Izrada 3D trkaće igre u unity engine-u

Hranić, Adrian

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI STUDIJ RAČUNARSTVO

ADRIAN HRANIĆ

IZRADA 3D TRKAĆE IGRE U UNITY ENGINE-U

ZAVRŠNI RAD

Čakovec, 2023.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
STRUČNI STUDIJ RAČUNARSTVO

ADRIAN HRANIĆ

IZRADA 3D TRKAĆE IGRE U UNITY ENGINE-U

DEVELOPMENT OF A 3D RACING GAME IN UNITY ENGINE

ZAVRŠNI RAD

Mentor:
Nenad Breslauer, viši predavač

Čakovec, 2023.

MEĐIMURSKO VELEUČILIŠTE U ČAKOVCU
ODBOR ZA ZAVRŠNI RAD

Čakovec, 22. veljače 2022.

Polje: **2.09 Računarstvo**

ZAVRŠNI ZADATAK br. 2021-RAC-R-34

Pristupnik: **Adrian Hranić (0313021948)**
Studij: redovni preddiplomski stručni studij Računarstvo
Smjer: Inženjerstvo računalnih sustava i mreža

Zadatak: **Izrada 3D trkaće igre u unity engine-u**

Opis zadatka:

Igra se sastoji od razina i nekoliko različitih automobila na izbor korisniku, kao i protivnike u vidu AI igrača.
Potrebno je izraditi scene te sve potrebne elemente, koristiti mogućnosti koje pruža podsustav za osvjetljenje scene, animacije te koristiti simulaciju fizikalnih svojstava. Izraditi početnu scenu s izbornikom. Koristiti platformu Unity, programski jezik C# te dodatne programske alate. Završni rad mora sadržavati sažetak, sadržaj i uvod, nakon čega slijedi poglavlje u kojem je potrebno navesti i pojasniti osnovne ciljeve rada te očekivani rezultat. U narednom poglavlju potrebno je opisati primijenjene postupke, alate i metode. Poglavlje koje slijedi obrađivati će postignute rezultate nakon čega slijedi poglavlja u kojem se kritički raspravlja o primijenjenim metodama i postupcima te se u narednom poglavlju iznose glavni zaključci rada. Rad se završava poglavljima s popisom literature te prilozima

Zadatak uručen pristupniku: 22. veljače 2022.
Rok za predaju rada: 20. rujna 2022.

Mentor:

Nenad Breslauer, v. pred.

Predsjednik povjerenstva za završni ispit:

ZAHVALA

Prije svega želim se zahvaliti svom mentoru Nenadu Breslaueru na velikoj pomoći, strpljenju i razumijevanju kod izrade završnog rada. Također želim se zahvaliti svim profesorima na veleučilištu na maksimalnom trudu i pomoći u svakom koraku mojeg studiranja.

Adrian Hranić

SAŽETAK

Videoigre danas su jedan od najčešćih oblika zabave. Razvoj tehnologije tako prati i razvoj videoigara pa su videoigre danas postale dosta kompleksne i grafički vrlo napredne. Ne čudi stoga činjenica da je razvoj videoigara u današnjici postao vrlo složen proces u kojem sudjeluje čitav tim stručnjaka, a sam razvoj često zna trajati i do par godina prije nego li je igra u potpunosti spremna za izlazak na tržište. Ovaj završni rad služi da bi se opisao cjelokupan proces u izradi 3D videoigre, a kod izrade koristi se razvojno okruženje Unity. Naravno postoji još dosta drugih okruženja za izradu videoigara kao što su CryEngine i Unreal Engine, ali Unity je i dalje najpoznatiji. Uspješnost Unity razvojnog okruženja prije svega leži u njegovoj jednostavnosti i odličnoj povezanosti s Visual Studiom, koji služi za pisanje kodova da bi igra bila funkcionalna i radila onako kako je zamišljeno. Konkretno, u ovom radu, skripte za funkcionalnost igre pisane su u C# programskom jeziku, a mogu se koristiti i mnogi drugi jezici kao što su C++, JavaScript, Python i ostali. Kao što je već spomenuto, u današnje vrijeme nije dovoljno biti dobar programer da bi se izradila videoigra, već je u samoj izradi potrebno kombinirati znanja iz različitih područja. Upravo zbog toga veliku pomoć kod izrade ovog rada predstavlja Unityeva službena trgovina s već unaprijed izrađenim elementima, Unity Asset Store. Upravo putem Asset Storea moguće je preuzeti brojne plaćene, ali i besplatne elemente da bi se dodatno olakšao i ubrzao postupak izrade same igre. Potrebno je napomenuti da će u ovom radu biti detaljno objašnjen postupak izrade različitih razina na izbor korisniku, kao i brojni drugi elementi kao što su brojač krugova, vrijeme trenutnog kruga i vrijeme najboljeg kruga. Sve su ovo samo osnovni elementi koji čine uspješnu trkaću igru, a kroz ovaj rad upoznat ćemo se s još mnogo drugih. Naravno bilo koju videoigru nemoguće je zamisliti bez glavnog izbornika kao i podizbornika koji služi da igrač odabere željeni automobil, željenu stazu i željeni način igre. Trenutno je igra zamišljena samo za PC, odnosno Microsoft Windows platformu.

Ključne riječi: *Unity, 3D, Visual Studio, videoigra, utrka*

SADRŽAJ

1. UVOD

Tema ovog završnog rada je izrada 3D trkaće igre koristeći programski alat Unity i programski jezik C#. Igra će se sastojati od nekoliko razina i nekoliko različitih automobila na izbor korisniku a sadrži i protivnike u vidu AI igrača. Potrebno je izraditi scene te sve potrebne elemente, koristiti mogućnosti koje pruža podsustav za osvjetljenje scene, animacije te isto tako koristiti simulaciju fizikalnih svojstava i izraditi početnu scenu s izbornikom. Cilj igre će, naravno, biti završiti utrku prvi, odnosno pobijediti sve AI protivnike. Nadalje će biti opisan programski alat Unity kao i detaljni proces stvaranja same igre u njemu, a bit će opisan i Microsoft Visual Studio koji nam služi za pisanje samog koda. Za početak treba spomenuti sam razvoj računalnih igara tijekom godina kao i žanr trkaćih igara.

2. RAZVOJ RAČUNALNIH IGARA KROZ POVIJEST

Sredinom prošloga, a osobito krajem 20. stoljeća, videoigre postaju novi i sve češći oblik zabave te u novije vrijeme polako postaju jedan od najpopularnijih oblika zabave u cijelom svijetu. Kada se govori o povijesti računalnih igara svakako treba spomenuti legendarni „Spacewar“ iz 1961. godine. Igra je bila razvijena na mini računalu DEC PDP-1, a osmislila su je dva studenta Massachusettskog instituta za tehnologiju (MIT). Sama igra prikazivala je borbu između dviju letjelica na vektorskom zaslonu, a cilj igre bio je uništiti drugu „neprijateljsku“ letjelicu. Treba napomenuti da su u početku videoigre bile dosta primitivne te uglavnom samo tekstualnog oblika. To su bile igre u kojima je igrač na raspolaganju imao nekoliko zapovijedi te je njihovim unošenjem određivao kuda se želi kretati. Međutim razvitkom sklopovlja, videoigre su postajale sve kompleksnije. Tijekom 80-ih godina prošlog stoljeća tekstualne igre polako odlaze u prošlost, a zamjenjuju ih nove igre koje sadrže grafičko sučelje. Razvoj tehnologije pratio je i računala koja su postala dovoljno snažna da renderiraju 3D računalne grafike, što je ponovno dovelo do razvoja novih žanrova videoigara kao što su simulacije letenja

i vožnje automobila. Takve igre često su na granici sa sportskim igrama. Od igara koje spadaju u žanr simulacije letenja svakako treba spomenuti Microsoftov simulator letenja („Microsoft Flight simulator“). Također 80-ih godina na tržište je lansirana možda najpopularnija akcijska igra svih vremena te ujedno igra koja je napravila pravu revoluciju u svijetu videoigara. Riječ je o „Super Mario Bros“ iz 1985. godine za „Famicom“ i „Nintendo Entertainment System“(NES). Tijekom 90-ih, lansirane su čak dvije igre koje će ostaviti dugotrajne posljedice na svijet videoigara. Prva takva igra je Wolfenstein 3D iz 1992. godine koja je definirala FPS žanr („First person shooter“). Druga takva igra je „Doom“ iz 1993. godine koja je u ono vrijeme bila prijelomna točka u grafici i dizajnu. [1].

