Izrada 2D Android igre u Unity okruženju

Crnković, Edi

Sveučilište Jurja Dobrile u Puli
Odjel za informacijsko komunikacijske tehnologije

EDI CRNKOVIĆ

IZRADA 2D ANDROID IGRE U UNITY OKRUŽENJU

Završni rad

Pula, 2016 godine

Sveučilište Jurja Dobrile u Puli
Odjel za informacijsko komunikacijske tehnologije

EDI CRNKOVIĆ

IZRADA 2D ANDROID IGRE U UNITY OKRUŽENJU

Završni rad

JMBAG: 0303033026, redoviti student

Studijski smjer: Informatika

Predmet: Programiranje

Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: inInformacijski sustavi i informatologija
Mentor: doc. dr. sc. Tihomir Orehovački

Pula, rujan 2016 godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Edi Crnković, kandidat za prvostupnika Informatike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraţivanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student:

U Puli, __. __. 2016. _____

IZJAVA

o korištenju autorskog djela

Ja, Edi Crnković dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom izrada 2D android igre u Unity okruženju koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

Potpis

U Puli, _____ _____

SADRŽAJ

UVOD

Tehnologije mobilnih uređaja se neprestano razvijaju i usavršavaju, što dovodi do sve većeg približavanja mobilnih uređaja prijenosnim računalima.

Razvojem mobilnih uređaja, igre postaju još dostupnije igračima, nudeći mogućnost preuzimanja golemog broja igara s online servisa poput *GooglePlay*-a za Android uređaje ili *AppleStore* za Apple uređaje. Takvi servisi omogućuju svojim korisnicima da igre koje su sami napravili stave dostupnima za preuzimanje na servisu kako bi ih drugi igrači mogli igrati. Takav pristup uvelike je pridonio popularnosti i jednostavnosti izrade igara. U današnje vrijeme, korištenjem već gotovih engine-a i *framework-a* za izradu igara, moguće je izraditi vlastitu igru s minimalnim troškovima te na takvom proizvodu ostvariti profit.

Industrija video igara danas predstavlja najrašireniju i najprofitabilniju granu zabavne industrije, više nego filmska i glazbena industrija zajedno. Rezultat ovog trenda su stotine milijuna korisnika Android operativnog sistema, preko 600 različitih mobilnih uređaja koje pokreće Android, kao i preko 700 000 aplikacija pravljenih za Android i dostupnih za *download*. Postoji veliki broj različitih vrsta Android aplikacija, od igara i aplikacija zabavnog sadržaja, do onih poslovnih I edukativnih.

Zbog sve većeg rasta I potražnje za zaposlenicima u grani video igra izabrana je ova tema za završni rad, koja će pomoći u usavršavanju potrebnih vještina za izradu igre ali i mogućnosti zarade.

Aplikacija koja je opisana u radu spada u kategoriju mobilnih video igara te su u ovom radu obrađeni osnovni koncepti potrebni za izradu 2D android igre pomoću *Unity* 3D tehnologije. Igra je zamišljena kao jednostavna logička igra koja bi mogla služiti kao pomoć pri razvijanju logike.

Glavna inspiracija za odabir mobilne logičke igre proizlazi iz igara kao što su: *Limbo* (Playdead ApS, 2010), *Where's My Water* (Disney Mobile, 2011), *World of Goo* (2D Boy, 2008) i *Braid* (Jonathan Blow, 2010) koji za razliku od ostalih igara zahtijevaju punu koncentraciju i razmišljanje igrača da bi ih mogao riješiti, te time potiču razvijanje mozga za razliku od nekih ostalih igara koje služe samo za razbijanje dosade.

U prvom će poglavlju biti riječi općenito o Androidu te kratko o njegovoj povijesti nastanka i razvoja.

Drugo poglavlje bavi se razvojnim alatom Unity 3D koji se koristi za stvaranje igara na *Android* platformi. Riječi će biti ukratko o nastanku i razvoju, sučelju te najčešće korištenim mogućnostima programa.

Od trećeg do petog poglavlja biti će objašnjen sam proces izrade igre - od stvaranja ideje do programiranja.

Šesto poglavlje bavit će se objavljivanjem aplikacija na *Google Play* trgovini te mogućnostima zarade, što je jedan od ciljeva programiranja.

1. Android

Android je operacijski sustav (OS) baziran na *Linux kernelu*[1] , primarno namijenjen za korištenje na mobilnim uređajima s ekranima osjetljivima na dodir. Android je *open-source* project (AOSP) što znači da svatko može uzeti objavljeni kod Android sustava te ga uređivati, unaprjeđivati te uljepšavati. Također omogućava i da sami 'prčkate' po sustavu i time postanete pravi vlasnik svog uređaja.

Android je cjelokupna platforma otvorenog koda dizajnirana prvenstveno za mobilne uređaje. Odvaja *hardware* uređaja od *software-a* kojeg pokreće te tako omogućuje velikom broju uređaja pokretanje istih aplikacija, što stvara bogati ekosustav i za korisnike i za programere.

Android uređaji su već nekoliko godina najprodavaniji na tržištu mobilnih uređaja i njihov broj iz godine u godinu raste kao sto se vidi u tablici 1.

Tablica 1:Tržišni udio mobilnih operativnih sustava u drugom kvartalu od 2012 do 2014.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	69.3%	16.6%	3.1%	4.9%	6.1%

Izvor: IDC: Smartphone OS Market Share 2015

1.1 Povijest

Android je nastao 2003. g. u kalifornijskom gradu Palo Alto od strane četvorice samostalnih programera: Andy Rubin, Rich Miner, Nick Sears i Chris White. Temeljna ideja je bila napraviti pametne mobilne uređaje s naglaskom na lokaciji korisnika.

[1] Jezgra operativnog sustava koja spaja aplikacije s hardwareom

U kolovozu 2005. g. Android je postao podružnica tvrtke *Google Inc*, na čijem čelu su ostali njegovi utemeljitelji. Ovim potezom *Google* je uspješno proširio svoje djelovanje na tržište pametne mobilne uređaje. *Android-om* upravlja *Open handset Alliance*, grupa od osamdesetak tehnoloških kompanija među kojima su *Google, HTC, T-mobile* i drugi. Cilj ovog konzorcija je ubrzati inovacije na području mobilnih operativnih sustava s naglaskom na javnu dostupnost koda. Javnom dostupnošću koda želi se potrošačima ponuditi bogatije, jeftinije i bolje iskustvo [Gargenta 2011, 3-4 str.].

1.2 Verzije

Evolucijski napredak Androida je zaista izvanredan i jedinstven te, za razliku od *iOS*-a koji je u principu ostao isti od 2007. godine, doživljava funkcionalni i dizajnerski preporod. Od primitivnih verzija poput 1.6 ili 2.1 do revolucionarnih 4.0 i 4.1 (4.2).

Google je poznat po tome što Android verzijama daje imena po slatkišima (developer.android 2016)

- 1.0 (rujan 2008.) – Web preglednik, podrška za kamere, Google usluge (pretraga, karte, sinkronizacija, dopisivanje), reprodukcija multimedije
- 1.1 (veljača 2009.) – poboljšane karte, prikaz i skrivanje numeričke tipkovnice, spremanje privitaka elektronske pošte
- 1.5 (*Cupcake*, travanj 2009.) – prva verzija nazvana po desertu; podrška za video, početni zaslon, widgeti8, copy-paste operacije, slike kontakata, animirane tranzicije, automatska rotacija zaslona
- 1.6 (*Donut*, rujan 2009.) – poboljšana podrška za gestikulacije s više prstiju, integrirana aplikacija za kameru i galeriju slika
- 2.0 (*Éclair*, listopad 2009.) – podrška za više Google računa, Bluetooth 2.1, Microso*ft Exchange i više veličina ekrana te pretraživanje SMS i MMS poruka*
- 2.2 *(Froyo*, svibanj 2010.) – poboljšana brzina i korištenje memorije, novi JavaScript engine za Chrome preglednik, USB dijeljenje mrežne veze i Wi-Fi hotspot
- 2.3 (*Gingerbread*, prosinac 2010.) – posljednja verzija namijenjena

isključivo mobilnim telefonima; poboljšano oslobađanje memorije, audio i video izvođenje, copy-paste poboljšanja i NFC9[2] podrška

- 3.0 (*Honeycomb*, veljača 2011.) – optimiziran za tablet uređaje; Fragments API i akcijska traka za modernizaciju aplikacija
- 4.0 (*Ice Cream Sandwich*, listopad 2011.) – ujedinjenje telefonskih i tabletnih SDK-ova; poboljšan video i omogućeno mijenjanje postavki pokretača programa
- 4.1 (*Jelly Bean*, veljača 2012.) – prvi od tri Jelly Bean izdanja, uveo poboljšanja u brzini izvedbe sučelja
- 4.2 (studeni 2012.) – poboljšanja kamere, više korisnika na tabletima, jedinstven dizajn sučelja početnih ekrana
- 4.3 (srpanj 2013.) – smanjena potrošnja Bluetootha, bolja grafika za video igre
- 4.4 (*KitKat*, listopad 2013.) – bolje upravljanje memorijom i energijom, bolja podrška za NFC i upravljanje spremljenim podacima, razna SMS i multimedijska poboljšanja
- 5.0 (*Lollipop*, studeni 2014.) – material design – novi način dizajniranja sučelja, redizajnirane notifikacije, ART runtime zamijenio Dalvik10[3] [Developer Android, 2016].

[2] eng. Near Field Communication – mogućnost međusobne komunikacije i prijenosa podataka kompatibilnih uređaja koji se gotovo dodiruju
[3] Virtualni strojevi za izvođenje Java aplikacija

2. Unity 3D

Unity 3D jedan je od vodećih i trenutno najpopularnijih game *engine-a* na svijetu koji se koristi pri razvoju igara za PC, konzole, mobilne uređaje i web stranice. Prvi puta je prikazan 2005. godine, te je bio podržan samo na OS X operativnom sistemu. Sam početak *Unity-a* usko je vezan uz tvrtku *Unity Technologies* koja je osnovana zbog razvitka samog pogona. Tvrtku su 2004. godine osnovali David Helgason, Nicholas Francis i Joachim Ante s ciljem razvitka pogona prihvatljive cijene za uporabu šire javnosti. Fokus tvrtke je "demokratizirati razvitak igara" te ga učiniti pristupačnim ljudima diljem svijeta. 2009. godine *Unity Technologies* lansirao je besplatnu verziju pogona nedugo nakon čega je zabilježen znatan porast registriranih korisnika.Danas ta brojka prelazi milijun registriranih korisnika[Jon Brodkin. 2013].
Unity dolazi u dvije inačice:

1. Osobna inačica (eng. *Personal edition*)
2. Profesionalna inačica (eng. *Professional edition*)

Personal edition je besplatan i dostupan je svakoj osobi ili tvrtki. Može se koristiti u komercijalne svrhe sve dok zarada korisnika nije veća od 100,000 dolara godišnje. Ako je zarada veća od gore navedenog iznosa potrebno je kupiti Professional edition.

