

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE
ODSJEK ZA INFORMACIJSKU TEHNOLOGIJU

LUKA ATELJ

ZAVRŠNI RAD

**IZRADA PACMAN 3D IGRE NA UNITY
PLATFORMI**

Split, srpanj 2016.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE
ODSJEK ZA INFORMACIJSKU TEHNOLOGIJU

PREDMET: Programske metode i apstrakcije

ZAVRŠNI RAD

KANDIDAT: Luka Atelj

**TEMA ZAVRŠNOG RADA: Izrada Pacman 3D igre na
unity platformi**

MENTOR: Ljiljana Despalatović

Split, srpanj 2016.

Sadržaj

Sažetak	1
1 Uvod	2
2 Korištene tehnologije	3
2.1 Unity	3
2.2 Blender	3
3 Unity	4
3.1 Sučelje	4
3.1.1 Prozor projekta	4
3.1.2 Alatna traka	4
3.1.3 Pogled na scenu	4
3.1.4 Hierarhiski prozor	4
3.1.5 Konzola	4
3.1.6 Inspektor	5
3.2 Assets	5
3.3 Objekti	5
3.4 Komponente	5
3.5 Montaža (Prefab)	6
3.6 Pisanje koda	6
3.6.1 Programski jezici	6
3.6.2 Petlje	6
3.7 Fizika	7
3.8 Kamera	7
4 Scene	8
4.1 Main menu	8
4.2 Game	8
5 Objekti	9
5.1 Player	9
5.2 PickUps	9

5.3	Ghost	9
5.4	Bonuses	10
5.5	Ground	10
5.6	BackGround	10
5.7	Canvas	10
5.8	Direction light	10
5.9	Main Camera	10
6	Klase	11
6.1	Main Menu	11
6.2	Camera Controller	11
6.3	Rotator	11
6.4	Yrotator	11
6.5	Ghost Controller	11
6.6	Packman Controller	12
7	Animacije	14
7.1	Animator kontrola	14
8	Zvuk	15
8.1	Postavljanje zvuka	15
8.2	Audio Clip	15
9	Zaključak	16
	Bibliografija	17

Sažetak

Za završni rad sam izabrao temu "3D Pacman" koja je kreirana na Unity platformi. Cilj igre je skupiti sve bodove prije isteka vremena te igrač mora paziti da se tokom igre ne dogodi kolizija sa ostalim objektima. Unity nam omogućuje da se nakon izrade igre ona može igrati na skoro svim platformama. Za igru su potrebni modeli kako bi vizualno bila zanimljivija. Za modele sam koristio Blender alat te sam kreirane modele u njemu uključio u Unity.

Summary

3D game in Unity platform

For the final work I have chosen the theme "3D Pacman" that was created on the unity platform. The aim of the game is to collect all the points before time runs out and the player must pay attention that the player does not collides with other objects. Unity allows us after making game that it can be played on almost all platforms. For the game we need models to make the game visually more interesting. For models i used Blender tool and included created models in Unity.

1 Uvod

Tema završnog rada je izrada igre na Unity platformi. Unity platforma je za mnoge najbolja platforma za izradu igara jer je jednostavna za shvatiti te uz malo znanja c sharp ili java script programskog jezika svako može kreirati željenu igru. U svom radu koristio sam c sharp programski jezik jer pruža više mogućnosti za rad. Kolegiji koji su mi pomogli za izradu igre su: programske metode i apstrakcije, programiranje u c sharp-u, napredno windows programiranje i strukture podataka i algoritmi. Kroz izradu igre sam se postepeno upoznao sa Unity platformom koja je iznimno korisna i jednostavna za izradu igara na željenim konzolama. Za ovakav rad me motivirala želja za izradom igara te trenutno tržište prodaje igara koje raste iz dana u dan. U drugom poglavlju su navedene i kratko opisane tehnologije koje su se koristile pri izradi igre. To su Unity i Blender. Treće poglavlje opsežno opisuje unity platformu kako bi se predočila kompleksnost unity-a. Osnove koje se trebaju znati za izradu igre se nalaze u ovom poglavlju. U četvrtom poglavlju su opisane scene u igri. Ova igra sastoji se od dvije scene "MainMenu" i "Game". Svaka scena za sebe ima svoje objekte i skripte. Peto poglavlje prikazuje i opisuje sve objekte koji su korišteni u igri. Bez objekata igra nije igriva te su sve komponente svakog objekta objašnjene u petom poglavlju. Šesto poglavlje opisuje klase. Klase upravljaju objektima i u ovom poglavlju se vide sve funkcionalnosti svakog objekta. Sedmo poglavlje su animacije. U ovom poglavlju je opisana izrada animacija i objašnjenje zašto su animacije važne. Osmo poglavlje objašnjava važnost zvuka i njegovu upotrebu.

