Izrada 2D igre kroz Unity3D programski alat

Slunjski, Tihomir

Sveučilište
Sjever

Završni rad br. 469/MM/2016

Izrada 2D igre kroz Unity3D programski alat

Tihomir Slunjski, 4761/601

Varaždin, rujan 2017. godine

Sveučilište Sjever

Multimedija oblikovanje i primjena

Završni rad br. 469/MM/2016

Izrada 2D igre kroz Unity3D programski alat

Student

Tihomir Slunjski, 4761/601

Mentor

dr. sc. Andrija Bernik, pred.

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za multimediju, oblikovanje i primjenu

PRISTUPNIK: Tihomir Slunjski | MATIČNI BROJ: 4761/601

DATUM: 16.02.2016 | KOLEGIJ: 3D modeliranje

NASLOV RADA: Izrada 2D igre kroz Unity3D programski alat

MENTOR: Andrija Bernik, dipl. inf. | ZVANJE: Predavač

ČLANOVI POVJERENSTVA:

1. pred. Robert Geček,dipl.ing. - predsjednik
2. doc.dr.sc. Dean Valdec - član
3. pred. Andrija Bernik, dipl. inf. - mentor
4. mr.sc. Vladimir Stanisavljević, v. predavač - zamjenski član
5.

Zadatak završnog rada

BROJ: 469/MM/2016

OPIS:

Opis teme: Unity3D je alat za izradu 2D i 3D igara koji je spojem pristupačnosti i naprednih funkcionalnosti osvojio ne samo mlade entuzijaste već i profesionalce koji dulji niz godina programiraju i stvaraju video igre. Jednom razvijena igra može se lako objaviti na različitim platformama. Unity3D ima mogućnost kreiranja igara za platforme: Windows, Mac, Linux, IOS, Android, Sony Playstation, Xbox, Web, i još mnogo drugih. Sve ovo samo su neki od razloga zašto je Unity3D najpopularniji program za izradu igara današnjice i zašto je odaban za temu ovog rada. Rezultat rada biti će funkcionalna 2D igra.

Rad će prikazati:
- Upoznavanje Unity engine-a
- Osnove manipulacije sa objektima
- Objekte i komponente
- Osnove objektno orijentiranog programiranja
- Dizajn 2d video igara
- Fiziku u Unity engine-u
- Import modela i tekstura
- Dodavanje zvuka u igru
- Korisničko sučelje igre
- Manipulaciju nad terenom

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

ZADATAK URUČEN: 09.09.2016. | POTPIS MENTORA: Bernik

Predgovor

Temu ovog završnog rada odabrao sam iz osobnih interesa i istraživanja kako se rade igre i koji proces moraju proći da bi dobili finalni sadržaj. Cilj je bio steći novo znanje i upoznati se sa novim programom za izradu igre, te kroz razne stručne radove i knjige saznati što više o tome području.

Zahvaljujem mentoru dr. sc. Andriji Berniku koji je pratio cijeli proces nastajanja završnog rada i svojim savjetima i entuzijazmom me usmjeravao kako da riješim probleme koji bi se pojavili prilikom izrade završnog rada. Također zahvaljujem roditeljima i prijateljima koji su mi bili podrška tijekom studiranja.

Summary

This paper presents the process of making simple 2D games, as well as the tools and programs used in its creation. The basic concepts of each program used, their requirements, basic parts, history and the like are explained. Video games in today's world have become one of the main types of gambling, and in the video game industry, billions of US dollars are turning, and the industry itself has also recently become bigger than the movie and music industry. This paper will describe the development of computer games, the industry's current game industry, numerous game development segments, the development game, and the game engine Unity, as well as an example of game development in the aforementioned starter.

Keywords: *2D Game, Game Engine, Graphic User Interface, Scripts, Particle System, Unity, Video Game*

Sažetak

U radu je prikazan proces izrade jednostavne 2D igre, te alati i programi koji se koriste u njezinoj izradi. Objašnjeni su osnovni pojmovi svakog korištenog programa, njihovi zahtjevi, osnovni dijelovi, povijest i drugo.

Video igre u današnjem svijetu su postale jedne od glavnih vrsta razbibriga, također u industriji video igara okreću se milijarde američkih dolara, a sama industrija nedavno je također postala veća od industrije filmova i muzike. Zbog ovih razloga te i zbog toga što i sam igram video igre odlučio sam se za temu za završni koja se bavi izradom video igre.

U ovom radu bit će opisani razvoj računalnih igara, položaj industrije igara u sadašnjosti, brojni segmenti kod razvoja igara, razvojno okruženje, odnosno pokretač (engl. game engine) Unity i prikazan primjer razvoja igre u spomenutom pokretaču.

Ključne riječi: 2D igra, Game engine, Grafičko korisničko sučelje, Skripte, Sustav čestica, Unity, Video igra

Sadržaj

1. Uvod

U današnje vrijeme multimedijske igre predstavljaju najčešći oblik virtualne zabave koja često zamjenjuje druge oblike provođenja vremena. Suvremene igre obuhvaćaju integraciju video materijala, zvuka, modela, animacija, teksta, pokreta, korisničkog sučelja te konačne interakcije igrača s virtualnim svijetom. Njihov je razvoj uz pomoć suvremenih alata i razvojnih okruženja, potpomognutih širokom zajednicom te dostupnošću velike količine informacija i edukativnih materijala, u današnje doba poprilično olakšan unatoč početnim dojmovima izuzetne kompleksnosti. Sudeći prema trendu konstantnog poboljšanja razvojnih okruženja, njihova intuitivnost te jednostavnost korištenja u budućnosti će biti sve izraženija. Postoje brojna razvojna okruženja, od kojih je većina razvijena od strane velikih tvrtki za vlastite potrebe. Međutim, svakako su najpopularnija ona dostupna u komercijalne svrhe te će u ostatku ovoga rada biti objašnjeno trenutno najpoznatije razvojno okruženje Unity 5.

U ovom je radu opisano razvojno okruženje Unity te njegove osnovne karakteristike, mogućnosti te implementirane funkcionalnosti koja ga čine preferencijalnim izborom nad sličnim konkurentskim proizvodima. Kao što je već napomenuto, Unity je razvojno okruženje (eng. *Game Engine*) za dizajniranje i razvoj svih potrebnih aspekata video igara. Podržava velik broj sustava za koje konačna igrica može biti namijenjena. Osnovni proces kreiranja igre započinje kreiranjem ili preuzimanjem gotovih objekata koji se ubacuju u promatranu scenu. Pod objektom se u ovom slučaju smatra svaka komponenta koja se koristi u izgradnji te scene. Iz toga možemo zaključiti da je scena trodimenzionalni prostor u kojemu pozicioniramo objekte video igre te im dodjeljujemo željene funkcionalnosti, od animacija i zvukovnih zapisa pa sve do predefiniranog ponašanja u određenim situacijama. Isti princip koristi i druga razvojna okruženja poput Unreal Engine 4, kojega ćemo u ovome radu isto tako spominjati kako bismo uvidjeli međusobne prednosti i nedostatke u odnosu na Unity. Uz teoretsku analizu razvojnih okruženja, u Unity-u je izrađen i vlastiti primjer video igre koji sadržava sve potrebne komponente jedne klasične igre.

2. Igre i razvoj računalnih igara

Alexnader Douglas 1952. godine napravio je igru „tic-tac-toe“, odnosno XOX igru. Douglas je igru prikazao na tadašnjem CRT zaslonu koji je bio veličine 35 x 16 piksela. Igra se igrala tako da, korisnik igra protiv računala, odnosno stroja. Sam projekt je Douglas počeo istraživati jer je tražio interakciju između računala i čovjeka. Igra se igrala preko telefona, tako da koji se broj pritisnuo na to mjesto išao je X ili O [26].

Prva 2D igra napravljena je 1958. godine, kada je nuklearni fizičar William Higinbotham napravio simulaciju tenisa na osciloskopu. Igra se igrala tako da je svaki igrač imao osciloskop na kojemu su bila dva gumba preko kojih je s jednim mogao udariti loptu u bilo koje vrijeme po izboru, a drugi gumb bio je za odabir kuta kojim se lopta udarala drugom igraču. Igra je napravljena pomoću reklaja, tranzistora i pojačala, što je u ono vrijeme tehničarima i Williamu trebalo nekoliko tjedana za projektiranje i izgradnju igre. Kasnih 70-ih N. Bushnell i T. Dabnex osnovali su tvrtku Atari koja je odgovorna za stvaranje konzola i komercijalnu upotrebu za svaki dom. Prva od tih konzola s više igara bila je Atari 2600 VCS i imala je 128 bajtova radne memorije [7].

Tvrtku Apple osnovali su Steve Jobs, Steve Wozniak i Ronald Wayne. Apple II 1977. godine postao je jedan od najpopularnijih računala ikad. Bez obzira na veliki napredak u odnosu na Apple I, Apple II je sadržavao isti procesor i radio je na istoj brzini. Apple II je bio jedan od prvih računala sa zaslonom u boji, a također je bio i prvi razumljiv sustav. Apple II je imao osam utora za proširenje, što je bila najvažnija značajka jer druga računala nisu imala ovakve mogućnosti. Spomenuta proširenja, omogućavala su jednostavan pristup sustavu matične ploče i proširenje utora [6].

Sci-fi igra po imenu Ultima Underworld napravljena je početkom 1989. godine. Igra je napravljena od istoimenog softvera za izradu igre (eng. *game engin-a*), ali kada je izašla igra Space Roungue, Origin System je preuzeo Ultima Underworld. Origin System je razvila algoritam za mapiranje tekstura koje su se mogle primijeniti na podove, stropove, zidove i slično. Velika revolucija u svijetu zabave bila je kad su došla prva 16 – bitna računala 1992. godine te je tvrtka id Software stvorila igru Wolfertein 3D. To je bila prva igra u kojoj su se teksture na sve predmete oslikavale metodom billboarding. Tako se dobivao dojam da se igra 3D igra, bez obzira na to što je ona bila 2D [8].

1990. godine id Software povlači granice i izlazi igra DOOM koja je izgledala drugačije i imala je više mogućnosti. Igra DOOM je radila u višim rezolucijama, a teksture su se koristile na krovu i podu, sobe nisu više bile ravne, kamera se pomicala gore – dolje prilikom kretanja,

postojala je rotacija oko dvije osi. Igra je uvodila mrežu Deathmatch i kooperativni mod. John Carmack i id Software postali su dio povijesti, jer su izazvali revoluciju i pomakli ljestvicu igara stupanj više. Igra DOOM je imala nastavak igru DOOM2. 1995. godine je izdana sljedeća revolucija igara napravljena od tvrtke Parallax. Igra je omogućavala kretanje i rotaciju u svakom smjeru. XnGine je bio prvi 3D engine koji je razvijen u DOS bazi. To je omogućilo kasnije da se upotrebljavaju grafike visoke rezolucije i da budu kompatibilne sa 3dfx grafičkim karticama. Quake engine je bio prvi pravi 3D game engine, koji je imao jedinstven kapacitet prerade za prikaz mape koji su vidljivi igraču koji je samo to prikazivao i očitavao. Najpopularnije razvojno okruženje (eng. *game engine*) za multiplatform igre bio je Renderware, koji podržava PlayStation 2, Wii, GameCube, Xbox, Xbox 360, PlayStation 3 i PSP platforme. OpenGL je višeplatformska specifikacija s podrškom za niz programskih jezika, čiji je cilj omogućiti pisanje aplikacija koje rade s 2D i 3D grafikom. Quake II, id Tech 2 engine podržava OpenGL. Uspješne igre koje su se temeljile na OpenGL su Half-Life i Counter Strike. Najpopularnija igra je Unreal Engine koja integrira vlastiti skriptni jezik i map editor [8].

Za igricu Serious Sam zaslužan je Croteam. Igrica je napravljena Serious razvojnim okruženjem i omogućavala je velike prostore i veliki broj modela na zaslonu u bilo kojem trenutku. Serious razvojno okruženje je vrlo djelotvoran i podržava desetke animiranih modela čak i na skromnim konfiguracijama računala i nadmašuje sposobnosti popularnih Quake, Unreal i Half-life engine-a. Podržava DirectX i OpenGL, a optimiziran je za DirectX 7. Današnje razvojno okruženje je Serious Engine 4 i prvi je istinski multi-platform razvojno okruženje s podrškom za mobilne platforme i nove buduće generacijske igre [10].

Game razvojno okruženje sadrži grafiku, zvuk, logiku, igre i druge elemente. Kada se rade alati za razvoj neke igre na razvojnom okruženju, ti alati trebaju biti kompatibilni s pogonom igre da bi informacije između programskog i dizajnerskog djela bile bolje. Povezanost spomenutih dijelova dijelimo na svjetla, 3D modele, animacije, zvuk i slično. Svaki od tih alata treba biti i kompatibilan s drugim hardverima i softverima. Kod igra, kvaliteta grafike nije ista kao u filmskoj, ali se radi na tome da se smanji hardversko opterećenje i da se pritom pruži visoka kvaliteta igre [8].

3. Unity alat

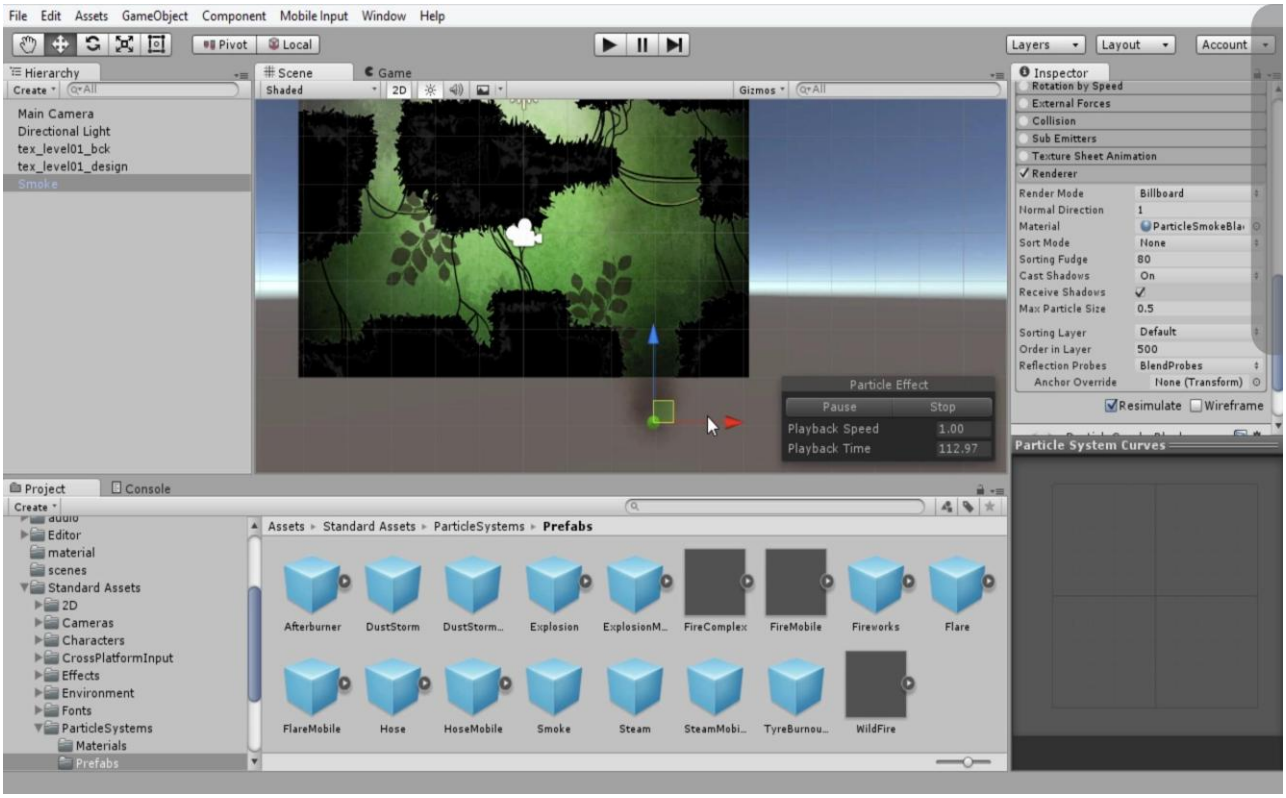
Unity razvojno okruženje je razvio Unity Technologies. To je razvojno okruženje za igre i računalnu grafiku, odnosno alat za izradu video igara. Prva verzija Unitya izašla je 9. kolovoza 2005. godine. Primarna karakteristika razvojnog okruženja je da ga je moguće izvoditi na svim platformama koje su danas na računalnom tržištu. Upravo iz tog razloga Unity je moguće koristiti na osobnim računalima, koji imaju Windowse ili Linux za operacijske sustave te u preglednicima kao web pluginove, konzolama i mobilnim uređajima. Zbog korištenja Unity-a na različitim uređajima to je osiguralo popularnost na tržištu. Razvojno okruženje je u cijelosti napisan u C i C++ programskim jezicima. Kako bi se olakšao razvoj korisnicima, odnosno developerima igara na jezgri sustava sagrađen je omotač (eng. *wrapper*) koji postavlja sloj za pristup .NET jezika. Tako se olakšava skriptiranje i korištenje grafičkog sučelja koje Unity pruža. Kako bi se dodatno proširila raznovrsnost same platforme, osiguran je API za razvoj u Javascriptu, C# i Boo-u [4].

Unity Technologies je poduzeće koje je osnovano 2004. godine od strane Davida Helgason-a, Nicolas Francis-a i Joachim Ante-a. Poduzeće je osnovano u Danskoj, u Kopenhagenu. Primarni razlog za osnivanje poduzeća je bio relativan neuspjeh igre koju su izradili, GooBall, a potom i želja da se stvori razvojno okruženje koji će biti dostupan svima. Njihov projekt je odmah doživio uspjeh kod ulagača te je privukao mnogo ventura kapitala. Mnogo važnije od ulagača i kapitala, bilo je privlačenje pozornosti na programerske zajednice, odnosno krajnjih korisnika softvera. S obzirom na to da su softveri skupi i razvoj vlastitog ne dolazi u obzir, to im je uspjelo. Ono što je najviše pridonijelo razvoju je iPhone. Unity je bio prva platforma koja je u cijelosti podržavala sve mogućnosti uređaja i operacijskih sustava [4].