Osnovni koncepti Uni*ty3D engine-a* su:

- Aktiva (engl. *assets*)

- Scena

- Objekt igre (engl. *GameObject*)

- Komponenta

- Skripte

- Predlošci (engl. *prefab*)

Aktive (engl. *assets*) su : modeli, teksture, zvuk, video, animacije. Koriste ih: engine za iscrtavanje (engl. *Render engine*), engine za zvuk i video (engl. *Sound/video engine*) i engine za animaciju (engl. *Animation engine*)

Scena- sadrži objekte igre. Scene se koriste za izradu razina igre (engl. *levels*) ili menija. Pri tome svaka scena je posebna razina igre ili meni. Unutar scene gradimo igru postavljajući naše objekte. Kreiranjem igre u više scena postiže se mogućnost raspodjele učitavanja dijelova igre, te mogućnost testiranja određenih dijelova igre zasebno.

Objekti - svaki objekt u igri je objekt igre, u *Unity-u* poznatiji pod izrazom *GameObject*. Objekti igre su spremnici komponenti. Svaki objekt sadrži barem jednu komponentu–*Transform* komponentu. *Transform* komponenta jednostavno daje *engine-u* informaciju o svojoj poziciji u 3D prostoru opisane pomoću X, Y, Z koordinate, te njegovu veličinu i rotaciju.

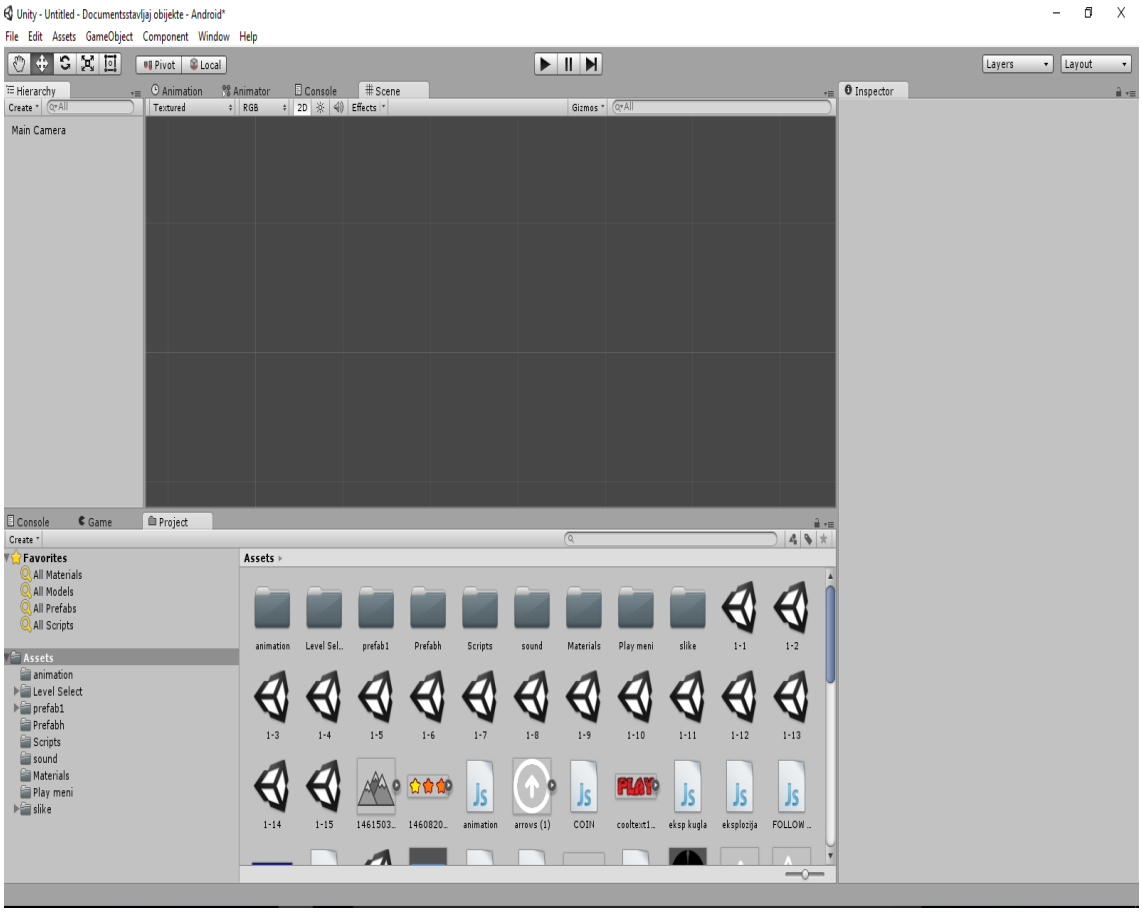
Komponente su funkcionalni dijelovi svakog objekta igre. One dolaze u različitim oblicima. Mogu utjecati na ponašanje objekta koji ih sadrži, definirati pojavu i iscrtavanje ili utjecati na neke druge funkcije objekta unutar igre.

Skripte - Za kreiranje skripti *Unity* podržava programske jezike *JavaScript, C#, Boo.* Skripte su važan i ključan koncept svakog razvoja igre i unutar *Unity3D engine-a* se smatraju komponentama. Skriptama možemo upravljati objektima igre na način da unutar skripte napišemo željeno ponašanje te tu skriptu dodijelimo objektu igre

Prefab - je tip aktive koji je moguće koristiti u više scena te neograničeno mnogo puta unutar jedne scene.

2.1 Sučelje programa *Unity 3D*

Sučelje programa *Unity 3D* može se podijeliti na nekoliko elemenata (slika 1):



Slika 1 Sučelje Unity 3D

Izvor: snimka zaslona autora

1. Preglednik projekta (eng. *Project browser*) - prikazuje datoteke koje se koriste za igru.

2. Preglednik scene (eng. *Scene browser*) - prikazuje trenutno otvorenu scenu

3.Inspektor (eng. *Inspector*) - pokazuje komponente odabranog objekta u sceni

4. Hijerarhija projekta (eng. *Hierarchy*) -popisuje sve elemente koje ste dodali na sceni.

5. Alatna traka (eng. *Tool Bar*) - Alatna traka se sastoji od sedam osnovnih kontrola.

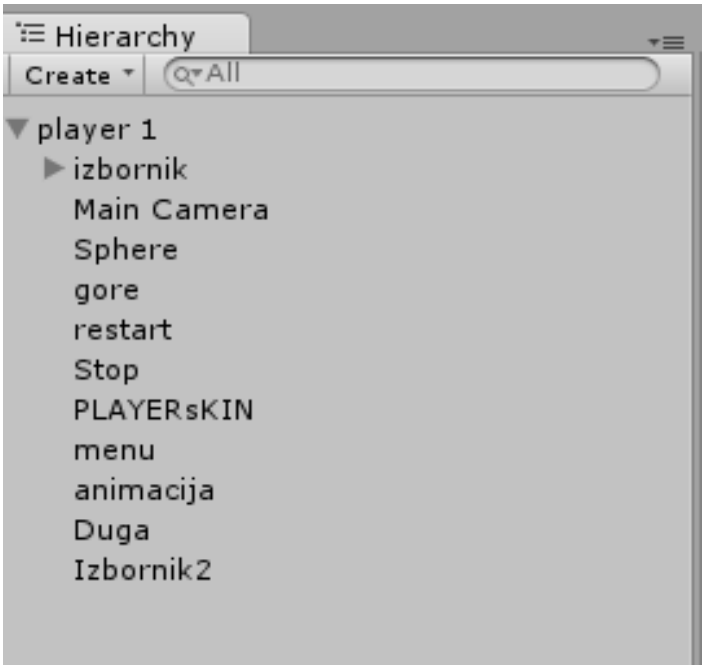
Alatna traka (slika 2) sastoji se od pet osnovnih kontrola, svaka povezana s drugim dijelom uređivača (eng. *Editor*). Alati za transformaciju (1 i 2) služe za

manipuliranje prikazom scene, *Play, Pause* i *Step* tipke (3) služe za prikaz igre, tj. animirane scene, padajući izbornik *Layers* (4) služi za odabir objekta koji se prikazuje u prikazu scene dok padajući izbornik *Layouts* (5) služi za kontrolu svih prikaza.

Slika 2 Alatna traka

Izvor: snimka zaslona autora

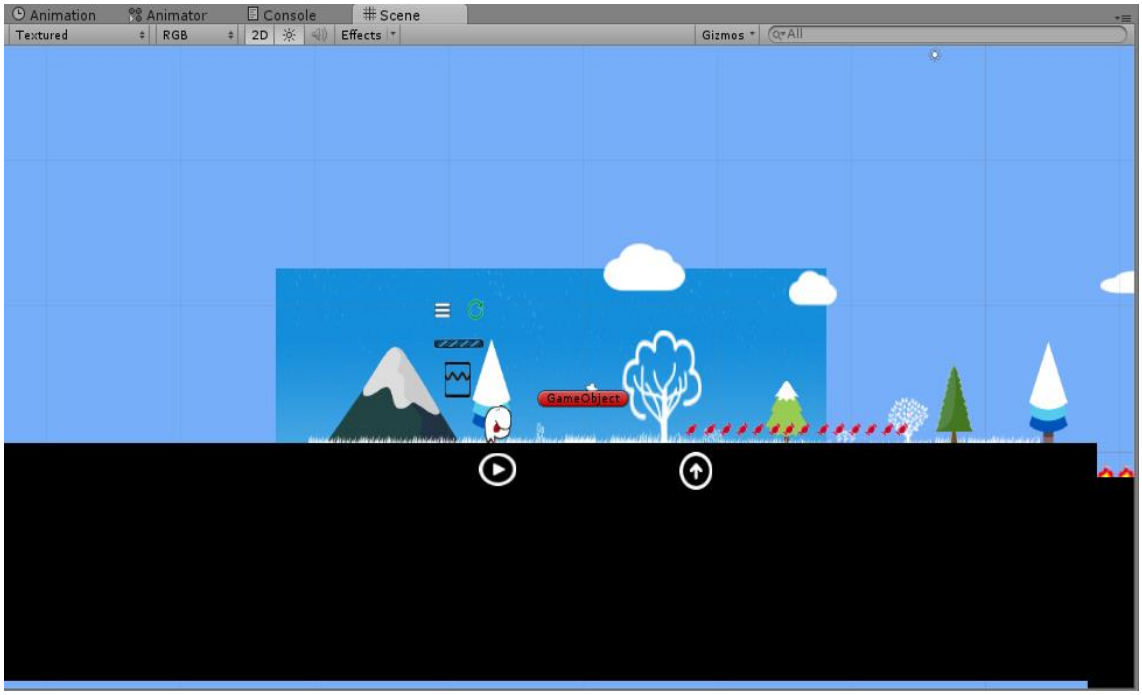
Hijerarhija (slika 3) sadrži sve objekte projekta u trenutnoj sceni. *Unity 3D* koristi koncept nazvan roditeljstvo (eng. *Parenting*) koji koristimo kada želimo da neki objekt naslijedi svojstva drugog objekta. Tada objekt koji nasljeđuje nazivamo dijete (eng. *Child*) dok je početni objekt roditelj (eng. *Parent*). Objekt-roditelj može imati više objekata - djece, dok objekt - dijete može imati i svoje objekte - djecu.



