# Izrada videoigre utrkivanja u programskom alatu Unity

**Lakoseljac, Rafael Dominik**

**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

**Rafael Dominik Lakoseljac**

# Izrada videoigre utrkivanja u programskom alatu *Unity*

**ZAVRŠNI RAD**

**Varaždin, 2022.**

**SVEUČILIŠTE U ZAGREBU**

**FAKULTET ORGANIZACIJE I INFORMATIKE**

**V A R A Ž D I N**

**Rafael Dominik Lakoseljac**

**Matični broj: 00161431413**

**Studij: Informacijski sustavi**

# Izrada videoigre utrkivanja u programskom alatu *Unity*

**ZAVRŠNI RAD**

**Mentor:**

doc. dr. sc. Mario Konecki

**Varaždin, rujan 2022.**

*Rafael Dominik Lakoseljac*

**Izjava o izvornosti**

Izjavljujem da je moj završni/diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

*Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi*

**Sažetak**

Završni rad sastojat će se od dva dijela. U prvom će dijelu teorijski biti opisana izrada videoigre. Taj dio sastojat će se od opisa alata, tehnologija i metoda koje će se koristiti prilikom izrade videoigre. Prolazit će se kroz različite funkcionalnosti alata *Unity* koje će omogućiti izradu igranih modela, terena i ostalih elemenata, kao što su programski kodovi u jeziku C#, koji su potrebni kako bi se igrica izradila. Drugi dio je praktični dio, odnosno sama videoigra, izrađena u alatu *Unity*. Prikazat će se njezine funkcionalnosti, njezine specifične karakteristike i sam način rada, odnosno igrica će biti odigrana da se prikaže njezina igrivost.

**Ključne riječi**: *Unity*, igrani model, videoigra, C#

# Sadržaj

# 1. Uvod

Kroz ovaj će završni rad biti prikazana izrada videoigre – teoretski i praktično. Videoigra će biti kreirana pomoću poznatog **alata *Unity***, koji nam omogućava mnoge funkcionalnosti – od kreiranja igranih modela do kreiranja okruženja te samog sučelja koje je namijenjeno korisničkom iskustvu njezinog korisnika, u ovome slučaju igrača videoigre. Kreiranje same videoigre zahtijevalo je određene metode i tehnike koje su dostupne u alatu *Unity* zahvaljujući kojima je kreiran finalni proizvod, a sve će metode i tehnike biti opisane u sljedećim poglavljima rada. To su poglavlja **Alati i tehnologije** u kojima će biti riječi o alatima koje je moguće koristiti za izradu programa i videoigra i o alatu *Unity*; zatim **Povijest videoigara** u kojemu ćemo nešto reći o nastanku i razvitku videoigara od prve izumljene pa sve do današnjih dana; i poglavlje **Kreiranje videoigre** u kojemu će biti prikazano stvaranje videoigre korak po korak i objašnjeni odabiri objekata i svega onoga što igrici daje dodatan značaj. Na kraju ćemo sve to **zaključiti**.

Završni se rad sastoji od **dvaju dijelova**. Prvi dio rada, odnosno **teoretski dio**, odnosi se na opisivanje tehnologija, primjerice programski jezik koji je korišten, igrači stroj (*engine*) i alate pomoću kojih su kreirani automobili, staza i ostali elementi kako bi proizvod bio izrađen. Drugi dio rada je **praktični dio** i njime se prikazuje sama videoigra, a dojam o videoigri najjasnije će biti prikazan ako je pokušamo zaigrati. Na taj će način biti prikazane njezine mogućnosti, pravila i logika koja je zaslužna pri brojanju krugova i vremena te sličnih funkcionalnosti.

# 2. Alati i tehnologije

## *2.1. Unity*

*Unity* je besplatan višeplatformski *game engine*[1], koji je razvila **tvrtka za razvoj softvera – *Unity Technologies***, prvi je put predstavljen 2005. godine na *Apple worldwide Developers Conference*[2] kao *game engine* za Mac OS X.

Otada se razvijanjem postigla razina na kojoj se *Unity* nalazi danas te se pomoću njega mogu izrađivati igre za mnoge platforme kao što su Windows, Linux, Mac OS, Android, iOS i mnoge druge platforme. *Engine*, ili u prijevodu motor, *Unity* kreiran je kako bi se izrađivale 2D i 3D igre, ali zbog njegove raznovrsnosti i kvalitete, prihvaćen je i drugim industrijama poput filmske – za izradu CGI modela i scena, automobilske industrije – za kreiranje i testiranje budućih prototipova te arhitektonske industrije, gdje se vjerno može prikazati izgled zgrada i precizno izračunati njihove dimenzije. Ovo su samo neke od mogućih primjena *Unityja*.

Program *Unity* kroz godine je redovito ažuriran te je svakom verzijom dodavano, unaprjeđivano i poboljšavano korisničko sučelje te same funkcije kako bi se proširile mogućnosti koje korisnik može iskoristiti izrađujući igru.

Prilikom izrade igre ili drugih primjena programa moguće je koristiti ***Assets***, odnosno **imovinu**, koja može biti besplatna ili ju je potrebno prije kupiti u *Asset Storeu* kako biste je mogli koristiti u izradi programa na kojemu radite. Za potrebe izrade videoigre u sklopu ovoga rada korištena je besplatna imovina dostupna svim korisnicima besplatne verzije *Unity* programa. Imovina u *Unityju* može biti **tekstura** kao što su zemlja, trava, asfalt ili druge površine, ali i **modeli automobila, aviona** te **model čovjeka** koji ima pokretne zglobove kako bi se mogao kretati i obavljati zadatke koji se u igrici mogu postaviti. Imovina, osim što se može kupiti, može se i prodavati. Primjerice, ako ste prilikom kreiranje vlastite igre, kreirali neki igrivi model (recimo automobil), a takav ne postoji u *Storeu,* možete je spremiti i staviti na prodaju kako bi je drugi **kreatori** koristili. Kako bi se izradile funkcionalnosti, potrebno je **skriptiranje** ili **pisanje programskog koda**. U *Unityju* se, odnosno u starijim verzijama, koristio program *Mono*

[1] *Game engine* softverski je okvir dizajniran s ciljem razvoja videoigara.
[2] Konferencija *Apple worldwide Developers Conference* namijenjena je čitavoj industriji **informatičke tehnologije** (IT) i stručnjacima iz IT područja. Na njoj se predstavljaju najnovije platforme, tehnologije i alati, a organizira je upravo kompanija *Apple* s ciljem promocije kreiranih noviteta. (preuzeto prema - https://developer.apple.com/wwdc22/)

*Develop*[3] koji je danas zamijenjen s *Visual Studio Codeom*[4]. Njega danas koristi veći broj ljudi i bolje je podržan od Monoa.