Slika 1. Doom

Izvor:https://www.imdb.com/title/tt0286598/mediaviewer/rm2776213505/(22.12. 2022)

3. TRKAĆE IGRE

S obzirom na to da je ideja ovog završnog rada izraditi 3D trkaću igru, u ovom poglavlju valjalo bi se osvrnuti i detaljnije opisati taj žanr videoigara. Osnovni cilj ovakvog tipa igara uvijek je bio isti, a to je odvoziti najbrže vrijeme ili prvi doći do cilja. Godine 1973. tvrtka *Atari* izdala je po mnogima prvu trkaću igru svih vremena „Space Race“. Grafika spomenute igre bila je vrlo jednostavna i primitivna, ali u ono vrijeme činila se veoma naprednom. U igri su tako postojale dvije rakete, a cilj je bio da igračeva raketa prva dođe do cilja pritom izbjegavajući sve prepreke na putu. Danas postoji nekoliko podžanrova trkaćih igara kao što su arkadni, simulacijski, kart i futuristički. Arkadne igre uvijek su bile nešto jednostavnijeg tipa i dostupne svima, s nešto jednostavnijom fizikom upravljanja vozilom. Najpoznatija arkadna trkaća igra je „Need for Speed“. Simulacijske igre potpuna su suprotnost. One uglavnom vrlo vješto repliciraju ponašanje vozila iz stvarnog života, simuliraju aktivnu potrošnju guma i goriva prilikom utrke. Također sustav oštećenja na vozilima u takvim igrama nije samo vizualnog tipa, već oštećenja direktno utječu na samu upravljivost vozila. Najpoznatije igre iz tog žanra su „Assetto Corsa“, „Gran Turismo“, „Iracing“. Kart igre ponovno su nešto jednostavnijeg tipa. Takve igre obično imaju staze neobičnog oblika i jednostavnu mehaniku upravljanja, a daleko najpopularnija igra ovakve vrste je „Mario Kart“. Futurističke trkaće igre svoj su vrhunac uživale 80-ih i 90-ih godina prošlog stoljeća, dok su danas uglavnom rijetke. Takve igre uglavnom sadrže brojne sci-fi elemente, a obično budu vrlo zabavne. Neke od igara iz tog tipa su: „F-Zero X“, „Wipeout“, „STUN Runner“.[2].

Slika 2. Need for Speed
Izvor: https://techreen.com/wp-content/uploads/2021/07/Racing-Games-768x432.jpg.webp (22.12.2022)

4. ALATI

Za potrebe izrade ovog završnog rada koristit će se Unity i to konkretno verzija 2021.2.15, dok će se za pisanje koda koristiti Microsoft Visual Studio. Koristit će se još neki alati kao što su blender, ali oni će biti korišteni u vrlo maloj mjeri pa ih u ovome radu nije potrebno dodatno opisivati. Unity se temelji na objektno orijentiranom jeziku C#, a prije njega koristio se Boo koji je uklonjen nakon što je izdana verzija Unity 5.

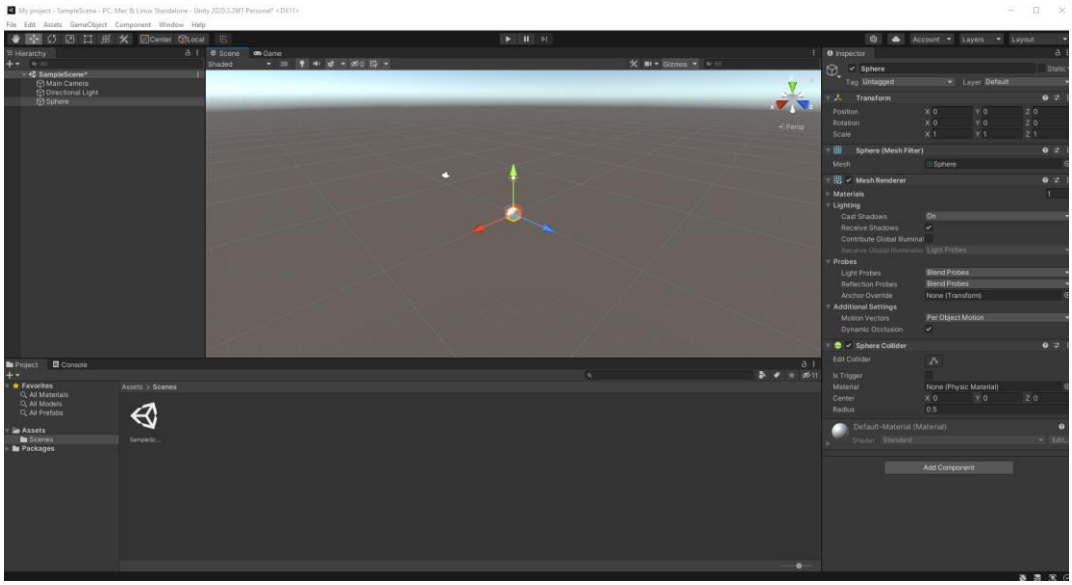
4.1 Unity

Unity (pokretač igre) prvi put je najavljen 2005. godine samo za Mac OS X platformu, a kasnije je ipak dodana podrška za najpopularniji Microsoft Windows. Unity se tako koristi za stvaranje trodimenzionalnih i dvodimenzionalnih igara kao i igara za proširenu i virtualnu stvarnost. Uspješnost Unity pokretača najbolje se vidi u tome što su ga s vremenom prihvatile i počele koristiti brojne industrije izvan videoigara, kao što su filmska industrija, automobilska industrija, arhitektura, građevina i inženjerstvo. Osnovni cilj Unity razvojnog okruženja bio je učiniti razvoj igara dostupnijim većem broju programera, a u tome su i uspjeli jer je Unity u kratkom razdoblju postao najkorišteniji „engine“ u izradi videoigara. Kao što je već napomenuto, Unity se koristi u izradi 2D i 3D videoigara. Unutar 2D igara Unity korisnicima omogućuje uvoz „spriteova“(grafički objekti u 2D nazivaju se „sprites“). Za 3D igre Unity omogućuje brojne opcije kao što su specifikacija kompresije tekstura, „mipmaps“, kao i postavke razlučivosti za svaku platformu koju mehanizam neke igre podržava, a pruža nam i podršku za mapiranje udara, preslikavanje refleksija, ambulantnu okluziju prostora zaslona (SSAO) i još mnoge druge opcije. Neki od najpopularnijih naslova koji su razvijeni u Unity okruženju su: „Super Mario“,“Temple Run“, „Rust“, „Among us“, „Angry Birds“ i „Subnautica“.

