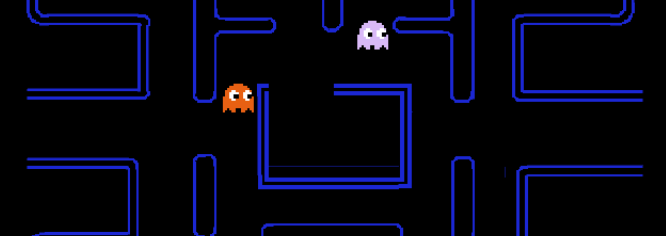
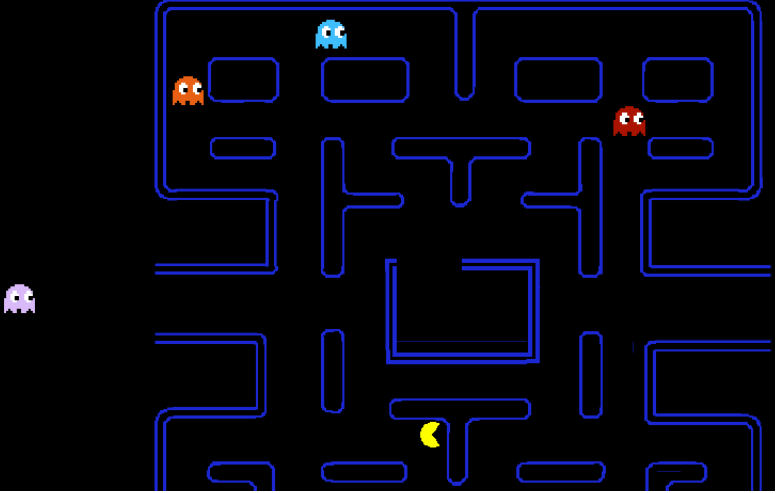


Fig. 20

Fantoma care ajunge la una dintre cele două porți ale portalului are două opțiuni. Fie se întoarce înapoi în labirint, fie încearcă să evadeze prin continuarea drumului. De menționat că ea nu este „conștientă” de alegere, ea doar alegând la întamplare una dintre cele două opțiuni.

Ea poate încerca la nesfârșit să evadeze, dar în zadar deoarece va fi redirecționată înapoi în labirint de fiecare dată.



În figura alăturată se observă cum Pinky încearcă să evadeze.

10. Cronometrul jocului.

Un ciclu de joc se realizează odată cu trecerea celor 90 de secunde în care playerul se poate

juca. Pentru această funcție a jocului, se utilizează o nouă clasă care are numele „Timer.cs”, lucru învățat prin vizionarea tutorialului [11]. Pe de altă parte, cronometrarea jocului necesită și crearea unui obiect de joc nou, un „textbox” în care se va afișa mesajul „Timp rămas: ” împreună cu un număr care reprezintă secundele rămase până când jocul se va sfârși.

În clasa tocmai creată, vom declara două variabile de tip întreg, „timeLeft” și „pause”, prima variabilă reținând numărul de secunde rămase, iar cea de-a doua va reține valoarea zero dacă jocul nu se află în modul pauză sau unu în caz contrar. Variabila de tip Text, specifică aplicției Unity, care se numește „countdownText” va reține mesajul menționat mai sus, „Timp rămas: timeLeft”. Dacă jocul este oprit prin activarea modului de pauză, variabila aceasta nu va mai reține textul respectiv, ci textul „Pauză!”. În funcția din „Timer.cs”, Start(), se va inițializa variabila „timeLeft” cu 90, iar variabila „pause” cu zero, pentru că la început modul de pauză este ianctiv inițial. Tot în cadrul acestei funcții se pornește enumeratorul(figura 21), „Losetime” care decrementează variabila „timeLeft” cu unu la fiecare secundă(figura 22). În funcția Update(), care se apelează per cadru, vom verifica dacă nu cumva jocul este „în pauză”. În fond, se verifică dacă variabila „pause” este egală cu zero, iar în caz afirmativ „countdownText” primește textul prin care jucătorul este informat cât timp mai are până când se sfârșește jocul. Când s-au scurs cele 90 de secunde, se va afișa textul: „Ups! Timpul s-a scurs. Apasă tasta N pentru a vâna din nou fantome.”, dar și enumeratorul este oprit prin comanda utilizată din figura 23.