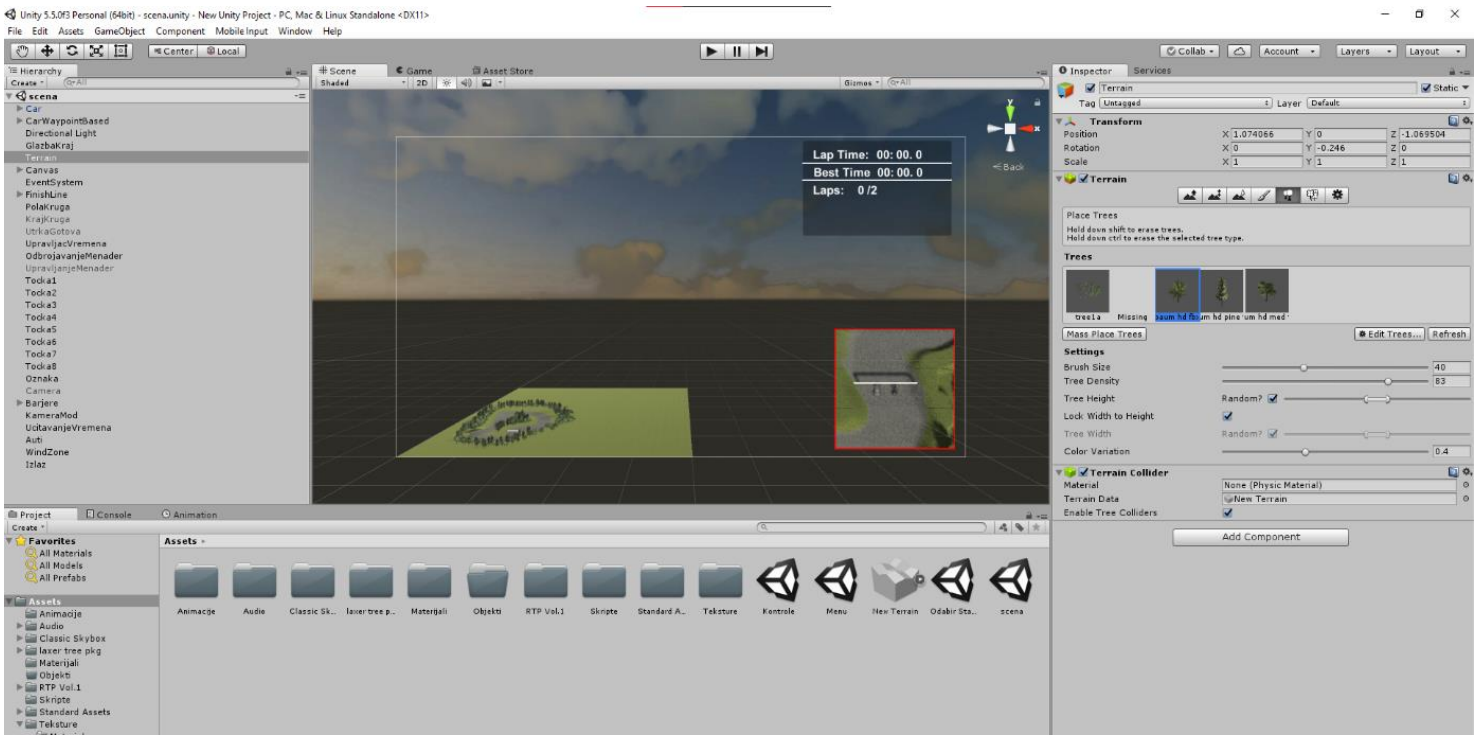
*Unity* je besplatan za pojedince i manje kompanije koje ne prelaze određenu svotu primanja, a prilikom prelaska potrebno je platiti *Unity Pro* verziju koju koriste veće kompanije. Mnoge igre kreirane su pomoću Unityja , a neke od njih su: Pokemon Go, Cuphead, Among us i Fall Guys. [1]

## 2.1.1. Unity 5.5.0f3

Verzija *Unityja* koja se koristila prilikom izrade ove igrice je **5.5.0f3**. To nije najnovija verzija, ali je vrlo kvalitetna. Najnovija se verzija razlikuje u sitnim detaljima i poboljšanjima. U nastavku će biti opisane neke funkcionalnosti koje su korištene pri izradi ovoga završnog rada, odnosno videoigre, te pobliže dočarati njihove karakteristike. [2]

### 2.1.1.1. Glavne funkcionalnosti

Kako bi se korisnik koji nikada nije koristio *Unity* mogao se u njemu snalaziti, potrebno je znati neke **osnovne funkcionalnosti programa**.




Slika 1: Program *Unity* - tijek izrade videoigre (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

[3] Program *MonoDevelop* razvojno je okruženje za pisanje programskoga koda.
[4] *Visual Studio Code* je, baš kao i *Mono Develop*, razvojno okruženje za pisanje programskoga koda.

Na prethodno priloženoj slici mogli ste vidjeti kako izgleda početno sučelje programa *Unity*, a sada ćemo opisat neka od glavnih svojstava toga programa.

**Glavni izbornik** na vrhu prozora sastoji se od opcija koje nam pružaju kreiranje nove scene ako je to potrebno, uređivanja i dodavanja svojstava, objekata i komponenti koje su nam potrebne u igri. Prilikom kreiranja bilo kakvog elementa moguće je na desnoj strani prozora, pomoću **inspektora**, vidjeti svojstva tog elementa i opcije koje nam se za određeni element nude. S lijeve je strane prikazan **prozor scene** na kojoj se trenutno nalazimo i svih elemenata od kojih se ta scena sastoji.

Na slici je prikazan teren koji je kreiran za potrebe održavanja utrke u videoigri utrkivanja te sva svojstva koja on sadrži i koja se mogu dodati ili promijeniti. Na dnu prozora vidljiv je odjeljak u kojem se nalaze sva svojstva i scene unutar igre koja su sortirana u mape. U te mape moguće je spremanje skripti koje se koriste za neke funkcionalnosti, pozadinske slike, audio, materijali i slično, koje se zajedno ujedinjuju u gotov proizvod.

### 2.1.1.2. Kreiranje terena i okruženja igre

Izradu videoigre započinjemo **kreiranjem terena** na kojemu će se odvijati igra utrkivanja i upravo je to prva funkcionalnost koju ćemo opisati. Reći ćemo ponešto o mijenjanju i dodavanju karakteristika izrađenome terenu, a kako bi se teren kreirao potrebno je kreirati novi *GameObject* pod nazivom *Terrain*. Kada smo to kreirali, dobivamo na ekranu praznu „ploču“, odnosno teren koji je kreiran samo je jedna ploha koju možemo uređivati prema vlastitim željama ili prema željama klijenta ako program / videoigru izrađujemo poslovno. Neke od mogućnosti su **podizanje i spuštanje terena, zaglađivanje površine, bojanje terena, postavljanje drveća i postavke samog terena**.

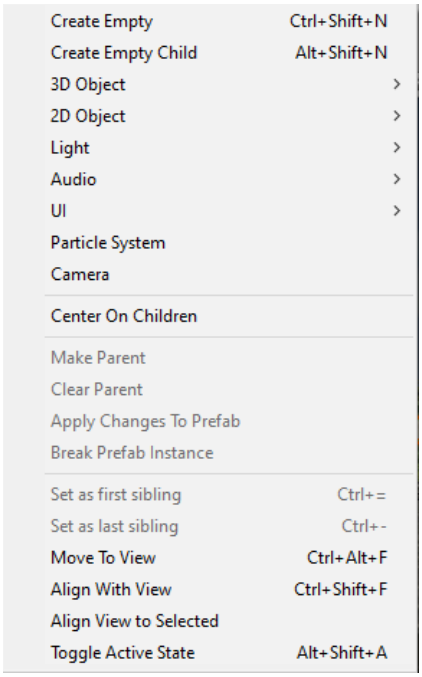
Slika 2: Program *Unity* - Opcije uređivanja terena (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Prva opcija na izborniku je **podizanje i spuštanje terena**. Kod ove opcije moguće je pomoću kista podizati i spuštati teren toliko koliko se dugo drži lijeva tipka na mišu na određenoj površini. Ta postavka je neprecizna i može dovesti do prevelikog podizanja terena ako to u tom slučaju nije bilo potrebno. Druga je opcija slična prvoj, ali je mnogo preciznija. Radi se o **bojanju visina**, odnosno korisnik odluči brojčano koliko površinu želi podignuti i kistom prolazi na onom mjestu na terenu gdje to želi primijeniti te tako podiže teren. Treća je opcija **zaglađivanje površina**. Ova je opcija korisna u slučaju naglog podizanja terena, odnosno

kada je teren potrebno zagladiti kako ne bi bilo oštrih rubova. Četvrta je opcija **bojanje tekstura**. Ono je potrebno kako bi teren imao boju, odnosno teksturu pa tako možemo kreirati travnata površina, asfaltna površina ili neku drugu površina koja je korisniku potrebna kod izrade igre. Ta se tekstura nanosi, kao i prijašnje opcije, pomoću kista kojemu odabiremo veličinu i dodjeljujemo mu željenu teksturu i boju. Peta je opcija **postavljanje drveća**. Ovom je opcijom moguće postaviti željeni model drveća koji smo pronašli na internetu ili koji smo mi sami kreirali u nekom drugom alatu.  Dodijeljeni se model, isto kao i sve prijašnje opcije, postavlja na željenu površinu uz pomoć kista, ali jedina razlika kod ove opcije je da se može birati gustoća, visina i varijacija u boji drveća koja se nanosi na površinu. Kasnije će biti prikazano kako je u kreiranju ove videoigre odabran i postavljen teren, njegova tekstura i drveće koje okružuje pistu na kojoj se automobili utrkuju.