2 Korištene tehnologije

Za izradu ovog rada koristili su se Unity i Blender alati.

2.1 Unity

Unity je najbrže rastući game engine za izradu 2D i 3D igara na svjetu. Jednostavan je za korištenje, može se raditi u timu, podržava bitne platforme kao što su: PC, Mac, Linux, Android, iOS, xBox, Play Station, itd. Sve što od vas zahtjeva je znanje c sharp programskog jezika. Može se koristiti i java script programski jezik, ali puno je jednostavnije zbog pronalazaka pogreški tokom pisanja skripte, koristiti c sharp programski jezik. Skripte se mogu pisati u bilo kojem editoru. Ukoliko korisnik želi napraviti unikatnu igru bilo bi poželjno da zna samostalno izrađivati 3D ili 2D objekte. Samostalno izrađivanje objekata nije nužno jer danas postoji mnogo gotovih objekata koje korisnik može besplatno preuzeti i koristiti se s njima. Unity se može besplatno koristiti te se može na njemu zarađivati. Ukoliko autor neke igre zaradi preko 100.000 dolara tada mora platiti profesionalnu verziju Unity-a.

2.2 Blender

Blender je besplatan alat za izradu 3D računalne grafike koji se koristi za izradu animiranih filmova, vizualnih efekata, modela za 3D printere, itd. Blender je razvila softverska kuća "Not a Number Technologies". Radna površina blendera je podjeljena na tri djela: glavni meni na vrhu, središnji 3D prostor za modeliranje i prostor sa dugmićima i opcijama na dnu prozora. Nakon što se kreira model potrebno je kreirati datoteku sa 3ds ekstenzijom te je takvu ubaciti u Unity alat.

3 Unity

3.1 Sučelje

Glavno sučelje unity-a sadrži više prozora koji se mogu grupirati po želji korisnika. Sučelje se sastoji od alatne trake, hierarhijskog prozora, pogleda na scenu, inspektora i prozora projekta.

3.1.1 Prozor projekta

Prozor projekta prikazuje sve datoteke koje se koriste u projektu. Preporučava se napraviti direktorije u kojima se raspoređuju skripte, zvukovi, animacije i ostali objekti. Kada uključujemo neke segmente u projekt oni će se pojaviti u prozoru projekta.

3.1.2 Alatna traka

Alatna traka omogućuje najbitnije funkcionalnosti za kreiranje igre. Na lijevoj strani se nalaze alati pomoću kojih možemo upravljati sa scenom i sa objektima koji su u sceni. Objektima možemo postaviti visinu, dužinu, koordinate trenutnog položaja itd. Na sredini se nalaze "play", "pause" i "stop" kontrole koje omogućuju testiranje igre. Na desnoj strani se nalaze kontrole za oblak servis i osobni unity račun te kontrole za postavljanje rasporeda sučelja po korisnikovoj želji.

3.1.3 Pogled na scenu

Scena nam prikazuje 3D ili 2D objekte, ovisno o vrsti projekta na kojem radimo. Pogled na scenu omogućava programeru da vizualno upravlja svojom igrom. Osim pogleda na scenu imamo i pogled na igru (game view) koji programeru omogućva testiranje svoje igre te pronalženje pogrešaka.

3.1.4 Hierarhiski prozor

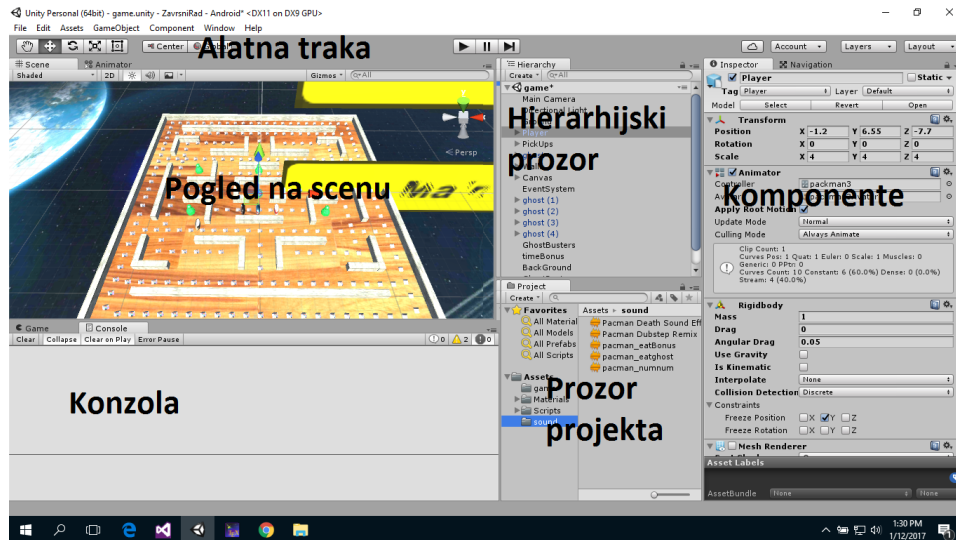
Ovdje možemo vidjeti kako su objekti pridruženi drugim objektima vizualno, što pomaže pri pisanju koda. Hierarhiski prozor će prikazati one objekte koji su u sceni koju trenutno uređujemo.

