UNIVERZITET SINGIDUNUM

Fakultet za informatiku i računarstvo

Razvoj lokalne multiplayer igrice

Food Feud

- diplomski rad -

Mentor: **Kandidat:**

prof.dr ***Mlađan Jovanović*** ***Sanja Golubović***

Beograd*, 2021.*

Sadržaj

Uvod **3**
1. Priprema razvojnog okruženja **6**
2. Korišćeni programi **8**
2.1. Blender 8
2.2. Adobe Illustrator 9
2.3. Adobe Photoshop 9
3. Struktura igrice u Unity-ju **11**
3.1. Project 11
3.2. Hierarchy 12
3.3. Inspector 14
3.4. Scene 14
3.5. Game 15
3.6. Console 16
4. Multiplayer u Unity-ju **17**
5. Korisnički interfejs **18**
5.1. Početna stranica igrice 18
5.2. Unos imena 19
5.3. Odabir hostovanja ili priključivanja serveru 20
5.4. Unos IP adrese 21
5.5. Odabir karaktera 22
5.6. Odabir oružja 23
5.7. Kontrole 24
5.8. Pobeda i poraz 25
6. Funkcionalnosti igrice **26**
6.1. Main menu 26
6.2. Osnovni gameplay 27
6.3. Upravljanje hostovanjem i priključivanjem serveru 30
7. Testiranje **31**
8. Poboljšanje igrice u budućnosti **33**
9. Zaključna razmatranja **34**
10. Literatura **35**

Uvod

Istorija video igrica predstavlja dugačak period kroz koji su se smenjivale i unapređivale različite tehnologije razvoja, od pojave prve video igrice 1950. godine pa do današnjeg dana. Svaki period imao je značajne doprinose što je učinilo da video igrice budu ono što danas jesu. Tehnološka ograničenja uticala su na sam ishod igrica, pa se tako prva igrica igrala na računaru koji je bio veličine 4 metra, i imala je samo jednu funkciju čiji unos se vršio preko jednostavnije verzije današnje tastature. Savremeno doba omogućava potpuno drugačiji doživljaj igranja igrica, pa su samim tim danas virtuelni svet i realnost jako usko povezani.

Kako je potražnja igrica rasla, tako se pojavljivalo i interesovanje da se doda mogućnost zajedničkog igranja igrica u realnom vremenu, pa je tako nastao koncept multiplayer igrica. Multiplayer, što u prevodu znači više igrača, prvi put se pojavio oko 1973. godine, kada je izašla igrica pod nazivom *"Empire"*. Danas uglavnom svaka igrica ima lokalni ili online multiplayer režim, što igračima pruža dodatne pogodnosti. Online režim omogućava povezivanje korisnika iz svih delova sveta. Neke od mana ovakih tipova igrica, jesu to što zahtevaju aktivnu internet konekciju, kao i to što može doći do određenih poteškoća ukoliko internet konekcija nije stabilna. Takođe posedovanje i održavanje servera mogu predstavljati veliki trošak, koji srazmerno raste sa veličinom igrice. Ovaj trošak se odnosi samo na online igrice, zato što lokalni režim dozvoljava jednom od igrača da na svojoj mreži napravi server, a ostali igrači se povezuju preko lokalne IP adrese serverskog uređaja. Multiplayer se zasniva na klijent-server arhitekturi, pa se sve informacije razmenjuju komunikacijom između servera i svih povezanih klijenata.

Kada je reč o razvoju igrica, danas postoji veliki broj programa koji su napravljeni tako da omoguće korisnicima da na što lakši način dobiju finalni proizvod. Dva najzastupljenija programa za razvoj igrica su Unity i Unreal Engine. Oni nude najviše pogodnosti kako za osobe sa iskustvom, tako i za početnike. Za odabir odgovarajućeg razvojnog okruženja potrebno je uzeti u obzir neke od zahteva igrice, kao što su na primer ciljana platforma, veličina, funkcionalnosti, itd..