### 2.1.1.3. Kreiranje elemenata

Kod kreiranja igre bit će nam zasigurno potrebni neki **elementi**, bili to objekti koji su već sastavni dio programa ili koje ćemo negdje preuzeti. Elemente koji su već sadržani u programu dodajemo preko opcije *GameObject* na vrhu prozora, ondje nas dočekuje padajući izbornih s velikom paletom elemenata koje možemo odabrati.




Slika 3: Padajući izbornik *GameObject* (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

***Create Empty*** (Kreiraj prazni) najčešće se koristi prilikom povezivanja skripte sa scenom, odnosno taj se element kreira te se na njega poveže skripta koja je zadužena za neku funkcionalnost i ona je prazna kako ne bi zauzimala resurse kod performansi igre. Moguće je

kreiranje **3D i 2D objekata** koji su nam potrebni. Moguće je kreirati sljedeće 3D objekte: kocku, kuglu, cilindar i ostale osnovne geometrijske likove, a njih je pomoću inspektora i ostalih opcija moguće uređivati po želji. Opcija ***Light*** služi nam kako bismo mijenjali osvjetljenje prostora u kojem se igra odvija te dodavanje ostalih svjetala za koje mislimo da su potrebni na nekom djelu terena. ***Audio*** nam služi za dodavanje zvukova, bilo to glazba ili neki drugi audio efekti koji su nam potrebni za dočaravanje atmosfere igre. *UI* ili ***user interface*** (korisničko sučelje) elementi su koji nam služe za korisničko sučelje, primjerice poruke, slike, gumbovi ili slični elementi koji igraču igre omogućavaju primanje neke vrste informacija ili omogućavanje kontroliranja odigravanja igre, odnosno kretanja kroz igru. Posljednja opcija koju ćemo spomenuti u ovome djelu je ***Camera***. Ta opcija omogućava nam dodavanje nove kamere koju korisnik postavlja i prilagođava korisniku tako da mu bude korisna, bila to kamera iz ptičje perspektive, fokalizator u prvom licu ili trećem ili neka druga opcija koja je potrebna za određeni žanr igre.

#### 2.1.1.4. Povećavanje, pomicanje i rotiranje svojstava i scene

Nakon što dodamo neki objekt (kocku, kuglu, automobil ili nešto drugo) unutar scene, potrebno je taj objekt i urediti kako bi se on što bolje uklopio u finalni proizvod.

Slika 4: Opcije uređivanja (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Prva opcija služi nam kako bismo se mogli **kretati kroz okruženje** povlačenjem lijevo desno po prozoru. Druga opcija omogućava nam **pomicanje elementa** kroz prostor, odnosno do određene lokacije na kojoj je on zamišljen. Treća opcija služi nam kako bismo element mogli **rotirati**. Zadnje dvije opcije odnose se na **promjenu veličine elementa**, njegovo povećavanje ili smanjivanje, ovisno o željama korisnika.

#### 2.1.1.5. Uvoženje svojstava

Prilikom kreiranja igre moguće je koristiti neke od već postojećih svojstava kao što su npr. vozila, teksture, materijali i neke od ostalih svojstava koje je neki programer ili dizajner kreirao prije nas. Ta sredstva moguće je pronaći na trgovini koja je integrirana u sam program te pomoću te opcije moguće je prilikom kupovine ista svojstva odmah i uvesti u sam program. Ta svojstva mogu se plaćati ili mogu i biti besplatna, ovisno o kreatoru tih svojstava.

## 2.2. MonoDevelop

*MonoDevelop* besplatni je i *open-source* program koji je integriran u *Unity*. Pomoću njega pišu se skripte koje se koriste prilikom izrade igre. Podržan je od strane Windowsa, Linuxa i Mac OS-a. Temelji se na .Net Frameworku[5] za skriptne jezike kao što su C#, JavaScript i mnogi drugi jezici. Predstavljen je 2004. godine, nakon što je tim zadužen za njegovo kreiranje uvidio budućnost u .NET Frameworku, koji je 3 godine ranije, u ljeto 2000. godine, predstavio Microsoft. Inicijalna verzija bila je osmišljena kako bi se kreirale Linux aplikacije, ali su s vremenom pronađene mnoge druge primjene kao npr. skriptiranje u Unityju.

*MonoDevelop* arhitektura sastoji se od dijelova za izvršavanje koda, biblioteke klasa, sakupljača smeća i ostalih komponenti kako bi program olakšao i programerima poboljšao pisanje koda za njihove projekte.

## 2.3. C#

C# programski je jezik koji se koristi u mnoge svrhe kao što su izrada aplikacija, videoigra i slično. Kreiran je 2000. godine u Microsoftu te ga je ECMA kasnije, 2002. godine, priznao kao standardni programski jezik. Prilikom predstavljanja C# predstavljen je i .NET Framework zajedno s *Visual Studiom*. Sintaksa jezika C# slična je jezicima kao što su C, C++ i Java.

## 2.4. JavaScript

*JavaScript* je jezik više razine, još znan i kao Js, koji se koristi prilikom izrada funkcionalnosti za web stranice uz pomoć HTML-a i CSS-a. Prva verzija JavaScripta zvala se *LiveScript* i predstavljena je 1995. godine. *JavaScript* se prvobitno koristi kod izrade web stranica, ali njegova se primjena može pronaći i kod nekakvih drugih proizvoda kao npr. u ovom radu kod izrade video igre.

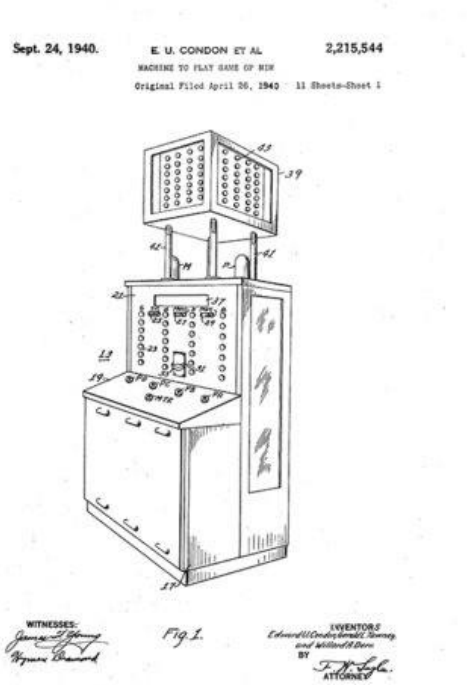
[5] *.Net Framework* upravljačko je izvršno okruženje za Windows, koje programerima omogućava stvaranje softverske aplikacije u jednom programskom jeziku.

# 3. Povijest videoigara

Danas je tehnologija mnogo revolucionarnija, a izrada videoigara puno laganija, dostupnija i sam je postupak izrade razvijeniji nego što je to bilo u 50. godinama 20. stoljeća kada je nastala prva videoigra. Ipak, da bismo došli do situacije u kojoj smo danas, bilo je potrebno puno **istraživanja, učenja i usavršavanja**. Kako bismo došli do ovoga što imamo danas, važno je znati kako je to nekada izgledalo i od čega je sve krenulo i upravo će o tome biti riječ u ovome poglavlju ovoga završnoga rada, naslovljenog ***Povijest videoigara***. Upoznat ćemo se s početcima i prvim igrama, prvim video konzolama, arkadnim mašinama i ručnim konzolama te sve sumirati kratkim pregledom konzola današnjice.