Unity razvojno okruženje podržava integraciju s mnogo alata za 3D modeliranje i uređivanje slika i slično. Neki od alata za izradu su [4]:

- 3D studio MAX,
- MAYA,
- BLENDER,
- AdobePS, Fireworks,
- Cheetah 3D,
- SoftImage.

Za navedene alate podrška se uglavnom ogleda u trenutnom osvježavanju datoteka koje su stvorene u ostalim aplikacijama te propagiranje novina kroz cijeli projekt. Omogućeno je unošenje nativnih formata. Time je omogućeno direktno integriranje vektorske grafike iz na primjer Photoshopa te korištenje 3D modela iz specijaliziranih alata kao što je Maya. To je također jedna od velikih prednosti same platforme [4].



Slika 3.1. Unity Sučelje

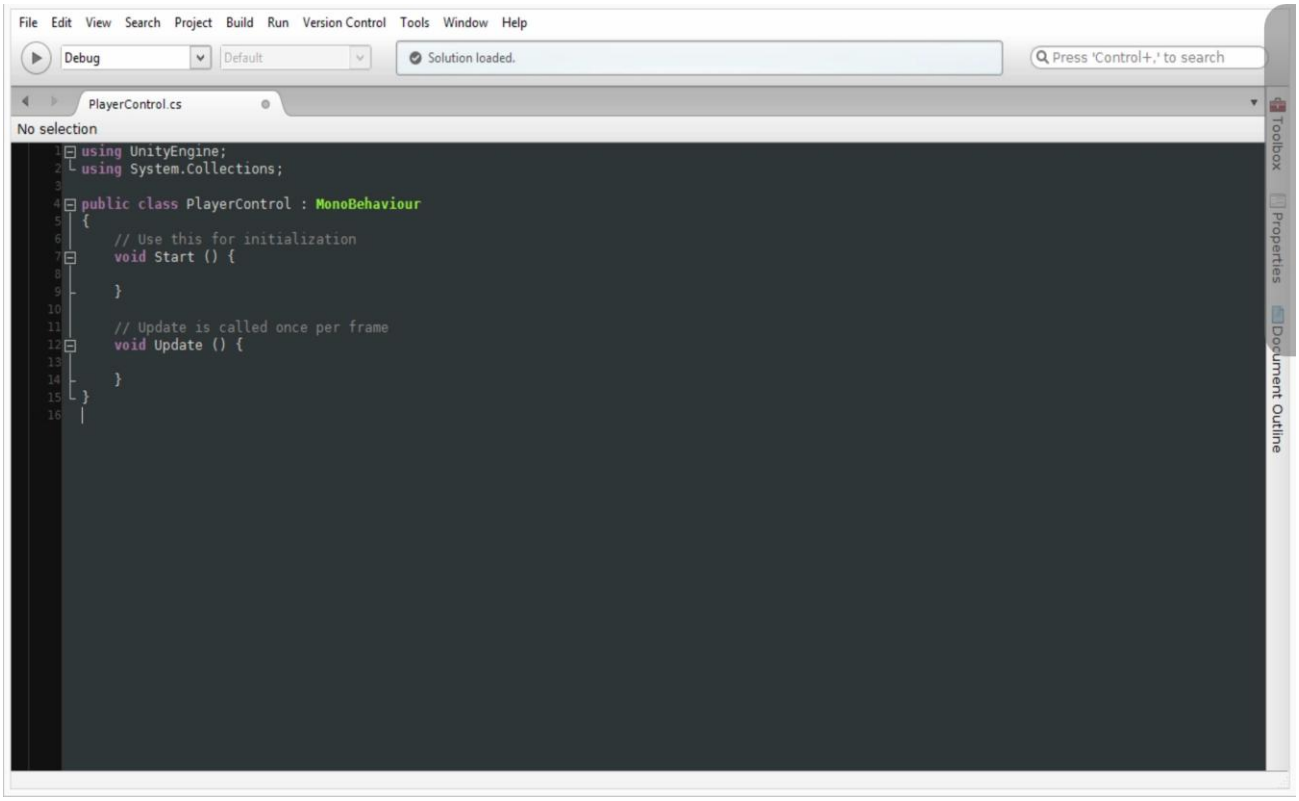
Unity je moćan alat koji puno olakšava izradu video igara zato što u sebi ima ugrađene mnoge funkcionalnosti koje su dostupne korisniku klikom miša, dok bi implementacija jedne takve funkcionalnosti otpočetka mogla trajati i nekoliko dana ili čak tjedana (kolizije, animacije, fizika, osvjetljenje….).

Kroz vrijeme izlazile su mnoge verzije Unity-a, a prva značajna verzija nakon 1.0. je bio Unity 2.0. koji je izašao 11. listopada 2007. godine, a nove funkcionalnosti koje je ova verzija donijela su: alati za rad s terenom, osjenčavanje u stvarnom vremenu te sustav za izradu korisničkog sučelja. Također 2.0. verziju je obilježila izdaja i Unity Asset Servera. Unity Asset Server je omogućavao lakše dijeljenje resursa među developerima.

Unity 3.0 je izdan 4. listopada 2010. godine, a donio je novine poput mapiranja svijetla, dodavanje zvučnih efekata i podršku za C# 3.5. Također, ova verzija dovodi podršku za izradu igre za android operacijske sustave.

U Unity 4.0. dolazi mnoštvo novina. Ova verzija razvojnog okruženja izašla je 13. studenog 2012. godine, a donosi podršku za DirectX 11, novi sustav za animaciju te osjenčavanje u stvarnom vremenu za mobilne uređaje. No 4.x verzije donose još novine, poput novog frameworka za izradu 2D igara te potpuno novog frameworka za izradu korisničkog sučelja. Također je uvedena podrška za windows phone 8 uređaje, windows store te blackberry.

Zadnja verzija Unity-a je 5.0 izdana 3. ožujka 2015. godine koja dovodi novine, poput mogućnosti osvjetljenja u realnom vremenu, novi audio mixer te poboljšanja u procesu animacije. Također, ova verzija podržava izradu igara za gotovo sve današnje platforme. Što se tiče programskog koda, Unity dolazi s ugrađenim IDE-om pod nazivom MonoDevelop koji podržava programske jezike C#, F#, Visual Basic .NET, C/C++ i Vala.



Slika 3.2. MonoDevelop

3.1. Osnovne značajke Unity alata

Unity je više platformski alat za razvoj video igara koji se koristi za izradu 2D i 3D igara. Svojom jednostavnošću i pristupačnošću i brojnim mogućnostima danas je jedan od vodećih među sve brojnijom konkurencijom. Dovoljno jednostavan za početnike i dovoljno moćan za napredne korisnike omogućava da ga koriste svi te da zadovolji sve zahtjeve. Doći do

instalacijskog paketa je također jednostavno i može se besplatno preuzeti s njihovih službenih stranica gdje postoje mnoge upute, materijali i dokumentacija za rad. Danas Unity nudi gotovo sve mogućnosti kao i drugi alati za razvoj video igara te tako ne odstupa ni na koji način po kvaliteti, a najbolji dokaz tomu su mnogi partneri koji su ujedno i velike i poznate tvrtke koje surađuju na razvoju Unity alata kao što su: Microsoft, Sony, Samsung, Nitendo, Oculus ili pak Intel.

Kao što je već spomenuto, Unity je besplatan alat, ali postoje i verzije koje se naplaćuju ovisno o zaradi i mogućnostima koje korisnik namjerava koristiti. U slučaju da se korisnik ili organizacija odluče za kupnju Unity alata, plaćanje se odvija na mjesečnoj bazi u obliku pretplate. Ovisno o količini mogućnosti postoje tri paketa koja se naplaćuju: Plus, Pro i Enterprise. Personal verzija je početna i besplatna. Potreba za korištenjem paketa koji se naplaćuju je ako zarada korisnika ili organizacije je veća od 100 000 američkih dolara te zatim se cijene kreću od 35 američkih dolara mjesečno za Plus verziju, 125 američkih dolara za Pro, a što se tiče Enterprise verzije cijena je po dogovoru [14].

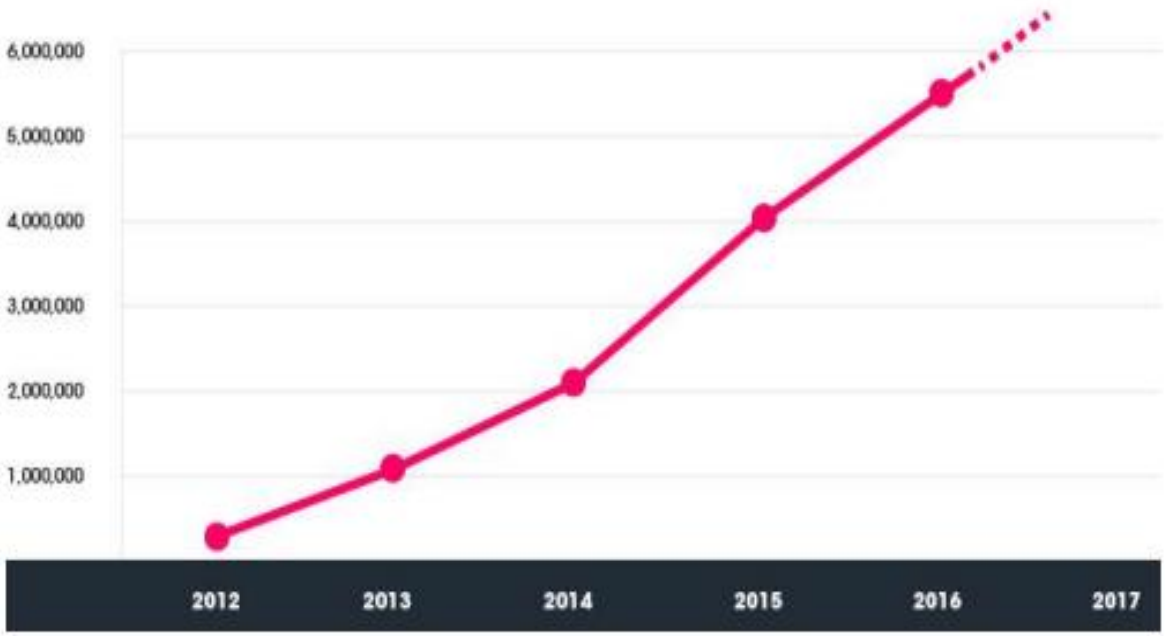
Danas Unity nudi razvoj na više od 21 različitih platforma, od raznih operacijskih sustava za računala, web, operacijskih sustava za mobitele, konzole, pametne televizore, sustava za virtualnu stvarnost i druge. Zahvaljujući tome i jednostavnom prebacivanju igara s jedne platforme na drugu. Unity ima toliku popularnost danas kada igre možemo pokretati gotovo na svim uređajima [14].

Slika 3.3. Logo znakovi svih platforma na kojima je moguć razvoj u Unity alatu (Unity Technologies, 2017)

Kvaliteta razvojnog okruženja Unity može se uvidjeti samom činjenicom da je još davne 2006. godine, dakle godinu dana nakon njegovog inicijalnog razvoja, od strane tvrtke Apple nominiran u kategoriji za najbolju upotrebu grafike Mac OS X sustava. To je prvi puta u povijesti da je razvojno okruženje za igrice bilo nominirano u toj kategoriji. Nadalje su se tijekom godina nizali samo uspjesi i brojne nagrade koje su samo potvrđivale početni uspjeh te kvalitetan daljnji razvoj razvojnog okruženja. U današnje vrijeme, sve je veći broj igrica koje objavljuju neiskusni programeri koji uz malo volje i truda te pomoći zajednice istomišljenika, koja se okupila oko razvojnog okruženja Unity, uspijevaju ostvariti odlične recenzije, što samo ide u prilog kvaliteti razvojnog okruženja Unity [2], [11].

3.2. Unity Technologies ltd.

David Helgason (CEO), Nicolas Francis (CCO) i Joachim Ante (CTO) osnovali su Technologies 2004. godine u Kopenhagenu, u Danskoj. Pokušavajući napraviti vlastitu igricu, shvatili su važnost razvojnih okvira te odlučili napraviti vlastiti koji bi objedinjavao brojne korisne funkcionalnosti. Glavni im je cilj bio jednostavnost i intuitivnost razvojnog okruženja u izradi dvodimenzionalnih i trodimenzionalnih igara te njegova prezentabilnost prosječnim ljudima diljem svijeta. Njihova namjera zasigurno je uspjela jer u današnje vrijeme Unity koristi više od 5.5 milijuna ljudi [9] [23].



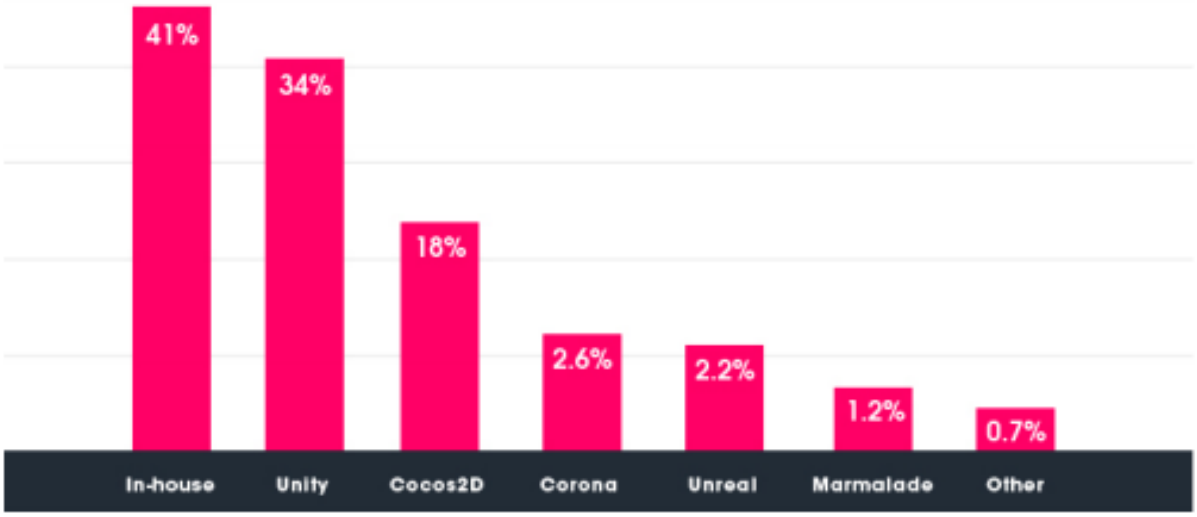
Slika 3.4. Broj registriranih programera koji koriste Unity
(Unity Technologies, 2017)

Broj korisnika se počeo povećavati 2010. godine, kada je službeno objavljena online trgovina za resurse (eng. *Unity Asset Store*). Na taj su način privukli ljude s raznim zanimanjima te objedinili niz informatičkih djelatnosti kao što su dizajniranje, modeliranje, programiranje, animiranje, glazbeno skladanje i tako dalje. Osim što su potaknuli zajednicu obožavatelja da doprinosi razvoju njihove inicijalne ideje, simulativno su smanjili novčane izdatke za razvoj brojnih potrebnih resursa koji su svakim danom sve brojniji i kvalitetniji. S vremenom su otkupili nekoliko manjih tvrtki s različitim djelatnostima te iskoristili njihovo prikupljeno znanje i iskustvo kako bi proširili raspon usluga koje pružaju u sklopu razvojnog okruženja Unity [23].

Unity u današnje vrijeme ima globalnu ulogu u rastućem tržištu igara te je najpoznatije razvojno okruženje za izradu istih. Najveći broj igara napravljen je upravo uz pomoć razvojnog okruženja Unity te s najveći broj programera odlučuje upravo za Unity [23].

Slika 3.5. Sjedišta Unity Technologies ltd.
(Unity Technologies, 2017)

Kao što je vidljivo na Slici 3.5., u današnje je vrijeme Unity rasprostranjen u svim informatički razvijenim zemljama te se postavlja kao standard za izradu računalnih, mobilnih i web igara. Prema najnovijim statistikama preuzetih sa službene stranice razvojnog okruženja Unity, ukupno 5.5 milijuna ljudi registriralo je račun na službenoj stranici, 770 je milijuna aktivnih ljudi koji igraju igre napravljene u Unity-u, dok 1.7. bilijuna mobilnih uređaja vrti igre napravljene u Unity-u. uz sve te korisničke statistike, između svih ponuđenih alata za razvoj igara, Unity je primarni izbor među programerima [3], [23].



Slika 3.6. 34% najboljih 1000 besplatnih mobilnih igara izrađeno je u Unity-u (Unity Technologies, 2016)

S obzirom na rastući trend popularnosti razvojnog okruženja Unity, njegov će se udio u cjelokupnom tržištu s vremenom samo nastaviti povećavati. Sve popularnije mobilno tržište igara već je značajno utjecano igrama izrađenim u Unity-u, što se može vidjeti iz gore navedenog grafikona na Slici 3.6. Unity je također vodeće razvojno okruženje u novijem trendu virtualne stvarnosti gdje se procjenjuje da je 90% Samsung Gear igara te 53% Oculus rift igara napravljeno uz pomoć Unity-a [23].

3.3. Unity 5 i Unreal Engine 4

Unity 5 i Unreal Engine 4 zadnje su stabilne inačice istoimenih razvojnih okruženja te će u nastavku biti objektivno uspoređeni. Realno gledajući, Unity i Unreal Engine 4 dva su najpopularnija razvojna okruženja dostupna široj javnosti te su iz tog razloga odabrane upravo ove dvije distribucije. S druge strane, postoje brojna primjerena razvojna okruženja koje su veći proizvođači igrica stvorili za svoje potrebe, no nisu toliko komercijalno zastupljeni. Unity i Unreal Engine 4 su oboje odlična razvojna okruženja te bi konačan izbor mogao ovisiti o tome što točno pojedinac želi raditi. U konačnici, možda najveće razlike odmah privuku pojedinca na neki od njih, primjerice, netko će preferirati jednostavnost korisničkog sučelja, dok će s druge strane netko preferirati integrirani programski jezik (Mayden, 2014; Unity 5 vs Unreal Engine 4, 2015). Prema trenutnim statistikama, oko 47% programera opredijeljenih za razvoj igara koristi Unity, dok se samo 13% opredijelilo za Unreal Engine 4 [12].