Slika 3. Unity logo
Izvor: https://www.gamesindustry.biz/unity-q1-2022-revenue-up-36-percent-to-usd320m(22.12.2022)

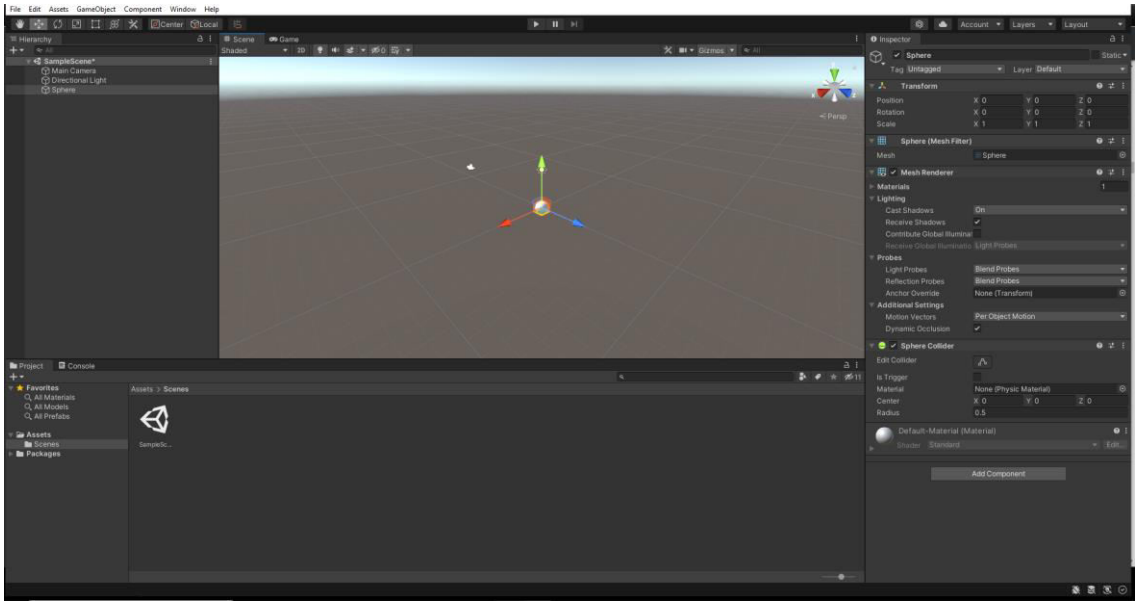
4.2 Unity sučelje

Unity sučelje isprva može izgledati dosta zbunjujuće i komplicirano. Iz tog razloga, ovo poglavlje služi da bi se približio način rada u Unity-u kao i izgled Unity sučelja. Osnovni i najbitniji prozor je „Project“ koji nam omogućuje navigaciju elementima potrebnim za izradu igre kao što su: skripte, zvuk, animacije.



Slika 4. Unity sučelje
Izvor: autor

Prozor Scena ili („Scene“) služi za stvaranje i postavljanje objekata koji se kasnije koriste u igri. Svaka scena u početku predstavlja jedan dio igre kao što su: glavni izbornik, igra, koje na kraju sastavljamo u jednu cjelinu. Sljedeća bitna opcija je opcija „inspector“ koja omogućuje da mijenjamo svaki objekt postavljen na scenu, a također svakome objektu možemo postavljati i dodatne komponente („Add Components“).



Slika 5. Objekti i inspektor

Izvor: autor

Sljedeći vrlo bitan element na sceni su kamere jer pomoću njih manipuliramo sa svim scenama u igri. Game prozor također uvelike ovisi o samoj kameri jer prikazuje ono što prikazuje kamera postavljena na sceni. Iznad alatne trake koja se nalazi iznad scene, postoje i gumbi pomoću kojih možemo pokrenuti ili pauzirati igru. Pomoću elemenata alatne trake možemo se kretati scenom i manipulirati objektima postavljenim na sceni. Svakako treba spomenuti i „animator Controller“ pomoću kojeg možemo dodavati animacije likovima ili objektima. Unity nudi dva različita načina programiranja a to su: JavaScript ili C#. U ovom radu koristi se C# u alatu Microsoft Visual Studio.

4.3 Unity Asset

Unity Asset je svaki element koji možemo koristiti u vlastitoj igri ili projektu na kojem radimo u Unityju. Treba spomenuti da asset ne mora nužno biti kreiran u samom Unityju, već to može biti neki 3D model napravljen u Blenderu, slika, audio datoteka ili bilo koja druga vrsta datoteke koju Unity podržava. Zbog toga postoji i Unity Asset Store u kojem direktno možemo pristupiti nekim assetima koje ćemo kasnije koristiti u našem projektu.

4.4 Unity Asset Store

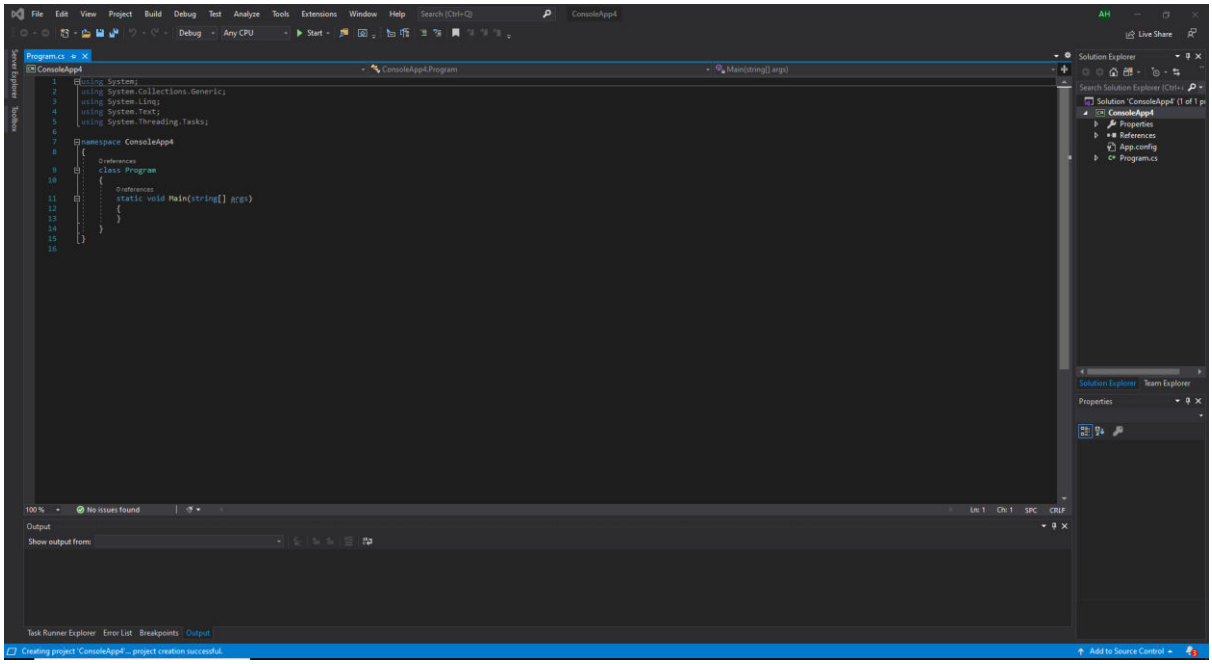
Unity Asset Store, kao što i samo ime govori, zapravo je trgovina s assetima. Postoji nekoliko vrsta asseta od najjednostavnijih tekstura, animacija pa sve do čitavih, već gotovih, projekata. Odlična stvar u vezi Asset Storea je što postoji i čitav niz potpuno besplatnih asseta, a valja spomenuti da same assete može postaviti bilo koji korisnik Unityja, a ne samo zaposlenici ili ljudi s posebnim dozvolama ili licensama. Asset Storeu moguće je pristupiti na dva načina. Prvi i najlakši način je direktno preko samog Unity Enginea, a drugi način je preko bilo kojeg podržanog web preglednika.