## 3.1. Početci i prve igre

Kada čujemo riječ ***videoigra***, prva stvar koja nam pada na pamet jest asocijacija na nekakvu igru koju igramo na računalu, mobitelu i sličnim modernim uređajima koji su danas vrlo rasprostranjeni i uobičajeni u našoj svakodnevici. U zadnjih nekoliko desetljeća tehnologija se toliko razvila i procvala da je nezamislivo kako je to sve izgledalo prije 80 godine kada se razvilo prvo „Računalo“. Tadašnja računala služila su samo za računanje brojeva, bilo to zbrajanje, oduzimanje, množenje, dijeljenje ili neka druga matematička operacija. Igre koje su se tada razvile bile su više za prikazivanje i testiranje mogućnosti računala za razliku od danas, kada je to isključivo zabava.



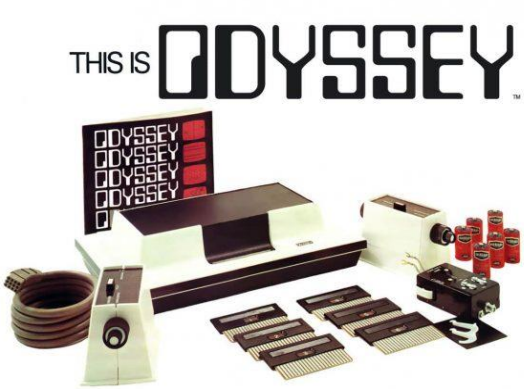

Slika 5: Prvo računalo iz 1940. godine (izvor: https://www.museumofplay.org/video_games/)

Tek je kasnije, u 50. godinama prošloga stoljeća, nastala prva videoigra. Točnije, smatra se da je ona nastala 1958. godine. Igru je kreirao **fizičar Willy Higinbotham** [3] te je to bila igra u kojoj se igrao tenis. Predstavljena je u laboratoriju jer se sama igrica igrala na osciloskopu[6]. Igra nije doživjela uspjeh te je ubrzo zaboravljena, ali je njezina pojava najavila buduće slične igre, kao npr. *Pong*. Nedugo nakon te igrice student s MIT-a kreirao je igricu u kojoj se prvotno nacrta labirint te nakon toga miš prolazi kroz taj labirint do izlaza.

## 3.2. Ubrzani razvoj i prve video konzole

Bržim razvojem tehnologije razvijale su se i same videoigre te i sama izrada tih igara. U 60-im je godinama došlo do razvoja **BASIC programskog jezika**, koji je omogućavao brzo i lako kreiranje igara. Posljedica toga je stasanje mnogobrojnih jednostavnih igara. BASIC je kreirao **John Kemeny** na *Dartmouth Collegeu*.

1972. godina bila je važna godina za videoigre zbog razvoja prve kućne konzole ***Magnavox Odyssey***. Ona se spajala na kućne televizore i korisnicima omogućavala igranje raznih igara kao što su stolni tenis, skijanje, tenis i mnoge druge igre koje su bile namijenjene igranju u paru te nadmetanju s protivnikom. Svaka konzola dolazila je u paru s dva upravljača koje su igrači koristili.




Slika 6: *Magnavox Odyssey* – prva kućna konzola (izvor: https://fontsinuse.com/uses/15155/magnavox-odyssey-game-console-logo-packaging)

Iste godine kreirana je i **prva arkadna mašina** koja je omogućavala igranje prije spomenute igre *Pong.* Tu je arkadnu mašinu razvila vrlo poznata kompanija *Atari*, koja je nekoliko godina

[6] Osciloskop je instrument koji promjene električnih veličina čini vidljivima na fluorescentnom ekranu. (Izvor: https://hjp.znanje.hr/index.php?show=search)

kasnije kreirala i započela s isporukom njihove kućne konzole ***Atari 2600***. Ta je konzola sadržavala prvu verziju današnjih klasičnih joystickova te igrice koje su bile u obliku kutija i prekidača koji su omogućavali promjenu težine igre koja se igra. [4]

## 3.3. Zlatno doba arkadnih mašina i ručne konzole

**Arkadne mašine** doživjele su svoj vrhunac u 80-im godinama prošloga stoljeća zbog velike popularnosti tadašnjih igara *Pac-Man* i *MS. Pac-Man*. Arkadne mašine svoju popularnost zadobile su ne samo zbog igara koje su se na njima mogle igrati nego i zbog lokacija na kojima su se te mašine nalazile, a to su bili trgovački centri, trgovine s igračkama i slična mjesta na koja su dolazila djeca koja su željela igrati igre, a nisu imala tu mogućnost kod kuće. Jedna od popularnih igrica toga vremena je i *Donky Kong* – igrica u kojoj igrač izbjegava bačve koje majmun (Donky Kong) baca. Korisnik je u ulozi *Jumpmana*, koji je danas bolje znan kao ***Super Mario***. To ime je dobio kasnije, kada se izradila igrica s njegovim danas poznatim imenom.

1984. godine lansirana je igrica *Tetris*. Ta se igrica igra i dan danas, ali u novijim verzijama te u boji. Tada je igrica bila kreirana u Rusiji, ali zbog stanja Hladnog rata i ostalih komunikacijskih problema igra nije doživjela brzi rast, nego je slavu doživjela tek nakon nekoliko godina, kada je procurila i kada je ju Nintendo objavio u kolekciji igara namijenjenu prvoj ručnoj konzoli – ***GameBoyju***.


Slika 7: *GameBoy* (izvor: https://sh.wikipedia.org/wiki/Datoteka:Game-Boy-FL.png)

Nakon originalne konzole došlo je do mnogih iteracija u boji i manjim veličinama u godinama koje slijede. Uvođenjem novih konzola uvele su se i nove tehnologije te se tako počinu kreirati 3D igre s boljim grafičkim svojstvima i opširnijeg razmjera nego igrice koje su bile prije njih te nastaju igrice kao *Doom.* [5]

Devedesetih godina igrice su se krenule nerazmjerno kreirati i prodavati te su ujedno tako i mnoge kompanije krenule s kreiranjem novih i boljih konzola. U to su vrijeme nastale konzole ***Playstation 1*** i ***Xbox*** te je čak i Microsoft, uz verziju Windowsa 3.0, pružao igrice poput *Solitarea* i mnoge druge koje se i danas nalaze na Windows računalima.

Arkadnim mašinama zbog navedenih konzola, koje su bile raznovrsne i pristupačne za kućnu uporabu, počinje opadati popularnost.

## 3.4. Današnje igre i konzole

Današnja industrija videoigara i video konzola je višemilijunska te se danas na razvoj videoigara troše velika novčana sredstva kako bi te igrice bile što prodavanije i što igranije. Danas su popularne takozvane *Free-to-play* igre. To znači da korisnik nije nužan kupiti igricu kako bi je igrao. Naravno u igri je omogućeno uređivanje igrača te to zahtjeva plaćanje pa je to jedan od novih načina kako kreatori zarađuju na igrici koju su kreirali.

Jedna od takvih igara je ***Fortnite***, koja je preko noći postignula veliki uspjeh i popularnost među svim dobnim skupinama jer je besplatna i nije teška za igranje te se može igrati s prijateljima u timu do 4 osobe. Igra omogućava veliki izbor različitih odijela za igrača, vozila te oružja. Taj je izbor mnoge igrače potaknuo da kupe neki predmet kako bi se isticali od ostalih igrača. Još jedan razlog njezine popularnosti je to da je to *Crossplay* igra, što znači da se može igrati s različitih konzola u isto vrijeme te je to omogućilo korisnicima koji imaju računalo i korisnicima koji imaju Playstation da igraju zajedno, što je prije bila rijetka pojava zbog različitosti u hardveru i zbog različitih mogućnosti koje korisnici različitih konzola imaju.

# 4. Kreiranje videoigre

Već je prethodno spomenuto kako je drugi dio ovoga završnog rada **praktične prirode**, odnosno sama videoigra koja je izrađena za potrebe završnoga rada. Stoga će ovo poglavlje biti posvećeno pobližem pojašnjavanju dijelova od kojih je igrica sastavljena te kako je tekao proces razvoja same igrice. Kreiranje igre bilo je podijeljeno u nekoliko faza kako bi se što bolje i brže mogla kreirati igra, a neke od faza su: **izrada terena i okruženja za utrkivanje**, **kreiranje igrača i protivnika** te **kreiranje sučelja** preko kojega će igrač zaprimati informacije o rezultatu odigrane igre. U nastavku će biti više riječi o tome.