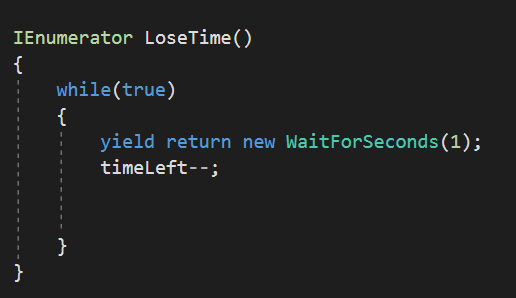




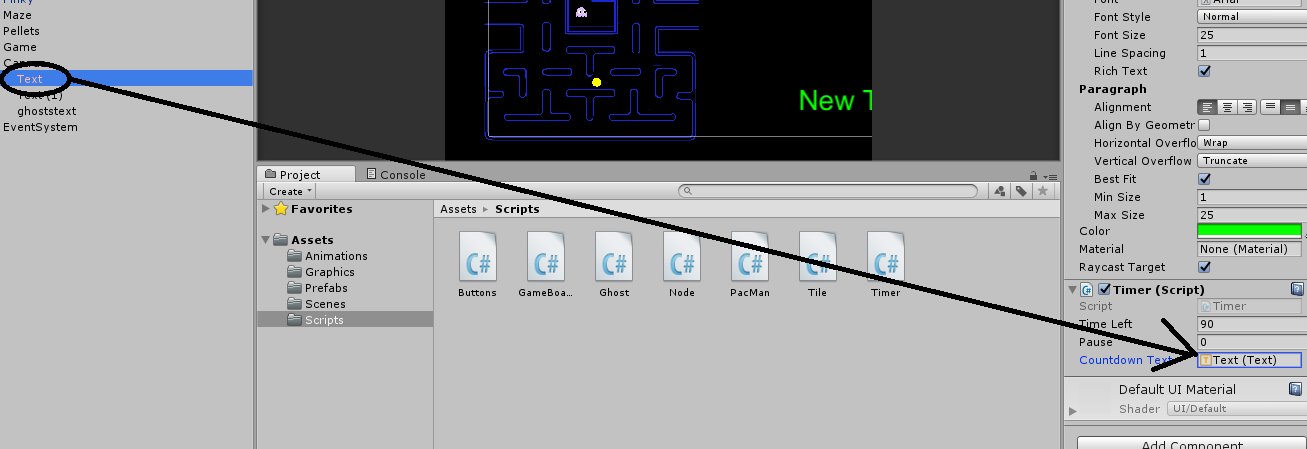
Fig. 21



Fig. 23

Fig. 22

Important! Odată ce s-a creat clasa „Timer.cs” și „căsuța” în care se va afișa timpul rămas sau mesajul „Pauză!” în caz de situație, trebuie în primul rând ca obiectului tocmai creat care reprezintă modalitatea de a afișa cele două mesaje să i se atribuie componenta „Timer.cs”. Fără asta, între această clasă și obiect n-ar fi existat nicio legătură. Următorul pas este să atribuim variabilei „countdownText”, obiectul de care are nevoie pentru a putea transpune cele două mesaje. Acest lucru este explicat în figura următoare:

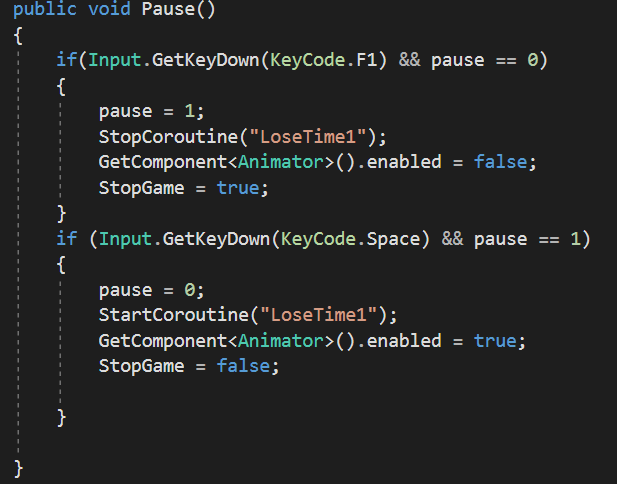


Pur și simplu, obiectul text în care vrem să apară cele două mesaje este adăugat la Timer(Script) unde apare „CountdownText” și scrie că nu îi este atribuit niciun text. Obiectul se poate adăuga prin tragere directă cu mouse-ul ca în figură, fie se apasă pe „cerculețul” aflat lângă variabilă prin care se deschide o fereastră care scrie „Add text”. Se caută în acea fereastră obiectul respectiv și se adaugă.

11. Modul pauză

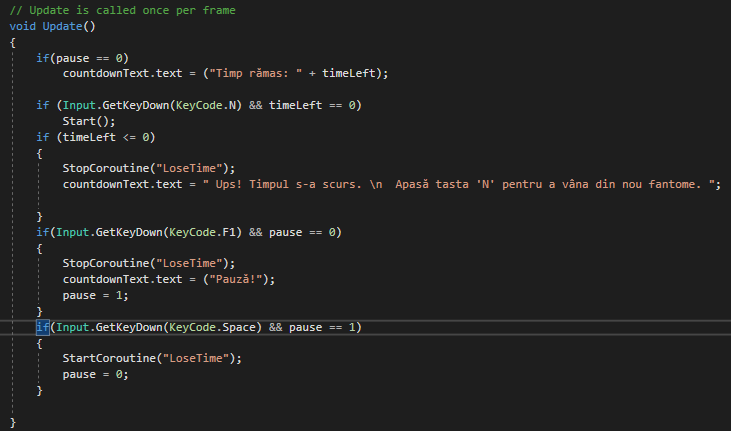
Joculețul creat beneficiează printre altele de opțiunea care îi permite jucătorului să pună jocul în modul de pauză oricând dorește. Pentru butonul de pauză trebuie inserate în clasele „Pacman.cs”, „Ghost.cs”, și „Timer.cs” câteva instrucțiuni. Acest lucru este necesar pentru a opri toate componentele din „mișcare” din joc, și anume PacMan-ul, cele patru fantome, dar și timpul care se scurge trebuie oprit pentru a evita un eventual „bug” caracterizat prin decalarea timpului. Practic, trebuie înghețate procesele care se actualizează permanent.

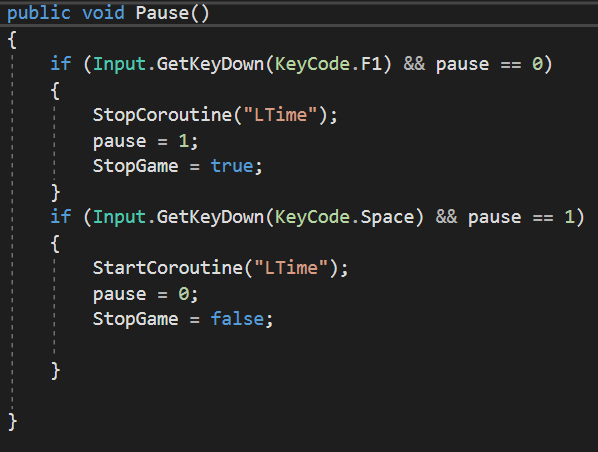
Ceea ce vom adăuga în cele trei clase este o nouă funcție prin care se activează și se dezactivează modul de pauză al jocului. Pentru această funcție este necesară declararea unor alte variabile, cum ar fi variabila de tip întreg „pause” care ia valoarea zero dacă modul de pauză este inactiv, iar valoarea unu în caz contrar. Cea de-a doua variabilă de care avem este de tip bool, adică reține doar valorile „false” și „true”, „StopGame” prin care se verifică dacă jocul este oprit sau nu. De exemplu, funcția care determină activarea modului de pauză din cadrului clasei „PacMan.cs” arată astfel:



Se observă că tasta prin care se activează modul de pauză este F1, iar cea prin care se dezactivează este SpaceBar. În cazul în care se apasă tasta F1, se activează modul pauză doar dacă variabila „pause” este egală cu zero, adică dacă nu este deja pauză. Dacă este deja activat acest mod, adică variabila este egală cu unu, apăsarea tastei F1 nu mai are niciun efect. Odată ce s-au îndeplinit cele două condiții din primul „if” din figura de mai sus, se execută operațiile: variabila „pause” primește valoarea unu, adică s-a activat modul, se oprește enumeratorul care îi permite PacMan-ului sa meargă doar 90 de secunde, iar cea de-a treia instrucțiune oprește animația PacMan-ului, în final variabila „StopGame” primind valoarea „true”. Procesul exact opus se aplică atunci când se apasă tasta Spacebar, care reprezintă dezactivarea acestui mod.

Asemănătoare sunt celelalte funcții folosite în celelalte două clase. Diferențe apar în cele două „if-uri”. În clasa „Ghost”, singura diferență este că nu trebuie oprită nicio animație, iar în clasa „Timer.cs”, pe lângă oprirea enumeratorului „LoseTime”, se face și operația prin care se afișează mesajul „Pauză”, procedeu explicat mai sus la punctul 10. Aceste funcții sunt apelate în funcția Update() corespondente fiecărei clase în parte; în clasa „Timer.cs” operația de activare și dezactivare a modului de pauză fiind făcută chiar în interiorul funcției Update().





Funcția „Pause” din cadrul clase „Ghost.cs”

Modul în care se realizează activarea și dezactivarea modului de pauză în clasa „Timer.cs”

12. Schimbarea vitezei de deplasare a PacMan-ului

Aplicația creată permite jucătorului să schimbe viteza PacMan-ului, mai exact îi poate mări viteza cu 50%. Acest lucru se realizează prin crearea unei noi funcții creată în interiorul clasei „PacMan.cs”, cu numele ChangeSpeed() care este apelată în interiorul funcției Update(). Ea este de tip void deoarece nu returnează nicio valoare, doar schimbă valoarea vitezei PacMan-ului prin apăsarea a două butoane, tastele „E” și „R”. „E” reprezintă tasta prin care i se mărește viteza, iar „R” tasta prin care se poate reveni la viteza inițială(funcția reprezentată în figura 24). În clasa „Ghost.cs”, unde se calculează scorul, dacă este mărită viteza PacMan-ului, scorul va scădea cu 1000, doar odată. Ne asigurăm că scorul scade doar odată la apăsarea tastei care determină creșterea vitezei, „E”, prin folosirea unei noi variabile „pressed”(figura 25).

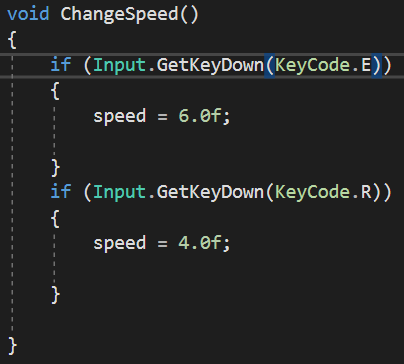


Fig. 24

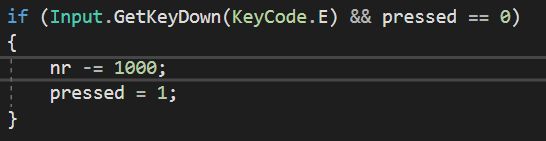
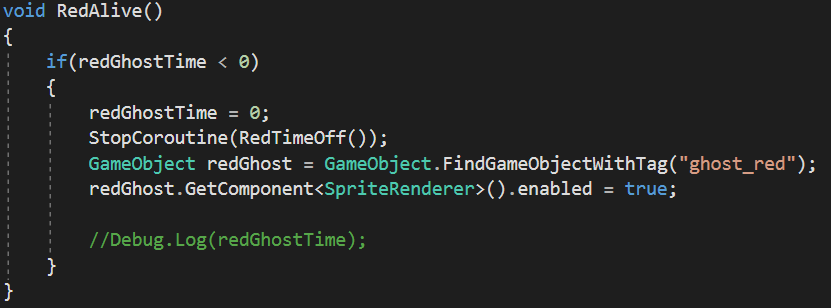


Fig. 25

13. „Învierea” fantomelor.