Vrste *asseta*

- 3D *asseti* – obuhvaćaju likove, vozila, rekvizite, animacije i vegetaciju
- *2D asseti* - obuhvaćaju *spriteove*, *teksture,* okolinu i likove
- Zvuk- Različiti zvučni efekti koji se mogu preuzeti s *Unity Asset Storea* ili kreirati samostalno
- *Add-ons* (Dodaci) – različite napredne funkcije koje je moguće dodati u projekt
- *Tools* (Alati) – obuhvaćaju primjerice vizualno skriptiranje i *AI (artificial intelligence)*
- *Templates* (Predlošci) – ova vrsta *asseta* namijenjena je prvenstveno početnicima, a odnosi se na preuzimanje unaprijed kreiranih dijelova projekta i slično
- *VFX* – vizualni efekti.[4].

4.5 Microsoft Visual Studio

Microsoft Visual Studio je integrirano razvojno okruženje (IDE) napravljeno od strane Microsofta. Visual Studio danas je jedan od najpopularnijih alata koji se koristi za izradu programa, web stranica, računalnih igara i web aplikacija. Također velika prednost Visual Studio alata je što službeno podržava čak 36 programskih jezika među kojima su neki od najpopularnijih kao što su C, C++, C#. Postoji i podrška za mnoge druge programske jezike: Python, Ruby samo putem dodatnih jezičnih usluga koje je potrebno dodatno instalirati. Još jedna velika prednost ovog alata jest to što nudi duboku integraciju s Unityjem tako da je dolazak do rješenja mnogo lakši te je osigurano poboljšanje produktivnosti od strane korisnika. Upravo ta odlična integracija između ova dva alata glavni je razlog što se koristi Microsoft Visual Studio za pisanje koda u ovom završnom radu. Za kraj, treba spomenuti da Microsoft nudi i besplatnu verziju Visual Studio pod nazivom „Community edition“ koja podržava sve dodatke bez ikakvih dodatnih troškova. [5].



Slika 6. Microsoft Visual studio sučelje
Izvor: autor

5. O IGRI

Ovo poglavlje služi da bi se opisala sama igra i neke njezine osnovne informacije kao i mehanike. Službeno ime igre je *Rally Extra*. Kao što je već spomenuto, igra je 3D trkaćeg žanra s naglaskom na reli (engl. *Rally*) utrkivanje. Igrač tako na raspolaganju ima tri različita automobila od kojih je jedan zaključan na startu, kao i tri različite staze za utrkivanje. Potrebno je napomenuti da u igri postoji i sustav novca, odnosno igrač nakon svake uspješno završene utrke dobiva određenu svotu novca kojom potom može kupiti i posljednji automobil za svoju kolekciju. Naravno jednom kupljeni automobil ostaje trajno u vlasništvu igrača pa ga nije potrebno ponovno kupovati. Postoje i dva različita načina igre. Prvi je klasični način utrkivanja u kojem je glavni cilj pobijediti AI protivnika za određen broj krugova, a drugi je *Time Trial* ili napad na što bolje vrijeme. Ovaj način igre nema AI protivnika niti određen broj krugova. Drugim riječima, igrač može odvoziti koliko god krugova želi da bi postigao što bolje vrijeme na određenoj stazi. Ovo je idealni način za one koji samo žele uživati u vožnji ili žele naučiti konfiguraciju staze prije nego li krenu u pravo utrkivanje, to jest u *Race Mode*. Nakon završene utrke, igraču se na ekranu prikaže izbornik s porukom da je utrka završena, a s istog izbornika može odabrati povratak u glavni izbornik ili odabrati sljedeću stazu. Naravno staze se razlikuju, a na njima postoji mnogo teških i nepreglednih zavoja kao i skokova da bi se maksimalno testirala igračeva vještina. Naravno, kao i svaka igra, i ova ima glavni izbornik iz kojeg je moguće ući u još jedan podizbornik iz kojeg igrač može odabrati automobil koji želi, kao i način igre i stazu. Na kraju postoji i pauza koja jednom kad je pritisnuta, u potpunosti zaustavlja tijek igre i otvara još jedan izbornik iz kojeg je potom moguće nastaviti igru ili se vratiti u glavni izbornik.

Slika 7. Izgled igre
Izvor: autor

6. UTRKIVANJE

Glavni smisao većine igara ovakvog žanra je utrkivanje dvaju ili više automobila po zadanoj mapi. Da bi ova funkcionalnost uopće bila moguća, na mapi je potrebno imati početnu liniju koja će predstavljati početak i kraj. Također potrebno je imati i kontrolne točke. Ako igrač uspješno prođe svim kontrolnim točkama, njegov prolazak početnom linijom računat će se kao uspješno odvožen krug. Jednom kad igrač napravi određen broj krugova, igra je gotova.

6.1 Odabir automobila, staze i načina igre

Prije nego li igrač uopće krene s utrkivanjem, potrebno je izabrati jedan od tri ponuđenih automobila, s time da posljednji, zeleni automobil, na početku igre bude zaključan te je potrebno sakupiti dovoljnu količinu novca za njegovu kupovinu.

```
public class OdaberiAuto : MonoBehaviour
{
   public static int TipAuta;
    public GameObject TrackWindow;
```

```
    public void CrveniAuto()  //1=crvena 2=plava 3=zelena
    {
        TipAuta = 1;
            TrackWindow.SetActive(true);
    }

    public void PlaviAuto()
    {
        TipAuta = 2;
        TrackWindow.SetActive(true);
    }

    public void ZeleniAuto()
    {
        TipAuta = 3;
        TrackWindow.SetActive(true);
    }

}
```

Kod 1. Odabir automobila

Izvor: autor

Nakon izbora automobila, potrebno je odabrati željeni način igre. Postoje dva različita načina. Prvi je onaj klasični način utrke (engl. *Race Mode*) u kojemu je, u određenom broju krugova, potrebno doći prvi do cilja. Drugi način igre nešto je jednostavniji te služi da bi se igrač upoznao sa samom konfiguracijom staze ili pokušao odvoziti što je najbolje moguće vrijeme u neodređenom broju krugova bez prisustva protivničkog automobila.

```
public class OdabirModa : MonoBehaviour
{
    public static int UtrkaMode;
    public GameObject OdabirStaze;


    public void TimeTrial()
```

```
    {
        UtrkaMode = 2;
        OdabirStaze.SetActive(true);
    }
        public void Utrka()
    {
        UtrkaMode = 0;
            OdabirStaze.SetActive(true);
    }
}
```

Kod 2. Odabir načina igre

Izvor: autor

Na kraju, potrebno je odabrati jednu od tri moguće staze. Jedna od staza vozi se noću. Staze su različite te dovoljno teške da bi bile izazovne i za nešto iskusnije vozače u ovakvom tipu igara.

```
public class OpcijeBotuna : MonoBehaviour
{
   //public void IgrajIgru()
    //{
      //  SceneManager.LoadScene(2);
    //}
    public void PokreniIgru()
    {
        SceneManager.LoadScene(1);
    }
    public void GlavniMeni()
    {
        SceneManager.LoadScene(0);
    }



    public void Staza01()
```

```
    {
        SceneManager.LoadScene(2);
    }

    public void Staza02()
    {
        SceneManager.LoadScene(3);
    }
    public void Staza03()
    {
        SceneManager.LoadScene(4);
    }


}
```