Unity, koji je takođe poznat i pod imenom Unity 3D, je razvojno okruženje za kreiranje digitalnih proizvoda, uglavnom video igrica. Iako je primarno osvrnut kao video igricama, Unity se sve više primenjuje u raznim oblastima kao što su arhitektura, umetnost i dizajn, a takođe se može koristiti za razvoj aplikacija. Lansiran 2005. godine kao pristupačan profesionalan softver za razvoj igrica, Unity je bio dostupan samo za Mac OS[1], uz samo par ponuđenih platformi. Danas je on podržan i na Mac i Windows OS, a korisnik se može opredeliti za veliki broj platformi koje su mu na raspolaganju, od kojih su najpoznatije:

- Mac
- Windows
- Linux
- IOS
- Android
- WebGL
- Playstation
- XBOXONE

[1] Operativni sistem

- Nintendo Switch
- Oculus
- Steam VR
- Google Cardboard

U toku donošenja odluke o tome šta najbolje odgovara potrebama jednog korisnika, često prva impresija nekog programa bude baš ono što na korisnika ostavi najveći utisak. U tom pogledu Unity je to rešio kreiranjem veoma pristupačnog korisničkog interfejsa, koji ostavlja malo mesta za greške i podstiče korisnika da nastavi sa daljim korišćenjem. Uredno raspoređeni prozori i alatke, kao i jednostavno dodavanje komponenti na objekte, omogućavaju brz i lak razvoj. Pored tradicionalnog pisanja skripti, postoji i mogućnosti vizuelnog programiranja. Velika razlika između Unity i Unreal Engine programa je to što prazan Unity projekat ima svega 22.5 MB, dok Unreal prazan projekat ima preko 100 MB, što čini da kada je u pitanju razvoj mobilnih aplikacija Unity zauzima prvo mesto.

Pored svih navedenih pogodnosti koje Unity pruža, jako je bitno napomenuti da je on dostupan i kao besplatna verzija, koja se jako malo razlikuje od profesionalne verzije, i iz tog razloga se većina korisnika opredeli baš za ovo razvojno okruženje. Takođe dostupna je detaljna oficijalna dokumentacija koja se može koristiti za otklanjanje nedoumica i rešavanje problema. Postoje i oficijalni kursevi napravljeni od strane iskusnih programera, ali se može naći i veliki broj kurseva od strane pojedinaca na internetu. Unity poseduje i svoj sajt koji se zove Unity Answers, i on predstavlja forum na kojem korisnici mogu postavljati i odgovarati na pitanja od strane drugih korisnika. Važno je reći da se za razvoj uspešne igrice treba obratiti posebna pažnja ne samo na kod, već i na druge bitne elemente kao što su animacije, grafika, muzika i modeli. Kada je reč o velikim kompanijama to ne predstavlja problem, ali za pojedince koji nisu obučeni za sve ove oblasti ovo može zvučati kao obeshrabrujuća vest. Unity kako bi omogućio da svaka osoba dobije iste šanse da svoju ideju pretvori u stvarnost, otvorio je i svoju virtuelnu prodavnicu pod imenom Unity Asset Store, koja sadrži veliki broj plaćenih i besplatnih proizvoda. Kada se sve od navedenog uzme u obzir, ne dovodi se u pitanje zašto baš ovaj program drži monopol na tržištu razvojnih okruženja video igrica.

Kada se pomene industrija video igrica, može se reći da sve veći procenat ljudi prelazi sa kompjuterskih igrica na igrice za mobilne telefone. Ovaj tip igrica doživeo je veliki porast sa razvojem prenosivih uređaja, koji su ljudima bili dostupni u bilo kojem trenutku. Jedna od prvih igrica za mobilne telefone bila je tetris, i može se reći da je upravo ona kamen temeljac onome što je danas postalo jedna od najvećih industrija sveta.

Dve najpoznatije platforme za mobilne igrice danas su Android i IOS. Korisnička baza, kao i mogućnost ravzoja za obe platforme su veoma slične, ali neke od bitnih razlika koje se mogu navesti su to da:

- Pretplata za oficijalnu Android prodavnicu košta 25$, i kada se korisnik jednom pretplati on ima mogućnost neograničenog postavljanja svojih proizvoda doživotno, dok za oficijalnu Apple prodavnicu pretplata košta 99$ godišnje.
- Korisnici Android telefona pored Play Store[2]-a mogu instalirati igrice sa drugih prodavnica ili izvora, dok su svi oni koji poseduju IOS telefon ograničeni isključivo na App Store[3]

[2] Oficijalna Android prodavnica kompanije Google
[3] Oficijalna IOS prodavnica kompanije Apple

- Android je tipa otvorenog koda što omogućava da se lako prilagodi velikom broju uređaja, dok je IOS tipa zatvorenog koda i radi samo sa Apple uređajima
- Procenat korisnika Android telefona je znatno veći od procenta korisnika IOS telefona

U ovom radu opisan je postupak razvoja lokalne multiplayer igrice za Android platformu. Razvijena je korišćenjem Unity razvojnog okruženja i C# programskog jezika. Pored toga korišćeni su i programi za kreiranje 3D modela Blender, kao i programi za vektorsku i rastersku grafiku Adobe Illustrator i Adobe Photoshop.