Jocul se bazează pe capturarea repetată a celor patru fantome timp de 90 de secunde. Fiecare fantomă, după ce este capturată, dispare timp de cinci secunde, lucru explicat mai sus, când s-a discutat despre capturarea acestora.

Pentru ca fanotmele să apară după cinci secunde din nou în labirint, se face referire la funcția de tip enumerator care numără cele cinci secunde în care fantoma este „absentă”. În cadrul noii funcții, se verifică dacă variabila care reținea cele 5 secunde și care este decrementată în enumeratorul respectiv este mai mică decât zero, adică dacă au trecut cele 5 secunde. În caz afirmativ, variabila care reținea timpul de inactivitate a fantomei este actualizat, primind valoarea zero, iar fantoma este făcută vizibilă din nou. De exemplu, următoarea secvență de cod prezintă funcția prin care este înviată fantoma roșie:



Se observă reactualizarea variabilei „redGhostTime” cu zero, variabilă care reține timpul în care Blinky este inactivă, se oprește enumeratorul care determina trecerea celor 5 secunde de inactivitate, iar următoarele două instrucțiuni determină apariția în scenă a fantomei roșii.

Analog, se face și pentru celelalte trei fantome, dar cu modificările de rigoare.

14. Sfârșitul jocului. Joc nou.

După trecerea celor 90 de secunde trebuie ca jocul să poată fi pornit din nou. Pentru aceasta, se integrează în clasa „PacMan.cs” o funcție care determină acest lucru. Odată ce cele 90 de secunde s-au scurs, se creează posibilitatea începerii unui joc nou. Tasta care va face posibil acest lucru este tasta „N” și apăsarea ei va avea efect doar când cele 90 de secunde s-au scurs.

Când cele 90 de secunde s-au scurs, variabila StopGame primește valoarea true(figura 26) care face ca PacMan-ul să se oprească din mers prin integrarea unei instrucțiuni în funcția Move() care face posibilă deplasarea PacMan-ului(figura 27).

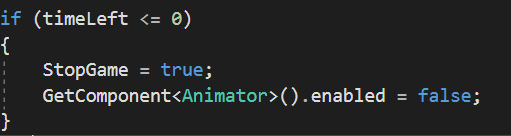


Fig. 26

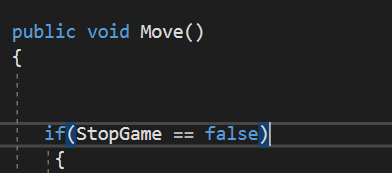
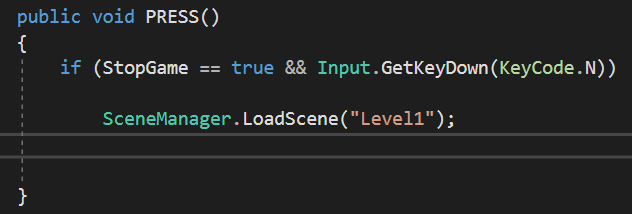


Fig. 27

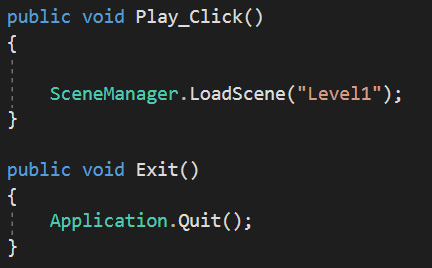
Reluarea jocului de la capăt se rezumă doar la un „if” în care se află doar o instrucțiune preluată din [12]:



Instrucțiunea „SceneManager.LoadScene(”Level1”);” încarcă din nou scena în care se află jocul efectiv, reinițializând toate variabilele, dar și pozițiile PacMan-ului și a celor patru fantome. Această instrucțiune necesită folosirea unei noi biblioteci, numindu-se astfel: „UnityEngine.SceneManagement”.