3.1.5 Konzola

Konzola nas obavještava kada se dogodi neka greška, upozorenje ili iznimka. Konzola je dosta praktična jer preko nje možemo saznati što nam se u kojem trenutku događa u projektu i to preko funkcije "Debug.Log()".

3.1.6 Inspektor

Inspektor omogućuje da imamo uvid u sva svojstva trenutno odabranog objekta. Preko njega možemo mjenjati komponente objekta. Odabirom nekog objekta inspektor će automatski komponente tog objekta prikazati u svom prozoru.



Slika 1: Prikaz sučelja

3.2 Assets

Assets-i su uključeni ili kreirani objekti. Assets-i mogu biti 3D modeli, zvuk, tekst itd. Kada neki objekt uključimo u unity on će se automatski pojaviti u asset direktoriju.

3.3 Objekti

Sve što se nalazi na sceni je objekt. Objekti se mogu kreirati u unity-u ili pridružiti ih iz drugog alata. Svaki objekt u sebi sadrži sljedeće osobine: ime, tag, sloj i static. Autor sam može proizvoljno dati ime svom objektu po želji. Tagovi su jako poželjni kada se referenciraju objekti u kodu. Traženje objekata preko tagova olakšava posao programeru pogotovo kada se radi na većim projektima. Kada više različitih objekata rade istu stvar onda je poželjno koristiti tag. Objekti mogu biti dodjeljeni drugom sloju. Static se koristi za objekte koji su statični.

3.4 Komponente

Svaki objekt mora sadržavati neke komponente. Pri kreiranju objekta njemu će se dodjeliti "Transform" komponenta koja prikazuje poziciju, rotaciju i visinu objekta. Bez ove komponente objekt bi bio nevidljiv na sceni. Neke od komponenata su "Character Controller", "Nav Mesh Agent", "Rigidbody" itd.

3.5 Montaža (Prefab)

Prefab možemo gledati kao objekt na koji je korisnik postavio instancu. Kada se napravi neka promjena na prefabu ta promjena će se primjeniti za instance tog objekta.

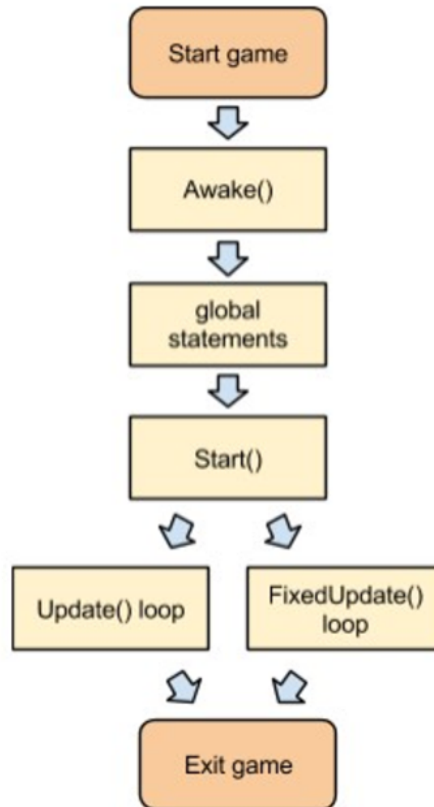
3.6 Pisanje koda

3.6.1 Programski jezici

U unity platformi se može kreirati igra korištenjem jednog od tri programska jezika, a oni su c sharp, javaScript i Boo. Prednosti c sharp programskog jezika prema java scriptu je to što se lakše otkrivaju pogreške u kodu. Java script se često koristi u Unity-u. Često se naziva i Unity Script. Prednosti java scripta prema c sharp-u su jednostavnija sintaksa java script programskog jezika i mnogo primjera koje se mogu naći na internetu. Programski jezik Boo se jako malo koristi u Unity-u i ima samo nekolicinu primjera na internetu što korisnika automatski udaljuje od ovog jezika.