Kod 3. Odabir staze

Izvor: autor

6.2 Početak i kraj

Nakon što je igrač odabrao željenu stazu i automobil, počinje odbrojavanje od tri sekunde prilikom kojeg igrač nema nikakvu kontrolu nad automobilom. Jednom kad odbrojavanje završi, igrač dobiva potpunu kontrolu te utrka može početi. Utrka se sastoji od dva kruga, a pobjeđuje onaj tko ih prvi uspije odvoziti. Nakon što jedan od automobila prođe ciljem kao prvi, pokreće se kratka scena koja prati vodeći automobil iz koje je zatim potrebno izaći u glavni izbornik te pokrenuti novu utrku.

```
public class Countdown : MonoBehaviour
{

    public GameObject CountDown;
```

```
        public AudioSource GetReady;
        public AudioSource GoAudio;
        public GameObject LapTimer;
        public GameObject CarControls;
        public AudioSource Muzika;

        void Start()
        {
              StartCoroutine(CountStart());
        }

              IEnumerator CountStart()
              {
              yield return new WaitForSeconds(0.5f);
              CountDown.GetComponent<Text>().text = "3";
              GetReady.Play();
              CountDown.SetActive(true);
              yield return new WaitForSeconds(1);
              CountDown.SetActive(false);
              CountDown.GetComponent<Text>().text = "2";
              GetReady.Play();
              CountDown.SetActive(true);
              yield return new WaitForSeconds(1);
              CountDown.SetActive(false);
              CountDown.GetComponent<Text>().text = "1";
              GetReady.Play();
              CountDown.SetActive(true);
              yield return new WaitForSeconds(1);
              CountDown.SetActive(false);
              GoAudio.Play();
              Muzika.Play();
              LapTimer.SetActive(true);
              CarControls.SetActive(true);
}
}
```

Kod 4. Odbrojavanje

Izvor: autor

6.3 Kontrolne točke

Obvezna stavka svake trkaće igre su i kontrolne točke. Svaka staza sadrži minimalno jednu kontrolnu točku na sebi. Glavna svrha ovih kontrolnih točki je da igrač ne pokuša varati te skratiti duljinu staze i tako dobiti bilo kakvu nepravednu prednost u odnosu na protivnički automobil. Nakon što je automobil prošao startno ciljnom linijom, provjerava se je li igrač uspješno prošao svim kontrolnim točkama. Ako je igrač prošao svim točkama i ciljnom linijom, priznaje mu se krug kao odvožen nakon čega sve kontrolne točke ponovno automatski postavljaju vrijednost varijabli na false jer se njima mora ponovno proći u novome krugu.

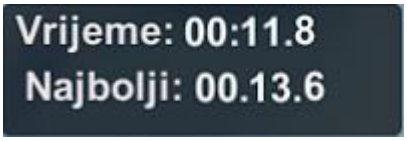
```
public class HalfPointTrigger : MonoBehaviour
{
   public GameObject LapCompleteTrig;
    public GameObject HalfLapTrig;

    void OnTriggerEnter()
    {
        LapCompleteTrig.SetActive(true);
        HalfLapTrig.SetActive(false);
    }
}
```

Kod 5. Kontrolne točke

Izvor: autor

6.4 Najbrži krug

Kao što je već spomenuto, igra ima razvijenu logiku bilježenja trenutnog i najbržeg kruga. Igrač u svakome trenutku može vidjeti svoj trenutni i najbrži krug u gornjem desnom kutu ekrana. Naime, čim igrač započne sa svojim krugom, automatski se pokreće i vrijeme trenutnog kruga, a jednom kad igrač prođe startno ciljnu liniju njegov odvoženi krug se zaustavlja te se to odvoženo vrijeme odmah uspoređuje s dotad najbrže odvoženim krugom na nekoj stazi. Ako je igrač uspješno odvozio novi najbrži krug, to vrijeme se automatski sprema pod najbolji krug. Svakako bitno je spomenuti da najbrži krug ostaje spremljen čak i kada igrač izađe iz igre te u nju ponovno uđe kasnije. S obzirom na to da igra ima tri različite staze, bitno je da najbrži krug na nekoj stazi ostane zabilježen samo kao najbrži na stazi na kojoj je postavljen, a ne na sve tri staze.



Slika 8. Trenutno i najbolje vrijeme
Izvor: autor

Da bi igra imala sposobnost da bilježi najbrže vrijeme, potrebno je napisati skriptu koja će uspoređivati odvožene krugove.

```
public class NajboljeVrijeme : MonoBehaviour


{
    public int Minute;
    public int Sekunde;
    public float MiliSekunde;
    public GameObject MinuteDisplay;
    public GameObject SekundeDisplay;
    public GameObject MiliSekundeDisplay;
```

```
void Start()
        {
            Minute = PlayerPrefs.GetInt("MinuteSave");
            Sekunde = PlayerPrefs.GetInt("SekundeSave");
            MiliSekunde =
   PlayerPrefs.GetFloat("MiliSekundeSave");

            MinuteDisplay.GetComponent<Text>().text = "" +
   Minute + ":";
            SekundeDisplay.GetComponent<Text>().text = "" +
   Sekunde + ".";
            MiliSekundeDisplay.GetComponent<Text>().text =
   "" + MiliSekunde;
        }
    }
```

Kod 6. Spremanje najbržeg kruga

Izvor: autor

6.5 Protivnički automobil

Glavni cilj ne samo ove, već i brojnih drugih igara ovakvog žanra, jest pobijediti protivnike, odnosno doći prvi do cilja. U ovoj igri igrač ima jednog suparnika kojeg kontrolira umjetna inteligencija. S obzirom na to da je sama kreacija umjetne inteligencije u igrama vrlo složen proces, ovdje je protivnički automobil sa svim bitnim svojstvima umjetne inteligencije preuzet s već spomenute trgovine assetima. Nakon toga u Unityju dobijemo niz opcija kojima možemo prilagoditi AI kako god želimo, možemo mu primjerice smanjiti ili povećati maksimalnu brzinu ili mu modificirati količinu gripa koju će imati u zavojima. Međutim, da bi sve to funkcioniralo, na svakoj je stazi potrebno kreirati „nevidljive“ markere kojima će protivnički automobil prolaziti i tako završiti svoj krug te samim time i utrku. Markeri moraju biti dovoljno veliki da bi protivnički automobil mogao bez problema voziti njima, ali moraju biti postavljeni na dobrom razmaku dase ne bi događalo da protivnik zapne na nekom zavoju ili na nekoj prepreci. Naravno, da bi sve to funkcioniralo, potrebno je napisati skriptu koja će voditi protivnički automobil markerima. Ovakva vrsta umjetne inteligencije je dosta

predvidljiva, ali s druge strane i dalje će pružati sasvim dovoljan izazov za sve igrače pa čak i one nešto iskusnije.

```
public class AIAuto01 : MonoBehaviour {
   public GameObject TheMarker;
public GameObject Marker01;
public GameObject Marker02;
public GameObject Marker03;
public GameObject Marker04;
public GameObject Marker05;
public GameObject Marker06;
public GameObject Marker07;
public GameObject Marker08;
public GameObject Marker09;


   public int MarkTracker;

void Update()
{
   if (MarkTracker == 0)
   {
        TheMarker.transform.position =
Marker01.transform.position;
   }
   if (MarkTracker == 1)
   {
        TheMarker.transform.position =
Marker02.transform.position;
   }
   if (MarkTracker == 2)
   {
        TheMarker.transform.position =
Marker03.transform.position;
   }
```

```
        if (MarkTracker == 3)
        {
              TheMarker.transform.position                    =
   Marker04.transform.position;
        }
        if (MarkTracker == 4)
        {
              TheMarker.transform.position                    =
   Marker05.transform.position;
        }
        if (MarkTracker == 5)
        {
              TheMarker.transform.position                    =
   Marker06.transform.position;
        }
        if (MarkTracker == 6)
        {
              TheMarker.transform.position                    =
   Marker07.transform.position;
        }
        if (MarkTracker == 7)
        {
              TheMarker.transform.position                    =
   Marker08.transform.position;
        }
              if (MarkTracker == 8)
             {
                   TheMarker.transform.position               =
   Marker09.transform.position;
              }
}

     IEnumerator OnTriggerEnter(Collider collision)
     {
        if (collision.gameObject.tag == "AIAuto01")
        {
```

```
            this.GetComponent<BoxCollider>().enabled        =
false;
            MarkTracker += 1;
            if (MarkTracker == 9)
            {
                  MarkTracker = 0;
            }
            yield return new WaitForSeconds(1);
            this.GetComponent<BoxCollider>().enabled = true;
        }
    }
    }
```