Slika 3 hijerarhija projekta

Izvor: snimka zaslona autora

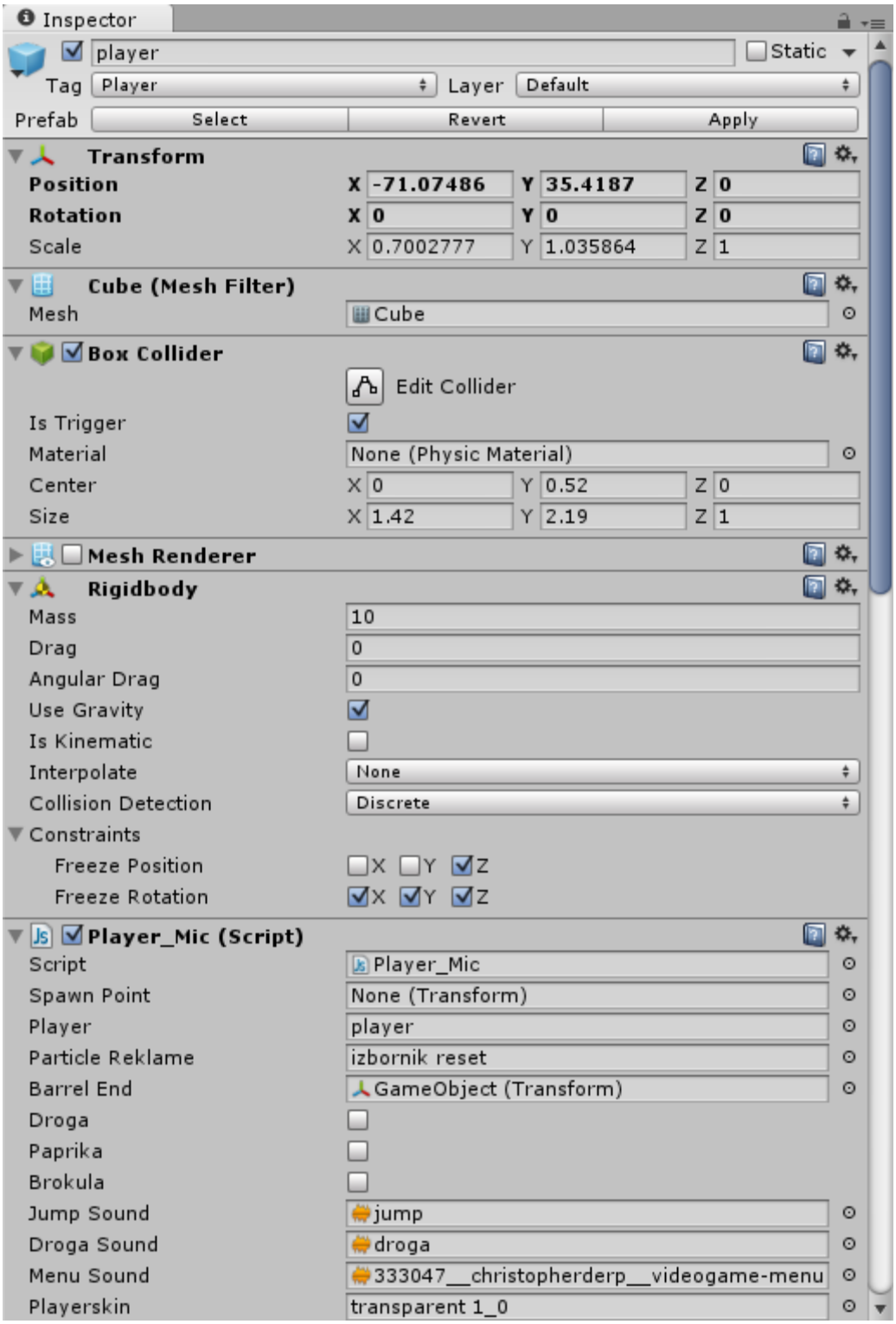
Preglednik scene (slika 4) je mjesto gdje se igra izrađuje. Prikazuje koje objekte imamo u igri i gdje su u prostoru u odnosu jedan na drugog.



Slika 4 Preglednik scene

Izvor: snimka zaslona autora

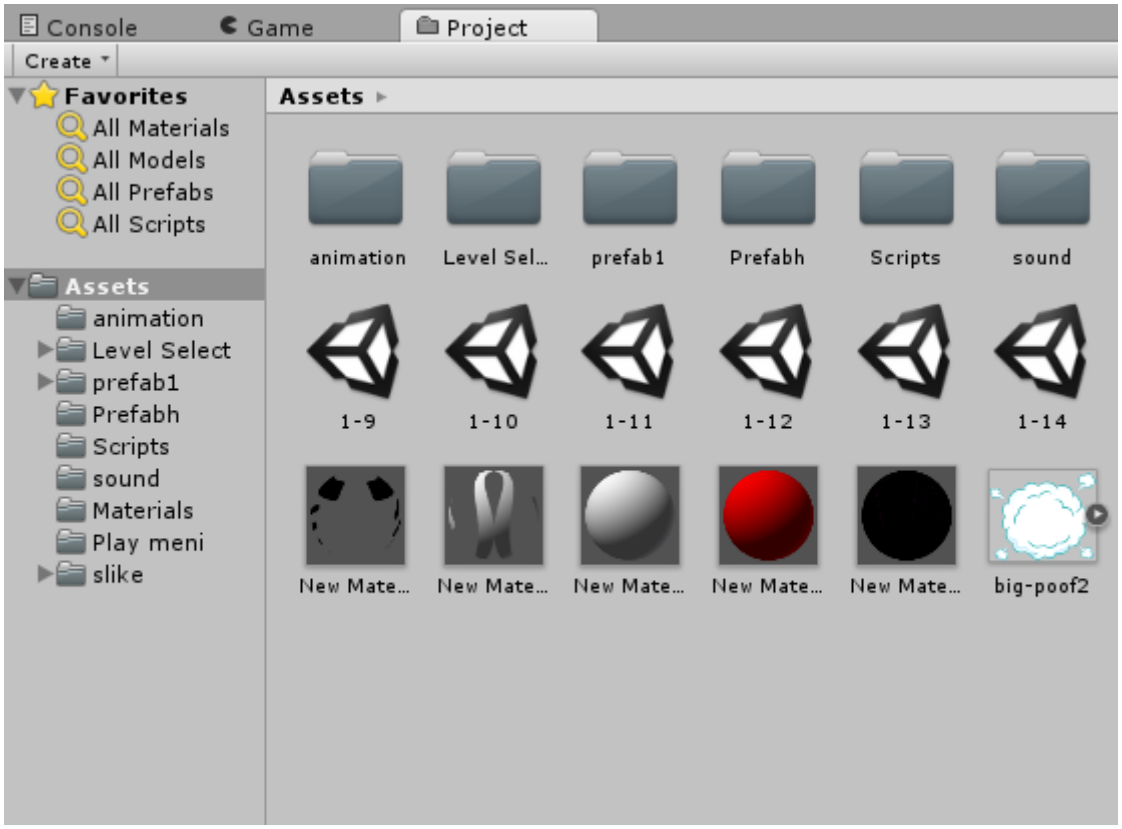
Inspector (slika 5) je mjesto gdje možete prilagoditi aspekte svakog elementa koji je na sceni. Kad se odabere objekt u prozoru hijerarhije ili dvostruko klikne na objekt u prozoru scene taj će objekt pokazati svoje atribute na inspektor ploči te u njemu možemo upravljati tim atributima.



Slika 5: Inspektor

Izvor:snimka zaslona autora

Posljednji element sučelja je preglednik projekta (slika 6). U pregledniku korisnik može pristupiti svim objektima i ostalim datotekama koje su sadržane u projektu [Sue Blackman 2011,50-57 str.].



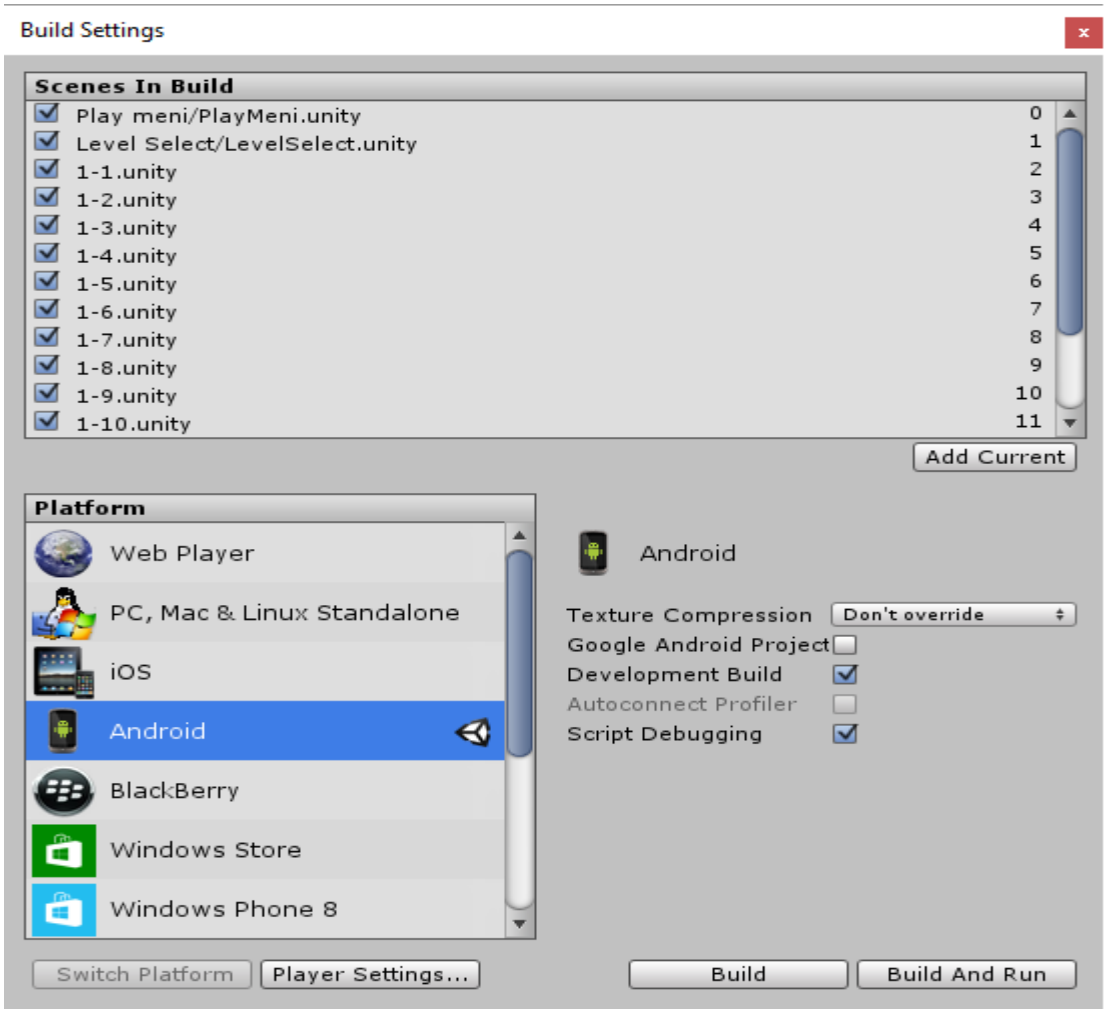
Slika 6: Preglednik projekta

Izvor: snimka zaslona autora

2.2 Izlazni formati

Unity 3D ima nekoliko različitih izlaznih formata (slika 7) pomoću kojih se igra može pokretati na raznim uređajima. Ako se igra želi pokrenuti u internetskom pregledniku postoje opcije da igra bude izrađena u formatu *Unity Web Player* ili *WebGL*. Ako igru želimo zaigrati na osobnom računalu onda ćemo igru izraditi u izvršnoj datoteci (eng. *executable*) formatu. Igre se također mogu izraditi za pametne telefone, pa se tako može dobiti format .apk koji služi za instalaciju igre na Android uređaju, odnosno .ipa format ako igru želimo pokrenuti na uređaju koji ima IOS operativni sustav. U postavkama za izradu igre također se mogu odabrati formati za konzole (Play Station, X-BOX). Ako posjedujete pametni televizor

marke Samsung, postoji i opcija da se igra izradi u formatu koji je podržan od strane Samsungovih pametnih televizora.