Kada je u pitanju razvoj mobilnih igara, tu definitivno Unity pokazuje svoju dominaciju. Brojne već razvijene igre u Unity-u pokazale su programerima raspon mogućnosti te implementiranih funkcionalnosti koje nadmašuju raspon Unreal Engine 4 [5] [12]. Brojni dodaci koji su već preinstalirani u Unity-u omogućuju brzu i jednostavnu implementaciju reklamnih materijala, kupnje unutar igrica, raznih statistika i još mnogo toga [1].

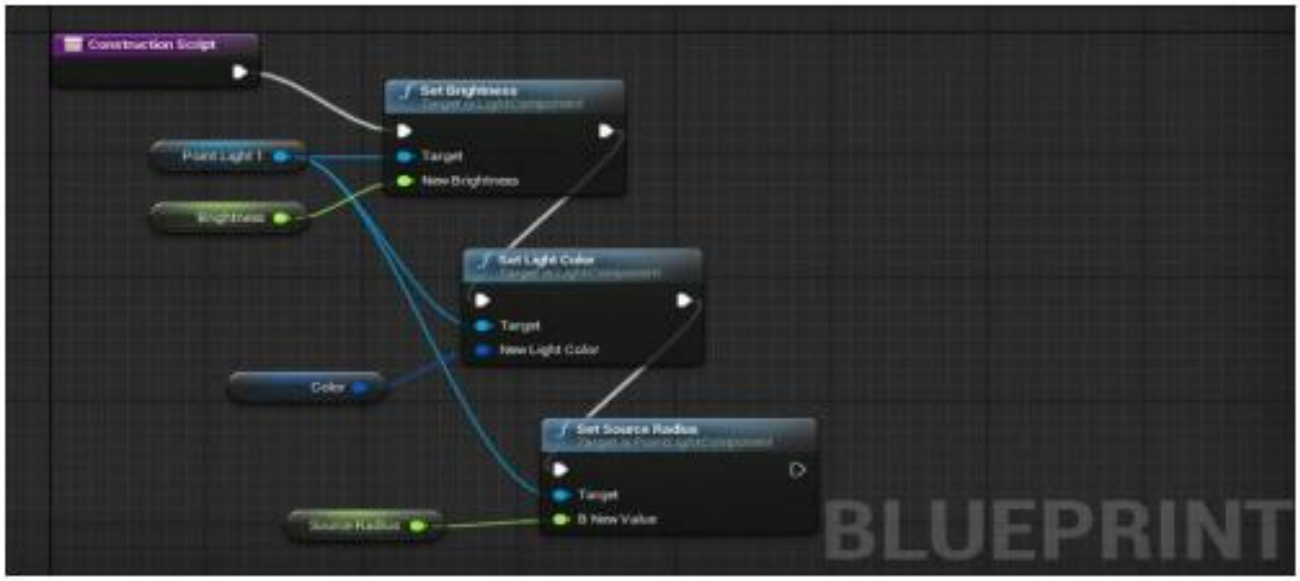
Dvodimenzionalne igre također su pretežito više razvijane u Unity-u jer sadrži brojne pogodnosti uz čiju pomoć se lako kreće s razvojem 2D igrice, dok je Unreal Engine 4 tek nedavno pokušao progurati svoj razvojni okvir za 2D igrice, koji isto sada sadrži mnoštvo funkcionalnosti. Trodimenzionalne igre su svakako primarna svrha oba razvojna okruženja, no kada bismo promatrali kvalitetu grafičkih detalja, onda je Unreal Engine 4 poprilično bolji u gotovo svim aspektima. S druge strane, ako za igricu nije potrebna tolika količina grafičkih detalja, onda se za početnike preporučuje Unity. Unatoč razlikama unutar implementacije i funkcionalnosti razvojnih okruženja, započeo je zanimljiv trend proširenja osnovnih mogućnosti kako bi se na interaktivan način korisnicima pružilo što bolje iskustvo. Iz tog razloga brojni umjetnici u informatičkoj industriji koriste oba razvojna okruženja za kreiranje navedenih interaktivnih iskustava [5] [12].

Slika 3.7. Usporedba grafičke kvalitete Unity i Unreal Engine 4 razvojnih okruženja (Game & Assets Development, 2017)

Cijena može biti presuđujući faktor u odabiru konačnog razvojnog okruženja. Unity u ponudi ima besplatnu osobnu verziju (eng. *Personal Edition*) razvojnog okruženja koja ima sve

potrebne mogućnosti za razvoj igre. Profesionalna verzija (eng. *Professional edition*), odnosno Unity Pro plaća se 1 500 dolara za svako novo izdanje programa ili 75 dolara mjesečno te daje brojne dodatne napredne mogućnosti. Ako razvijate mobilnu igricu, ponovno je potrebno platiti 1 500 dolara ili 75 dolara mjesečno za Android ili iOS licencu. Nakon što službeno izdate svoju igricu, nije potrebno plaćati nikakvu posebnu naknadu. S druge strane, s besplatnom verzijom niste u mogućnosti postavljati vlastite resurse na online trgovinu resursa, pa samim time niti zarađivati. Unreal Engine 4 nema besplatnih i profesionalnih verzija. Prije je koštao 19 dolara mjesečno, no od ožujka 2015. godine potpuno je besplatan te se uz njega dobiva otvoren kod napisan u C++ programskom jeziku. Onoga trenutka kada službeno izdate igricu, nakon prvih 3 000 zarađenih dolara, tvrtki koja stoji iza Unreal Engine 4 morate platiti 5% ukupnih prihoda za svaki proizvod koji ostvaruje profit. Osim toga, isti postotak naplaćuje se i za sve ostale prihode, kupnje i reklame unutar igre. Unreal Engine 4 je potpuno besplatan za fakultete i sveučilišta te nema pravnih problema oko toga. Koristiti ga mogu profesori i studenti, no ista ona naknada od 5% ostaje kao i za svakog korisnika. Generalno gledajući, za osobu koja profesionalno razvija igrice, obje su opcije relativno povoljne te je teško u dugom roku procijeniti koje bi razvojno okruženje bilo isplativije [1], [5], [23].

Za razvoj svake igre potrebno je poznavanje nekog programskog jezika. Ovisno o preferencama pojedinca, ovo bi čak mogao biti presuđujući faktor pri odabiru razvojnog okruženja. Unity koristi C#, JavaScript ili Boo, no nije potrebno koristiti isključivo jedan od navedenih, mogu se ukomponirati sva tri, što daje veći opseg mogućnosti i prilagodljivosti prilikom programiranja. Unreal Engine prije je koristio UnrealScript, koji i nije bio previše poznat u programerskoj zajednici. Danas koristi svima dobro poznati C++ programski jezik za kojega mnogi tvrde da je pomalo zastario, dok i dalje postoji velika baza ljudi koji ga preferiraju i podržavaju. Za sve one koji ne znaju naprednu razinu C++ , Unreal Engine 4 ponudio je Blueprint, vizualni editor skripti (eng. *Visual Script Editor*). Tehnički, nije potrebno napisati niti jednu liniju koda, što je pogodno za implementiranje brzih prototipova igara, no čak se mogu razviti potpune igre na ovaj način. Naravno, pritom postoje određena ograničenja koja na ovaj način nije moguće izvesti, no ako igrica sadrži vizualizacije i arhitekturalni pristup, onda je Blueprint odličan izbor. Unity još uvijek nema integriran ovakav sustav, no postoje odlične alternative u online trgovini resursa (eng. *Unity Asset Store*). Primjerice, najpoznatiji su Playmaker te u Script Professional, no potrebno je izdvojiti određenu svotu novca za iste. Iz navedenog možemo zaključiti da je Unreal Engine 4 možda bolji izbor početnicima koji žele besplatno razviti igricu bez pisanja koda, no ako pojedinac traži posebne mogućnosti za koje je potrebno pisanje koda, onda bi Unity možda bio praktičniji izbor [1], [5], [12], [25].



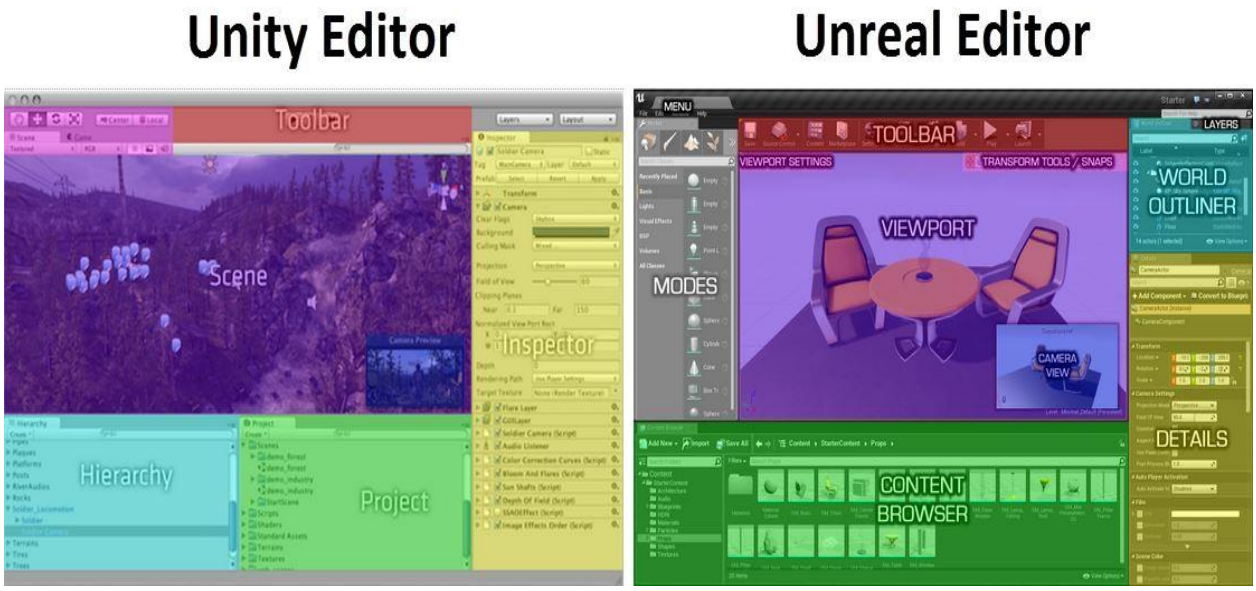
Slika 3.8. Blueprint povezivanje u Unreal Engine 4

(Mayden, 2014)

Unity i Unreal Engine 4 imaju implementiranu online trgovinu s resursima (eng. *Asset Store*) u kojoj se mogu kupovati ili prodavati resursi potrebni za izradu igrice, pa čak i u potpunosti implementirana rješenja. Postoje i brojni besplatni resursi koji se mogu preuzeti i jednostavno implementirati u vlastitu igricu. Primjerice, trodimenzionalni humanoidni karakteri s gotovim animacijama, nasumični objekti koji mogu biti potrebni, okoline koje možete koristiti za razvoj igre, brojni efekti, teksture i zvukovi. Unity-eva trgovina s resursima ima preko petnaest tisuća veoma povoljnih resursa te na njemu 1.5 milijuna aktivnih korisnika. Resursi su uistinu raznoliki, 3D modeli, animacije, audio isječci i glazbene pozadine, gotovi projekti, ekstenzije za razvojno okruženje, brojni efekti, isprogramirane skripte za razne funkcionalnosti, teksture i materijali te mnogo drugih stvari. Postoje gotovi generatori za animacije, grafičko sučelje, umjetnu inteligenciju neprijateljskih objekata pa čak i razvojni okviri za specifične tipove igara, kao što su strategijske i potezne igre. Unreal Engine 4 trgovina s resursima je puno kasnije implementirana te je stoga puno manja. Generalno gledajući i resursi su malo skuplji, no svi su izuzetno visoke kvalitete. Ukoliko pojedinac želi prodavati resurse putem online trgovine s resursima, oba razvojna okruženja od prodaje uzimaju određeni profit koji iznosi 30% od postavljene cijene [1], [5], [12].

Učenje oba razvojna okruženja jednostavno je uz malo volje, truda i motivacije zahvaljujući aktivnim zajednicama koje će odgovoriti na svako postavljeno pitanje na službenim stranicama. Unity i Unreal Engine 4 pobrinuli su se za mnoštvo dostupnih lekcija kojima se na jednostavan i interaktivan način upoznaje s njihovim razvojnim okruženjem. Postoje i pisane dokumentacije popraćene adekvatnim grafičkim prikazima koji olakšavaju snalaženje unutar

programa. Sudeći prema subjektivnom mišljenju većine, Unreal Engine 4 ima transparentniju i kvalitetnije napisanu dokumentaciju s brojnim objašnjenjima i priloženim slikama. S druge strane, Unity ima prostora za popravljanje određenih propusta u dokumentaciji. Međutim, taj propust nadoknadili su brojnim objašnjenjima i lekcijama putem video materijala gdje pokrivaju svaki aspekt izrade brojnih vrsta igara. Unity i Unreal Engine 4 s vremena na vrijeme znaju održavati otvorena predavanja kroz koja se razvija konkretna igra te ih često snimaju i koriste kao edukativni materijal početnicima koji tek kreću učiti u svojim domovima [1], [5], [12].



Slika 3.9. Korisničko sučelje Unity i Unreal Engine razvojnih okruženja (Unity 5 vs Unreal Engine 4, 2015)

Korisničko sučelje stvar je subjektivne procjene, no ipak je Unity generalno poznatiji kao lakši za korištenje te puno intuitivniji. Unreal Engine svojim četvrtim izdanjem izuzetno se popravio, no i dalje se nalazi na drugom mjestu sudeći prema korisničkim iskustvima. Oba razvojna okruženja imaju slična korisnička sučelja s alatnim trakama i postavkama unutar skalabilnih, pomičnih prozora. Glavna zamjerka Unreal Engine 4 korisničkom sučelju jest višak koraka i trošenje dodatnog vremena za vrlo jednostavne zadatke te potrebno dulje vrijeme da se uvezu i spreme resursi, što su najčešće radnje prilikom izrade igrice. Unity je po tom pitanju puno brži, a korisničko sučelje je vrlo responzivno i intuitivno. Po pitanju performansi također nije zahtjevno jer se može pokrenuti na Windows XP (SP2), dok je Unrealu Engineu 4 potreban barem Windows 7 (64 bitni). Iako možda konačan proizvod izgleda puno bolje u Unreal Engineu 4, put do njega je puno dulji te je potrebno uložiti puno više truda, pogotovo za početnike [12].

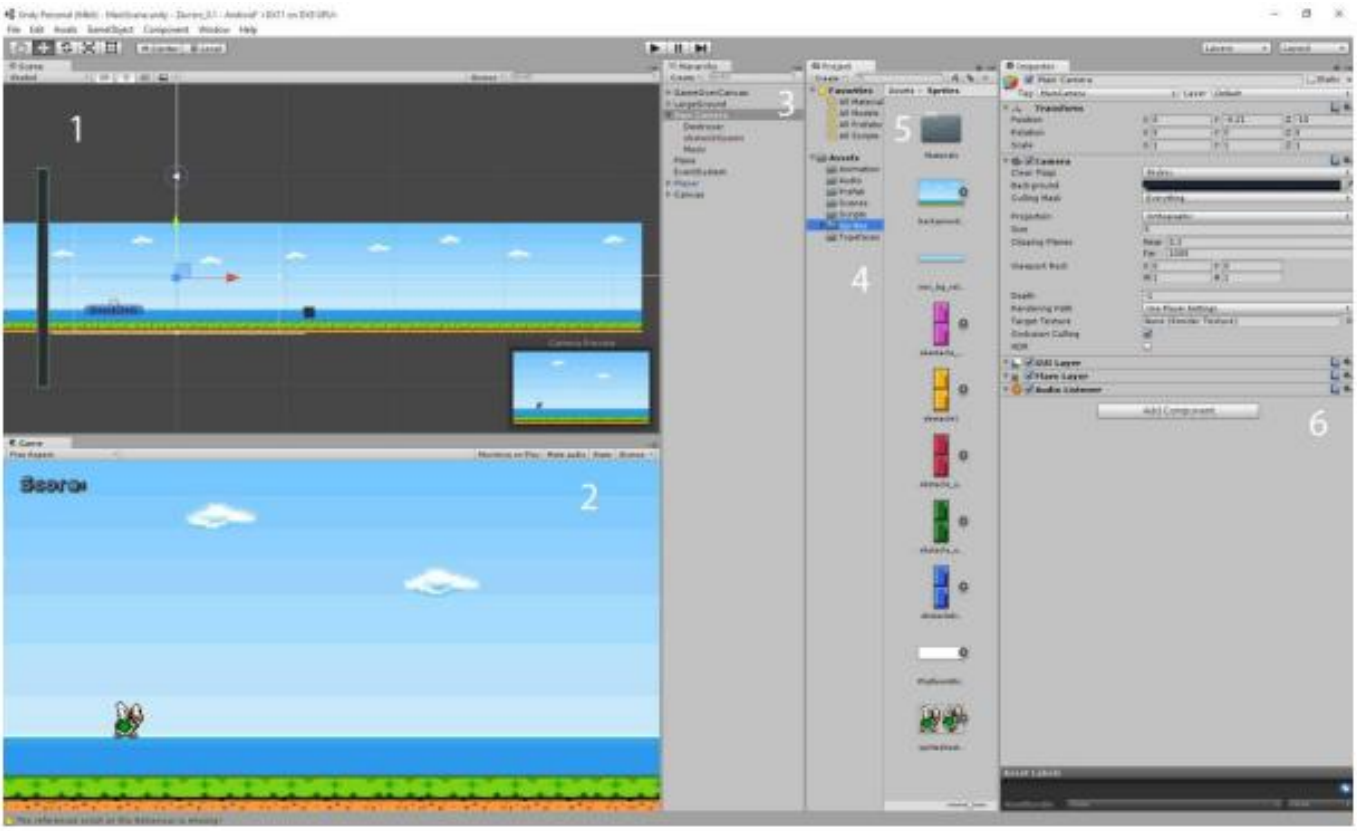
Za kraj bi valjalo samo napomenuti da Unity podržava daleko više platformi. Unity podržava ukupno 21 platformu, kao što je već i spomenuto. Platforme su: iOS, Android, Windows Phone, Tizen, Windows, Windows Store Apps, Mac, Linux ili Stream OS, WebGL, PlayStation 4, PlayStation Vita, Xbox One, Xbox 360, Wii U, Nintendo 3DS, Oculus Rift, Google Cardboard, Stream VR, PlayStation VR, Gear VR, Microsoft Hololens, Android TV, Samsung Smart TV te tvOS [23].