15. Meniul principal.

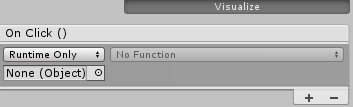
Pentru a crea meniul principal al jocului, vom crea o nouă scenă în folderul „Scenes” din proiect, crearea meniului principal fiind posibilă după ce s-a vizionat [13]. În aceasta vom introduce două butoane. Primul buton va face să apară scena jocului când este apăsat cu mouse-ul de către jucător. Cel de-al doilea buton prezent în scena meniului principal reprezintă butonul care permite ieșirea din aplicație. Pentru a putea folosi butoanele create vom crea o nouă clasă, „Buttons.cs”. În aceasta vom crea două funcții, una pentru butonul care permite pornirea jocului, iar una pentru butonul care închide aplicația. Cele două funcții sunt reprezentate în imaginea următoare:



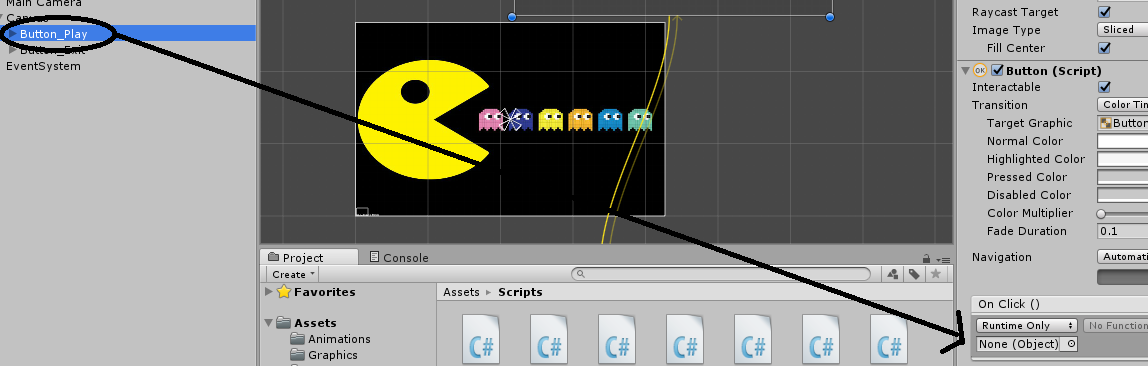
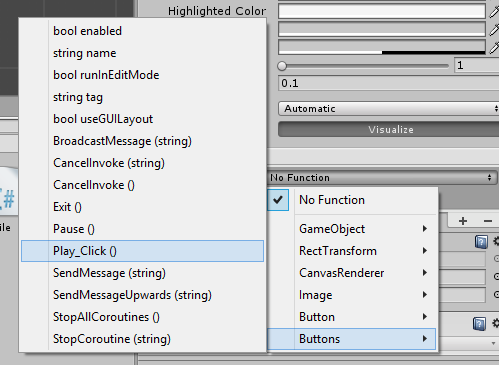
Fiecare funcție conține câte o instrucțiune predefinită de clasa „MonoBehaviour” care subordonează toate „scenariile” folosite de Unity. Practic, ea este mama clasă a tuturor claselor folosite într-un proiect realizat în aplicația de dezvoltare a jocurilor, Unity. Prima funcție conține comanda preluată din [12] și presupune, așa cum s-a precizat și mai sus când s-a vorbit de începerea unui joc nou, încărcarea scenei în care se află jocul. Cea de-a doua funcție are în alcătuire instrucțiunea „Application.Quit();” care face ca jocul să se închidă.

După ce s-a realizat clasa de care avem nevoie cu cele două funcții aferente, „Play\_Click()” și „Exit()”, trebuie atribuită celor două butoane create în Unity. Se dă click pe unul dintre cele două butoane și se observă că se deschide fereastra „Inspector”. În partea de jos a acestei ferestre apare butonul „Add Component” prin care se atașează butonului clasa „Buttons.cs”.