Slika 7: Izbornik formata

Izvor: snimka zaslona autora

2.3 Prednosti i nedostaci

Unity je najbrže rastući 3D game *engine* na svijetu i u samom startu je ciljano rađen za manje timove, tj. indie segment tržišta, a stalni razvoj ohrabruje tvrdnju da bi ubrzo mogao prestići i naprednije *3D engine* po pitanju funkcionalnosti

Prednosti:

- jednostavniji za korištenje od usporedivih 3D enginea
- prikladan za timove

- podržava bitne platforme: PC, Mac, Linux, Xbox, Playstation, Android, iOS, itd.
- veliko integrirano tržište gotovih grafika, modela i skripta
- daleko jeftiniji od konkurencije
- besplatan je i dostupan svima (osim ako prihodi od korištenja programa iznose više od 100 000 dolara godišnje - onda se obavezno mora kupiti profesionalnu inačicu)
- postoji jako puno dokumentacije pomoću koje se program lakše savladava

Nedostaci:

- zahtijeva znanje programiranja (Javascript, C# ili Boo)
- “zatvoren” izvorni kod *engine-a*
- loša grafička svojstva u odnosu na neke druge game *engine* (npr. Unreal Engine)

3. Proces izrade igre

Tema opisuje procese i probleme na koje se nailazi prilikom izrade i plasiranja virtualne igre. Virtualne igre trenutno su industrija jača od filmske i glazbene industrije zajedno, stoga nas ne čudi da je danas znanje izrade igara vrlo traženo. Izrada virtualne igre je kompleksan rad i sastoji se od niza procesa. Generalna podjela tih procesa bila bi:

1. izrada ideje

2. izrada same igre

3. prodaja igre

Ova tri područja raščlanjuju se u još manje segmente i opisuje se što je potrebno znati za svaki pojedini proces, kako najlakše naučiti potrebno i što se dobije nakon odrađenog procesa. Osnovni problemi se javljaju između svakog navedenog procesa.

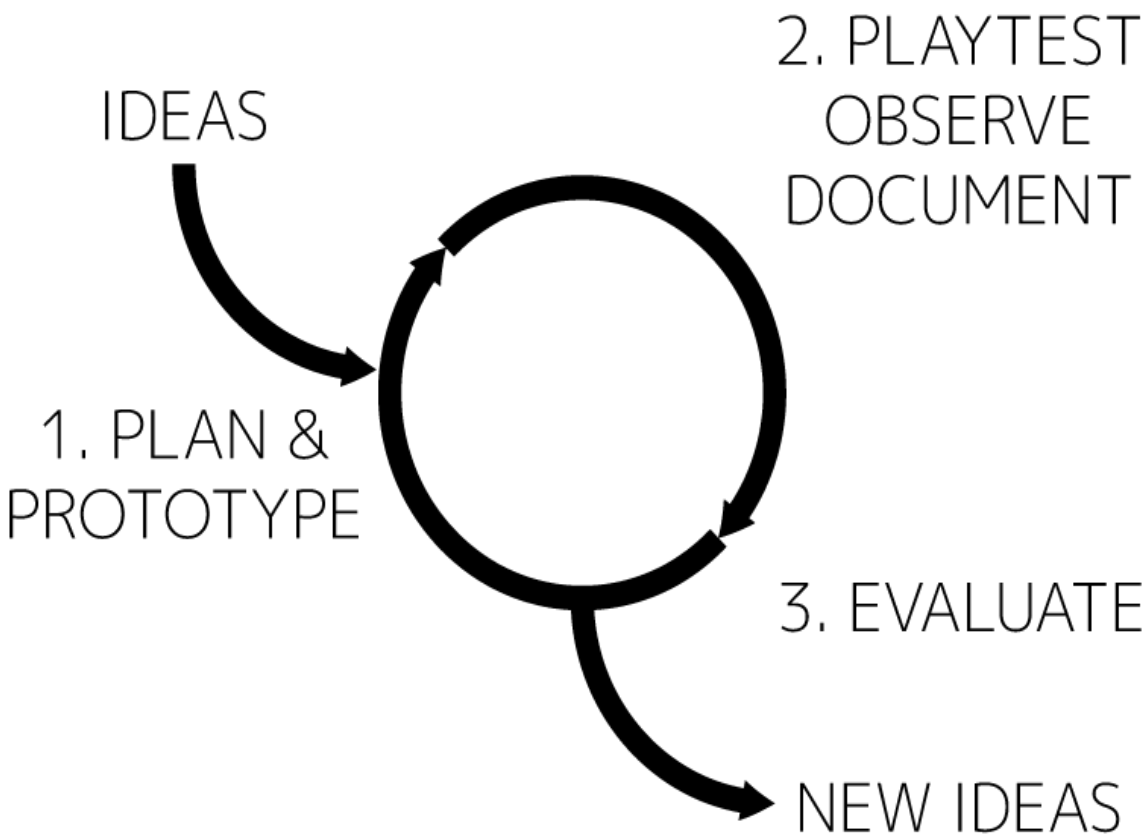
Nije potrebno puno znanja kako bi se napravila ideja igre, no problem koji se javlja nakon napravljene ideje je potrebno znanje da se realizira ta ideja. To je prvi glavni problem osobama koji se upuštaju u izradu igre. Drugi glavni problem nastaje nakon što se ideja realizira, a taj problem je plasiranje i prodaja igre. Tu se ulazi u drugu dimenziju znanja koja je opisana u radu. Cilj rada je obrazovati osobe željne znanja na području razvoja igara i dati im savjete koji su vrlo potrebni u tom poslu. Problematika rada je u tome što se nerijetko mladi ljudi odlučuju za izradu virtualne igre obično u timovima, no takvi projekti u velikoj većini slučajeva propadnu zbog manjka osnovnog znanja o takvim projektima ili zbog vrednovanja pojedinih procesa manje bitnim od drugih zbog njihove jednostavnosti.

4. Izrada ideje

Prvi korak u ovom procesu je validacija ideje na tržištu. Kada smo osmislili neku novu igru, to jest ideju za igru, treba provjeriti da li takvih igara već ima. Ako postoje, koliko su uspješne? Ako su uspješne, možemo li postići istu ili veću razinu kvalitete kao naša konkurencija? Ako pak takve igre ne postoje, zašto ne postoje? Da li se nitko nije sjetio toga ili nema interesa za takav tip igre? Tek će odgovori na ova pitanja ukazati na to da li su nam ideje dobre ili loše.

Drugi korak je razrada ideje gdje neku inicijalnu skicu ideje širimo sve dok ne vidimo manje-više cijelu igru raspisanu do zadnjih detalja. Ovdje je vrlo bitno dijeliti ideju sa što više ljudi – posebice s ciljanim igračima.

Na Slici 8 se vidi kako se svaka ideja treba dorađivati nizom stalnih planiranja, dorađivanja i komuniciranja, ali i jednostavno puštanjem ideje te prelaskom na novu ako ona jednostavno ne funkcionira onako kako smo zamislili.



Slika 8: Izrada ideje

Izvor: malmo james too.prototyping-advice

Ideja ove igrice je da igrač dovede svoje *Candy Freak* čudovište sigurno nazad kući da bi mogao otključati sljedeću stazu pritom skupljajući sto više bombona koji vam pomažu pri stvaranju objekata pomoću kojih rješavate prepreke. Svaki nivo ima određene objekte koji otežavaju dolazak do cilja odnosno do kuće. Kako napredujete prema višem nivou prepreke su teže, pa samim time igra od vas traži više koncentracije i mozganja.

5. Izrada igre

Proces izrade igre može se podijeliti u nekoliko koraka:

1.Prikupljanje i izrada 2D modela – pri ovom koraku prikupljaju se ili modeliraju svi objekti koji će biti potrebni za izradu nivoa odnosno mape na kojoj će se igra odvijati

2.Oživljavanje modela - programiraju se i „oživljavaju“ objekti.

3.Izrada mape – pri ovom koraku izrađuje se okruženje u kojem će se igra odvijati

4.izrada sučelja - ovo je korak u kojem se dizajnira grafičko sučelje odnosno GUI (eng. *Graphical user interface*)

5. Podešavanje grafičkih svojstava - na kraju izrade igre dodaju se efekti kako bi igra izgledala realnije i ljepše

5.1 Prikupljanje i izrada 2D modela

Svaka igra mora imati objekte odnosno 2D modele od kojih je ona napravljena te koji oblikuju izgled same igre.

Za ovu igru bilo je potrebno sakupiti sljedeće modele:

1.Glavni lik - kako će izgledati i animirati glavni lik u igri

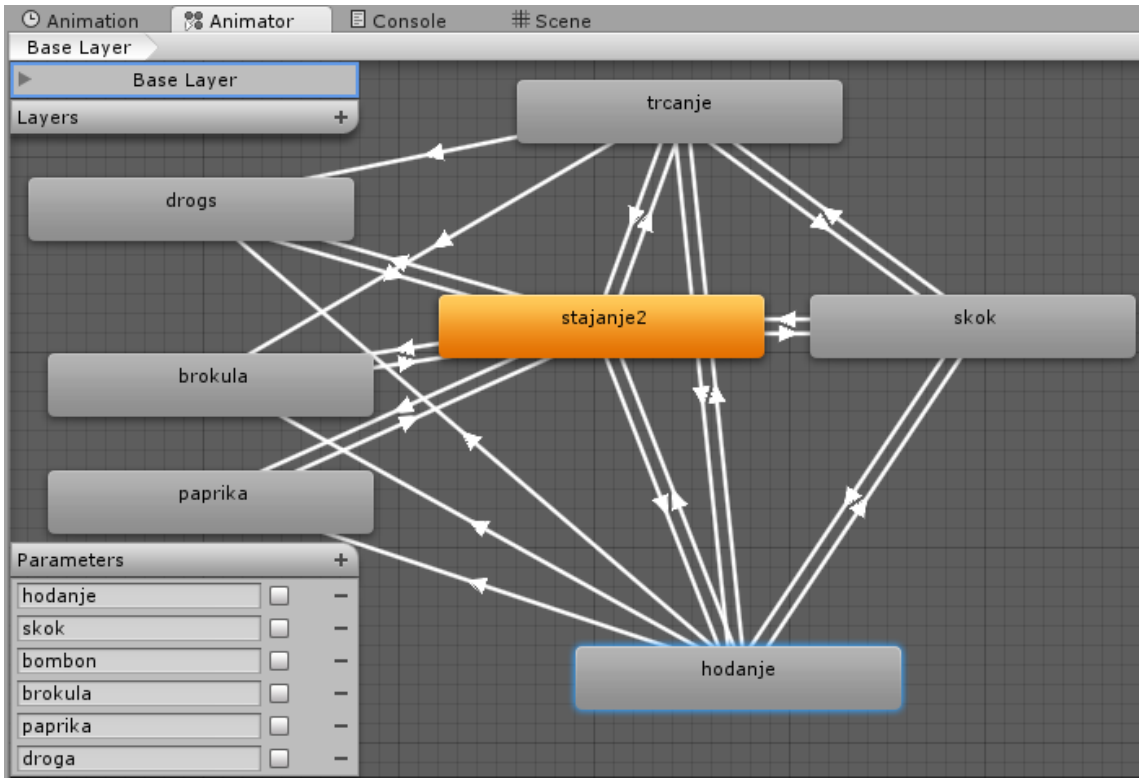
2.Modeli za izradu mape - ograde, drveća, protivnici i slično