## 4.1. Opis videoigre

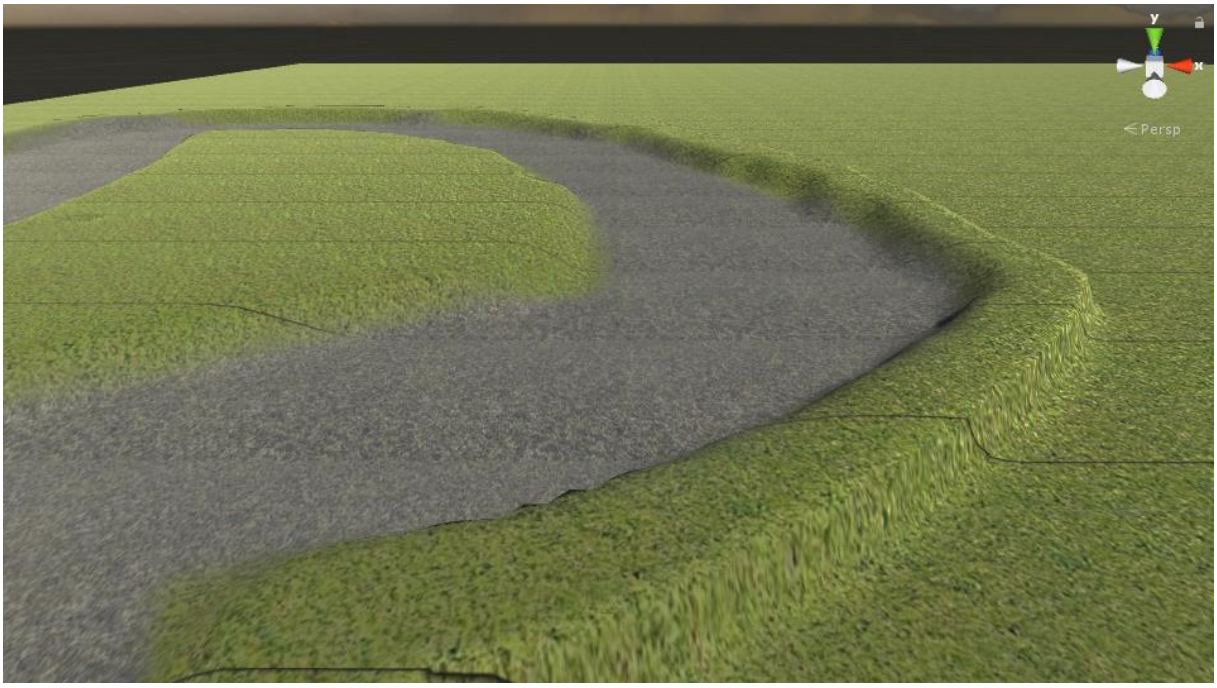
Izrađena videoigra jest **igra utrkivanja automobilima**. Ona se sastoji od igrača koji se natječe protiv protivnika na stazi. Igra se odigrava u dva kruga te se pamti najbolje vrijeme ostvareno u ta dva kruga. Kontrole su uobičajene – upravlja se tipkovnicom, odnosno strelicama ili tipkama *wasd*. Prilikom završetka igre ovisno o rezultatu, bio on pozitivan ili negativan, igraču se prikazuje prikladna poruka.

## 4.2. Kreiranje okruženja

Kod kreiranja videoigre jedna od ključnijih komponenti jest **kreiranje okruženja**. U ovome je slučaju to trkaća staza na kojoj će se sama igra igrati, odnosno na kojoj će se automobili utrkivati.

Prvi zadatak kod kreiranja staze bio je **odabir podloge** po kojoj će se automobili kretati. Odabir je pao na klasičan asfalt kao stazu, a za njezinu je okolinu odabrana travnata površina. Kako bismo mogli koristiti te teksture potrebno ih je bilo pronaći i preuzeti s interneta. Nakon preuzimanja tekstura, bilo ih je potrebno uvesti u program te ih prenamijeniti tako da se mogu koristiti kao boja, odnosno kako bi se podloga mogla obojiti tim teksturama. Nakon bojanja terena potrebno je terenu pridodati dubine i visine, a to je postignuto prethodno opisanim alatom za podizanje i spuštanje terena[7].

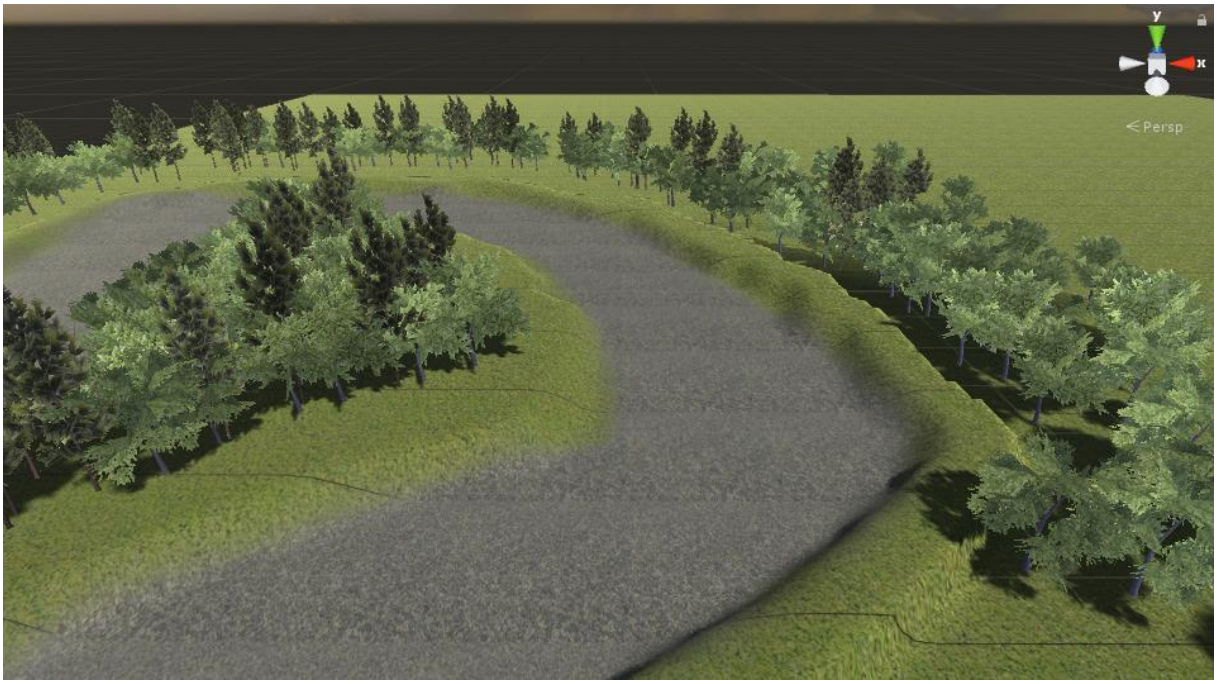
[7] Čitav je postupak opisan unutar poglavlja 2., potpoglavlja **2.1.1.2. Kreiranje terena i okruženja igre**, na 4. stranici ovoga rada.




Slika 8: Prikaz terena videoigre (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Nakon bojanja terena potrebno je nadodati elemente koji će popuniti stazu kako ne bi izgledala prazno. Odlučeno je da će stazu okruživati drveće. Ono se u program dodaje preko integrirane trgovine – *Unity Asset Store*. U trgovini se pronađe željeni objekt koji je potrebno dodati te ga se preuzme i uveze u program kao imovinu koju se može koristiti, o tome je već bilo riječi u jednome od prethodnih poglavlja[8]. Nakon toga je drveće potrebno dodati u alat koji je kreiran baš u svrhu dodavanja drveća. Prilikom dodavanja važno je odabrati model kao objekt, a ne materijal ili teksturu jer bi u protivnom došlo do grešaka i problema u ponašanju programa. Nakon uspješnog dodavanja objekata oni se dodaju na principu bojanja gdje se odabire veličina i gustoća drveća koju ćemo dodavati pomoću alata.