Kod 7. Protivnički automobil

Izvor: autor

6.6 Određivanje mjesta u utrci

Još jedan vrlo bitan faktor je da igrač u svakome trenutku zna na kojoj se poziciji trenutno nalazi. Kada vozač uspješno prijeđe ispred protivničkog automobila, njegova pozicija odmah će se promijeniti u 1. mjesto, a isto će se dogoditi i ako igrač izgubi poziciju, samo što će se tada njegova pozicija promijeniti u 2. mjesto. Samim time potrebno je osigurati mjesto na HUD-u da bi igrač u svakom trenutku znao na kojoj poziciji se nalazi. Sve to funkcionira tako da je na igračev automobil, kao i na protivnički, potrebno postaviti dva razmaknuta zida koji će pokrivati oba automobila. Jednom kad jedan zid bude ispred ili iza, igra će pomoću skripte odrediti na kojoj poziciji se igrač nalazi.

```
public class PozicijaIspred : MonoBehaviour
{
    public GameObject PrikazPozicije;

    void OnTriggerExit(Collider other)
    {
        if (other.tag == "PozicijaAuta")
        {
```

```
                PrikazPozicije.GetComponent<Text>().text    =
"1. Mjesto";
            }
        }
    }
```

Kod 8. Pozicija ispred

Izvor: autor

```
public class PozicijaIza : MonoBehaviour
{
    public GameObject PrikazPozicije;

    void OnTriggerExit(Collider other)
    {
        if (other.tag == "PozicijaAuta")
        {
            PrikazPozicije.GetComponent<Text>().text    =
"2. Mjesto";
            }
        }
    }
```

Kod 9. Pozicija iza

Izvor: autor

6.7 Novac

Velika količina igara u današnje vrijeme ima razvijen sustav novca ili bodova koji kasnije služe za otključavanje novih stvari u igri, kao što su primjerice novi automobil, novo oružje i slično. Ovakav sustav vrlo je popularan i prisutan u gotovo svim današnjim igrama jer se tako motivira igrača na daljnje odigravanje same igre. Ovaj rad tako također sadrži u sebi element novca koji funkcionira na sljedeći način. Igrač nakon svake uspješno završene utrke dobiva određenu količinu novca pomoću koje kasnije može otključati i posljednji zeleni automobil za svoju kolekciju. Bitno je naglasiti da jednom otključani automobil ostaje trajno u vlasništvu igrača pa ga nije potrebno ponovno kupovati. Količinu novca moguće je vidjeti u svakome trenutku i to u glavnom izborniku u gornjem lijevom kutu ekrana.

Slika 9. Količina novca
Izvor: autor

```
public class Pare : MonoBehaviour



{
    public int Novac;
    public static int TotalniNovac;
    public GameObject NovacDisplay;

    // Start is called before the first frame update
    void OnEnable()
    {
```

```
            TotalniNovac = PlayerPrefs.GetInt("OsvojeniNovac");
        }


        // Update is called once per frame
        void Update()
        {
            Novac = TotalniNovac;
            NovacDisplay.GetComponent<Text>().text = "Novac $"
+ Novac;
        }
    }
```

Kod 10. Sustav novca

Izvor: autor

7. HUD

Ovo poglavlje služi kao objašnjenje HUD-a (eng. *Heads-Up Display*) unutar samog rada. HUD služi da bi igrač u svakom trenutku imao pristup osnovnim informacijama koje se odnose na stanje u igri u kojoj se nalazi. Bitno je da HUD elementi budu dovoljno veliki i vidljivi, ali opet donekle transparentni da ne bi odvlačili pažnju prilikom igranja.[6]. HUD u ovoj igri prikazuje nam broj odvoženih i broj ukupnih krugova u utrci u gornjem lijevom kutu ekrana. Gornji srednji dio ekrana zadužen je za prikaz pozicije na kojoj se igrač trenutno nalazi, dok gornji desni kut ekrana služi za prikaz trenutnog i najboljeg vremena na nekoj stazi. Glavni izbornik sadrži tek jedan HUD element i to količinu novca u gornjem lijevom kutu. Razlog zašto je taj element samo u glavnom izborniku jest taj što upravo preko tog izbornika vozač kupuje novi automobil pa ne bi imalo previše smisla da isti element bude stalno prisutan na ekranu tijekom utrke. Bitna napomena je da pod HUD ne spadaju izbornici u igri kao što su

izbornik za izlaz iz igre ili izbornik koji nam služi za odabir staze, automobila i željenog načina igre.

Slika 10. Izgled cjelokupnog HUD-a
izvor: autor

8. KAMERA

Vjerojatno jedan od najvažnijih elementa svakog projekta je kamera. Upravo kamera određuje ono što igrač vidi na ekranu te, ako je loše konfigurirana, može se dogoditi da igrač neće vidjeti dovoljno ili neće uopće vidjeti sliku koja mu je potrebna za uspješno savladavanje neke razine. U ovome završnome radu fokus je na kameri koja cijelo vrijeme prati igračev automobil kada prolazi zavojima i ostalim preprekama. Glavni cilj jest prilagoditi kameru tako da igrač može dobro vidjeti svoj automobil, ali i situaciju oko njega da bi na vrijeme mogao uočiti sve zavoje i potencijalne zamke na stazi. Međutim, ako samo dodamo kameru na automobil, vidjet ćemo da se ona nekad nekontrolirano okreće kada se igračev automobil izvrti ili prevrne što bi moglo dovesti do toga da igrač izgubi potrebnu koordinaciju u prostoru. Zbog toga je potrebno napisati kratku skriptu koja nam omogućuje da kamera u svakom trenutku bude stabilna.

```
public class Kamera : MonoBehaviour
{

    public GameObject Car;
    public float CarX;
    public float CarY;
    public float CarZ;

);
    // Update is called once per frame
    void Update()
    {
```

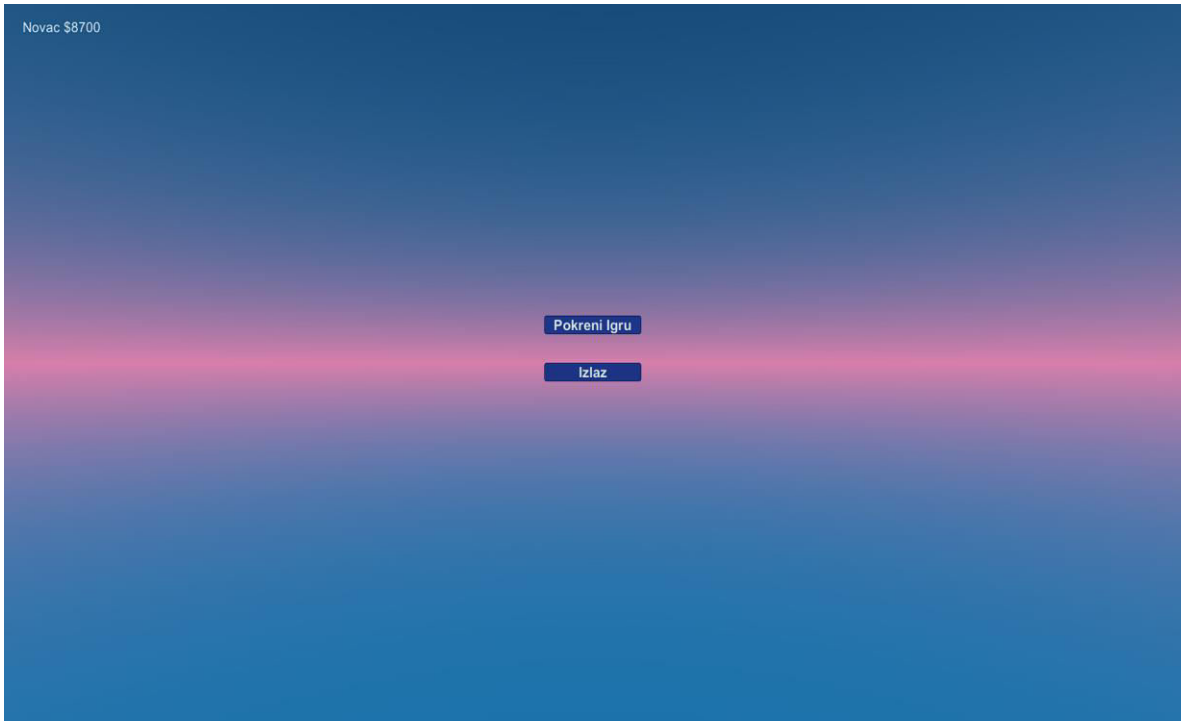
```
            CarX = Car.transform.eulerAngles.x;
            CarY = Car.transform.eulerAngles.y;
            CarZ = Car.transform.eulerAngles.z;
  transform.eulerAngles = new Vector3(CarX - CarX, CarY, CarZ -
Car}}}
```