3.6.2 Petlje

Kada se kreira skripta unity će automatski kreirati "Start()" funkciju i "Update()" petlju. "Awake()" funkcija je funkcija koja će se prva pozvati kada se igra pokrene i ona se poziva samo jednom. "Awake()" funkcija se najčešće koristi da se u nju postave reference između skripti jer se objekti još nisu izvršili. "Start()" funkcija će se pozvati nakon "Awake()" funkcije, ona se poziva kada se svi objekti iz scene izvrše. "Start()" funkcija će se također pozvati samo jednom. "Start()" funkcija je najbolje mjesto za postaviti reference objekata. Nakon funkcije "Start()" unity će pozvati petlju "Update()". Postoje dvije takve petlje a to su "Update()" i "Fixedupdate()". Petlja se izvršava onoliko puta po sekundi koliko je računalo može procesuirati. Razlika između ove dvije petlje je ta što se "FixedUpdate()" izvršava u određenom vremenu.



Slika 2: Arhitektura igre

3.7 Fizika

Svaki objekt koji ima veze sa fizikom mora sadržavati `RigidBody` i `Collider` komponente. Glavne komponente `RigidBody`-a su "Use Gravity" i "Is Kinematic". "Use Gravity" jednostavno znači da će se tom objektu postaviti gravitacija s obzirom na komponentu "Mass". Komponenta "Is Kinematic" se koristi kada korisnik želi tokom igre upravljati sa objektom preko transform funkcije. "Use Gravity" i "Is Kinematic" komponente su tipa `bool` i njih postavljamo u inspektoru.

3.8 Kamera

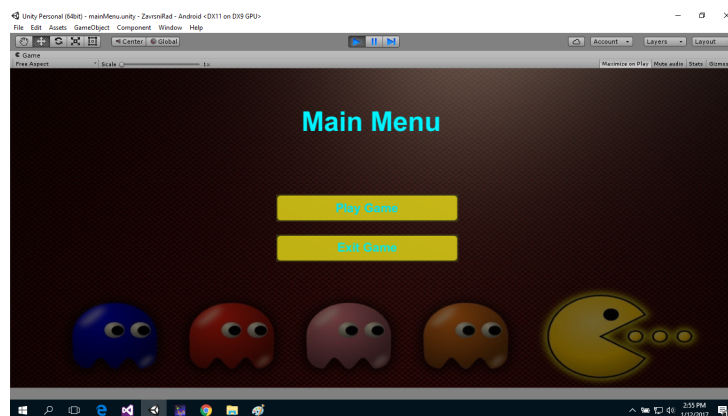
Svaka scena mora sadržavati barem jednu kameru. Kada se scena kreira kreira se i objekt glavna kamera. Preko kamere određujemo da li je igra "first person game" ili "third person game".

4 Scene

Scena sadrži sve objekte igre koji se koriste. Scena nam može služiti za kreiranje izbornika, ukoliko igra ima više levela, svaki level će se nalaziti u svojoj sceni. Preko scene uređujemo okolinu, postavljamo objekte i prepreke. Scena zapravo sve sklopljene elemente pretvara u jedan sadržaj.

4.1 Main menu

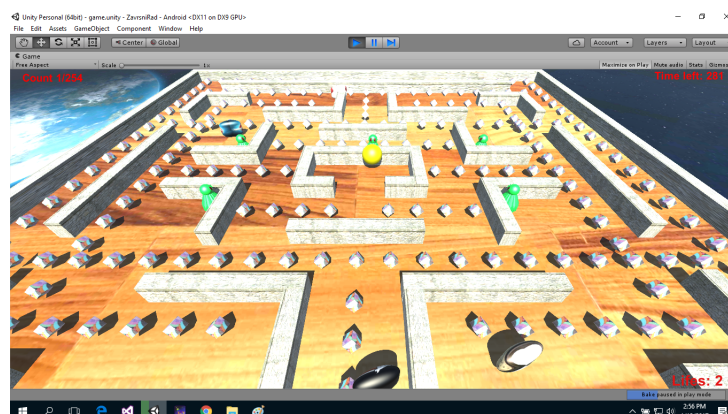
Ovo je početna scena u igri. Scena sadrži platno sa tekstom i dva botuna te objektom "plane" na kojega je zaljepljena slika radi boljeg ugođaja.