U igrama su teksture jedan od najvažnijih čimbenika koji utječu na to kako igrač doživljava igru. To je posebno važno u 2D igrama, gdje nema potrebe za iznimno detaljnim modelima, već se svi objekti sastoje od teksturi ravnih površina. Drugim riječima, teksture određuju izgled igre. Model i animacija glavnog lika (slika 9) je napravljen pomoću *paint-a* i *photoshop-a* dok su modeli za izradu mape (drveća, pozadina) preuzeta s interneta [*Mobile Game Graphics].*

Slika 9 Animacija glavnog lika

Izvor: snimka zaslona autora

Za animiranje modela mora se konstruirati *Animation Controller* (slika 10) koji služi za podešavanje animacija te upravljanje animacijama unutar skripte. *Animation Controller* koristi automat stanja u kojem pojedina stanja predstavljaju osnovne animacije (eng. *animation clips*). U automatu stanja povezuju se animacije kako bi prijelaz među njima bio što prirodniji.



Slika 10 Prikaz automata stanja

Izvor: snimka zaslona autora

Skripte koristimo za upravljanje animacijama unutar scene. Unutar skripte možemo definirati, tj. kreirati događaje koji se izvode kada napravimo određenu akciju.

U sljedećem primjeru koda vidimo kada se pomoću skripti pokreće određena animacija, to jest pritiskom na skok mijenja se tvrdnja iz netočne u točnu čime se u animatoru pokreće određena animacija - u ovom slučaju skok.

```
if(MisKliknutGORE2 == true){
anim.SetBool("skok",true);
}
if(MisKliknutGORE2 == false){
anim.SetBool("skok",false);
}
```

Programski kod 1.Pokretanje animacije

Izvor:Autor

5.2 Oživljavanje modela

U izradi svake igre programeru mora biti poznat cilj igre, kao i tok događaja, kako bi mogao apstraktne pojmove i njihova svojstva implementirati u objekte koji će, kao jednostavne i smislene cjeline, simulirati zadano ponašanje i svojstva. U ovoj igri naša logika počinje od igrača, i to bi mogao biti prvi objekt za implementaciju. Igrač treba vidjeti neku prezentaciju sebe na ekranu (budući da je igra dvodimenzionalna), kako bi znali gdje se nalazi, kuda ide i što je u njegovoj neposrednoj okolini. Također, trebat ćemo implementirati kretanje i druge radnje koje bi igrač trebao kontrolirati kako bi ispunio cilj.

U ovoj igri igrač ima pet glavne karakteristike (slika 11) a to su:

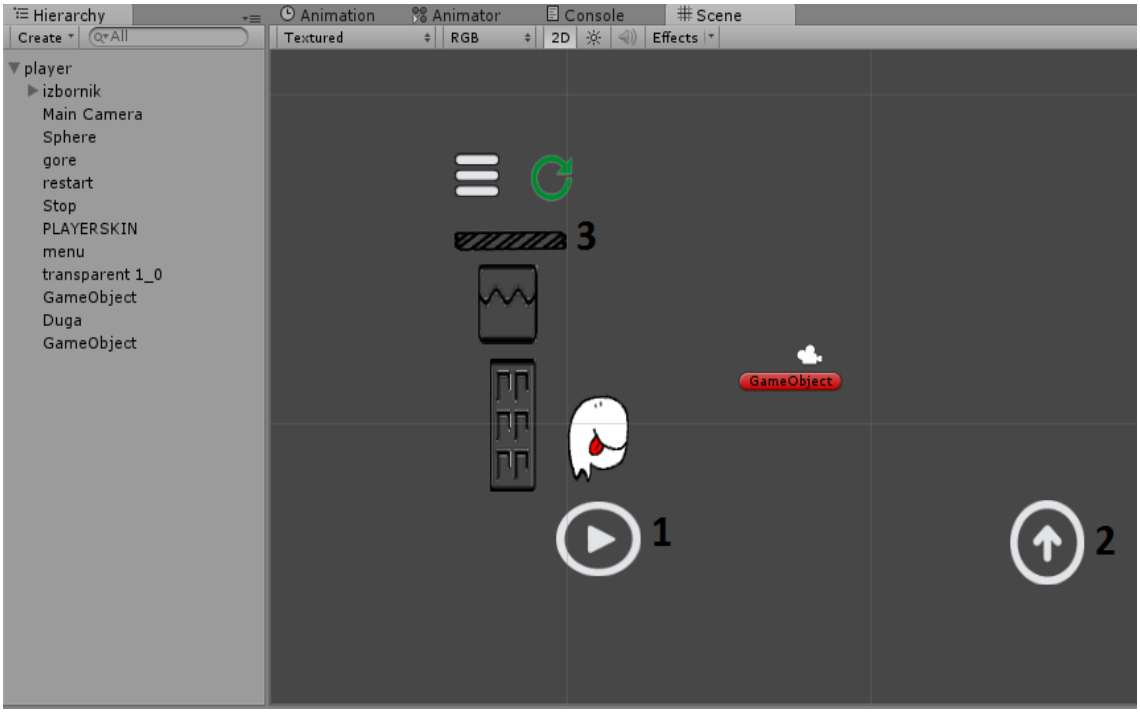
1.horizontalno hodanje samo u smjeru plus x

2.skakanje u smjeru plus y

3.stvaranje objekata

4.uništenje objekata

5.interakcija s drugim objektima u igri

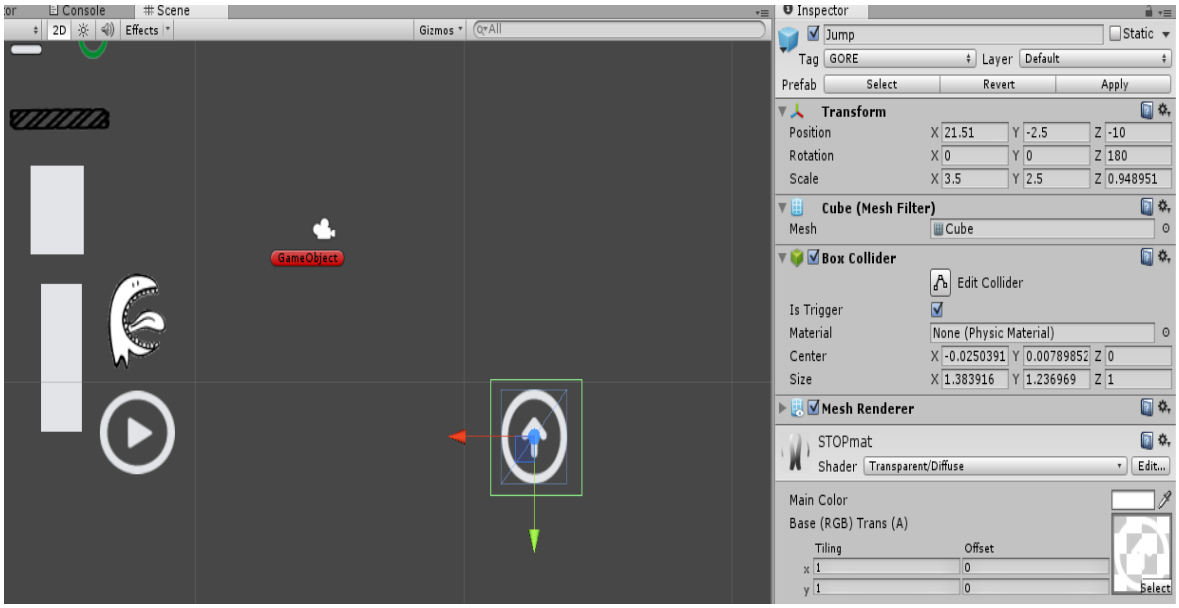


Slika 11 Funkcije igrača

Izvor: snimka zaslona autora

Da bi to mogli ostvariti prvo sto trebamo je napraviti igrača koji će na sebe imati povezane ostale objekte (eng. *parent*) kao što su kamera, dugme za skakanje, objekte za stvaranje itd., koji se zovu *child* posto su povezani s drugim objektom.

Nakon sto smo to napravili svaki od tih objekata mora dobiti svoju oznaku ili *tag* koji će biti jedinstven tom objektu, da se može prepoznati pri pritisku na njega. Kao sto se vidi na slici 12. dugme za skok ima *tag* GORE.



Slika 12 Dugme za skok

Izvor: snimka zaslona autora

5.2.1 Skripta

Da bi se igrač mogao kretati i reagirati na pritisak dugmeta, moramo napraviti skriptu koja će njime upravljati. *Unity* sadrži ugrađen IDE[4] za kreiranje skripti nazvan *Monodevelop*, no skripte se mogu kreirati u bilo kojem razvojnom okruženju koje podržava jezik u kojem se one kreiraju. Najjednostavniji način je korištenje samog *Monodevelop* alata kako je on već integriran i prilagođen *Unity*-u. U ovom radu će se koristit *JavaScript.*

Važno je spomenuti 3 osnovne funkcije koje sadrži skripta :

- Start – koristi se za inicijalizaciju varijabli prilikom stvaranja objekta kojem je skripta pridružena

- Update – sadrži svu logiku koja se izvršava u stvarnom vremenu Poziva se jednom pri svakom iscrtavanju (engl. *per frame*).

-OnTriggerEnter - aktivira se kad se dva objekta dodirnu

[4] IDE-Integrirano razvojno okruženje

Unutar skripti moći ćemo pristupiti svim elementima objekta igre te po potrebi mijenjati ili upravljati njihovim sadržajem. Također, važno je spomenuti kako su skripte igračevo „sučelje“ s igrom jer su odgovorne za upravljanje ponašanjem objekta u ovisnosti na ulaz (*input*) putem dodira na ekran.

U nastavku vidimo primjer koda za pritisak na dugme skok koje ima *tag* GORE. Pritiskom na dugme varijabla MisKliknutGORE postaje istina pri čemu će igrač, ako je na tlu, skočiti.