Kod 11. Postavke kamere

Izvor: autor

9. IZBORNICI

Gotovo je nezamislivo imati videoigru bez barem jednog izbornika. Danas svaka igra sadrži i nekoliko različitih izbornika koji služe za lakšu navigaciju igrom, ali i za prezentaciju. Prije nego što igrač može početi s igrom, u većini slučajeva dočekat će ga glavni izbornik (engl. *Main Menu*). Ovaj Projekt sadrži ukupno tri izbornika. To su: glavni izbornik (engl. *Main Menu*), izbornik za odabir željene staze, automobila i načina igre te izbornik kod pauziranja igre (engl. *Pause Menu*). Glavni izbornik u ovoj igri izgleda poprilično jednostavno te sadrži samo dvije opcije. Prva opcija je „pokreni igru“ koja nam otvara sljedeći izbornik u kojem igrač bira željenu stazu, automobil i način igre, a druga opcija služi za izlazak iz igre. Da bi ove tipke uopće radile, potrebno je za svaku od njih napisati funkciju u skripti. Također, kao što je već spomenuto, u gornjem lijevom kutu ekrana nalazi se količina novaca koju je igrač sakupio prilikom dosadašnjeg igranja.



Slika 11. Glavni Izbornik
Izvor: autor

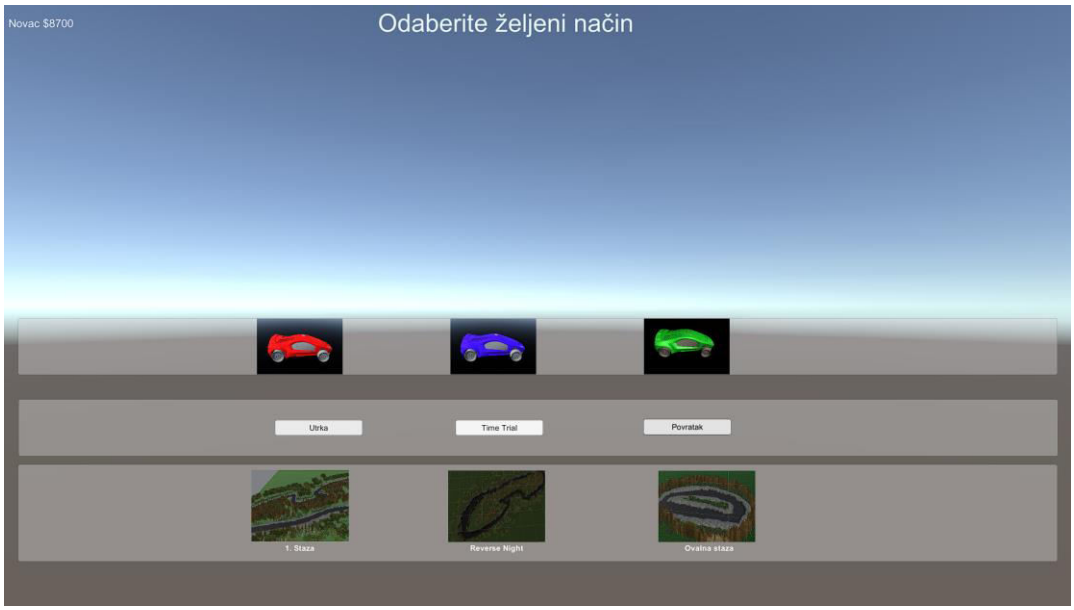
```
public class OpcijeBotuna : MonoBehaviour
{
   //public void IgrajIgru()
    //{
      //  SceneManager.LoadScene(2);
    //}
    public void PokreniIgru()
    {
        SceneManager.LoadScene(1);
    }
    public void GlavniMeni()
    {
        SceneManager.LoadScene(0);
    }

    public void EndGame()
    {
        Application.Quit();
```

```
        }
    }
```

Kod 12. Glavni izbornik

Izvor: autor

Nakon glavnog izbornika slijedi, u ovom slučaju još važniji, izbornik u kojem igrač bira automobil i stazu na kojoj će voziti kao i željeni način igre. Bitna napomena je da se svaki podizbornik ovdje otvara pojedinačno, odnosno potrebno je najprije izabrati željeni automobil da bi se otvorio podizbornik za željeni način ili za povratak u glavni izbornik, a posljednji se otvara podizbornik za biranje staze. Jednom kad igrač odabere stazu, utrka može početi.



Slika 12. Odabir Željenog načina

izvor: autor

Igru je moguće pauzirati u svakom trenutku pritiskom na tipku Esc na tipkovnici te se time otvara izbornik koji nudi mogućnost nastavka igranja, povratak u glavni izbornik ili izlazak iz cijele igre. Otvaranjem izbornika pauze vrijeme u igri se u potpunosti

zaustavlja te se zatamnjuje trenutna scena umjesto otvaranja novog izbornika.

Slika 13. Meni pauze
Izvor: autor

```
public class PauseMeni : MonoBehaviour {

    public GameObject PauseMenu;
public static bool IsPaused;


    // Start is called before the first frame update
    void Start()
    {
        PauseMenu.SetActive(false);
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Escape))
        {
            if (IsPaused)
```

```
            {
                ResumeGame();
            }
            else
            {
                PauseGame();
            }
        }

    }

    public void PauseGame()
    {
        PauseMenu.SetActive(true);
        Time.timeScale = 0f;
        IsPaused = true;
    }

    public void ResumeGame()
    {
        PauseMenu.SetActive(false);
        Time.timeScale = 1f;
        IsPaused = false;

    }

    public void GoToMainMeni()
    {
        Time.timeScale = 1f;
        SceneManager.LoadScene("OdabirStaze");
    }

    public void QuitGame()
    {
        Application.Quit();
    }
```

```
}
```