Slika 3: Scena "Main Menu"

4.2 Game

"Game" je glavna scena na koju korisnik dolazi pritiskom botuna "Play" na "Main Menu" sceni. Sadrži sve doli napisane objekte i njihove komponente. Također sadrži i platno koje se pojavljuje po potrebi i preko kojeg se može vratiti na početnu scenu.



Slika 4: Scena "Game"

5 Objekti

5.1 Player

Ovo je glavni objekt u igri. Napravljen je od više objekata u blenderu. S ovim objektom upravlja klasa "playerController". Sadrži sljedeće komponente:

- Transform → Zadana komponenta svakog objekta koja određuje poziciju, rotaciju i veličinu objekta.
- Rigidbody → Kontrolira poziciju objekta kroz fizičku simulaciju. Ovaj objekt ne koristi fiziku, ali je iskorištena njegova komponenta da se onemogući kretanje objekta po y osi.
- Sphere collider → Služi za detekciju kolajdera u obliku kule. Time možemo postaviti veličinu radiusa za kolajder.
- Audio source → Pridružen je objektu zbog ispuštanja zvuka kojeg objekt kontrolira.
- Animator → Služi za kontrolu animacije objekta. U ovoj igri animacija je napravljena u Blender alatu. Objekt preko ove komponente upravlja sa svojom animacijom.

5.2 PickUps

Ovo je prazan objekt u kojemu su smješteni objekti "PickUp" koje igrač skuplja u igri. S ovim objektom upravlja klasa "Rotator". Njegove komponente su:

- Transform
- Box Collider → Služi za detekciju kolajdera u obliku kocke. Možemo postaviti x, y i z veličinu kolajdera.
- Mesh renderer → Pomoću njega dodjeljujemo materijale objektu(boja), možemo definirati veličinu i sjenu.

5.3 Ghost

Postoje pet istih objekata "Ghost". Njima je pridružen tag "ghost" i "ghostCollor". Preko taga možemo upravljati s njima iz skripte drugog objekta. Njima upravlja klasa "ghostController" i svi imaju iste komponente:

- Transform
- Mesh renderer
- Capsule collider → Služi za detekciju kolajdera u obliku kapsule. Možemo postaviti visinu i radijus kolajdera. Kolajder sadrži svojstvo "isTrigger" koje se koristi za triggering događaja. Npr: void OnTriggerEnter(Collider other) other.gameObject.SetActive(false);
- Character controller → Omogućuje kretanje objekta bez da se koristi Rigidbody i da se sudara sa drugim objektima tj. da ne prolazi kroz njih.
- Audio source

- Animator
- Nav Mesh Agent → Omogućuje nam da se objekti ne sudaraju međusobno i sa drugim objektima tokom kretanja.

5.4 Bonuses

Imamo četiri različita objekta bonus koji imaju iste komponente ali služe za različite stvari koje se kontroliraju u "playerController" klasi. Ovim objektima je pridružena skripta Yrotator koja ih rotira po y osi radi boljeg ugođaja tokom igranja igre. Nazivi objekata su: "ghostBusters", "ghostBuster", "lifeBonus" i "timeBonus". Njihove komponente su:

- Transform
- Mesh renderer
- Capsule collider

5.5 Ground

Ovaj objekt je podloga na kojoj se igra odvija. Komponente podloge su:

- Transform
- Mesh renderer

5.6 BackGround

Ovaj objekt je podloga koja se nalazi ispod glavne podloge na koji je nalijepljena slika zbog boljeg ambijenta tokom igranja. Imam iste komponente kao objekt "ground".

5.7 Canvas

Ovaj objekt u sebi sadži pet tekst komponenti i dva botuna. Tekst komponente služe za ispisivanje određenog sadržaja, a botuni za navigaciju kroz scene.

5.8 Direction light

Služi nam za postavljanje pozicije svjetla, namještanje sjene, jačina svjetla itd. Kao i svi drugi objekti sadrži transform komponentu.

5.9 Main Camera

Glavna kamera u igri koja prati igrača tokom njegove kretanja. Kretanju tokom igre mu omogućuje klasa "cameraController" koja je priključena na ovaj objekt.

6 Klase

Svakom objektu je dodjeljena svoja zasebna klasa. Svaka klasa se nalazi u zasebnoj skripti. Klase koje nasljeđuju `MonoBehavior` nesmiju imati konstruktore. Klase koje su kreirane za ovu igru su "mainMenu", "cameraController", "Rotator", "Yrotator", "ghostController" i "packman-Controller".