[8] Pogledati potpoglavlje **2.1.1.3. Kreiranje elemenata** na 5. stranici ovoga završnog rada.




Slika 9: Prikaz terena s drvećem (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Kako bi igra bila realističnija potrebno je dodati dvije ključne stvari. Prva je vjetar, a druga je svjetlo, odnosno promjena vremena.

**Vjetar** se dodaje tako da se u programu kreira **novi 3D objekt** – *WindZone* – pomoću kojega je moguće namještanje jačine, smjera i turbulencije samog vjetra. Prilikom kreiranja, zona vjetra namještena je na normalne postavke te je potrebno i modelima drveća koje smo prijašnje dodali dodijeliti *Bend faktor* koji im omogućava pomicanje prilikom utjecaja vjetra. Nakon toga drveće bi trebalo biti aktivno prilikom igranja igre što će se moći vidjeti pri pristupanju samoj videoigri.

**Svjetlo** ili **promjena vremena** još je jedan od elemenata koji je moguće skinuti putem integrirane trgovine. Nakon procesa koji je prethodno objašnjen odabire se vrijeme koje želimo te se ono dodaje i primjenjuje na izgled scene, odnosno igre.

## 4.3. Kreiranje Igrača

Nakon kreiranog okuženja potrebno je **kreirati igrača** koji će se kretati u tom okruženju. Igrač, ili u ovom slučaju automobil, uvezen je putem trgovine gdje je preuzeto sve potrebno kako bi automobil funkcionirao u jednom paketu. Taj paket sadržava predodređene kontrole, zvukove i fiziku koja je zaslužna za **ponašanje automobila**. Nakon preuzimanja automobila

potrebno ga je pozicionirati na stazu, u ovom će slučaju automobil biti pozicioniran na mjesto gdje se smatra da će biti početak kruga, odnosno **startna pozicija za trkanje**.

Slika 10: Automobil na stazi unutar videoigre (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Nakon postavljanja automobila na stazu, prilikom pokretanja igre nemamo pregled nad samim automobilom jer kamera koja je zaslužna za igranje nije namještena da prati automobil, već je postavljena na početno mjesto. Kako bismo to promijenili, potrebno je samom automobilu dodati objekt, u ovom slučaju kocku, koja će biti povezana s automobilom. Na tu kocku sada po želji pozicioniramo kameru kako bi bila optimalna za igrača.

Nakon **pozicioniranja glavne kamere** bilo je potrebno dodati još dvije opcije pogleda. Prva opcija je bila u prvom licu, a druga opcija slična je glavnoj opciji samo malo udaljenija da igrač ima veći pregled terena na kojem se nalazi. Kako ova opcija nije podržana u već prije postavljenim kontrolama, bilo je potrebno dodati novu kontrolu preko izbornika *Input*, gdje smo povećali brojku kontrola za jednu te smo odabrali koja će tipka na tipkovnici odgovarati promjeni kamere. Nakon što smo nadodali tipku potrebno je napisati kod koji je zadužen za funkcionalnost iste. Raspisani kod nalazi se u nastavku ovoga poglavlja.

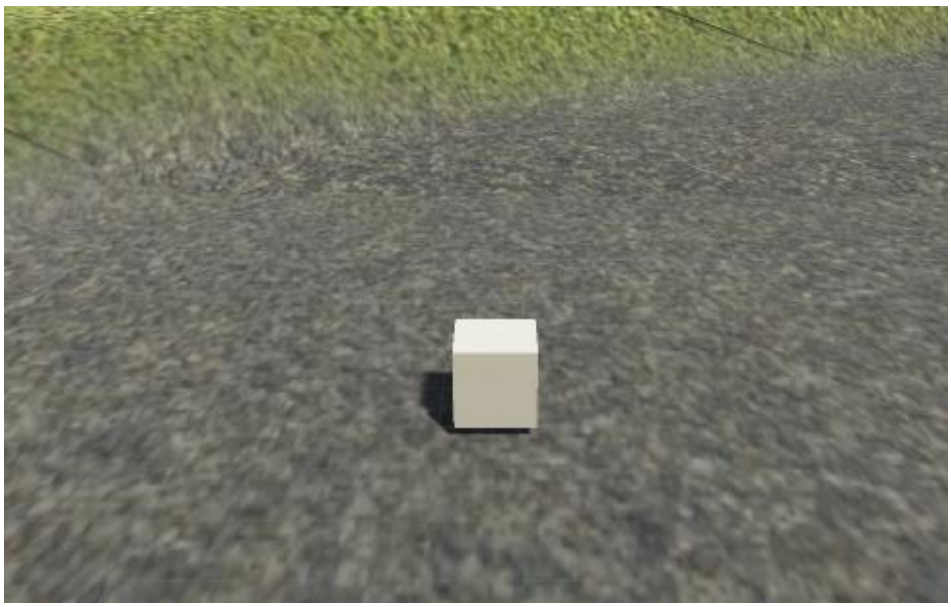
```
public class PromjenaKamere : MonoBehaviour {

    public GameObject NormalnaKamera;
    public GameObject DaljnjaKamera;
    public GameObject FirstPerson;
    public int Promjena;


    void Update () {
        if (Input.GetButtonDown ("Kamera")) {
            if (Promjena == 2) {
                Promjena = 0;
            } else {
                Promjena += 1;
            }
            StartCoroutine (Mjenjanje ());
        }
    }
    IEnumerator Mjenjanje(){
        yield return new WaitForSeconds (0.01f);
        if (Promjena == 0) {
            NormalnaKamera.SetActive (true);
            FirstPerson.SetActive (false);
        }
        if (Promjena == 1) {
            DaljnjaKamera.SetActive (true);
            NormalnaKamera.SetActive (false);
        }
        if (Promjena == 2) {
            FirstPerson.SetActive (true);
            DaljnjaKamera.SetActive (false);
        }
    }
}
```

Programski kod za promjenu kamere

Slika 11: Pogled igrača (izvor: snimka zaslona, Rafael Dominik Lakoseljac, 2022.)

## 4.4. Kreiranje protivnika

Kako bi igrica imala smisla, odnosno kako se ne bismo utrkivali sami protiv sebe, potrebno je dodati barem **jednog protivnika**. Taj protivnik kreira se na sličan način kao i igrač, ali uz razliku da se kod kreiranja ne uvozi sam automobil već automobil koji sadrži **umjetnu inteligenciju**, odnosno predodređeni program koji radi na principu oznaka koje programer dodaje te se protivniku zadaje da ih prati i vozi od jedne do druge kako bi završio krug.



Slika 12: Oznaka za praćenje (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

U ovoj je igrici kreirano osam takvih oznaka koje protivnik treba pratiti. Zbog veličine staze broj oznaka nije velik, ali s povećanjem kompleksnosti staze taj broj raste. Kako bi se smanjili problemi kod kretanja protivnika, veći broj oznaka pridonosi težem i preciznijem protivniku. Pomoću ove metode moguće je kreirati optimalni krug, odnosno vrijeme koje će protivnik ostvariti, te se ovisno o pozicijama tih točaka ostvaruju različite težine. Nakon pozicioniranja svih oznaka na stazu potrebno je napisati skriptu koja će biti zaslužna za promjenu cilja protivniku. Kako bi to bilo moguće, potrebno je kreirati još jednu oznaku koja će biti promjenjiva. To znači da će na tu oznaku biti povezana skripta koja je zadužena za promjenu. Skripta se temelji na grananjima koja primaju broj oznake i ovisno o broju prilikom „sudara“ s oznakom, ono se poveća za jedan i prelazi na sljedeću oznaku. Sudari se događaju tako da ne ometaju protivnika u njegovom kretanju jer su isključeni, odnosno nisu vidljivi te im je isključeno sudaranje s okolinom. Svaka oznaka kreirana je kao okidač što znači da se prilikom sudara odvija događaj koji mijenja poziciju promjenjive oznake.