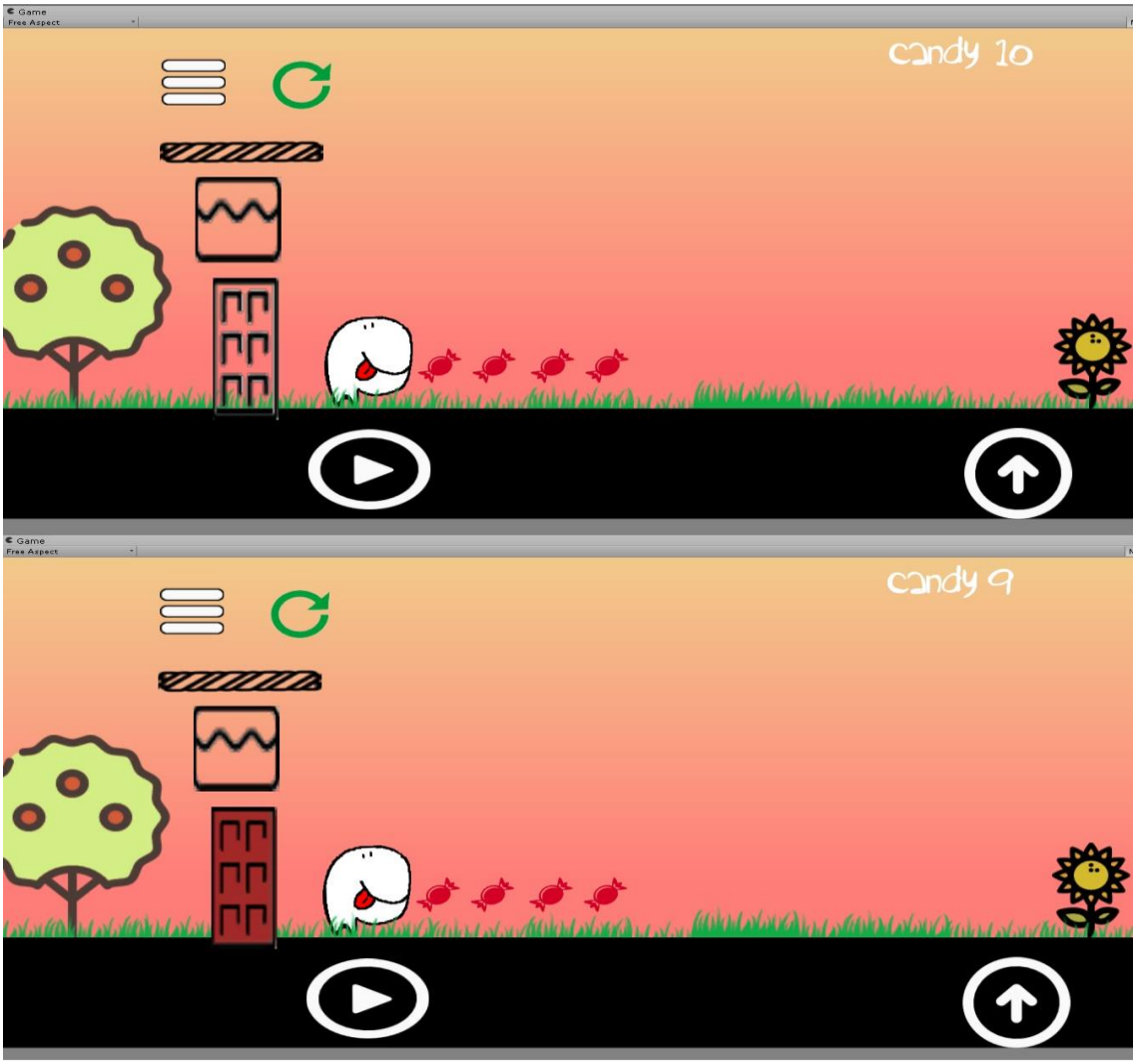
```
private var mousePoint : Vector3;
var Jump : float = 2.0;
var MisKliknutGORE: boolean = false;
var diraPod : boolean = false;
var MisKliknut : boolean = false;
var jumpSound:AudioClip;

function Update () {

var rayHit : RaycastHit;

if(Physics.Raycast(Camera.main.ScreenPointToRay(Input.mouse
Position), rayHit)) {

mousePoint = rayHit.point;

if(Input.GetMouseButtonDown(0) &&  rayHit.collider.tag ==
"GORE" && diraPod == true ){ MisKliknutGORE = true;
MisKliknut = true;  }
}
if(MisKliknutGORE == true  ){
AudioSource.PlayClipAtPoint(jumpSound, transform.position);
rigidbody.velocity.y = Jump;

MisKliknutGORE = false;

diraPod = false;

MisKliknut = false;

}}
```

Programski kod 2.Skok igrača

Izvor:Autor

Jedan od bitnijih elemenata ove igre je taj da igrač bilo gdje u prostoru može sam stvarati objekte pomoću kojih rješava prepreke kako bi mogao stići do cilja.

Elementi se stvaraju pritiskom na jedan od tri ponuđena elementa (lizalica, torta, čokolada), te kada je element označen moguće ga je stvoriti u prostoru uz uvjet da ima sakupljen bombon (*score*) koji mu to omogućava. Na slici 13 vidimo kako se prilikom odabira objekta tekstura mijenja kako bi korisnik znao šta je odabrao (tag. stup Izabran).



Slika 13. Prije i poslije Izabira čokolade u igri

Izvor: snimka zaslona autora

Sljedeći kod prikazuje skriptu pomoću koje je riješen problem. Kada korisnik odabere stup mijenja se njegova tekstura te se varijabla stup1 mijenja u istinu čime se kasnije, ako korisnik pritisne na prazno polje uz uvjet da ima *score*, veći ili jednak, stvara jedan objekt.

```
function Update () {
if (Input.GetMouseButtonDown(0) && rayHit.collider.tag ==
"stupIzabran") {

     stup1 = true;
     kutija1 = false;
     daska1 = false;

objectStup.renderer.material.mainTexture = textureStup1;
objectKutija.renderer.material.mainTexture =
textureKutija0;
objectDaska.renderer.material.mainTexture = textureDaska0;

AudioSource.PlayClipAtPoint(menuSound, transform.position);
                }
if (Input.GetMouseButtonDown(0) && stup1 == true &&
MisKliknut ==false  && imClickedOn == false && score >=1){
               mousePoint.z = 2;

     Instantiate(stup,mousePoint,transform.rotation);
               Instantiate(particle, mousePoint,
transform.rotation);
                score -=1;
               }}
```

Programski kod 3.Odabir objekta

Izvor:Autor

5.3 Objekti okoline

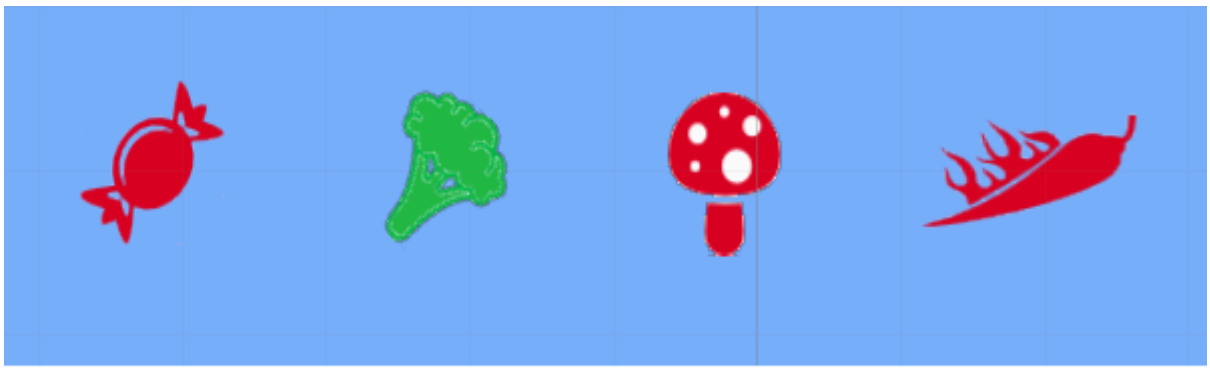
Osim glavnog lika potrebni su objekti okoline. Objektima okoline smatraju se svi objekti koje igrač ne može pokupiti. Ovakvi objekti mogu biti i statični i dinamički, ovisno o konkretnom objektu. Namjena ovakvog razreda je popunjavanje scene dodatnim izazovima ili pomaganje igraču na neki način. Pod objekte okoline spadaju šiljci, mostovi i trampolin, te je sa svim vrstama omogućena interakcija. Najjednostavniji objekti skupine su šiljci, statični predmeti koji će ozlijediti igrača.

5.4 Objekti za sakupljanje

U igri se nalazi nekoliko vrsta objekata koje igrač može pokupiti (slika 14). Jednom kad igrač pokupi nešto, taj objekt se mora prvo maknuti sa scene i zatim pokrenuti određenu animaciju i utjecaj na igrača. Na objekte koje igrač sakuplja ne djeluje gravitacija, s ciljem postavljanja nagrada za igrača na teže dohvativa mjesta.

Postoje četiri vrste objekata koji se mogu naći u igri i svaki od njih definira vlastitu funkcionalnost nakon što se objekt pokupi. Najčešći objekti koje će igrač pronalaziti su bomboni. Nakon sakupljanja, igračev rezultat (*score*) povećava se za vrijednost pokupljenog bombona.

Objekti koji se još mogu pronaći u igri su: paprika koja povećava igračevu brzinu te uzima mogućnost zaustavljanja igrača, brokula koja uzima bodove igraču i na kraju gljiva koja uzima mogućnost sakupljanja bombona.



Slika 14.Objekti za skupljanje

Izvor: snimka zaslona autora

Kada igrač dođe u doticaj s paprikom objekt se uništava, to jest miče se sa scene i igraču se dodaje brzina plus 8 po osi x u određenom vremenskom periodu. Kod je prikazan u nastavku:

```
var paprika : boolean = false;
var timerPaprika: float;

function OnTriggerEnter (other : Collider) {
     if (other.gameObject.tag == "paprika"){
          paprika = true;
               Destroy(other.gameObject);
```

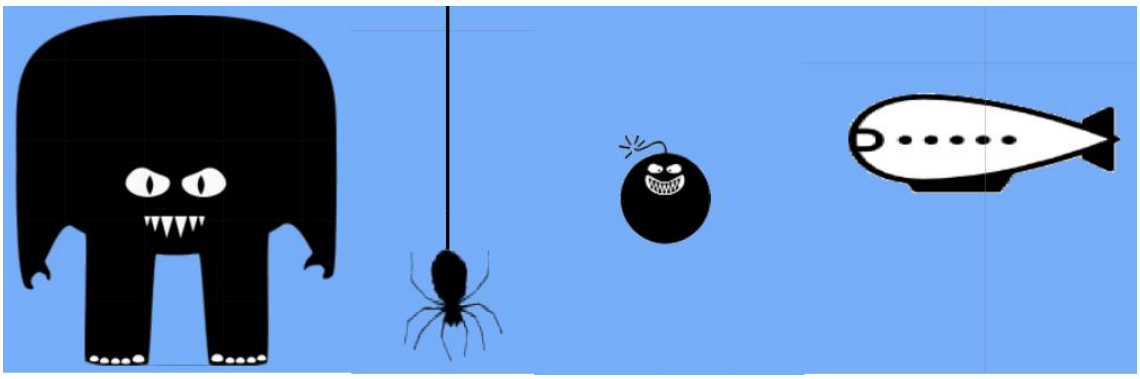
```
                Speed +=8;}}
function Update () {
 if(paprika == true){
     timerPaprika+= Time.deltaTime;
     if(timerPaprika > 3){
paprika = false;
timerPaprika = 0;
Speed -=8; }
}}
```

Programski kod 4.Skupljanje paprike

Izvor:Autor

5.5 Neprijatelji

U igri postoji i razred objekata koji predstavljaju opasnost za igrača (Slika 15). Karakteristika tih objekata je kretanje neovisno od akcija igrača. Trenutno je implementirano više vrsta neprijatelja: čudovišta, pauci, bombe i Zeppelin. Sve vrste će oduzeti život igraču.