6.1 Main Menu

Klasa "mainMenu" se nalazi u prvoj sceni. Ona upravlja botunima koji su postavljeni preko "onClick()" funkcije.

- void Start() -> Omogućava prikaz objekata: `mainMenu.enabled = true;`
- public void exit() -> Omogućava zatvaranje igre: `Application.Quit();`
- public void playGame() -> Otvara "game" scenu. `SceneManager.LoadScene("game");`
- public void menu() -> Otvara "mainMenu" scenu koja će se pozivati iz "game" scene.

6.2 Camera Controller

Klasa "cameraController" se nalazi u "game" sceni i ona kontrolira kretanje kamere tokom igre. Funkcije koje se koriste:

- void Start () -> Postavlja kameru na početnu poziciju.
- void Update() -> Prati kretanje Packmana i pomiče kameru u tom smjeru.

6.3 Rotator

Klasa "Rotator" koristi samo Update() petlju. Ova klasa služi za objekte koji će se rotirati po x, y, z osi.

6.4 Yrotator

Klasa "Yrotator" radi isto što i klasa "Rotator" ali rotira objekte samo po y osi.

6.5 Ghost Controller

Klasa koja upravlja s objektima "Ghost". Ti objekti imaju komponentu `NavMeshAgent`. `NavMeshAgent` komponenta omogućuje objektima koji se kreću ka istom cilju da se ne sudaraju.

- void Start () -> Postavlja `NavMeshAgent` komponentu objektima "Ghost". U skriptu je dodana komponenta `AudioSource` koja će izpustiti zvuk kada jedan od objekata "Ghost" se sudari sa objektom "Player". Postavlja se brzina objekta: `Ghost.speed = 30;` Objekt "Player" smo dohvatili preko svojstva "Tag": `packMan = GameObject.FindGameObjectWithTag("Player")`

- void Update() → Funkcija koja objektima "Ghost" postavlja destinaciju: Ghost.destination = packMan.transform.position;
- void OnTriggerEnter(Collider other) → Funkcija koja detektira kada će se objekti "Ghost" sudariti sa objektom "Player". U koliko se to dogodi aktivira se AudioSource komponenta. Također ukoliko se objekti "Ghost" sudare sa nekim objektom koji predstavlja bonus, aktivacija tog objekta će se postaviti na "false".

6.6 Packman Controller

Glavna klasa s kojom se kontrolira igrača i njegove komponente. Klasa je pridružena objektu "Player" koji je glavni objekt u igri.

Reference:

- public GameObject packman → Referenca na igrača.
- public GameObject ghostBusters → Referenca na bonus.
- public GameObject ghostBuster → Referenca na bonus.
- public GameObject bonusTime → Referenca na bonus.
- public GameObject bonusLife → Referenca na bonus.
- public AudioClip numnum → Referenca na određeni zvuk.
- public AudioClip eatGhost → Referenca na određeni zvuk.
- public AudioClip eatBonus → Referenca na određeni zvuk.
- public Text countObjects → Referenca na text koji ispisuje koliko je objekata skupljeno.
- public Text time → Referenca na text koji ispisuje koliko je vremena igraču ostalo.
- public Text winText → Referenca na text koji se ispisuje ako igrač skupi sve "PickUp" objekte.
- public Text lifes → Referenca na text koji ispisuje koliko igrač ima života.
- public Text countObjects → Referenca na text koji ispisuje koliko je igrač skupio bodova.
- public Text pause → Referenca na text koji ispisuje "Pause" kada korisnik pritisne bilo koji dio ekrana.
- private Vector3 up, right, left, down, currentDirection → Određuje smjer kretanja igrača.
- int count → Broji koliko je "PickUp" objekata ostalo.
- public float timeLeft → Broji koliko je vremena ostalo.
- public GameObject[] ghostColor → Niz djela "Ghost" objekta za bojanje.
- public GameObject[] ghosts → Niz svih "Ghost" objekata.
- private Color color → referenca na boju.
- int lifeCount → Broji koliko je života ostalo igraču.

- `int tempTime` → Broji koliko je vremena ostalo do kraja igre.
- `float bonusTimeLeft` → Broji koliko je vremena ostalo dok su bonusi aktivni.
- `private float currentAcceleration` → Broj koji određuje poziciju y osi.