Unreal Engine podržava svega 10 platformi: Windows, Mac OS X, iOs, Android, VR, Linux, SteamOS, Xbox One te PlayStation 4 [12].

U zaključku ove generalne usporedbe svakako bih rekao da su oba alata izvanredna te posebna na vlastiti način. Generalni odabir varirat će od osobe do osobe, ovisno o vrsti igre koju žele izraditi, bila to jednostavna dvodimenzionalna arkadna igrica ili trodimenzionalna pucačina, slagalica s osnovnom fizikom ili web igrica. Navedene su prednosti i nedostaci za svaku vrstu igre, no za dodatne detalje lako se obratiti zajednici jer su svima dovoljno dobro poznate pojedine prednosti svakog od razvojnih okruženja. Za izradu mobilnih igrica definitivno bih preporučio Unity zbog svoje dominacije u svim aspektima tog područja. Sve ostalo je dosta varijabilno, ovisno o željenim dugoročnim planovima. Unity je jako fleksibilan i puno se korisnika može vrlo jednostavno prilagoditi te je uz to jako dobro razvojno okruženje za prosječno računalo. Ima ogromnu bazu korisnika i ljudi koji su spremni pomoći pri svakoj prepreci, osim toga i odličnu edukacijsku pozadinu kroz online tečajeve i preglednu dokumentaciju. Brojni dostupni resursi olakšat će potrebno vrijeme izrade igrice, no uz to i C# programski jezik koji je dosta praktičan i prilagodljiv za razvoj igara. S druge strane, Unreal Engine 4 je bolji izbor za ljude koji ne žele pisati puno koda i dugo se zamarati pojedinim komponentama, ali i za iskusne programere koji žele da im krajnji rezultat grafički izgleda izuzetno detaljno i kvalitetno. Omogućuje apsolutnu slobodu za modifikacijom radnog prostora jer je razvojno okruženje otvorenog koda. Glavna razlika Unity-a i Unreal Engine 4 je činjenica da je Unity više profesionalno orijentiran, no zahtjeva poznavanje programske logike, odnosno određena predznanja u području programiranja. S druge strane, Unreal Engine 4 besplatno pruža gotove okoline s implementiranim protivnicima, trkaće staze, oružja i mnoge druge resurse, no problem je što se onda mora raditi s dobivenim detaljima bez previše modifikacija.

3.4. Korisničko sučelje Unitya

Korisničko sučelje Unity-a možemo podijeliti na šest različitih prozora, kao što je vidljivo na Slici 3.10. U ovom djelu će ukratko biti objašnjen svaki prozor.



Slika 3.10. Korisničko sučelje Unity-a – objašnjenje

Prozor pod brojem jedan je prozor scene. Ovaj prozor služi za kreiranje scene igre, odnosno za pozicioniranje objekata koje će se koristiti u igri. Ako se radi 2D igra, ovaj prozor je ograničen na 2D prostor, odnosno ima samo x i y os. Prebacivanjem prozora u 3D dobivamo z os, odnosno dubinu. Z os također može biti korisna u 2D igri jer pomoću nje možemo kontrolirati slojeve (eng. *layer*) u igri poput slojeva u Photoshopu.

Prozor broj dva je prozor igre. Ovaj prozor prikazuje izgled igre kada se pokrene. Točnije ovaj prozor prikazuje igru iz perspektive glavne kamere igre. U ovom slučaju vidimo kako igra izgleda kada se pokrene. Možemo primijetiti razliku između izgleda scene i igre.

Treći prozor možemo nazvati prozorom objekata. Ovaj prozor prikazuje sve objekte koji se nalaze u trenutnoj sceni. Također, unutar njega možemo kreirati nove objekte te ih grupirati unutar drugih objekata.

Četvrti prozor je prozor koji prikazuje sve mape za trenutni projekt, ovaj prozor možemo grupirati s petim prozorom koji prikazuje sadržaj mape koja je trenutno odabrana. Ukratko ovi prozori nam služe za organiziranje resursa igre te pristupanjem njima kako bi mogli modificirati neke njihove karakteristike. Što nas dovodi do šestog prozora.

Šesti prozor je prozor karakteristika (eng. *properties*), ovaj prozor u Unity-u služi za dodavanje, brisanje te kontroliranje svih karakteristika odabranog objekta. U ovom prozoru možemo dodavati karakteristike poput collidera koji služi da kontroliranje sudara s objektima, dodavanje tekstura ili sličica objekta za definiranje izgleda objekta, pa sve do dodavanja animacija i slično.

Ovo je bio ukratko pregled korisničkog sučelja Unity-a. Budući da je Unity velik program, detaljno opisivanje svake pojedinosti bi zauzelo previše mjesta, a i premašilo temu ovog rada. Više informacija i detalja o Unity-u se može naći na njihovim službenim stranicama (https://unity3d.com/learn) gdje postoje mnogi videi koji opisuju razne pojedinosti samog programa.

3.5. Osnovni koncepti Unity alata

Ovo poglavlje objašnjava osnovne koncepte koji su se koristili u izradi praktičnog dijela završnog rada. Radeći igru u Unity alatu koristi se puno različitih koncepata i bitno se razlikuju ako igra koju radimo je 2D ili 3D. Iz razloga što igra u ovom radu je 2D, sljedeća potpoglavlja će predstavljati osnovne koncepte bitne za kreiranje 2D igre.

3.5.1. Objekti u igri

Objekti u igri su osnovni i najvažniji objekt. Ako korisnik želi i razumjeti i znati raditi u Unity alatu, mora znati što su objekti u igri i kako oni rade. Svaki objekt koji se nalazi u igri je objekt, ali sam objekt sam po sebi ne radi ništa. Kako bi imali određenu funkciju nužno je da objekti imaju posebna svojstva koja im omogućuju da predstavljaju likove, dijelove okoliša ili efekte poput svjetla. Način na koji ti objekti funkcioniraju je jednostavan, oni su zapravo spremnici koji sadrže niz komponenata (eng. *Component*) koji svaki od njih ima određenu svrhu. S obzirom na to kakav objekt želimo napraviti, koristimo razne kombinacije komponenata i sam Unity ima mnogo različitih vrsta koji su unaprijed ugrađeni. Korisnici također mogu kreirati svoje vlastite komponente pomoću skripta i o tome će biti riječ u sljedećim poglavljima [16].

3.5.2. Kamera

Kamere su objekti koji bilježe i prikazuju svijet igraču. Prilagođavanjem i manipulacijom komponenata kamere, korisnik može prezentirati igre na razne jedinstvene načine. U jednoj sceni može biti neograničen broj kamera i mogu se primjenjivati u bilo kojem redoslijedu ili mjestu na ekranu. Kamere su neophodne za prikazivanje igre samom igraču [24].

3.5.3. Gotovi resursi

Prilikom izrade određene scene prikladno je kreirati objekt i dodati mu komponente i podesiti im svojstva. U slučaju kada želimo imati više takvih objekata poput lika kojim ne upravlja igrač ili dio okoliša scene, tada nam može biti komplicirano ponovno kreirati takve objekte u slučaju da želimo više istih. Kako bi se taj problem što lakše riješio, Unity nam nudi kreiranje gotovih (eng. *Prefab*) resursa koji omogućuju spremanje već kreiranih objekata kako bi ih mogli po potrebi ubacivati u scenu. Gotove resurse spremamo u mape projekta i kada želimo instancirati u scenu imaju određene vrijednosti komponenata objekta. Te vrijednosti se mogu prilagođavati po potrebi i svaki novi objekt ubačen u scenu može imati na taj način jedinstvena svojstva [18].

3.5.4. Scene

Scene sadržavaju određen broj i raspored objekata u određenom stadiju igre. Mogu se koristiti za izradu izbornika, pojedinih razina (eng. *level*) igre ili bilo čega drugog. Jednu individualnu scenu možemo gledati kao jednu jedinstvenu razinu. U svakoj sceni korisnik može polagati dijelove okoliša, likove, dijelove korisničkog sučelja ili bilo koje druge tipove objekata kako bi izradio željeni dio same igre. Scene se spremaju u obliku datoteka u mapama projektnih resursa [19].

3.5.5. Skripta

Skripte su jedne od najvažnijih dijelova svih igara. One sadržavaju programski kod koji pokreće određene akcije u igri te čak i najjednostavnije igre trebaju neku vrstu programskog koda. Najčešći primjer korištenja skripta je reagiranje na unos kontrola za kretanje igrača ili stvaranje događaja u određenim trenucima igre. Osim toga skripte se koriste i za kreiranje grafičkih efekata, kontrolu ponašanja objekata, pisanje umjetne inteligencije za likove kojima ne upravlja igrač itd. Ovo je možda najteži dio kreiranja igara te je potrebno općenito znanje programiranja i uloženo vrijeme da se savladaju funkcije i logika pisanja programskog koda za

Unity. Kako bi korisnik kreirao skriptu za Unity, potrebne su mu funkcije Unity alata i dokumentaciju koja objašnjava način njihovog korištenja. Dokumentacija se može naći na službenim stranicama Unity alata. Unity alat nam omogućuje pisanje skripta u tri jezika, UnityScript (JavaScript), C# i Boo. Kada želimo kreirati novu skriptu program nudi samo UnityScript i C# iz razloga što žele izbaciti programski jezik Boo zbog njegovog slabog korištenja među kreatorima igara [20].

3.5.6. 2D Sprite

Sprite je osnovni 2D grafički objekt. Ako je igra koja se izrađuje 3D, tada je Sprite obična tekstura koja se primjenjuje na neki 3D model. Svaki grafički elementi u 2D igri su zapravo 2D Spriteovi koje su dio projektnih resursa. Njih ubacujemo kao obične slike koje nakon toga izrezujemo pomoću Sprite Editor-a. Izrezane elemente primjenjujemo na objekte koji mogu biti likovi, dijelovi okoliša ili neki drugi [21].

Slika 3.11. Prikaz opcija nad slikom u kontrolnom prozoru koju možemo pretvoriti u Sprite (Unity Technologies, 2017)

3.5.7. GUI elementi

Slike korisničkog sučelja i tekstovi mogu biti prikazani pomoću „GUI Text“ i „GUI Texture“ objekata u Unity alatu. Svaki GUI element može imati svoju veličinu skalabilnost i poziciju. GUI elementi su zapravo objekti koji se prikazuju ispred svih ostalih objekata u sceni [22].

3.5.8. Zvuk

Baš kao i u stvarnom životu zvukovi imaju svoje izvore, a osluškivači ih primjećuju i slučaju. Zvukove koje želimo koristiti ubacujemo u projektne resurse kao i ostale, a možemo i unutar Unity alata izrađivati nove. Kako bi zvukove mogli iskoristiti potrebno je objektu koji treba biti izvor navedenog zvuka dodati Audio Source komponentu [15].

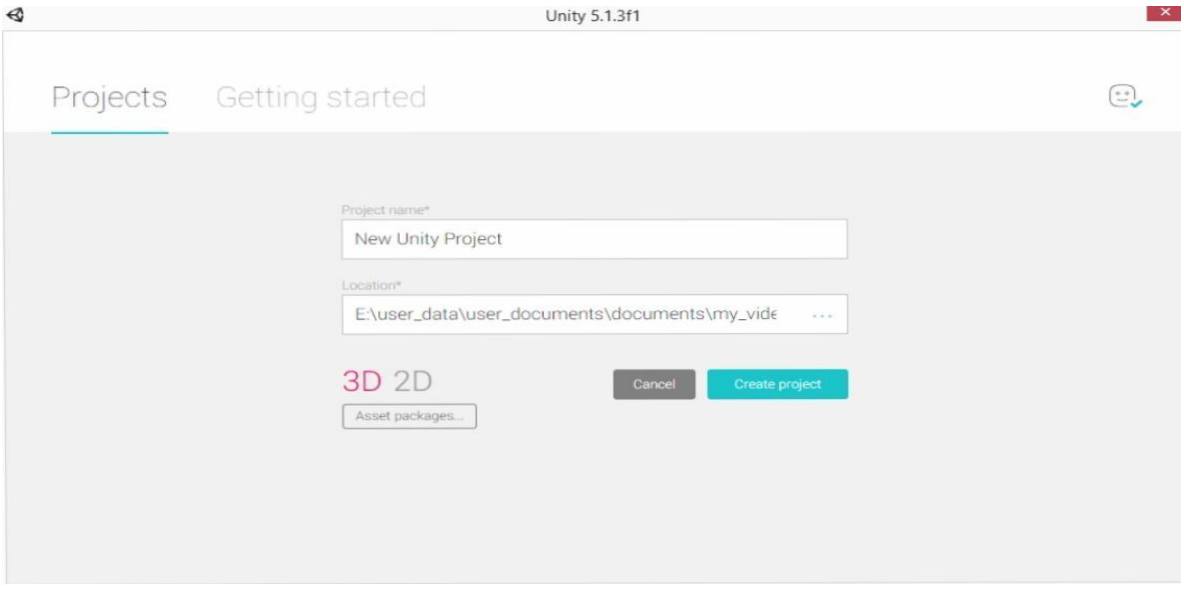
4. Izrada 2D igre

Kako bi izradili uspješnu 2D ili 3D igru moramo se pridržavati nekoliko ključnih koraka za izradu same igre.

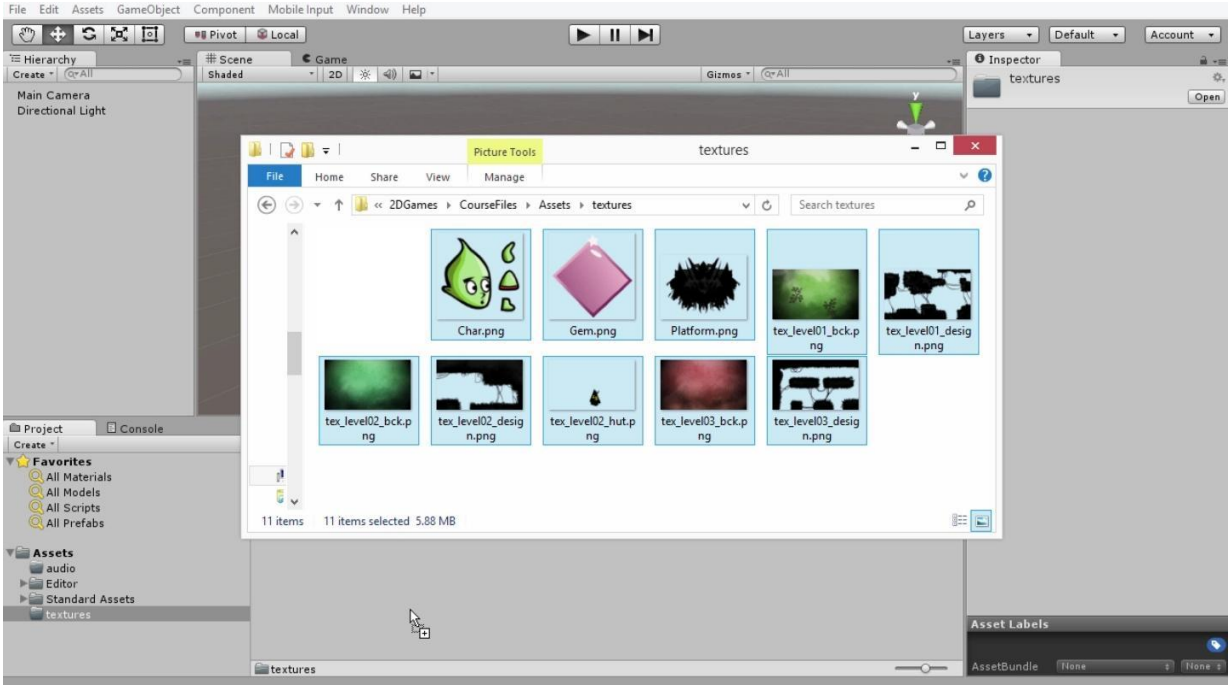
1. Izrada modela koji će se nalaziti u igri ili preuzimanje samih modela iz trgovine za modele koju je izradu unity (eng. *Asset store*)
2. Izrada i konfiguracija scena koje će se nalaziti u igri.
3. Animacija, programiranje objekata i igrača (eng. *Player*)
4. Dodavanje glazbenih efekata kako bi dodatno oživili igru.
5. Izrada grafičkog sučelja GUI (eng. *Graphical user interface*)

4.1. Izrada i prikupljanje 2D modela i objekata

Kao što smo prije napomenuli objekti su ključne sastavnice za izradu igre, u ovom primjeru igre izradio sam 2D glavnog lika, mapu te objekte u programu Adobe photoshop, te ih moramo uvesti u Unity kako bi ih mogli koristiti. Prvi korak koji moramo učiniti je naravno pokrenut program Unity i zatim odabrati dali radimo 2D ili 3D igru te pritisnuti Create project. Zatim kada smo kreirali projekt moramo uvesti naše objekte i glazbene efekte koji su rađeni u programu Adobe audio. Kako bi to uradili kreirat ćemo mape u projektnoj ploči (eng. *Project panel*) to uradimo tako da upotrijebimo desni klik i odaberemo create – folder za ubuduće lakše snalaženje, zatim otvorimo praznu mapu te jednostavno metodom povuci baci (eng. *Drag and drop*) ubacimo naše objekte, scene, i glazbene efekte.