Slika 15.Objekti neprijatelji.

Izvor: snimka zaslona autora

Čudovišta su prva vrsta neprijatelja koja je bila dodana u igru. Oni se ne miču te ako ih igrač dotakne, umre. Način na koji ih igrač prelazi je pritisak na njih koji ih briše iz igre. U nastavku je skripta pomoću koje se to dešava.

```
if (Input.GetMouseButtonDown(0)  &&  rayHit.collider.tag ==
"smrt" )

{

 Destroy(rayHit.collider.gameObject);
```

```
Instantiate(particleBlod, mousePoint, transform.rotation);

}
```

Programski kod 5.uništavanje protivnika

Izvor:Autor

Pauci su objekti koji se pomiču vertikalno ali za razliku od čudovišta pritiskom na njih igrač ih neće maknuti iz igre.

Bombe u doticaju s tlom eksplodiraju pri čemu uništavaju sebe i bilo što u njihovoj blizini osim tla. Kod je prikazan u nastavku.

```
#pragma strict

public var particle: GameObject;

public var barrelEnd: Transform;

private var dira : boolean  = false;

function OnTriggerStay (other : Collider){

 if (other.gameObject.tag == "Pod" || other.gameObject.tag
== "Player" || other.gameObject.tag == "Pod2"  ){

     dira = true;

 }}

function Update () {

if(dira == true){

Instantiate(particle,
barrelEnd.position,transform.rotation);

     Destroy(gameObject );

}}
```

Programski kod 6.Bomba

Izvor:Autor

Zeppelin su objekti koji se miču negativno po osi x te stvaraju bombe.

5.6 Izrada nivoa

Izrada nivoa korak je izrade igre koji uzima najviše vremena jer oni moraju biti pomno planirani te nakon toga svi moraju biti testirani kako bi bili sigurni da se svaki nivo može proći. Također, mora se obratiti pozornost na to da se uklone sve greške pomoću kojih bi igrač mogao stići do cilja nekom drugom putanjom.

Ova igra sastoji se od 15 nivoa koji su podijeljeni u nekoliko težinskih skupina:

- **jednostavni nivoi** (od 1 do 5) - služe da bi se igrač upoznao s mehanikama igre

-srednja težina (od 6 do 10) - za ove nivoe od igrača se očekuje visoka koncentracija te razumijevanje igre

-**teška razina** (od 11 do 15) - traži se od igrača visoko razumijevanje igre i logika.

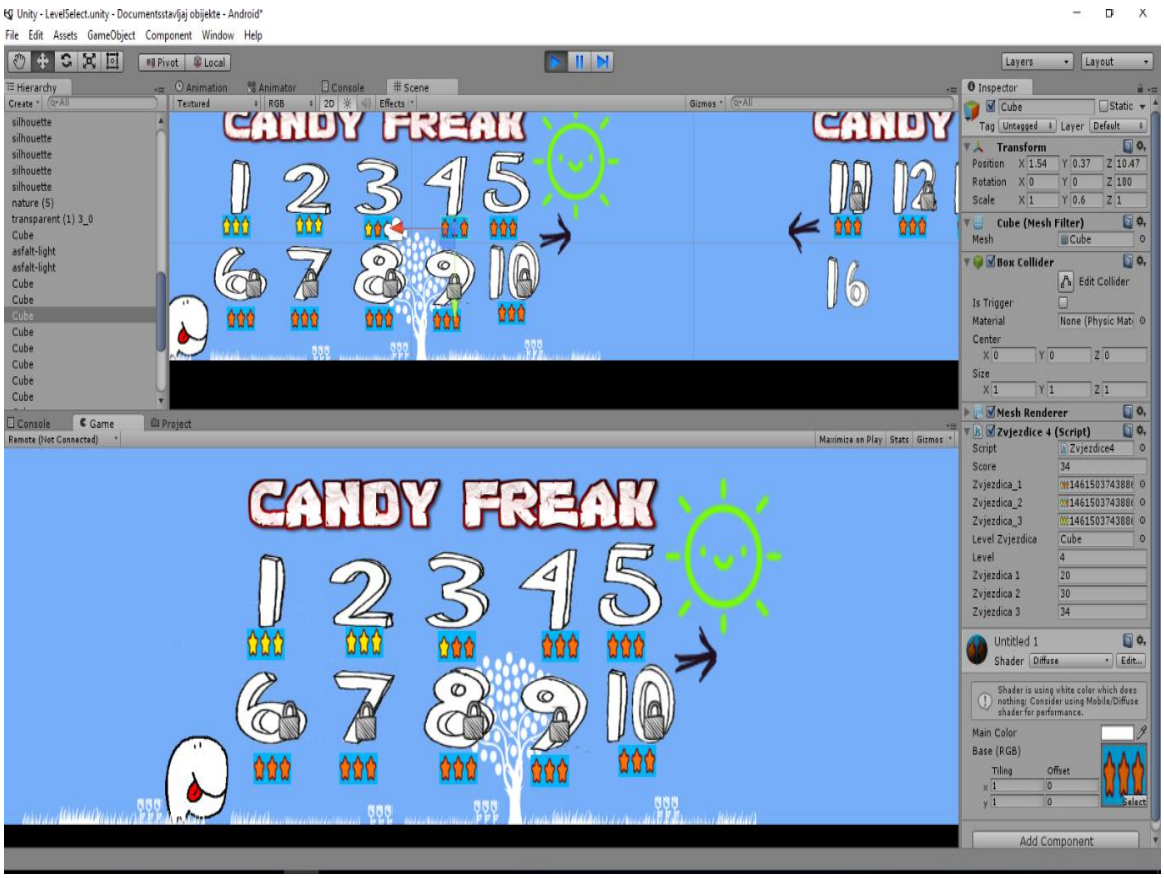
Svaki nivo sastoji se od startne pozicije (eng. *Starting Position*) te od završne točke (eng. *Target Spot*) što je u ovom slučaju kuća (slika 16).

Slika 16.Završna točka

Izvor: snimka zaslona autora

Svaki nivo mora biti različit od drugog te postaviti neke nove prepreke igraču kako bi na drugačiji način došao do cilja. Dolaskom do cilja igraču se računa koliko mu je bombona ostalo pri završetku igre te se time dodjeljuju zvjezdice (prikazano

slikom 17-od niti jedne zvjezdice što znači da je došao do cilja s vrlo malenim brojem bombona do tri što predstavlja vrlo uspješan dolazak do cilja s velikim brojem skupljenih bombona).



Slika 17.Dodjela zvjezdica

Izvor: snimka zaslona autora

5.7 Glavni izbornik

Izbornik staza jedan je od bitnijih dijelova igre pošto na njemu igrač vidi napredak kroz igru i sprema se njegov rezultat.

Glavne funkcije izbornika su:

1.Ponudi igraču level koji može odabrati

2.Sprema napredak kroz igru

3.Pokazuje uspješnost prelaska svakog levela.

Napredak igrača se pamti pomoću *PlayerPrefs.SetInt* ("Lev*elUnlock",*level); u koji se sprema napredak igrača, to jest nakon prelaska određene staze sprema broj levela koji otključava sljedeći level,,npr ako je igrač završio prvu stazu u *PlayerPrefs.SetInt,("LevelUnlock*",level); će se spremiti broj jedan koji označava da je prvi level prijeđen te će se time otključati sljedeći level. Kod je prikazan u nastavku.

```
function Update () {
if(PlayerPrefs.GetInt("LevelUnlock")>0){levelLock2.
renderer.material.mainTexture  = NOlock;}

if(Input.GetMouseButtonDown(0) &&  rayHit.collider.tag ==
"1-2" &&
PlayerPrefs.GetInt("LevelUnlock")>0){Application.LoadLevel
("1-2");}
}
```

Programski kod 7.Otključavanje staze

Izvor:Autor

Prvo se uspoređuje *PlayerPrefs* s nulom da se vidi da li je prvi level prijeđen. Ukoliko jest, miče se tekstura *lock* koja prikazuje da je level zaključan (slika 18).

Slika 18.Prije i poslije prelaska prve staze

Izvor: snimka zaslona autora

Ako igrač pritisne na određenu stazu (u ovom slučaju druga staza), ponovno se uspoređuje *PlayerPrefs* s nulom te, ako je veći od nule, omogućuje mu se ulazak na drugu stazu.

6. Plasiranje na Google Play trgovinu

Prije nego što se bilo koja aplikacija objavi potrebno ju je testirati na reprezentativnom broju različitih uređaja, te zatim zapakirati. Pakiranjem se dobiva apk datoteka koja je u stvari komprimirana zip datoteka i u sebi sadrži više direktorija i datoteka.

Za objavljivanje aplikacija na *Google Play-u* razvojni programeri moraju posjedovati *Google* korisnički račun te je prilikom prve prijave na *Google Play* potrebno uplatiti iznos od 25 američkih dolara na ime registracije. Programer osim apk datoteke treba pružiti dodatne informacije o njoj. Potrebno je odabrati naziv za prikaz, sastaviti kraći i dulji tekstualni opis aplikacije, te barem dva slikovna prikaza izvođenja aplikacije

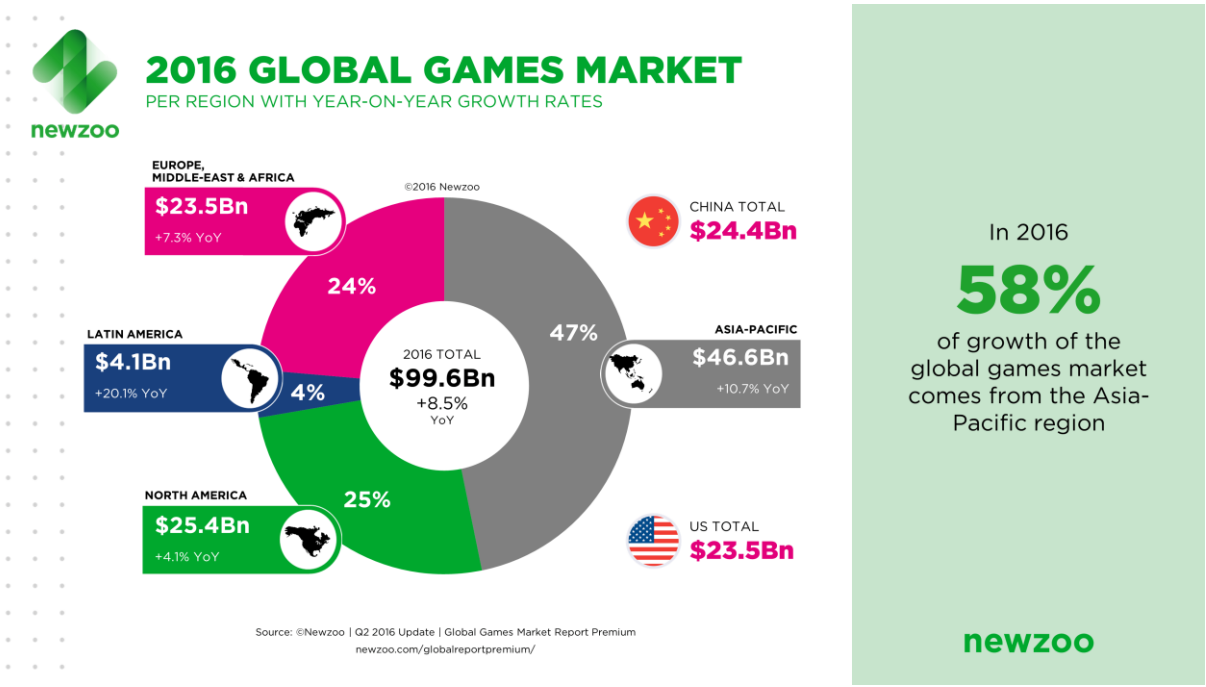
Također je potrebno stvoriti nekoliko promotivnih slika raznih veličina kako bi se aplikaciju lakše automatski promoviralo unutar *Google Play* trgovine. Moguće je i povezati videozapis korištenja aplikacije koji će biti prikazan među ostalim grafičkim prikazima. Prije objavljivanja potrebno je još ručno opisati aplikaciju birajući spada li u općenite aplikacije ili igre, kojoj kategoriji pripada te ispuniti upitnik o sadržaju kojim se dobiva certifikat odobrenja sadržaja