Spomenuti kod možete vidjeti u nastavku ovoga poglavlja.

```
public class OznakaTraker : MonoBehaviour {

    public GameObject Oznaka;
    public GameObject Tocka1;
    public GameObject Tocka2;
    public GameObject Tocka3;
    public GameObject Tocka4;
    public GameObject Tocka5;
    public GameObject Tocka6;
    public GameObject Tocka7;
    public GameObject Tocka8;
    public int BrojOznake;

    void Update () {
        if (BrojOznake == 0) {
            Oznaka.transform.position = Tocka1.transform.position;
        }
        if (BrojOznake == 1) {
            Oznaka.transform.position = Tocka2.transform.position;
        }
        if (BrojOznake == 2) {
            Oznaka.transform.position = Tocka3.transform.position;
        }
        if (BrojOznake == 3) {
            Oznaka.transform.position = Tocka4.transform.position;
        }
        if (BrojOznake == 4) {
            Oznaka.transform.position = Tocka5.transform.position;
        }
        if (BrojOznake == 5) {
            Oznaka.transform.position = Tocka6.transform.position;
        }
        if (BrojOznake == 6) {
            Oznaka.transform.position = Tocka7.transform.position;
        }
        if (BrojOznake == 7) {
            Oznaka.transform.position = Tocka8.transform.position;
        }

    }

    IEnumerator OnTriggerEnter(Collider sudar){
        if (sudar.gameObject.tag == "Protivnik") {
            this.GetComponent<BoxCollider> ().enabled = false;
            BrojOznake += 1;
            if (BrojOznake == 8) {
                BrojOznake = 0;
            }
            yield return new WaitForSeconds (1);
            this.GetComponent<BoxCollider> ().enabled = true;
        }

    }

}
```

Programski kod za sustav praćenja

## 4.5. Kreiranje sučelja

### 4.5.1. Brojač krugova i vremena

Igrice utrkivanja moraju sadržavati **nekoliko ključnih stvari** kako bi igrač znao što mu je cilj i koliko mu je vremena trebalo kako bi ostvario taj cilj. S obzirom na to da se radi o igri utrkivanja, igrač mora znati **koliko krugova je potrebno odvoziti** i **koliko mu je vremena bilo potrebno pri vožnji u jednome krugu** te **koje mu je najbolje prolazno vrijeme**.


Slika 13: Sučelje (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Budući da je riječ o kratkoj stazi, vrijeme će rijetko prijeći jednu minutu, ali ostavljena je ta opcija za buduće nadogradnje ili dodavanje većih i kompleksnijih staza. Kako bi se vrijeme spremilo, potrebno je bilo dodati oznaku, slično kao i kod protivnika, ali ovdje je ta oznaka morala pokriti cijelu ciljnu ravninu kako se ne bi dogodilo da se prođe ciljnom ravninom, a da se vrijeme nije spremilo. Na tu se oznaku dodala skripta koja je napisana kako bi pamtila vrijeme u minutama, sekundama i stotinkama kako bi ono bilo što preciznije. Prilikom završavanja kruga vrijeme kruga se sprema, a trenutno vrijeme postavlja se na nulu, odnosno počinje ponovno računanje vremena za svaki sljedeći krug. Nakon prelaska preko cilja, broj kruga izvršenog kruga povećava se za jedan te kada broj krugova bude jednak unaprijed zadanom/postavljenom broju krugova, utrka se završava. Prilikom ponovnog pokretanja igrice, učitat će se najbolje spremljeno vrijeme, a za to je zadužena druga skripta koja prima informacije koje su bile zapisane u prethodnom pristupanju videoigri.

```
public class UpravljacVrremena : MonoBehaviour {

    public static int Minute;
    public static int Sekunde;
    public static float Stotinke;
    public static string PrikazStotinke;

    public GameObject MinuteBox;
    public GameObject SekundeBox;
    public GameObject StotinkeBox;
```

```
    public static float Vrijeme;

    void Update () {
        Stotinke += Time.deltaTime * 10;
        Vrijeme += Time.deltaTime;
        PrikazStotinke = Stotinke.ToString ("F0");
        StotinkeBox.GetComponent<Text> ().text = "" + PrikazStotinke;

        if (Stotinke >= 10) {
            Stotinke = 0;
            Sekunde += 1;
        }

        if (Sekunde <= 9) {
            SekundeBox.GetComponent<Text> ().text = "0" + Sekunde + ".";
        } else {
            SekundeBox.GetComponent<Text> ().text = "" + Sekunde + ".";
        }

        if (Sekunde >= 60) {
            Sekunde = 0;
            Minute += 1;
        }

        if (Minute <= 9) {
            MinuteBox.GetComponent<Text> ().text = "0" + Minute + ":";
        }else{
            MinuteBox.GetComponent<Text> ().text = "" + Minute + ":";
        }
    }
}
```

Programski kod za računanje vremena

### 4.5.2. Minijaturna mapa (*Mini map*)

Kako bi igrač imao bolju perspektivu gdje se nalazi, odnosno na kojem je dijelu kruga i koliko mu je preostalo do samoga kraja, kreirana je **minijaturna mapa**. Minijaturna mapa pozicionirana je u desnom donjem kutu kako igraču ne bi zaklanjala ostatak ekrana. Minijaturna mapa je ustvari još jedna kamera koja prikazuje sliku kao i glavna kamera koja je povezana na automobil. Kamera je pozicionirana tako da automobil gleda odozgo, iz takozvane ptičje perspektive, kako bi igrač mogao vidjeti stazu, primjerice nadolazi li nekakav zavoj ili prepreka, ali i kako bi mogao pratiti protivnika.

Slika 14: Minijaturna mapa (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

### 4.5.3. Početni meni, odabir automobila i staze

Prilikom pokretanja igre, otvara nam se izbornik u kojem imamo opcije za odigravanje igre, odabir staze ili izlaz iz igre. Kod odabira kontrole otvara nam se nova scena na kojoj su prikazane sve kontrole koje se koriste za igranje igrice. To novome korisniku, odnosno igraču, omogućava da prije ulaska u samu igricu dobije uvid u sve ponuđene mogućnosti.

Prilikom odabira ***Odigraj igru*** zamišljeno je bilo nasumično biranje staze i boje automobila, ali s obzirom na to da u ovoj verziji igrice imamo samo dvije boje automobila i jednu stazu, gumb vodi na drugu scenu gdje se pritiskom na gumb odabire automobil i staza. Ovdje je, također, ostavljena mogućnost nadogradnje za buduće verzije videoigre.


Slika 15: Početni izbornik unutar videoigre (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

Nakon pritiska gumba dolazimo do izbornika za automobile i stazu. Prvi odabir koji nam je dopušten jest **odabir automobila**. Nakon što odaberemo željenu boju automobila – moguće je odabrati između crvene ili ljubičaste – nudi nam se novi odabir, a to je odabir staze. Kod odabira staze nudi nam se samo jedna staza, kao što je prethodno spomenuto. Ipak, ova je opcija postavljena s ciljem daljnjeg poboljšanja i nadodavanja staza. Posljednja opcija na ekranu je još i *Izlaz*, odnosno povratak na glavni izbornik koji se nalazi prije ove scene prikazane na *Slici 15*. Prilikom odabira staze igra se automatski pokreće i počinje odbrojavanje za kretanje utrke.


Slika 16: Izbornik automobila i staza unutar videoigre (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

## 4.5.4.Odbrojavanje

Svaka igra utrkivanja započinje s nekom vrstom odbrojavanja, bio to semafor ili klasično odbrojavanje 3,2,1 – kreni!