Igrica se zove *"Food Feud"* i nastala je kao ideja sedmogodišnjeg dečaka, pa samim tim ciljnu grupu korisnika ove igrice predstavljaju deca.

Igrica omogućava igračima da se takmiče jedan protiv drugog u borbi sa hranom. Postji više različitih karaktera i oružja između kojih korisnici mogu da biraju. Takođe svaki korisnik se raspoznaje svojim imenom koje unosi na početku igrice. Igrač koji uspešno pogodi drugog igrača više puta postaje pobednik.

Pošto se radi o multiplayer igrici, potrebno je da oba korisnika imaju pristup istoj internet mreži, kao i da znaju svoju lokalnu IP adresu.

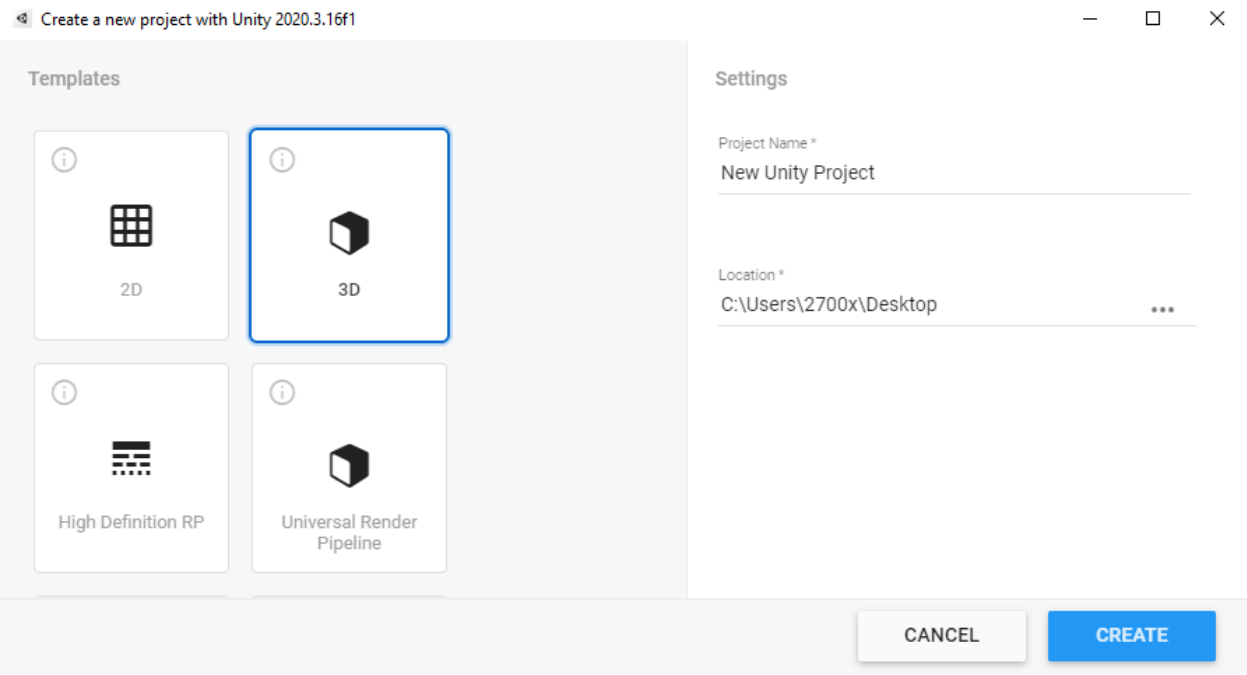
Podrazumevani jezik igrice je engleski, pošto on predstavlja univerzalni jezik koji se uglavnom koristi u svim video igricama.

1. Priprema razvojnog okruženja

Kao što je ranije napomenuto, ova igrica razvijena je pomoću Unity razvojnog okruženja. Ciljna platforma ove igrice je Android, i zbog toga se Unity pokazao kao najbolja opcija.

Programski jezi korišćen za razvoj ove igrice je C#, a pošto se radi o Android platformi korišćen je Android SDK[4], koji se može automatski instalirati uz Unity. Trenutno najveća verzija API[5] nivoa je 30, dok je minimalan API nivo ove igrice 19.

Postoji više opcija kreiranja projekta unutar Unity-ja od kojih su neke prikazane na sledećoj slici.