Nakon same distribucije igru je potrebno održavati i, ovisno o igri, nadograđivati. Održavanje igre podrazumijeva dodavanje novih dodataka igri kao što su nove staze te popravke nakon distribucije poznatije kao zakrpe (engl. *patches*). Nakon objave programeru su dostupne razne mogućnosti praćenja uspjeha aplikacije pomoću *Google Analytics-a*. Moguće je vidjeti broj instalacija te filtrirati prikaz po verziji Android sustava, nazivu uređaja, jeziku i zemlji korisnika.

6.1 Mogućnosti zarade

Igre za tablet i pametne telefona će generirati ukupno 36,9 milijarde $ prihoda ili 37 posto ukupnog tržišta koje se očekuje da će doći do 99,6 milijarde $ ove godine, prema istraživanju tvrtke *Newzoo* .
Za usporedbu , PC igre će donijeti 31,9 milijardi $ dok će igru za konzolu

generirati 29 milijardi $ prihoda (slika 19). (Newzoo,global games market 2016)



Slika 19. Prihod od video igara u 2016

Izvor: Newzoo.global games market 2016.

Plan zarade mora biti osmišljen na samom početku, jer će on bitno utjecati na razvoj i dizajn aplikacije. Treba precizno znati u kome dijelu će se pojaviti opcija za kupovinu unutar aplikacije, da bi se uhvatio pravi trenutak, znati koje poteze igrači obično koriste, koji su dijelovi igrice najteži, kada je najvjerojatnije da će korisnici kupiti virtualnu robu, i sl.

Da bi se reklama prikazivala u igri potrebno je imati *Google Wallet* račun i unutar aplikacije omogućiti korištenje *Google*-ovih API-ja za naplatu i/ili oglašavanje. Unos oglasa i reklama i plaćanje unutar aplikacije neće demotivirati obožavaoce igara. Važno je osmisliti i napraviti dobru igru koja će privući što veći broj korisnika na duže vrijeme. Mnogo korisnika - mnogo novca.

Dodatak za monetizaciju je urađen tako da radi samo na platformama s android operativnim sistemom, i poziva se iz Jave. Da bi ugradili monetizaciju u neku aplikaciju, potrebno je dodati dodatak koji poziva java funkciju koja prikazuje reklamu. Ovaj dodatak učitava reklamu s identifikacijom programera, prikazuje je

i ako kliknete na reklamu donosi predviđenu zaradu programeru. U zavisnosti od lokacije gdje se nalazi korisnik i vrste aplikacije, spomenuti dodatak bira jednu od reklama, i ako je sve u redu poslije prikaza reklame, bez obzira da li ste kliknuli na reklamu i vidjeli što Vam se nudi ili samo ugasili reklamu, glavna aplikacija nastavlja s radom, kao da ništa nije bilo.

6.2 Oglašavanje igre

Marketing je, poslije objavljivanja aplikacije, najvažnija faza koja vodi ka željenom cilju - ostvarivanju velikog broja preuzimanja aplikacije

Da bi došlo do preuzimanja aplikacije nije dovoljno samo napraviti dobru aplikaciju. Mnoge dobre aplikacije ostaju nepopularne sa samo nekoliko desetina ili stotina preuzimanja, što nije dovoljno za ostvarivanje neke ozbiljne zarade.

Najpopularnije metode marketinga android aplikacija: [Jin Kim, 2014]

1.Promocija na socijalnim mrežama kao što su *Facebook, Twitter, Google+* i druge

2. Reklamiranje aplikacije u što više grupa koje su sa sadržajem slične aplikaciji.

3. Reklamiranje aplikacije po forumima

4. Plaćene metode marketinga - postoje razni sajtovi i pojedinci koji nude svoje usluge tipa reklamiranje, ocjenjivanje, instaliranje i slično

Zaključak

Pametni mobilni telefoni postali su svakodnevni alati koje koristi veliki broj ljudi različitih godina, zanimanja i interesa. Oni više ne predstavljaju luksuz već su dostupni svima, te su time svima dostupne i aplikacije koje nude.
Aplikacije na pametnim telefonima mogu raditi razne stvari, od provjere vremenske prognoze, preko komunikacije s prijateljima i obitelji do zabave kroz igre koje od svih aplikacija donose najviše zarade, što je i bio jedan od glavnih pokretača kod odabira ovog završnog rada.
U ovom radu opisan je proces izrade mobilne igre. Kroz praktičan primjer prikazana je ideja igre, izrada igre i kako doći do zarade od mobilne aplikacije.
Pri izradi igre najvažnije je pronaći adekvatan alat kojim će biti moguće ostvariti sve zacrtane želje i zadatke. U ovom radu korišten je programski pogon *Unity* 3D, koji se može pronaći u dvije verzije besplatnoj i licenciranoj. Iako je naizgled intuitivan, za snalaženje unutar pogona potrebno je dosta vremena i znanja.
U radu određene su stvari mogle biti drugačije izvedene, ali trenutna struktura rada je proširiva, tako da se u budućim verzijama može jednostavno nadograditi.
U sklopu potencijalnih kasnijih verzija moguće je prekrajanje idejnog okruženja i dodavanje dodatnih funkcionalnosti, kao što su, primjerice:

- Estetski ljepši i pristupačniji dizajn elemenata i sučelja
- Mogućnost kupovine novih likova
- Više staza i novih prepreka
- Mogućnost da korisnik sam izradi stazu

Razvoj tehnologije omogućuje nam da pojedinac vlastitim trudom može napraviti igru, bez nekih prevelikih troškova, te da u današnje vrijeme izrada videoigara više nije rezervirana za velike razvojne timove.
Ovaj rad ukazuje i na to da izrada igre danas nije tolika nepoznanica te da se uz nešto truda i znanja u nekoliko programa može razviti kompletna igra, što nije bio slučaj prije nekih desetak godina.
Veliki postotak ljudi smatra video igre zabavnima te one postaju jaka platforma pomoću koje se može dobro zaraditi.

Literatura

1. Android.com.Launch Checklist
https://developer.android.com/distribute/tools/launch-checklist.html
(pristupio 3.9.2016)

2. Developer android. dashboards 2016.
https://developer.android.com/about/dashboards/index.html (pristupio 3.9.2016)

3. Gargenta Marko. *Learning Android.* Sebastopol: O'Reilly Media, Inc., 2011.

4. IDC: Smartphone OS Market Share 2015
http://www.idc.com/prodserv/smartphone-os-market-share.jsp
(pristupio 4.9.2016)

5. Jon Brodkin, How Unity3D Became a Game-Development Beast.03.06.2013
http://insights.dice.com/2013/06/03/how-unity3d-become-a-game-development-beast/ (pristupio 6.9.2016)

6. Jin Kim,App Marketing,This is Real Android Marketing:Itmagnet publishing 2014

7. Mobile Game Graphics -Background Builder Kit
https://mobilegamegraphics.com/product/vector-background-generator/
(pristupio 10.5.2016)

8. Malmo james too:prototping-advice
http://www.malmojamstoo.com/prototyping-advice/ (pristupio 8.9.2016)

9. Newzoo,global games market 2016.
https://newzoo.com/insights/articles/global-games-market-reaches-99-6-billion-2016-mobile-generating-37/ (pristupio 8.9.2016)

10. Sue Blackman, Beginning 3D Game Development with Unity:Apress Publishing 2011

11. Unity Documentation
https://docs.unity3d.com/Manual/UsingTheEditor.html (pristupio 6.9.2016)

Popis slika,tablica i koda

Tablica

Programski kodovi

Prilog

Radu se u digitalnom obliku prilaže cjelokupan izvorni kod Android aplikacije *Candy Freak* izrađene u sklopu pisanja rada.

Sažetak

U današnje vrijeme mobilne igre predstavljaju najčešći oblik digitalne zabave te su ušle u sve pore svakodnevnog života i ne poznaju dobne granice. Kroz ovaj rad je opisan postupak izrade videoigre za Android platformu naziva *Candy Freak* pomoću alata *Unity 3D*.

U prva dva poglavlja bit će opisani Android i program *Unity 3D* - kako su nastali, kako funkcioniraju i njihovo trenutno stanje na tržištu. Od trećeg do petog poglavlja opisan je postupak izrade igre, tj. koji su programi korišteni, kako doći do ideje za igricu te kako napraviti android igricu. U šestom poglavlju je objašnjen način objave neke aplikacije na *Google Play*-u ,kako je reklamirati i ono najbitnije - kako od nje zaraditi. Na kraju rada u zaključku su navedena iskustva stečena tijekom izrade igre te daljnji planovi u razvoju igre.

Ideja za izradu mobilne igre proizašla je prije godinu dana nizom pokušaja za pronalaskom igre na kojoj bi bilo zanimljivo raditi a da je drugačija od ostalih, te je iz toga nastao *Candy Freak*. Ozbiljan rad na igrici je počeo tek prije 6 mjeseci kao potreba za završni rad.

Za izradu ovoga rada bilo je potrebno proučiti velik broj izvora podataka te primijeniti stečena znanja o programiranju i grafičkom dizajnu.

Ključne riječi: Unity, engine, android, razvoj igara

Summary

Nowadays, mobile games are the most common forms of digital entertainment included in all aspects of daily life regardless of the age limit.

The process of making a video game for Android platform called *Candy Freak* by *Unity 3D* was described throughout this paper.

The first two chapters describe Android and *Unity* 3D programs - how they arised, how they work and their current status in the market. Chapters from three to five describe the process of creating games - which programs are used, how to come up with an idea for a game and how to make an Android game. The sixth chapter explains how applications are published on Google Play, how to advertise them, and most importantly, how to make a profit. At the end, in conclusion are given the experiences gained during the development of the game and further plans to develop games

The idea for creating a mobile game emerged a year ago from a number of attempts for finding a game that would be interesting to work on and that is different from the others, and from it emerged *Candy Freak*. Serious work on the game began 6 months ago, as finishing of this paper required.

To write this thesis, it was necessary to examine a large number of data sources and apply the acquired knowledge in programming and graphic design.

Key words: Unity, engine, android, game development