Slika 4.1. Kreiranje projekta

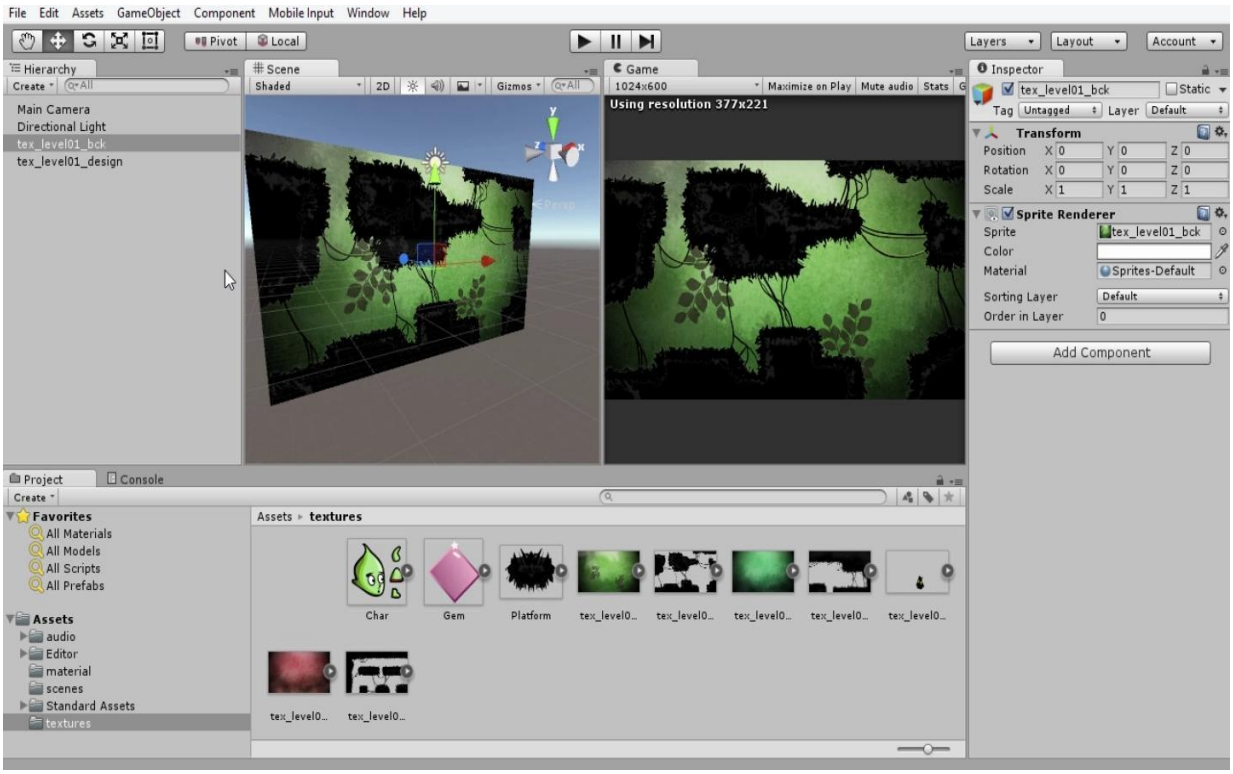


Slika 4.2. Ubacivanje elemenata

4.2. Izrada i konfiguracija scena

U prethodnim koracima ubacili smo potrebne materijale za izradu 2D igre te sada moramo naše slike (eng. *Texture*) prilagoditi programu Unity i konfigurirati ih kao 2D slike (eng. *Texture*). Kako bi to uradili prvo označimo sve naše slike pomoću ctrl tipke i klikom na sve slike, zatim u projektnoj ploči Inspector promijenimo Texture type u Sprite (2D and UI) te pritisnemo primjeni (eng. *Apply*). Unity će tada prepoznat prozirne (eng. *Transparent*) dijelove na slikama i prilagoditi slike kao 2D teksture.

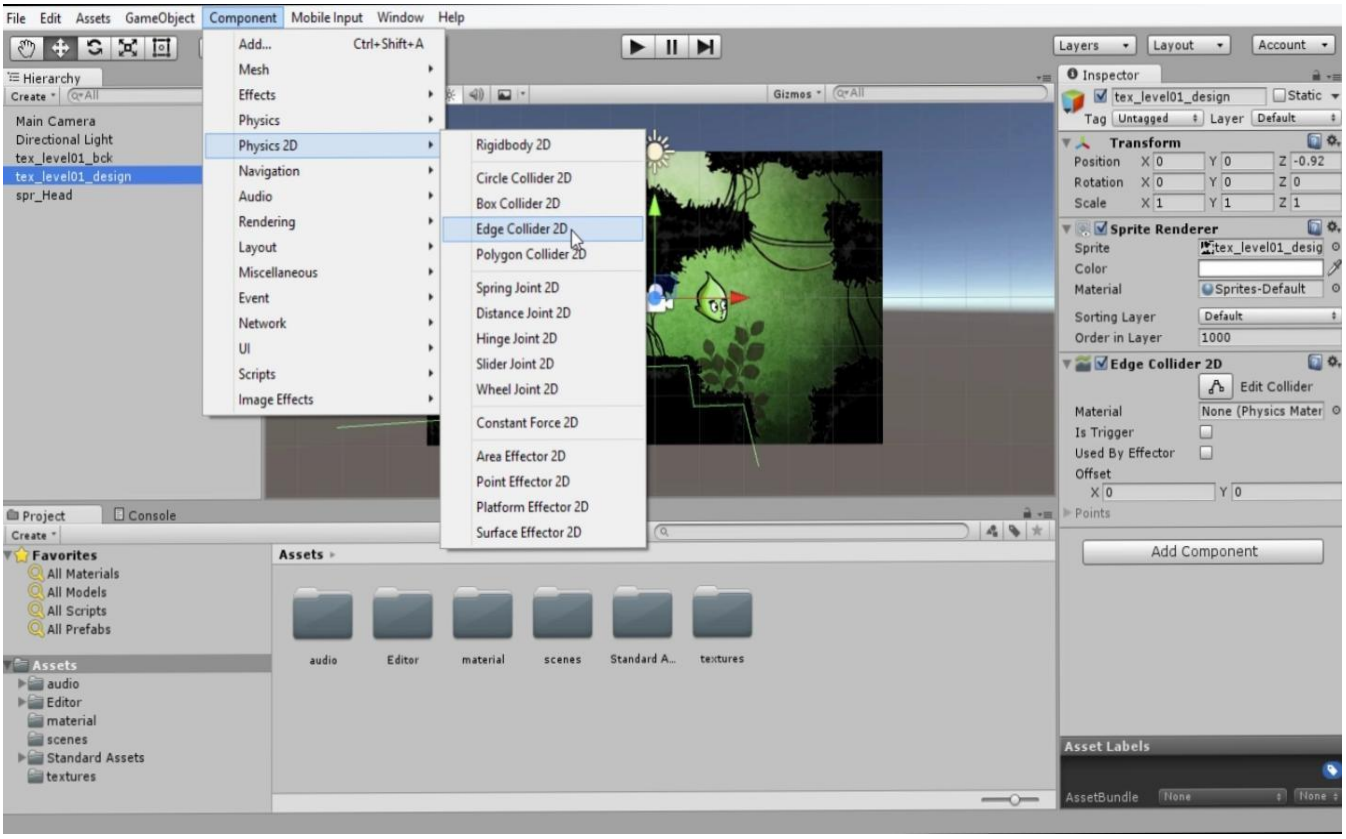
Slika 4.3. Konfiguracija slika



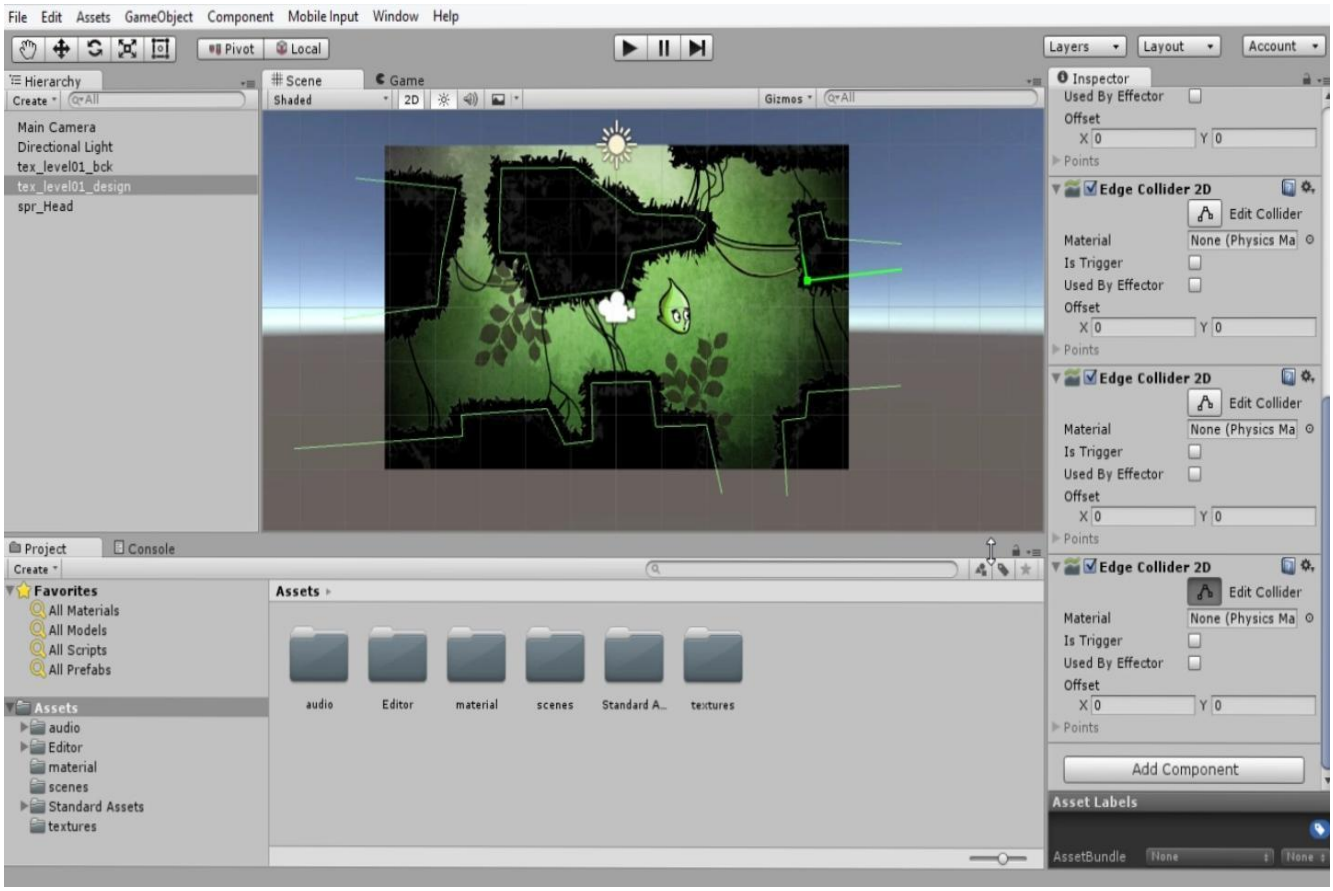
Slika 4.4. Konfiguracija slika

4.2.1. Edge Collider

Kako bi se naš lik mogao kretat po scenama moramo postavit linije (eng. *Edge collider*) po kojima će se moći kretati. U navigacijskoj traci odaberemo izbornik Component – Physic 2D - Edge Collider 2D – Edit Collider. Zatim linije (eng. *Edge Collider*) moramo postavit po rubovima naše scene, jednostavno pritisnemo desnom tipkom miša i počinjemo modelirat linije po rubovima scena.



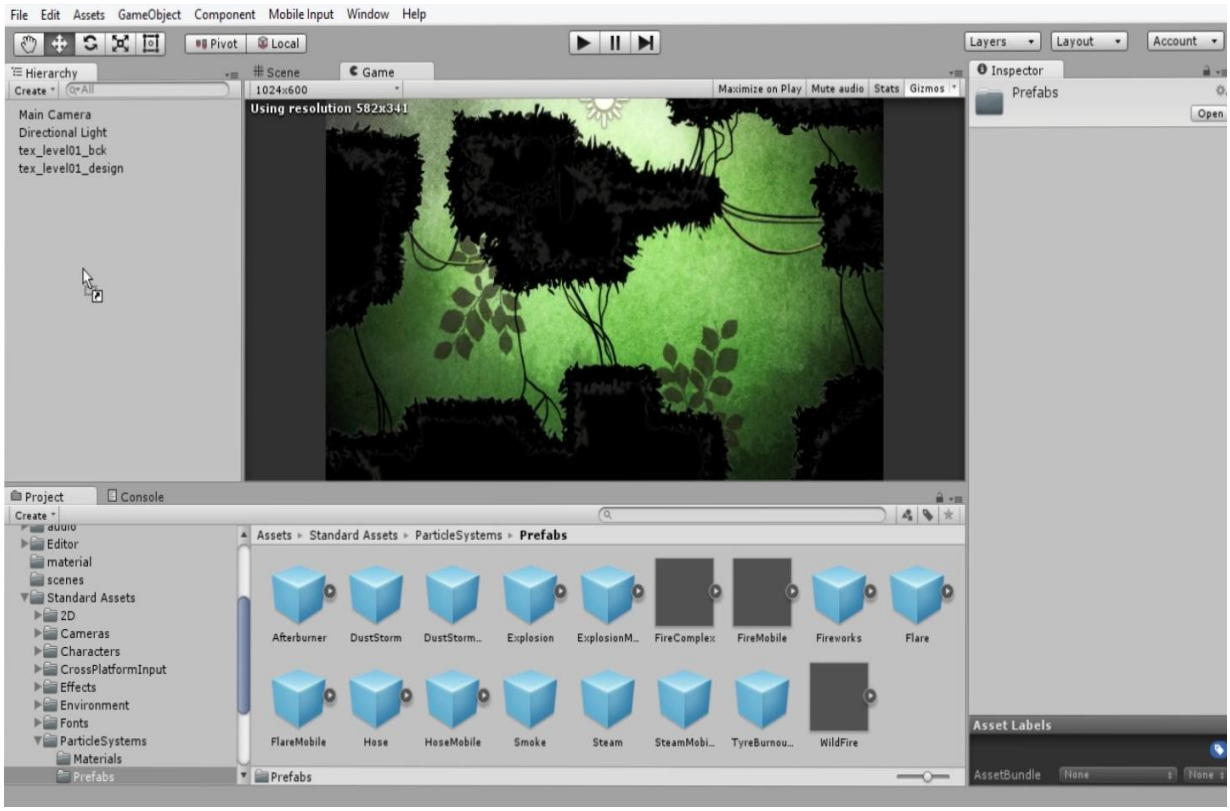
Slika 4.5. Kreiranje linija



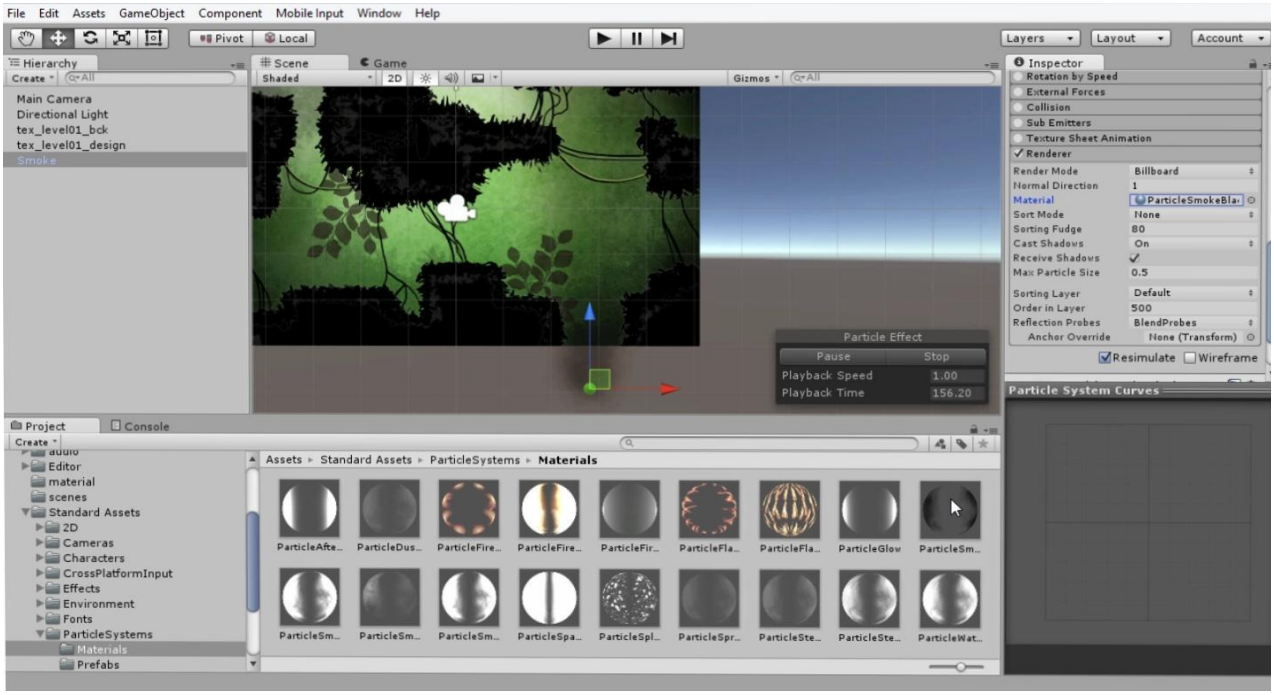
Slika 4.6. Kreiranje linija

4.2.2. Particle System

Da bi igra izgledala realnije i zanimljivije ubacit ćemo elemente kao što su dim, vatra itd. (eng. *Particle system*) u projektnoj ploči otvorimo mapu Standars Assets, te zatim otvorimo mapu Partycle System, unutra ćemo pronaći mnogo zanimljivih elemanata koje možemo jednostavno ubaciti metodom povuci baci. Također možemo manipulirati s elementima, dodavati im nove materijale, mjenjat boje itd.