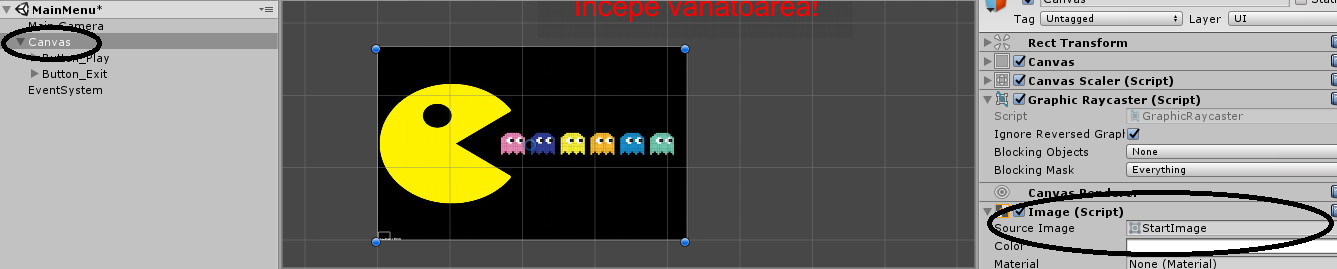
După ce aceasta a fost adăugată se procedează ca în schemele următoare care explică cum îi atribuim o acțiune butonului când se dă click pe acesta. Exact la fel se procedează pentru celălalt buton, doar că i se atribuie o altă acțiune când dăm click pe acesta.



Forma inițială



Așa se creează acțiunea pentru butonul Play. Analog se face și pentru butonul Exit.

 Pe butonul care permite intrarea în joc va scrie „Începe vânătoarea”, iar pe cel care închide aplicația, „Ieșire”. În final se adăugă imaginea de fundal descărcată de pe [14] al scenei care reprezintă meniul principal:

Se dă click pe „Canvas”, după care se deschide „Inspector”. Acolo apare „Image (Script)” de unde poți alege imaginea. Se caută numele imaginii descărcate, în cazul ăsta „StartImage” și se importă. Ne putem întoarce la meniul principal în timpul jocului prin apăsarea tastei „Esc”. La apăsarea acestei taste se execută comanda de la [12] integrată într-un „if” în una din clase.

Bibliografie

[1] https://weeklycoder.com/2017/01/08/pac-man-clone-using-unity-5-5-video-tutorial/

[2] https://www.youtube.com/watch?v=tjxKxZsofdk

[3] https://www.youtube.com/watch?v=lV-6lf0CxOQ

[4] https://www.youtube.com/watch?v=TyQLweQzhlM&t=73s

[5] https://docs.unity3d.com/ScriptReference/GameObject.GetComponent.html

[6] <https://weeklycoder.com/2017/01/08/pac-man-clone-using-unity-5-5-video-tutorial/>

[7] <https://www.youtube.com/watch?v=9nSRDbHaRRE&t=1401s>

[8] <https://docs.unity3d.com/ScriptReference/GameObject.Find.html>

[9] [https://forum.unity.com/th reads/transform-position-as-vector2s.498261/](https://forum.unity.com/th%20reads/transform-position-as-vector2s.498261/)

[10] <https://docs.unity3d.com/ScriptReference/MonoBehaviour.StartCoroutine.html>

[11] <https://www.youtube.com/watch?v=RLKQhpDxDlE>

[12] https://www.youtube.com/watch?v=eWp8zfau1XA&t=77s

[13] <https://www.youtube.com/watch?v=zc8ac_qUXQY&t=176s>

[14] <https://www.hivplusmag.com/cure/2017/10/22/tech-eats-hiv-pac-man-eats-ghosts>

Cuprins

Introducere...............................................................................2

Conținutul lucrării

1. Setările inițiale..................................................................4
2. Construirea labirintului.....................................................4
3. Mișcarea PacMan-ului......................................................5
4. Animația PacMan-ului și controlarea acesteia.................7
5. Introducerea fantomelor în joc.........................................9
6. Mișcarea fantomelor........................................................10
7. Capturarea fantomelor.....................................................11
8. Scor..................................................................................12
9. Momentul de evadare al fantomelor................................13
10. Cronometrul jocului.....................................................14
11. Modul pauză................................................................15
12. Schimbarea vitezei de deplasare a PacMan-ului.........18
13. „Învierea” fantomelor..................................................19
14. Sfârșitul jocului. Joc nou.............................................20
15. Meniul principal..........................................................21

Bibliografie..............................................................................23