U ovome je slučaju odabrana opcija odbrojavanje od 3 do 1. Kod kreiranja odbrojavanja trebalo je odabrati zvuk koji će nam naznačavati kada se odbrojavanje smanjilo za 1 te isto tako kada je vrijeme za pokret. Osim toga, bilo je potrebno i onemogućiti kontrole kako se igrač ne bi počeo kretati prije no što je odbrojavanje završilo. Kod toga nam je pomogla Js skripta koja prilikom njezinog poziva omogućava kontrolu nad igračevim automobilom i protivničkim automobilom, odnosno onemogućuje im prijevremeno kretanje. Za to nam je bila potrebna izrada programskog koda, a on izgleda ovako:

```
function Start () {
    CarControl.GetComponent("CarController").enabled = true;
    Protivnik.GetComponent("CarAIControl").enabled = true;
}
```

Programski kod za omogućavanje kontrola

Kako bi ta skripta funkcionirala potrebno ju je bilo pozivati kod skripte za odbrojavanje jer su kontrole od pokretanja programa onemogućene. Kako bi odbrojavanje imalo smisla, pomoću skripte postavljeno je to da se između promjene odbrojavanja čeka 1 sekunda te se za svaku sekundu broj smanjuje za 1. Kako se vrijeme ne bi pokretalo odmah pokretanjem igre, onemogućeno je dok se odbrojavanje nije završilo. Programski kod odbrojavanja je sljedeći:

```
public class Odbrojavanje : MonoBehaviour {

    public GameObject CountDown;
    public AudioSource AudioPriprema;
    public AudioSource AudioStart;
    public GameObject Timer;
    public GameObject Upravljanje;
    public AudioSource Soundtrack;
    public GameObject Automobil;

    void Start () {
        StartCoroutine (StartMetoda ());

    }

    IEnumerator StartMetoda(){
        Timer.SetActive (false);
        yield return new WaitForSeconds (0.5f);
        CountDown.GetComponent<Text> ().text = "3";
        AudioPriprema.Play ();
        CountDown.SetActive (true);
        yield return new WaitForSeconds (1);
        CountDown.SetActive (false);
        CountDown.GetComponent<Text> ().text ="2";
        AudioPriprema.Play ();
        CountDown.SetActive (true);
        yield return new WaitForSeconds (1);
        CountDown.SetActive (false);
        CountDown.GetComponent<Text> ().text = "1";
```

```
        AudioPriprema.Play ();
        CountDown.SetActive (true);
        yield return new WaitForSeconds (1);
        CountDown.SetActive (false);
        AudioStart.Play ();
        Soundtrack.Play ();
        Timer.SetActive (true);
        Upravljanje.SetActive (true);
        CarController.m_Topspeed = 200;
    }

}
```

## 4.6. Utrkivanje

Kao što je već spomenuto, nakon što smo odabrali automobil i stazu kreće odbrojavanje u obliku *3, 2, 1*. Kada se odbroji 1, igrač može kontrolirati automobil i krenuti s utrkivanjem. Prije nego što je odbrojavanje završeno, igrač nije u mogućnosti upravljati automobilom. To je omogućeno pomoću skripte koja tek nakon 3 sekunde omogućuje automobilu da krene. Ista skripta odnosi se i na protivnika. Nakon što je odbrojavanje završeno, korisnik kreće s utrkom te započinje mjerenje vremena. Prilikom završenog kruga, kao što je prethodno spomenuto, vrijeme se sprema te se analizira u usporedbi s već postignutim najboljim vremenom. Prilikom pokretanja igre započinje i **pozadinska glazba** koja poboljšava korisničko iskustvo. Glazba je preuzeta sa stranice Mixkit.co koja sadrži glazbu koja nema autorska prava, to je takozvana *free to use* glazba, odnosno nije potrebno platiti njeno preuzimanje i korištenje.


Slika 17: Korisnikova perspektiva pri igranju (izvor: slika zaslona, Rafael Dominik Lakoseljac, 2022.)

## 4.7. Kraj utrke

Prilikom završetka utrke korisnik dobiva poruku na ekranu ovisno o rezultatu koji je postigao. Ako je pobijedio, ispisat će se poruka ***Pobijedili ste!***, a u suprotnome ispisat će se poruka ***Izgubili ste!*** Uz prikladnu poruku, korisniku se nudi mogućnost povratka na glavni izbornik gdje može odlučiti hoće li ponovno zaigrati igricu ili je zatvoriti.

# 5. Predstavljanje videoigre

Nakon što je videoigra izrađena, potrebno je prikazati njezine **funkcionalnosti** kroz odigravanje. Pokretanjem videoigre dočekuje nas izbornik gdje nam je ponuđeno nekoliko odabira: **Kontrole, Odabir staze** i **Izlaz**.

Opcija **Kontrole** prikazuju nam kontrole koje su potrebne kako bismo kontrolirali automobil koji ćemo odabrati te kako se mogu promijeniti opcije kamere, odnosno pogled. Nadalje, opcija **Odabir staze** omogućava nam odabir automobila, a nakon toga i same staze na kojoj se želimo utrkivati. Nakon odabira automobila i staze prikazuje nam se ekran s odabranim automobilom na stazi te počinje odbrojavanje za početak utrke. Nakon što je odbrojavanje završeno, dobivamo mogućnost upravljanja automobilom te započinjemo s utrkivanjem. Kada prijeđemo oznaku od dva kruga, utrka završava s već prethodno spomenutom prikladnom porukom ovisno o ishodu same igre te opcijom za povratak na glavni izbornik.

Pomoću opcije **Izlaz** možemo izaći iz igre ili pomoću opcije **Odabir staze** nastaviti igrati videoigru, odnosno krenuti ispočetka.

# 6. Zaključak

Na samome kraju možemo zaključiti kako je *Unity* vrlo **praktičan alat za kreiranje velikog spektra videoigara**, bile to 2D, 3D ili čak danas VR ili AR igre koje nam približavaju virtualnu i artificijelnu stvarnost. Da bi se kreirala kvalitetna igra, potrebno je mnogo vremena i za jednog pojedinca to bi se moglo odužiti na nekoliko mjeseci, pa čak i nekoliko godina. Zbog toga su igre koje kreiraju pojedinci rijetko konkurente naspram igrara koje stvaraju velike kompanije. *Unity* bi, stoga, mogao poslužiti kako bi **mlade programere zainteresirao** da se bave tim poslom te da **razvijaju svoja znanja** kako bi se u budućnosti možda mogli zaposliti u nekoj od takvih tvrtki ili osnovati svoju.

U uvodnom smo dijelu obznanili da će se u ovome završnom radu izrađivati videoigra, prethodno je objašnjeno kako će to biti napravljeno, a potom je i prikazan svaki korak izrade stoga možemo reći da je najavljena videoigra **uspješno realizirana**. Također, ostavljena je i **mogućnost** njene **nadogradnje** – dodavanje većega broja staza, koje mogu biti i kompliciranije od postojeće, kreiranje većega broja modela automobila koje bi korisnik (igrač) dobio na izbor za igranje, ali i većega broja protivnika.

Posljednje, valja istaknuti kako su videoigre **nekada bile namijenjene samo mlađoj populaciji**, ali danas, povećavanjem razvoja i pristupačnosti tehnologija, može se vidjeti da i starija populacija igra videoigre. Tomu je pridonijelo povećanje razvoja tehnologija jer se danas programi za izradu igara, kao što je *Unity*, mogu besplatno instalirati i kreirati nekakve jednostavne igre te se danas tržište igara, pogotovo mobilnih, sastoji od velikog broja igara koje su kreirali pojedinci, a ne tvrtke kojima je to primarni cilj.

# Popis literature

[1] Peckham, Eric (2019) *How Unity built the world's most popular game engine*. Tech Crunch - https://techcrunch.com/2019/10/17/how-unity-built-the-worlds-most-popular-game-engine/ (posljednji put pristupljeno 5. rujna 2022.)

[2] Unity documentation – *Learning the Interface* – https://docs.unity3d.com/550/Documentation/Manual/LearningtheInterface.html (posljednji put pristupljeno 2. rujna 2022.)

[3] APS physics (2008) *This Month in Physics History – October 1958: Physicist Invents First Video Game* – https://www.aps.org/publications/apsnews/200810/physicshistory.cfm (posljednji put pristupljeno 25. kolovoza 2022.)

[4] The strong national museum of play – *Video Game History Timeline* – https://www.museumofplay.org/video_games/ (posljednji put pristupljeno 2. rujna 2022.)

[5] Technopedia – *Arcade Game* – https://www.techopedia.com/definition/1903/arcade-game (posljednji put pristupljeno 20. kolovoza 2022.)

# Popis slika