Kod 13. Pauza

Izvor: autor

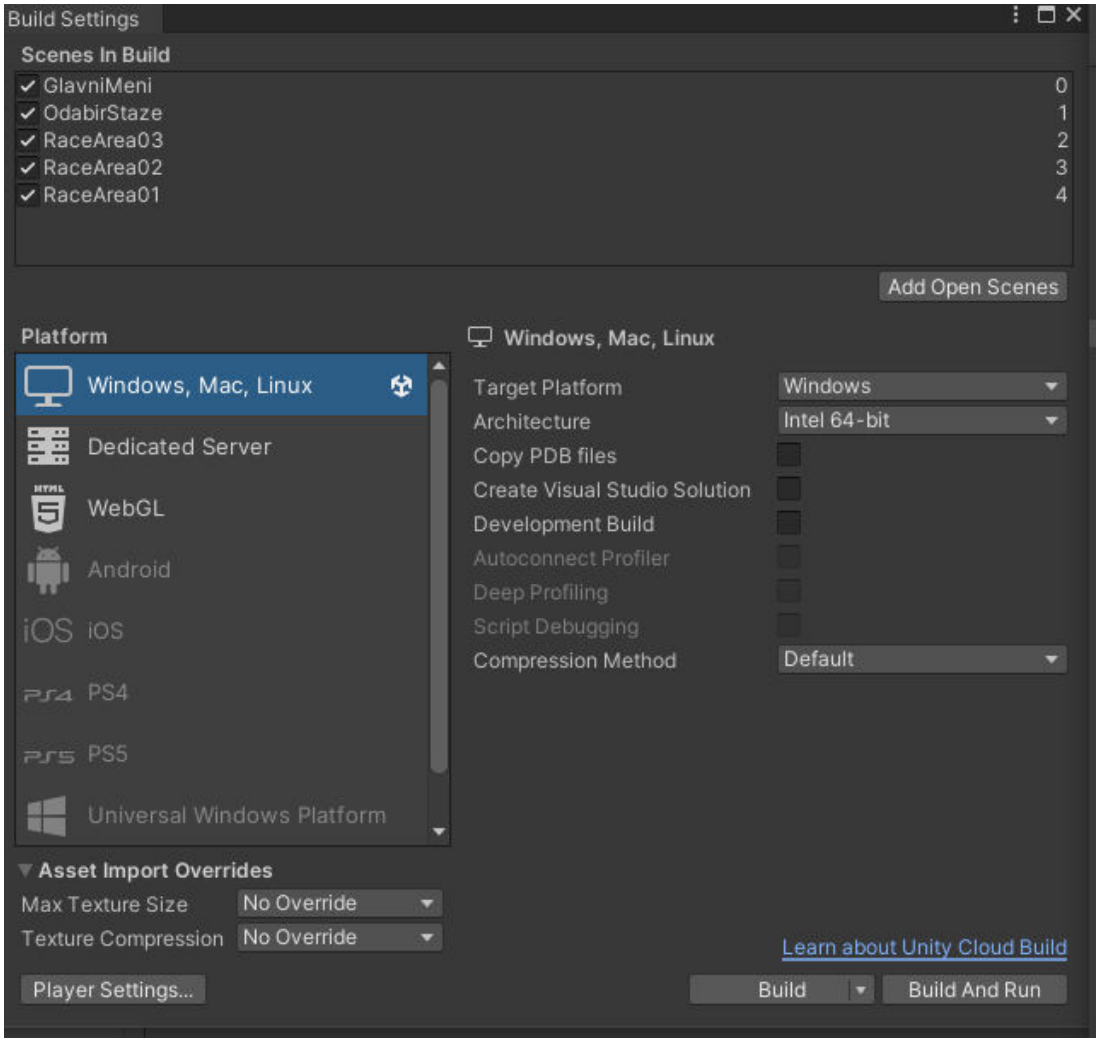
10.POZADINSKA GLAZBA

Kao što je to slučaj u filmovima i na televiziji, glazba u videoigrama obično se koristi za postavljanje tempa i atmosfere. Zvučni zapisi u videoigrama ipak idu i korak dalje jer se nalaze u osnovi svega što radimo u igri. Bilo da se upuštamo u neku tešku bitku ili samo mirno lutamo otvorenim svijetom, sve naše radnje prati glazba. Upravo zbog toga danas se često govori da glazba i zvučni efekti čine razliku između dobre i vrhunske videoigre.[7]. S obzirom na to da je izrada vlastite pozadinske glazbe i zvučnih efekata u igrama danas postala grana kojom se bave specijalizirani timovi, u ovom radu koristi se pozadinska glazba koja je preuzeta s već spomenutog Asset Storea. Da bi emitiranje glazbe uopće bilo moguće, Unity sadrži komponentu Audio Source u kojoj se nalazi polje Audio Clip u koje je potrebno dodati zvučnu datoteku. Još jedna pozitivna stvar u vezi Audio Source komponente je ta što nam omogućuje mnogo opcija i mogućnosti za podešavanje glazbe i zvučnih efekata.[8].

Slika 14. Audio Source
Izvor: autor

11.DISTRIBUCIJA

Posljednji korak u izradi videoigre svakako je njezina distribucija. Ovo je korak koji se radi tek na kraju kada je proces izrade igre u potpunosti završen. Unity u sebi već ima opciju distribucije koja nam omogućuje da pretvorimo Unity igru u već gotov proizvod za čije nam otvaranje nisu potrebni Unity ili Visual Studio. Drugim riječima, distribucijom zapravo dobijemo jednostavnu aplikaciju koja se pokreće dvostrukim klikom na ikonu. Potrebno je samo odabrati željeni operacijski sustav i neke od osnovnih podataka i postavki kao što su ime igre i slično. Konačno, potrebno je odabrati i sve scene koje će biti na raspolaganju igraču kao i platformu. Nakon toga potrebno je pritisnuti tipku Build koja stvara izvršnu datoteku igre.



Slika 15. Build postavke
Izvor: autor

12.ZAKLJUČAK

Gaming industrija nastavlja sa svojim velikim rastom gotovo svakodnevno. Pomalo zastrašujuće zvuči podatak da industrija videoigara danas vrijedi čak trostruko više nego filmska industrija. Upravo ove informacije služe kao dokaz da videoigre danas nisu samo izvor zabave, već posebna grana umjetnosti kojom autori pokazuju svoju umjetničku sposobnost.

Proces izrade videoigara danas je dodatno olakšan zahvaljujući brojnim alatima pomoću kojih se gotovo svaka osoba može iskusiti u ovom području te napraviti barem lakši projekt.. Međutim, kada je riječ o pravim ozbiljnim naslovima, situacija je bitno drugačija. Postoje brojni timovi unutar neke kompanije, primjerice timovi koji se bave isključivo programiranjem ili timovi koji se bave isključivo dizajnom. Upravo ovakav pristup i način izrade vrlo je zanimljiv, ali i dinamičan jer zahtijeva sposobnosti iz više grana izrade, a ne samo iz izrade likova, programiranja ili slično. Konačno, današnji proces izrade videoigara postao je vrlo skup, ali i dugotrajan proces pa se tako često može vidjeti da neka igra bude godinama u razvoju prije nego li je u potpunosti spremna za izlazak na tržište.

Ovaj projekt napravljen je tako da kombinira znanje iz programiranja i dizajniranja, dok su neke zahtjevnije stvari, poput izrade umjetne inteligencije, preuzete s Asset Storea.

13.POPIS LITERATURE

[1]. History of video games.

https://docs.unity3d.com/2018.2/Documentation/Manual/AssetStore.html (4.1.2023.)

[2]. Evolution of racing games.

https://antidote.gg/evolution-of-racing-games/ (6.1.2023.)

[3]. Lindsay Schardon, What is Unity? A guide for one of the top game engines
https://gamedevacademy.org/what-is-unity/ (15.1.2023.)

[4]. Using the Asset Store.

https://docs.unity3d.com/2018.2/Documentation/Manual/AssetStore.html
(15.1.2023.)

[5]. Welcome to the Visual Studio IDE.

https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022 (15.1.2023.)

[6]. Vangie Beal, HUD- Heads Up Display.

https://www.webopedia.com/definitions/hud/ (12.1.2023.)

[7]. The importance of Music in Video Games

https://www.anarapublishing.com/the-importance-of-music-in-video-games/
(13.1.2023)

[8]. Audio Source.

https://docs.unity3d.com/Manual/class-AudioSource.html (14. 1. 2023.)

14.POPIS KODOVA

15.POPIS SLIKA