Metode:

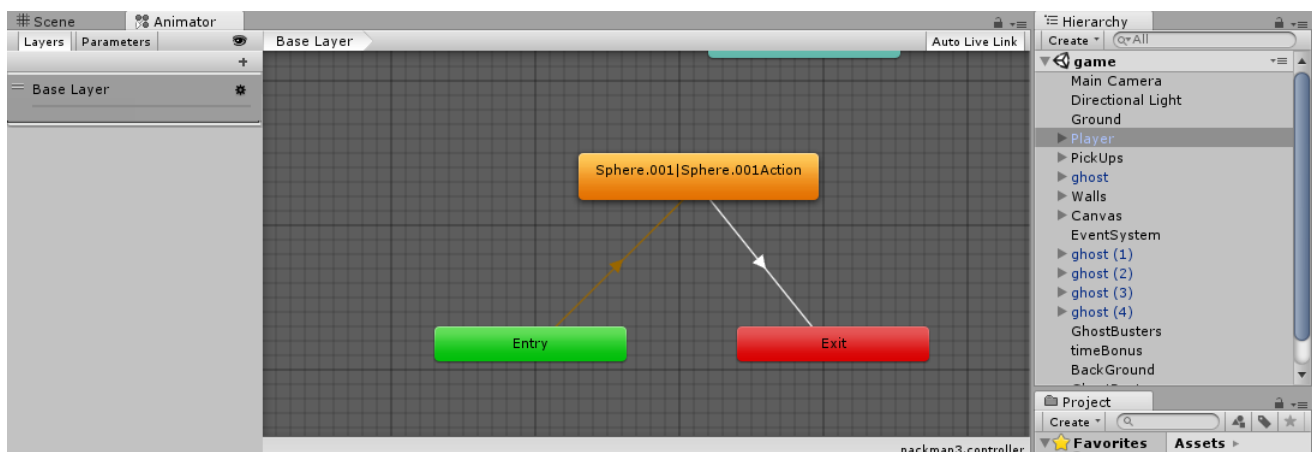
- `void Start ()` → Inicializira sve reference.
- `void finalText(int final)` → Metoda koja ispisuje jeli korisnik pobedio ili izgubio.
- `void ghostStatus(bool status)` → Postavlja aktivnost objekata Ghost.
- `void changeGhostColor(Color col)` → Postavlja boju objekata Ghost.
- `Vector3 randomPosition()` → Vraća random vektor 3 za postavljanje pozicija bonusa.
- `void pausePhone()` → Korisnik dodirom na ekran poziva ovu funkciju. Funkcija pauzira igru tako da zamrzne sliku i postavi aktivan plato koji ispisuje odgovarajući tekst i aktivira botun koji vodi korisnika na glavni izbornik. Ukoliko korisnik želi nastaviti igru treba opet pritisnuti dio ekrana na kojem se ne nalazi ovaj botun.
- `void activateBonuses()` → Funkcija koja svakih 30 sekundi aktivira bonuse i drži ih aktivnima 10 sekundi. Nakon što prođe 10 sekundi bonusi se deaktiviraju i pojavljuju opet za 20 sekundi.
- `void checkGhost()` → Funkcija koja provjerava aktivnost objekata "Ghost". Nakon što igrač skupi određeni bonus objekti "Ghost" se deaktiviraju na 10 sekundi. Ova funkcija provjerava dali je prošlo 10 sekundi od skupljanja bonusa te ako je aktivira objekte Ghost.
- `void setFinalEnviroment()` → Funkcija koja provjerava je li igrač ispunio uvjete da se pozove metodu `finalText(int final)`.
- `void setCanvas()` → Tokom igre na ekranu stoje 3 teksta. Jedan je za broj bodova koje je igrač skupio, drugi je za prestalo vrijeme igre a treći za broj života koji je igraču ostao. Funkcija prikazuje na ekranu reference "time.tex"t, "lifes.text", "countObjects.text".
- `float? getPhonePosition()` → Nakon svake dvije sekunde vraća poziciju y osi.
- `void playerMoveAcceleration()` → Korisnik upravlja sa glavnim objektom "Player" preko akceleracije mobitela. Problem ovakvog upravljanja sa objektima je taj što se mobitel treba držati uvijek u istom položaju. Zato u sebi ova metoda sadrži metodu `getPhonePosition()`. Tako korisnik može igrati igru u bilo kojem položaju jer se svake dvije sekunde trenutna pozicija y osi postavlja na zadanu poziciju.
- `void OnTriggerEnter(Collider other)` → Ova metoda detektira kada su se dva objekta sudarila. U ovoj metodi se nalaze najbitnije funkcionalnosti igre. Igrač se sudara sa objektima koje mora skupiti("Pick Up"), sa objektima koji ga hvataju("Ghost") i sa bonus objektima. Sa uvjetima je detektirano koji se sudar dogodio i time se vrši radnja nad tim objektima.