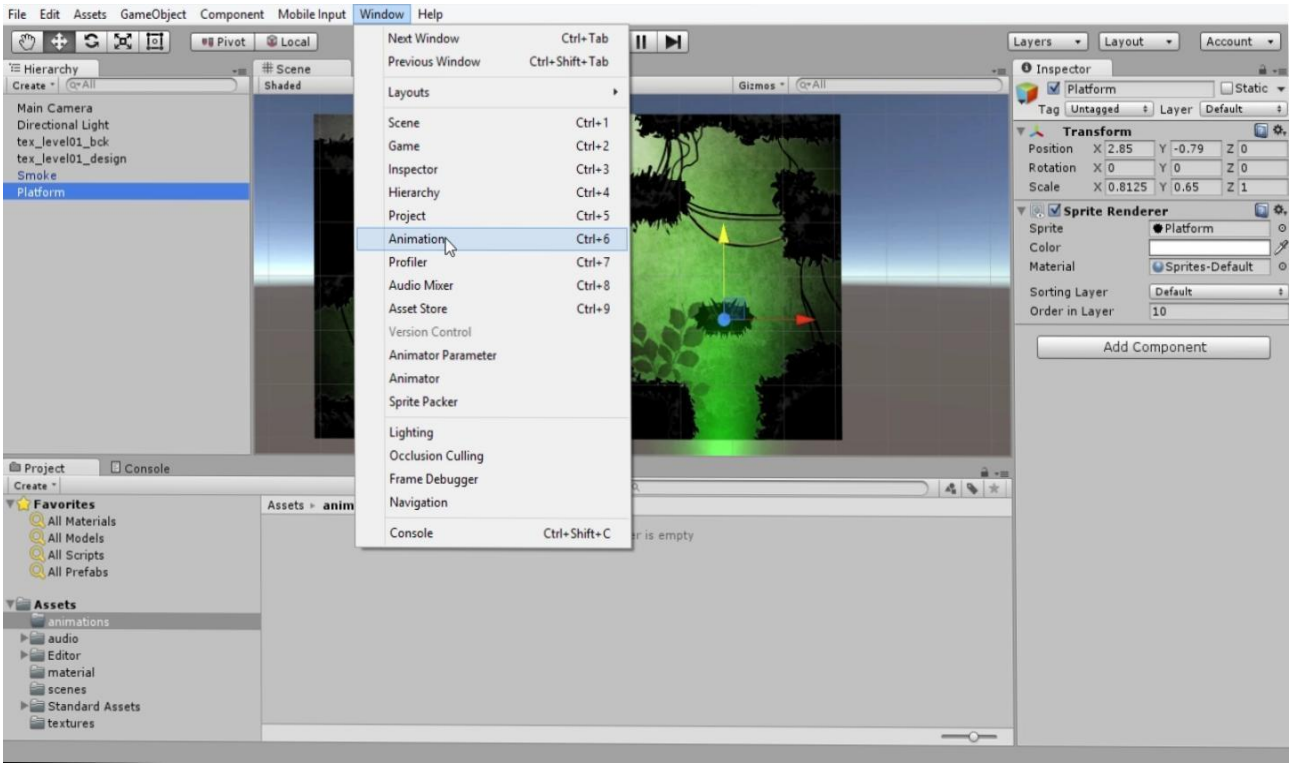
Slika 4.7. Ubacivanje elemenata



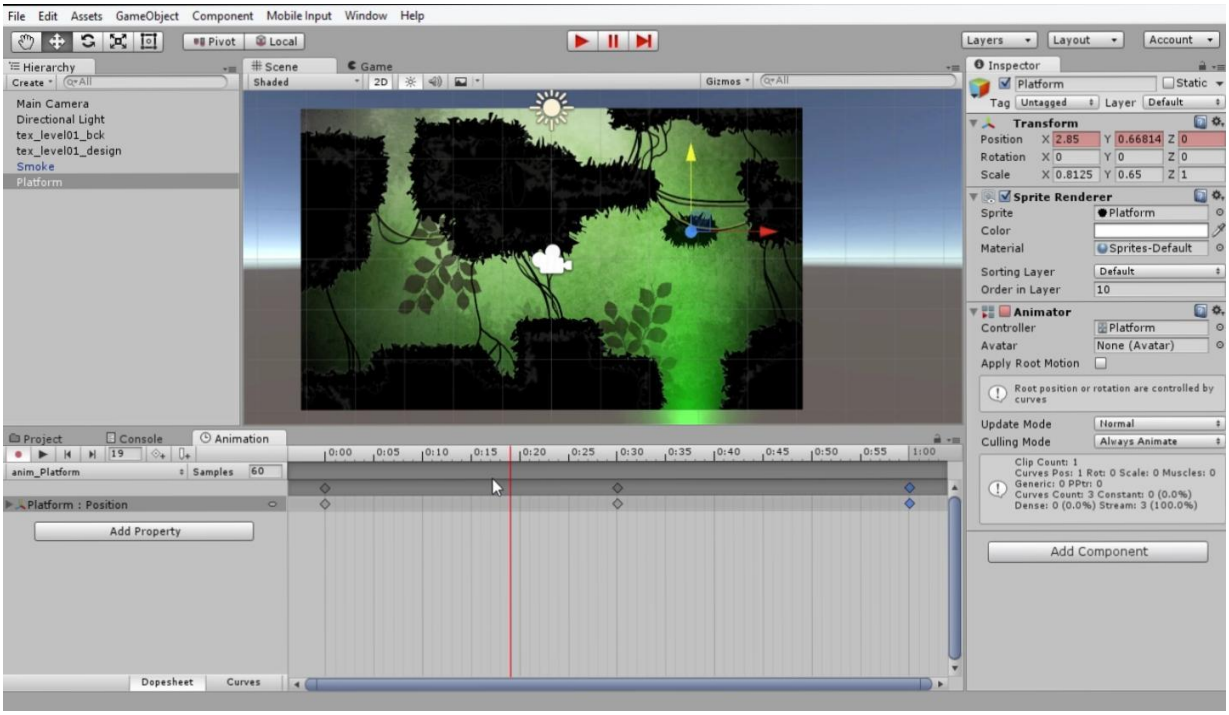
Slika 4.8. Ubacivanje elemenata

4.3. Animacija, programiranje objekata i igrača

U ovom poglavlju bit će objašnjeno kako animirati određene objekte i kako „oživiti“ lika. Prvo ćemo animirati objekt koji će se kretati dolje – gore između rupa kako bi igrač (eng. *Player*) mogao proći scenu (eng. *Level*). Kako bi napravili animaciju prvo kreiramo mapu u projektnoj ploči i nazovemo ju Animacije, zatim odaberemo objekt i otvorimo izbornik Window, – Animation otvorit će nam se novi prozor u kojemu je gumb (eng. *Button*) create animation pritisnemo ga kako bi snimili animaciju. Pojavit će se točkice (eng. *Key frame*) s kojima ćemo raditi animaciju. Prvi korak koji moramo učiniti je kopirati točkice i postaviti ih na početak i kraj vremenskog okvira (eng. *Timline*), zatim stavimo pokazivač miša u sredinu vremenskog okvira i pomaknemo naš objekt do najviše točke i snimili smo animaciju u kojoj će se objekt kretati gore – dolje. Zadnje što uradimo je u ploči Inspector promijenimo Speed na 0.5 da bi se objekt sporije kretao.



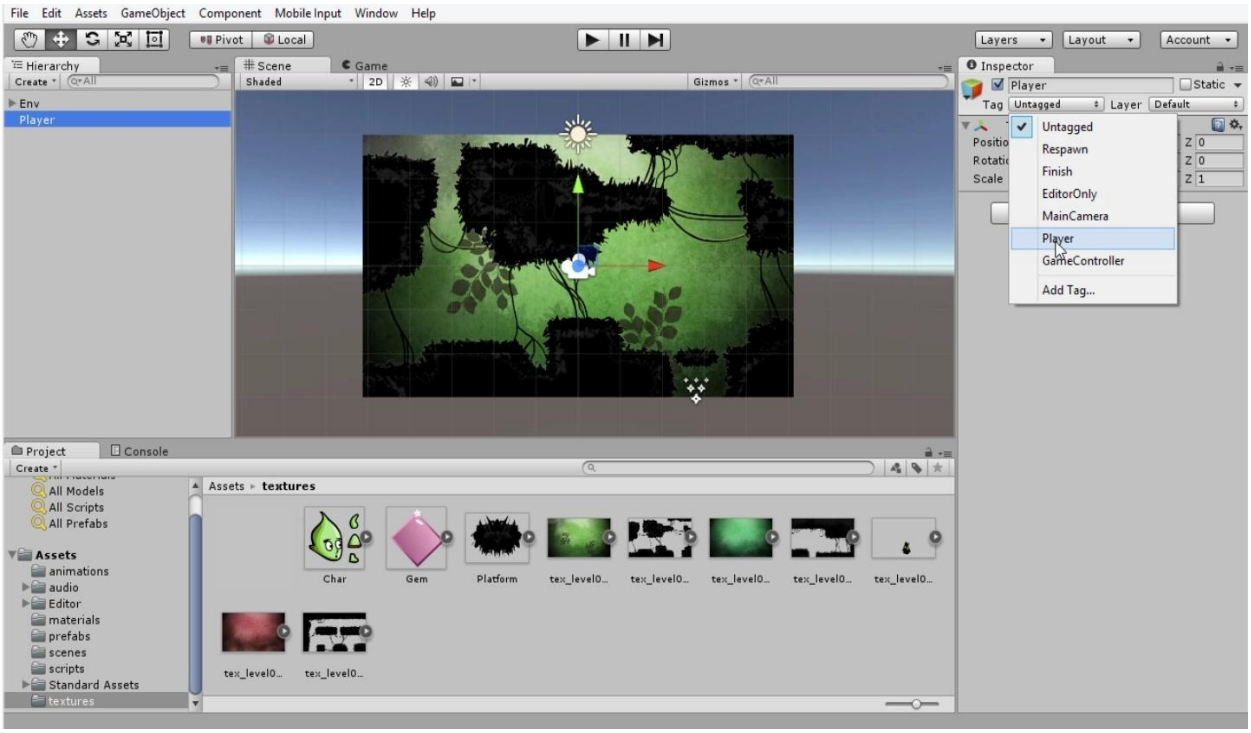
Slika 4.9. Animacija objekta



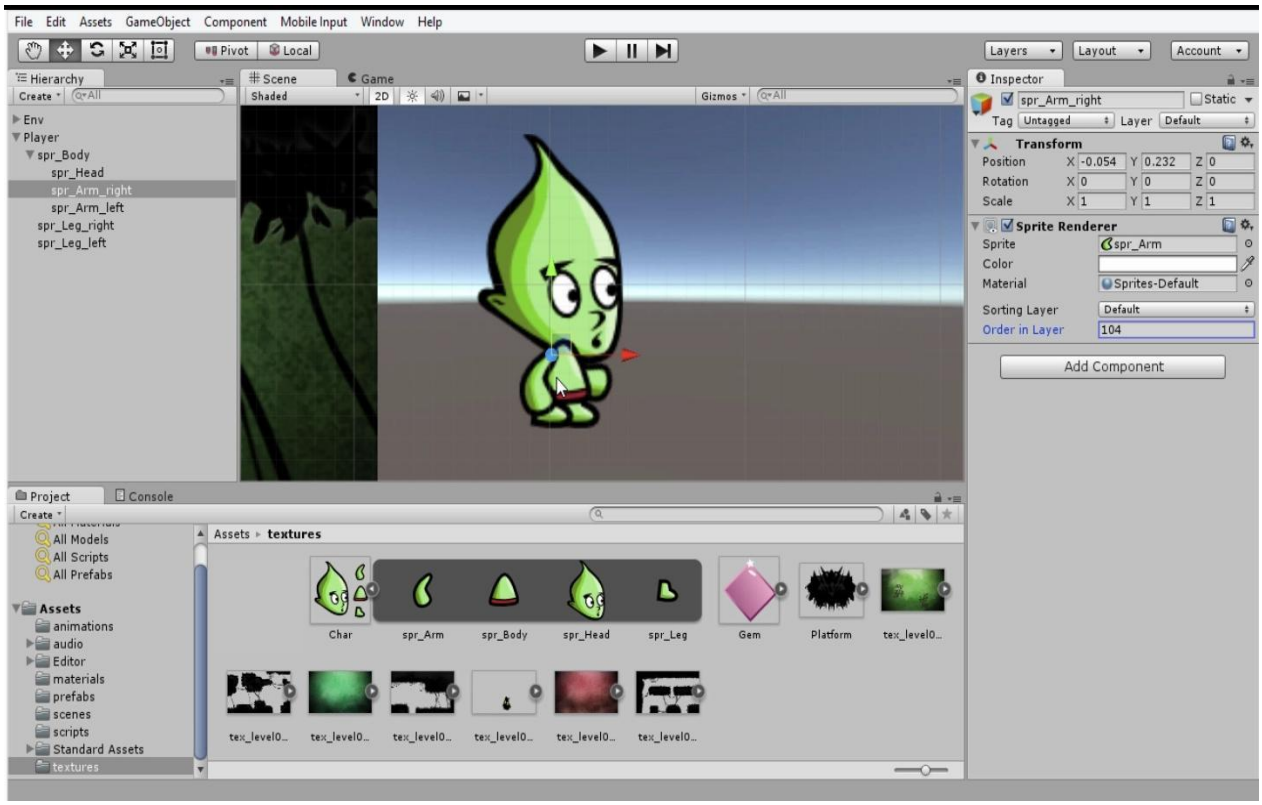
Slika 4.10. Animacija objekata

4.3.1. Izrada lika (eng. Building a charachter)

Da bi mogli animirati i programirati lika moramo prvo njegove dijelove (ruke, noge, tijelo, glavu) spojiti u jednu komponentu. To uradimo da prvo kreiramo objekt (eng. *Game object*) nazvat ćemo ga Player i zatim u Inspector ploči u izborniku Tag stavimo opciju Player. Sljedeći korak koji moramo učiniti je dodat elementu u objekt Player i tako izgradimo lika.



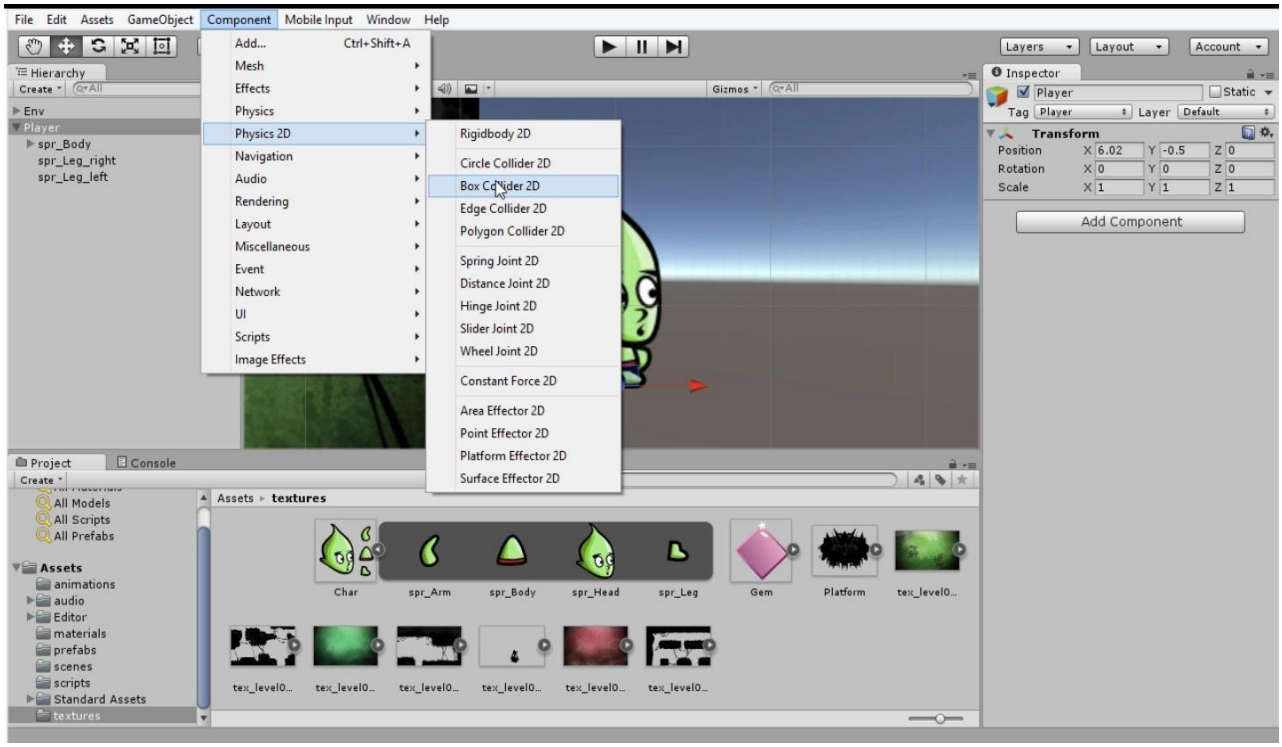
Slika 4.12. Izrada lika



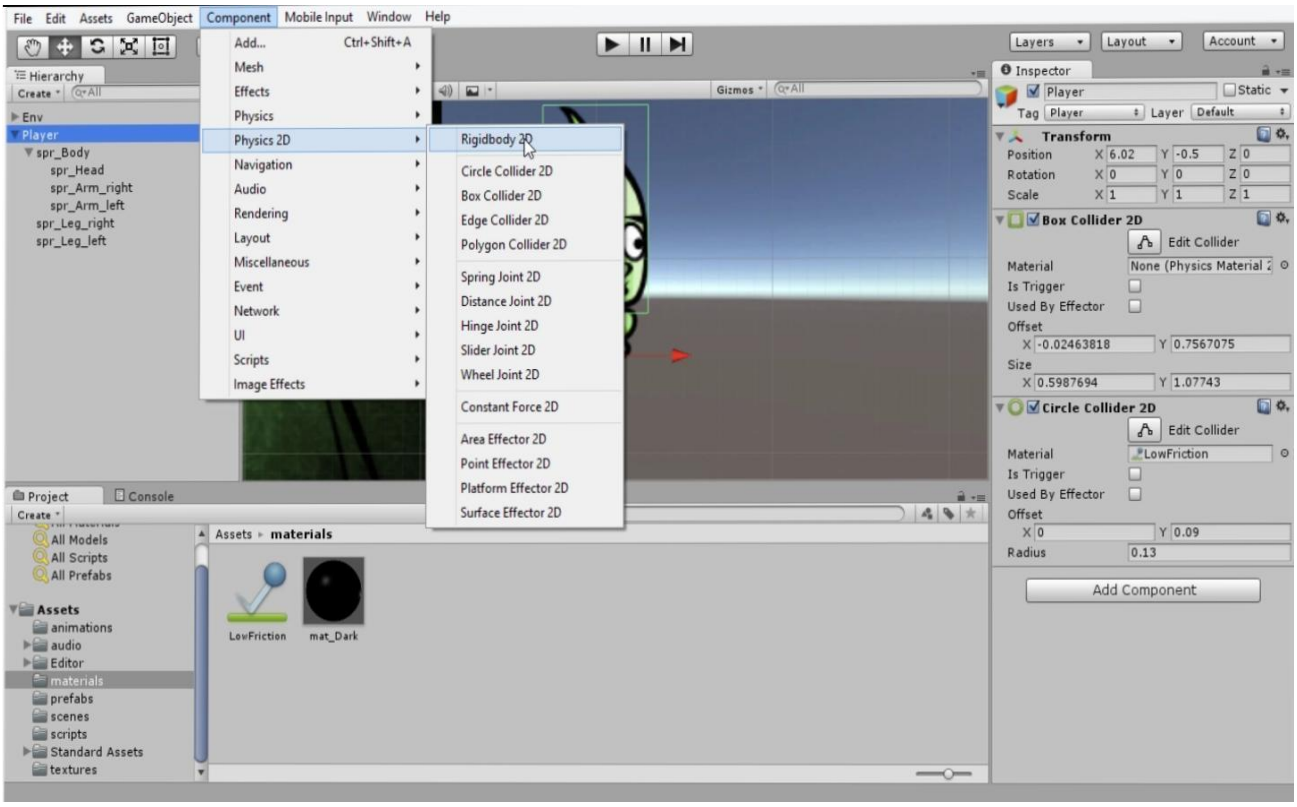
Slika 4.11. Izrada lika

4.3.2. Rubovi oko igrača (eng. Charachter Collider)

Lik ne može stajati na sceni, jer nismo odredili njegove rubove (eng. *Collider*) i nismo postavili gravitaciju (eng. *Gravity*) i fiziku (eng. *Physics*). Kako bi to uradili moramo u izborniku Component odabrati Physics 2D – Box Collider te zatim Edit Collider. Jednostavno postavimo rubove oko igrača tako da kliknemo na kvadrat i krug i povučemo na glavu i tijelo lika. Posljednje što moramo učiniti je dodati fiziku liku (eng. *Physics*) u izborniku Component – Pysics 2D – RigidBody 2D. Lik je sada spreman za programiranje kako bi se mogao kretati po scenama.