7 Animacije

Igre se većinom rade tako da korisnik upravlja sa nekim objektom. Da bi taj objekt bio vizualno zanimljiv potrebno mu je dodati neke animacije. Animacije se u Unity-u rade u animacijskom prozoru. Animacijski prozor sadrži vremensku crtu koja nam služi da odredimo vrijeme animacije. Jedan objekt može imati više animacija. Sučelje za kontrolu svih animacija objekta se zove Animator kontrola.

7.1 Animator kontrola

Sa animator kontrolom određujemo kada će se koja animacija izvršiti. Najčešće je slučaj da objekt ima više animacija koje čekaju svoj red za izvršavanje. Animator kontrola tu postavlja redosljed izvršavanja animacija. Npr. ukoliko je određena animacija aktivna i kad korisnik pritisne određenu tipku ta animacija prelazi na drugu određenu animaciju. Na slici je prikazana animator kontrola za objekt "Player".



Slika 5: Dodjeljivanje animacija objektu

8 Zvuk

Da bi igra bila korisniku što poželjnija potrebno je u nju postaviti zvuk. Zvuk daje korisniku osjećaj da stvarno upravlja sa nekim objektom. Glavna komponenta zvuka u Unity-u je "Audio Source". On reproducira zvuk u sceni preko "Audio Clip" svojstva. U tablici su prikazani formati zvuka koje podržava unity.

Supported formats

Format	Extensions
MPEG layer 3	.mp3
Ogg Vorbis	.ogg
Microsoft Wave	.wav
Audio Interchange File Format	.aiff / .aif
Ultimate Soundtracker module	.mod
Impulse Tracker module	.it
Scream Tracker module	.s3m
FastTracker 2 module	.xm

Slika 5: Formati zvuka koje podržava Unity

8.1 Postavljanje zvuka

Postavljanje zvuka u unity-u je vrlo jednostavno. Da bismo stvorili zvuk koji će se koristiti tokom igre potrebno je taj zvuk prvo prebaciti u unity projekt. Kada je zvuk u projektu dodjeli se objektu sa kojim ima komponentu "Audio Source". "Audio Source" komponenta sadrži mnogo svojstva. Jedno od svojstva je "Play on Awake" koje se koristi u "Main Menu" sceni. Ovo svojstvo pokreće željeni zvuk kada se igra pokrene. Unity svaku datoteku zvuka gleda kao "Audio Clip".

8.2 Audio Clip

"Audio Source" komponenta je beskorisna bez svojstva "Audio Clip". "Audio Source" je zapravo kontrola koja upravlja sa svojstvom "Audio Clip". Ako koristimo samo jedan zvuk dovoljno ga je samo ubaciti u objekt, a unity će sam napraviti komponentu i ubaciti taj zvuk u nju.

Primjer ispuštanja zvuka:

```
AudioSource audio;  
public AudioClip death;  
audio = GetComponent<AudioSource>();  
audio.PlayOneShot(death);
```

9 Zaključak

Cilj svake igre je da bude korisniku što jednostavnija za upotrebu i naravno da bude zanimljiva. Svaki programer može napraviti igru, ali pitanje je koliko će ta igra privlačiti korisnika. Da bi igra bila pristupačnija korisniku trebala bi biti lijepo dizajnirana jer većina korisnika koja želi igrati neku igru će prvo pogledati dizajn igre. Možemo reći da će dizajn igre privući korisnika a funkcionalnosti igre će pridobiti korisnika. Učenje svake tehnologije je u početku teško ali ukoliko je programer željan stvaranje nečeg novog i ima strast prema igrama unity je pravo rješenje za nega. Na internetu se može naći puno primjera i objašnjenja kako koristiti unity. Tijekom izrade ovog projekta sam se upoznao sa unity okolinom. Trebalo mi je nekoliko vremena da savladam glavne komponente unity-a, a nakon što sam to prošao samo sam trebao upotrijebiti vlastitu maštu kako bih igru napravio što pristupačnijom korisniku. Kod izrade igara za mobilne uređaje se javlja problem kontrole nekog objekta. Objekte možemo kontrolirati preko dugmadi, dodira na ekran i preko akcelometra. Ja sam se odlučio za akcelometar zato što time ostavljam korisniku čist pogled na igru, ali sve ovisi o kompleksnosti igre koje će autor kontrolu odabrati.

Literatura

- <https://unity3d.com/learn/tutorials>
- <https://www.youtube.com/watch?v=n0VspDUOErEt=452s>
- <https://unity3d.com/learn/tutorials/topics/scripting>