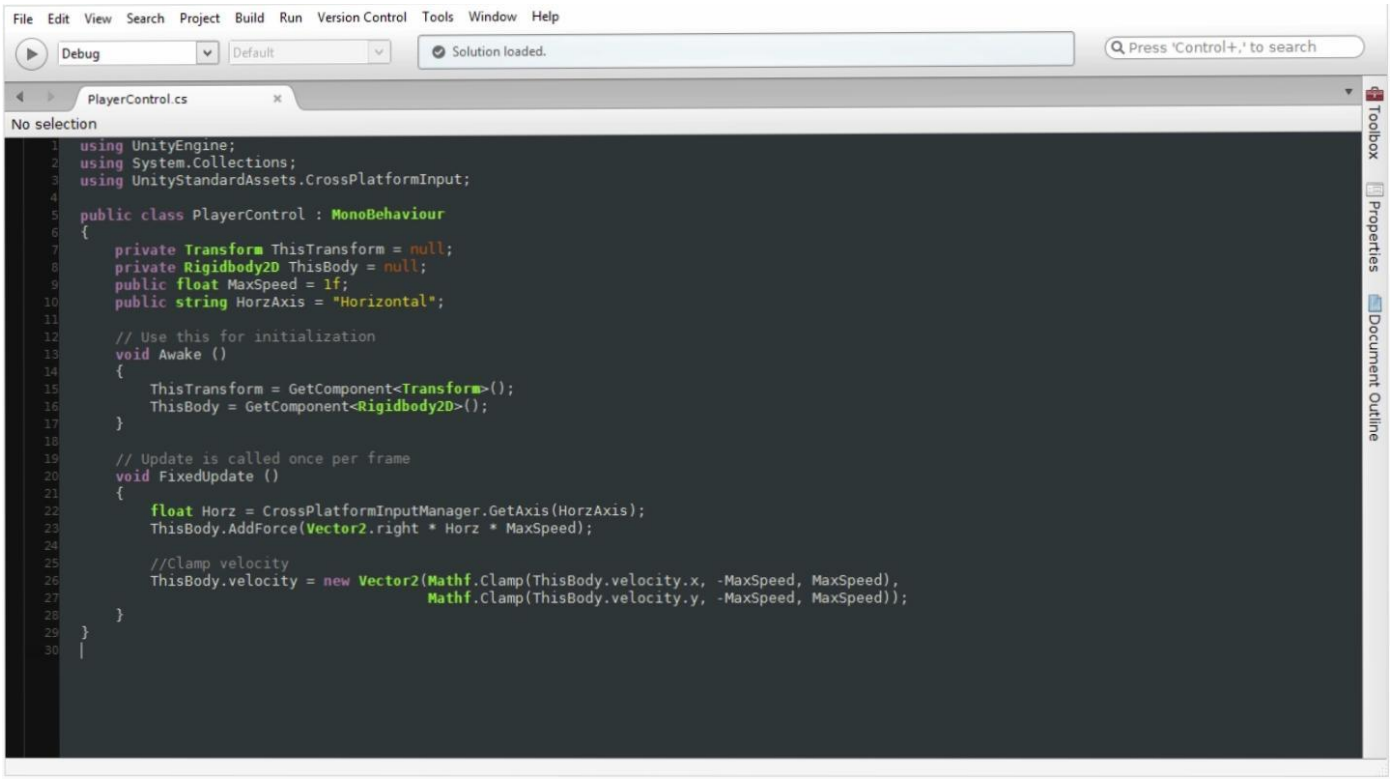
Slika 4.13. Određivanje rubova



Slika 4.14. Određivanje rubova

4.3.3. Programiranje lika (eng. *Player*)

Kako bi se lik mogao kretati po scenama moramo napraviti skriptu (eng. *Script*) u projektnoj ploči desnom tipkom miša i pritisnuti Create – C#Script-Player Script. Pojaviti će se novi prozor s početnim predloškom za kretanje lika (eng. *Template*) skriptom.



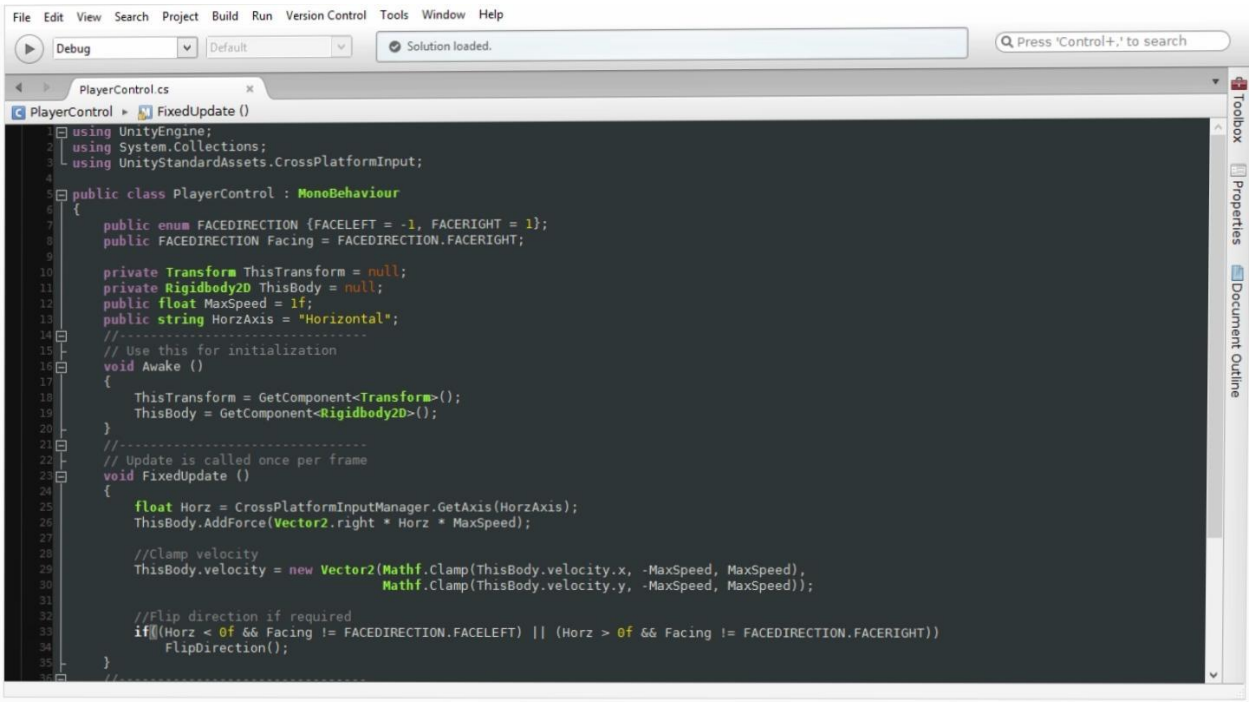

```
using UnityEngine;
using System.Collections;
using UnityStandardAssets.CrossPlatformInput;

public class PlayerControl : MonoBehaviour
{
    private Transform ThisTransform = null;
    private Rigidbody2D ThisBody = null;
    public float MaxSpeed = 1f;
    public string HorzAxis = "Horizontal";

    // Use this for initialization
    void Awake ()
    {
        ThisTransform = GetComponent<Transform>();
        ThisBody = GetComponent<Rigidbody2D>();
    }

    // Update is called once per frame
    void FixedUpdate ()
    {
        float Horz = CrossPlatformInputManager.GetAxis(HorzAxis);
        ThisBody.AddForce(Vector2.right * Horz * MaxSpeed);

        //Clamp velocity
        ThisBody.velocity = new Vector2(Mathf.Clamp(ThisBody.velocity.x, -MaxSpeed, MaxSpeed),
                                        Mathf.Clamp(ThisBody.velocity.y, -MaxSpeed, MaxSpeed));
    }
}
```

Slika 4.15. Programiranje lika

Kako bi kretanje lika izgledalo realističnije dodat ćemo par varijabli i petlju If da lik pogleda u desno kada se kreće i također u lijevo. Moramo kreirat varijablu:

```
public enum FACEDIRECTION { FACELEFT = -1, FACERIGHT = 1};
public FACEDIRECTION Facing = FACEDIRECTION.FACERIGHT;
If((Horz < 0f && Facing != FACEDIRECTION.FACELEFT) || (HORZ >0f && Facing
!=FACEDIRECTION.FACERIGHT)) Flip.Direction();
private.void Flipdirection()
{
Facing = (FACEDIRECTION) ((int) Facing *-1f);
Vector3 LocalScale = ThisTransform.LocalScale;
LocalScale.x *= -1f;
ThisTransform.LocalScale = LocalScale;
}
```

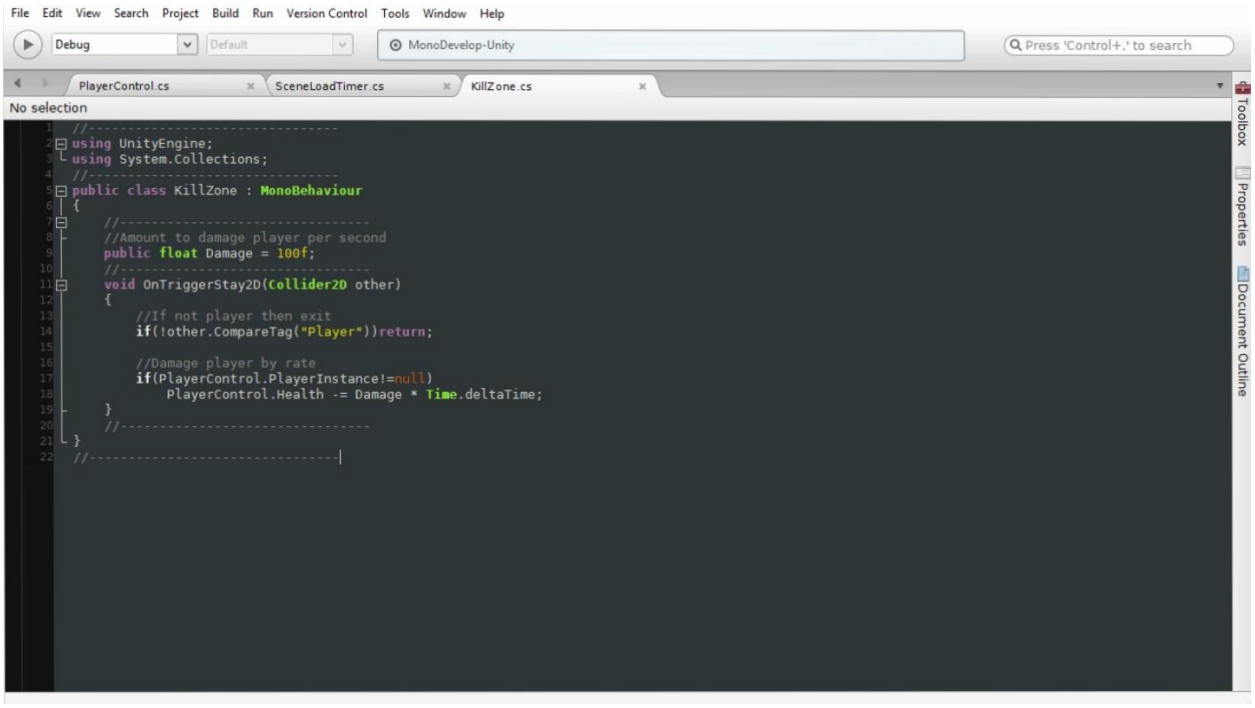
S ovim linijama koda postigli smo da naš lik gleda u lijevo i desno kada se kreće scenama.



Slika 4.16. Programiranje lika

4.3.4. Zona smrti (eng. *Kill zone*)

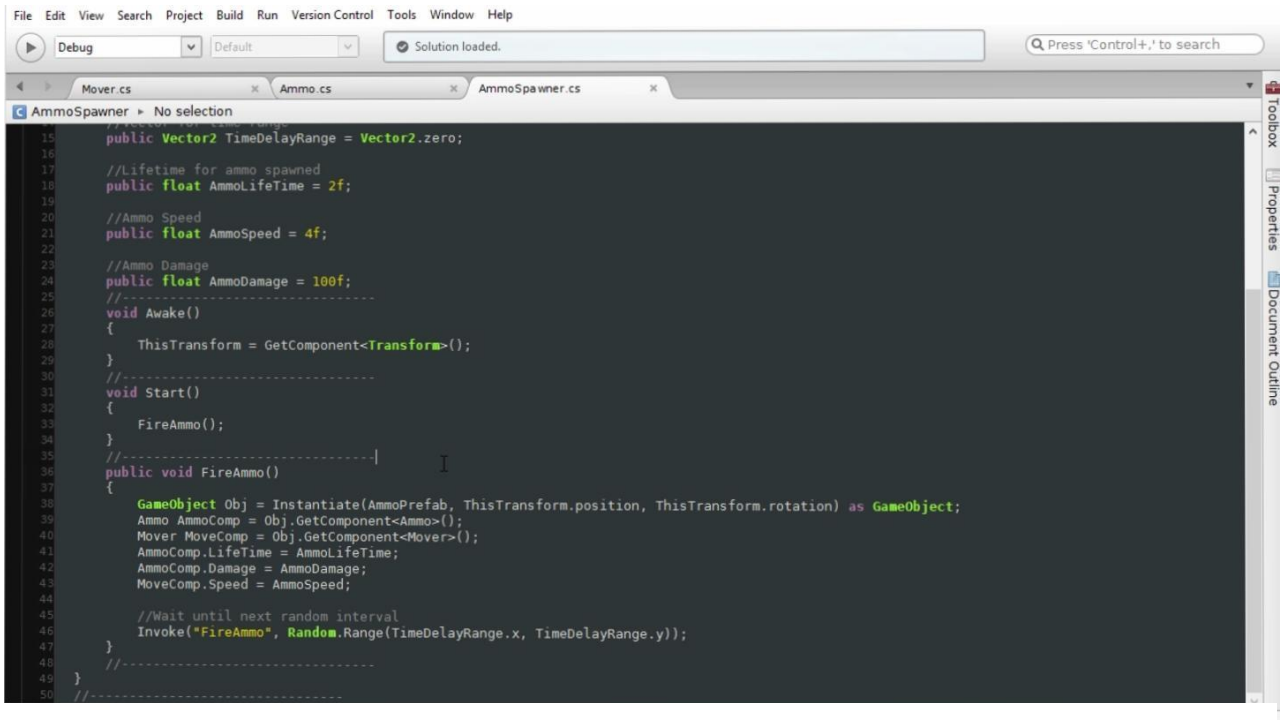
Kada lik upadne u rupu on mora umrijeti. Kako bi se to ostvarilo moramo kreirati skriptu i napisati kod, da se to dogodilo. Public float damage trebamo postaviti na maksimalnu razinu zdravlja (eng. *Health*) koja iznosi 100 kako bi se cijela razina zdravlja oduzela kada upadne u rupu nakon jedne sekunde, zatim postavimo petlju if koja će oduzeti zdravlje sa štetom i pomnožit s vremenom boravka u rupi što će dovesti do smrti lika.



Slika 4.17. Definiranje skripte za zonu smrti

4.3.5. Izrada topa

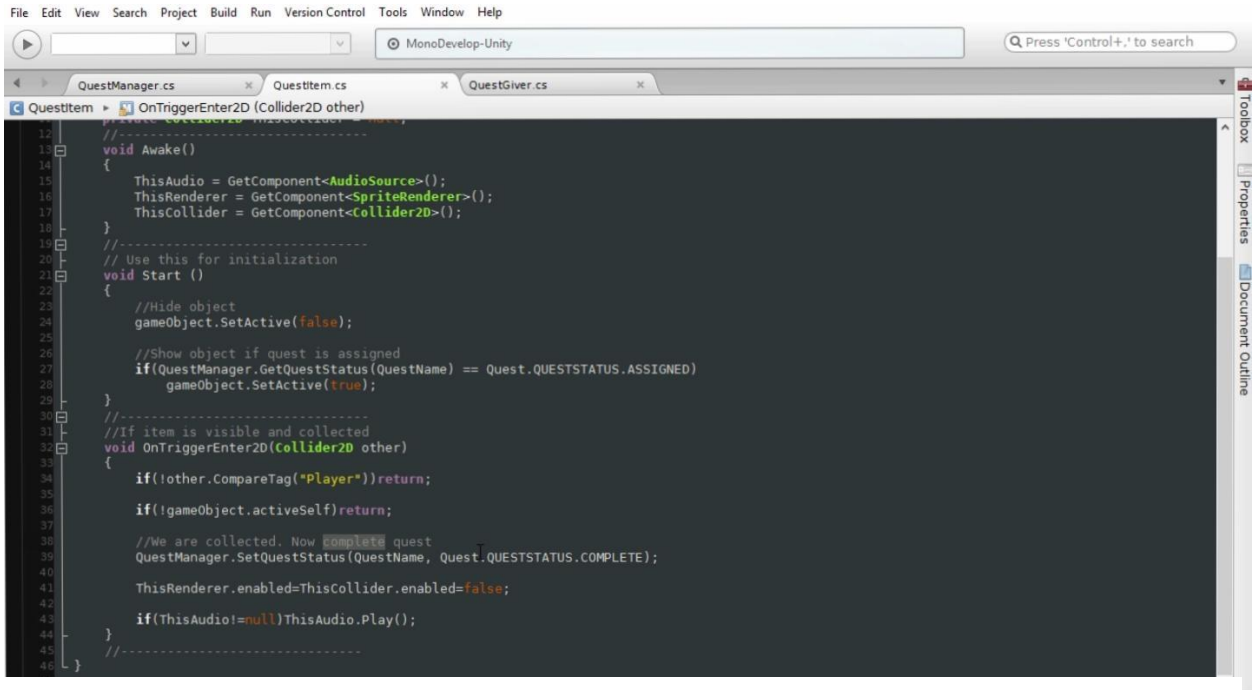
Da bi izradili top u izborniku Component izaberemo GameObject – Cube i kocku pozicioniramo gdje želimo da se nalazi. Sljedeći korak koji moramo uraditi je kreirati GameObject i odabrat PartycleSystem kako bi top ispucavao kugle. Zatim kreiramo skriptu koja će omogućiti pucanje i ozljeđivanje lika. Kreirati ćemo varijable za brzinu kugle (public float AmmoSpeed = 4f) vrijeme za ispucavanje kugle (public float AmmoLifeTine = 2f) i koliko će štete raditi kugla liku (public float AmmoDamage = 100f). Kako bi top pucao moramo kreirati varijablu void Start() FireAmmo(); }



Slika 4.18. Izrada topa

4.3.6. Izrada novčića

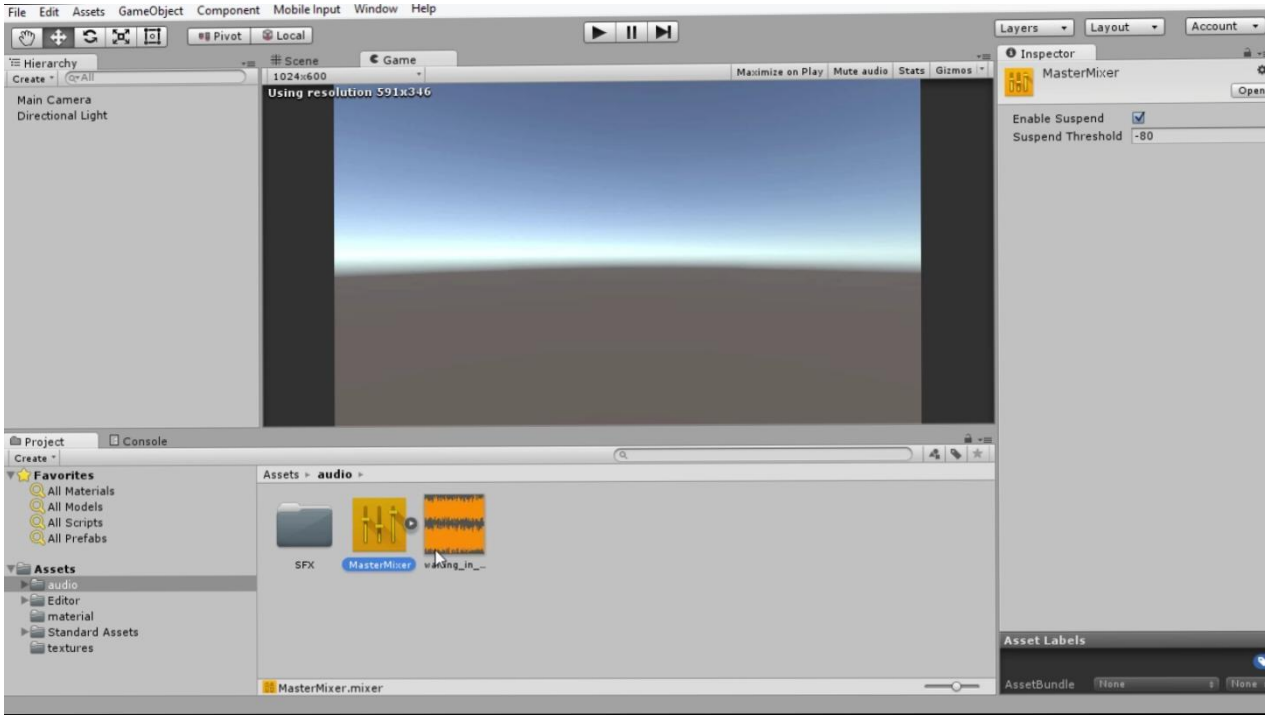
Objekt novčić liku donosi 100 golda ako ga pokupi. U skripti je dodano da kada lik pokupi novčić zadatak je gotov, i novčić se uništava sa scene. To uradimo tako da na objekt stavimo Box Collider i postavimo opciju Is Trigger.



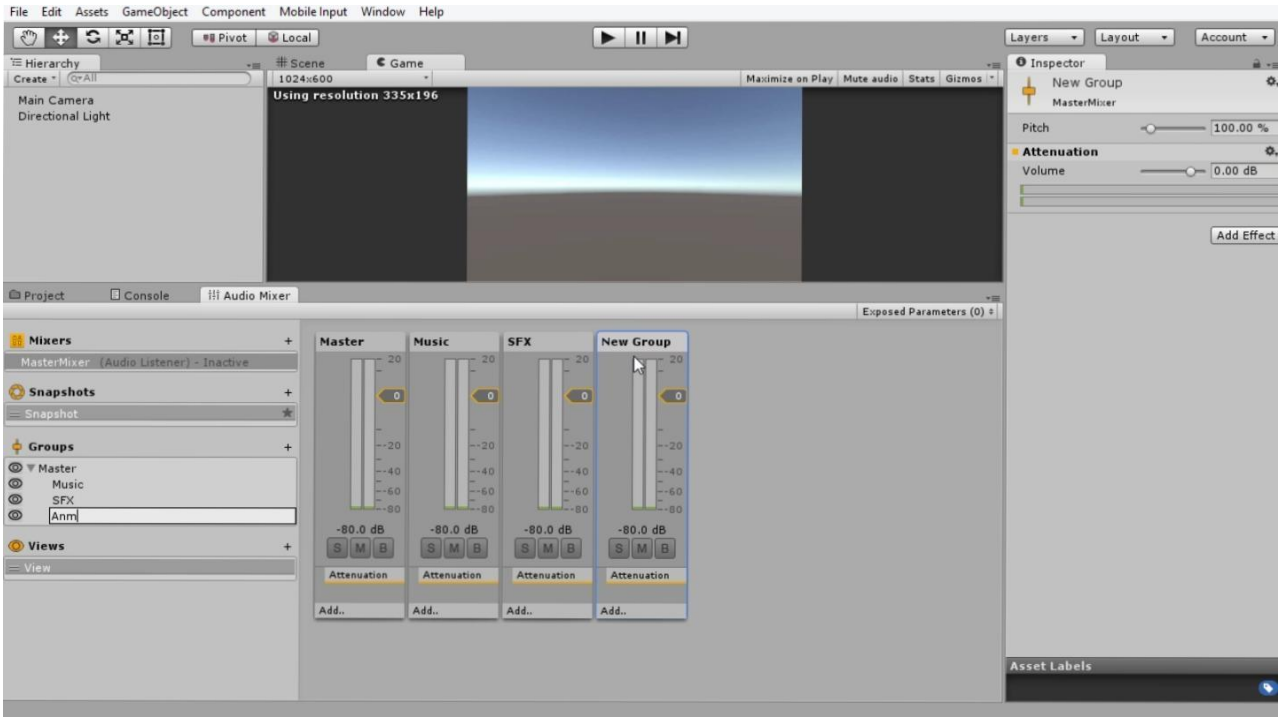
Slika 4.19. Izrada novčića

4.4. Zvuk

Da bi igra bila zanimljivija i uzbudljivija dodat ćemo zvuk u igru. U izbornik Assets odaberemo Standard Asstets i zatim Audio, vidjet ćemo u projektnoj ploči mapu Audio. Zatim otvorimo MasterMixer u projektnoj ploči po predlošku postoji samo jedan kanal kroz koji će se puštati svi glazbeni efekti, što može dovesti do prevelike buke i podudaranja glazbe kako bi to spriječili dodat ćemo još kanala koji će biti povezani s Master kanalom. Pritisnemo desnu tipku miša i stisnemo Create AudioChannel i nazove kanale kako želimo, tada možemo stavljati efekte, razinu glazbe i postaviti opciju ponavljanja (eng. *Loop*).



Slika 4.20. Dodavanje zvuka



Slika 4.21. Dodavanje zvuka

5. Zaključak

U ovom su radu opisane osnovne karakteristike razvojnog okruženja Unity te su navedene sve mogućnosti koje ono pruža prosječnom korisniku. Usporedba s Unreal Engine 4, koje je trenutno drugo plasirano razvojno okruženje na globalnom tržištu, poslužila je kako bismo shvatili generalne prednosti i nedostatke razvojnog okruženja Unity. S obzirom da sam uz malo strpljenja uspio naučiti princip rada Unity-a te samostalno razviti prototip klasične igre u vremenskom periodu od samo dva mjeseca, mogu samo potvrditi riječi pohvale široke zajednice ljudi.

Kao jedna od prednosti Unity-a navedena je velika količina dostupnih informacija i edukativnih materijala, što je zapravo temelj razumjevanja rada razvojnog okruženja te uvelike olakšava shvaćanje mogućih nejasnoća početnicima koji su tek krenuli izrađivati vlastitu igru. U dodatku, čak i kada se pojavi specifičan problem kojemu nije moguće pronaći rješenje u edukacijskim materijalima, vrlo se jednostavno obratiti širokoj zajednici prijateljski nastrojenih ljudi koji su putem službenog foruma voljni pomoći sa svakom preprekom koja se nađe na putu izrade video igre. Prolaskom kroz video tečajeve vrlo se brzo shvati generalni princip rada Unity-a te potom jedino preostaje poznavanje programerske logike za realizaciju specifičnih funkcionalnosti koje želite implementirati unutar vlastite igre.

Razumjevanje razvojnog okruženja ubrzano je i mogućnošću prilagodbe korisničkog sučelja prema vlastitim preferencama, no i bez personalizacije je izuzetno intuitivno i jednostavno za koristiti. Za razvoj video igre korištena je besplatna verzija razvojnog okruženja Unity. Time su uskraćene neke napredne mogućnosti, no smatram da početnicima nisu niti potrebne. Osnovni proces kreiranja video igre započinje vlastitom izradom ili preuzimanjem gotovih objekata koji se ubacuju u promatranu scenu. U ovom slučaju, pod objektom se smatra svaka komponenta koja se koristi u izgradnji te scene. Iz toga možemo zaključiti da je scena trodimenzionalni prostor u kojemu pozicioniramo objekte video igre te im dodjeljujemo željene funkcionalnosti, od animacija i zvukovnih zapisa pa sve do predefiniranog ponašanja u određenim situacijama.

Za potrebe ove igre, s Unity-eve online trgovine s resursima (eng. *Unity Asset Store*) preuzeti su neki osnovni modeli koji odgovaraju željenoj tematici igre. Prvi je korak bio kreiranje okoline unutar scene pozicioniranjem različitih objekata iz brojnih preuzetih paketa. Modeliranje i dizajniranje svakako su djelatnosti koja zahtjevaju puno strpljenja, znanja i iskustva, određenu količinu artističkog izražavanja te oko za detalje. Okolinu je bilo potrebno podesiti te definirati međuovisnost pojedinih objekata, kreirati i generirati brojne druge elemente

koji su temelj za realizaciju svih ostalih funkcionalnosti. S druge strane, posebno se konfiguriraju i modeli koji su u direktnoj interakciji s okolinom. Potrebno im je definirati animacije, uvjete promjene stanja te dodati brojne druge komponente kojima se programskim putem definira željeno ponašanje.

Funkcionalnosti su realizirane pisanjem odvojenih skripti koje se potom unutar razvojnog okruženja Unity povlače na objekte, ali i objekti na skripte koje su na nekom drugom objektu. Skripte kojima su realizirane specifične funkcionalnosti pisane su u C# programskom jeziku zbog dosta jednostavne interakcije s komponentama i objektima unutar scene. Postoje brojni gotovi algoritmi u sklopu edukacijskih materijala čiji dijelovi mogu služiti kao odlično polazište za realizaciju vlastitih funkcionalnosti. Skriptama se zapravo povezuju svi objekti unutar scene te se definira njihova međuovisnost sukladno komponentama koje pojedini objekti sadrže. Navedeni proces često rezultira ogromnim brojem pogrešaka unutar scene koje se mogu pojaviti iz brojnih razloga. Pritom je traženje pogreške poprilično zamarajući proces jer uzrok na prvu ruku može biti bilo što, pa iz vlastitog iskustva i greška u samom Unity-u koja se rješava izvan razvojnog okruženja.

U svakom slučaju, više sam nego zadovoljan novim stečenim znanjem i iskustvom te bih svakako preporučio Unity svakoj osobi koja želi baviti nekim aspektom izrade video igara. Samo je mašta granica svim funkcionalnostima koje se mogu implementirati unutar razvojnog okruženja Unity.

Literatura

1. Bevza, A. (2015). *Unity3D or Unreal Engine 4*. Preuzeto 5. 7. 2017. iz https://stfalcon.com/en/blog/post/unity3d-vs-unreal-engine-4
2. Dale, L. (2015). *Unity - does indie gaming's biggest engine have an image problem?* Preuzeto 4. 7. 2017. iz https://www.theguardian.com/technology/2015/jul/06/unity-indie-gamings-biggest-engine-john-riccitiello
3. Game & Assets Development. (2017). *Unity5 and UE4 Comparison*. Preuzeto 4. 7. 2017. iz http://not-lonely.com/blog/making-of/unity-ue-comparison/
4. Kudeljnjak , I., & Lozić, S. (2012). *Unity game engine*. Preuzeto 3. 7. 2017. iz http://rg.c-hip.net/2012/seminari/kudeljnjak-lozic/works.html
5. Mayden, A. (2014). *Unreal Engine 4 vs. Unity: Which Game Engine Is Best for You?* Preuzeto 5. 7. 2017. iz https://www.pluralsight.com/blog/film-games/unreal-engine-4-vs-unity-game-engine-best
6. Oldcomputers. (2015). *Apple II - 1977*. Preuzeto 2. 7. 2017. iz http://www.oldcomputers.net/appleii.html
7. Oskay, W. (2008). *Resurrecting Tennis for Two, a videogame from 1958*. Preuzeto 2. 7. 2017. iz http://www.evilmadscientist.com/2008/resurrecting-tennis-for-two-a-video-game-from-1958/
8. Paul, P., Goon, S., & Bhattacharya, A. (2012). History and comparative study of modern game engines. 246. - 247. Preuzeto 2. 7. 2017. iz https://www.researchgate.net/publication/236590476_HISTORY_AND_COMPARATIVE_STUDY_OF_MODERN_GAME_ENGINES
9. Rowland , A., & Clarke, C. (2011). *Unity Technologies Lands $12 Million in Series B Funding Led by WestSummit Capital and iGlobe Partners*. Preuzeto 4. 7. 2017. iz http://www.marketwired.com/press-release/unity-technologies-lands-12-million-series-b-funding-led-westsummit-capital-iglobe-partners-1540593.htm
10. *Serious Engine 4*. (2016). Preuzeto 2. 7. 2017. iz http://www.croteam.com
11. Takahashi, D. (2012). *Game developers, start your Unity 3D engines.* Preuzeto 4. 7. 2017. iz https://venturebeat.com/2012/11/02/game-developers-start-your-unity-3d-engines-interview/
12. *Unity 5 vs Unreal Engine 4*. (2015). Preuzeto 5. 7. 2017. iz https://create3dgames.wordpress.com/2015/09/07/unity-5-vs-unreal-engine-4/

13. Unity Technologies. (2016). *Thirty - four percent of top games are made with Unity*. Preuzeto 4. 7. 2017. iz https://unity3d.com/public-relations
14. Unity Technologies. (2017). *2D Unity tutorial*. Preuzeto 3. 7. 2017. iz https://unity3d.com/learn/tutorials/projects/2d-roguelike-tutorial
15. Unity Technologies. (2017). *Audio*. Preuzeto 2. 7. 2017. iz https://docs.unity3d.com/Manual/Audio.html
16. Unity Technologies. (2017). *GameObject*. Preuzeto 2. 7. 2017. iz https://docs.unity3d.com/ScriptReference/GameObject.html
17. Unity Technologies. (2017). *Multiplatform, Unity*. Preuzeto 3. 7. 2017. iz https://unity3d.com/unity/features/multiplatform
18. Unity Technologies. (2017). *Prefabs*. Preuzeto 3. 7. 2017. iz https://docs.unity3d.com/Manual/Prefabs.html
19. Unity Technologies. (2017). *Scenes*. Preuzeto 2. 7. 2017. iz https://docs.unity3d.com/Manual/CreatingScenes.html
20. Unity Technologies. (2017). *Scripting*. Preuzeto 2. 7. 2017. iz https://docs.unity3d.com/Manual/ScriptingSection.html
21. Unity Technologies. (2017). *Sprite Editor*. Preuzeto 3. 7. 2017. iz https://docs.unity3d.com/Manual/SpriteEditor.html?_ga=2.38418851.1455400083.1501021168-27780970.1495611186
22. Unity Technologies. (2017). *UI*. Preuzeto 3. 7. 2017. iz https://bitbucket.org/Unity-Technologies/ui
23. Unity Technologies. (2017). *Unity - Game Engine*. Preuzeto 4. 7. 2017. iz https://unity3d.com/
24. Unity Technologies. (2017). *Using Cameras*. Preuzeto 3. 7. 2017. iz https://unity3d.com/learn/tutorials/topics/graphics/using-cameras
25. *Unreal Engine VS Unity*. (2016). Preuzeto 5. 7. 2017. iz https://www.vrstatus.com/news/unreal-engine-vs-unity.html
26. Winte, D. (2013). *Noughts And Crosses - The oldest graphical computer gamer*. Preuzeto 2. 7. 2017. iz http://www.pong-story.com/1952.htm

Popis slika

Prilozi: CD

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Tihomir Slunjski *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Izrada 2D igre kroz Unity 3D alat *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
Tihomir Slunjski
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Tihomir Slunjski *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Izrada 2D igre kroz Unity 3D alat *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

(vlastoručni potpis)