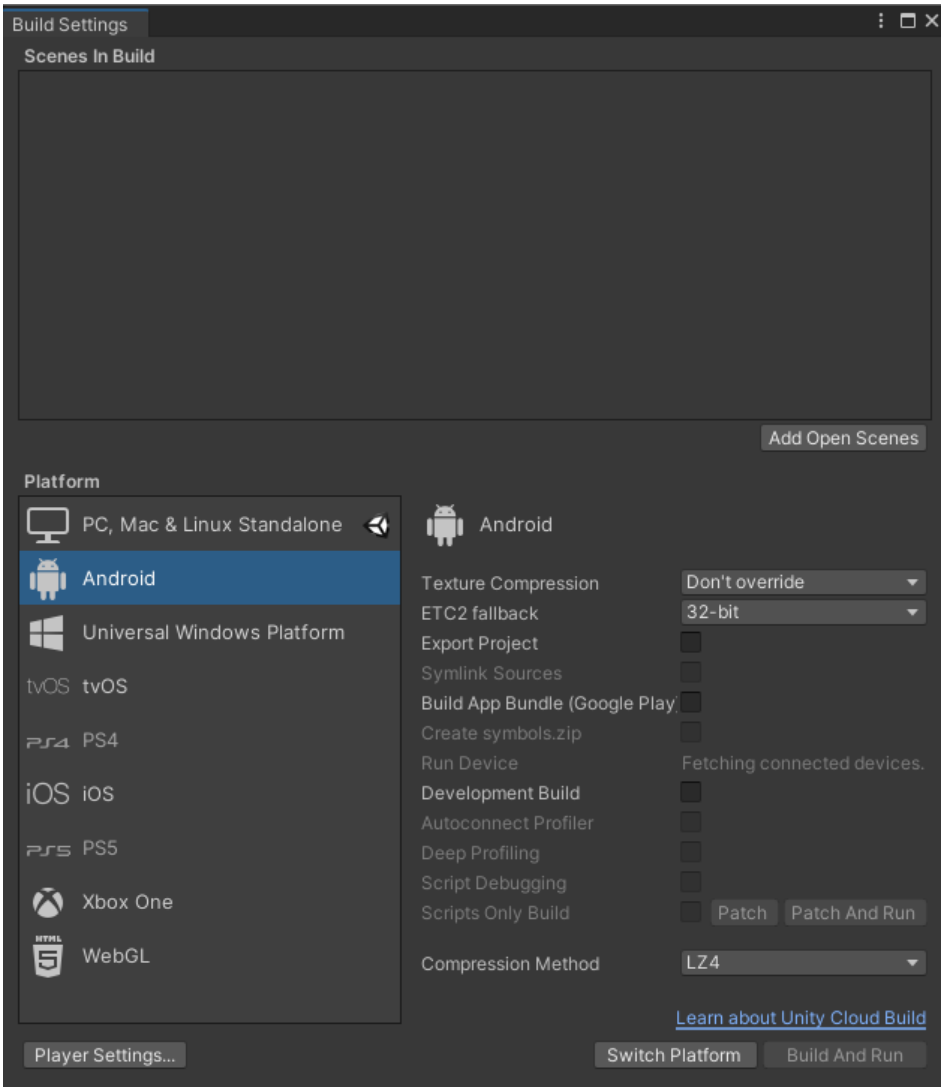
Slika 1 – Prikaz opcija kreiranja Unity projekta

Pored odabira odgovarajuće vrste projekta, takođe unosimo i ime projekta kao i lokaciju na kojoj će se projekat skladištiti.

Nakon uspešnog kreiranja i otvaranja projekta, sledi podešavanje odgovarajuće platforme. Unity je na početku podrazumevano podešen na Windows platformu.

Unutar podešavanja prikazane su sve dostpune instalirane platforme. Android platforma je automatski instalirana zajedno sa Unity instalacijom, pa sve što je potrebno da uradimo jeste da kliknemo na ponuđenu platformu i zatim na dugme switch platform.

[4] Software development kit
[5] Application programming interface



Slika 2 – Prikaz dostupnih platformi

Nakon odabira platforme, projekat je spreman za rad na njemu. Sva dalja podešavanja vrše se tek na kraju razvoja cele igrice, kada je ona spremna za izradu i dalje distribuiranje.

2. Korišćeni programi

Pored glavnog razvojnog okruženja, korisnici u većini slučajeva koriste i neke dodatne programe, koji omugućavaju rad sa elementima igrice, za koje Unity nije predviđen.

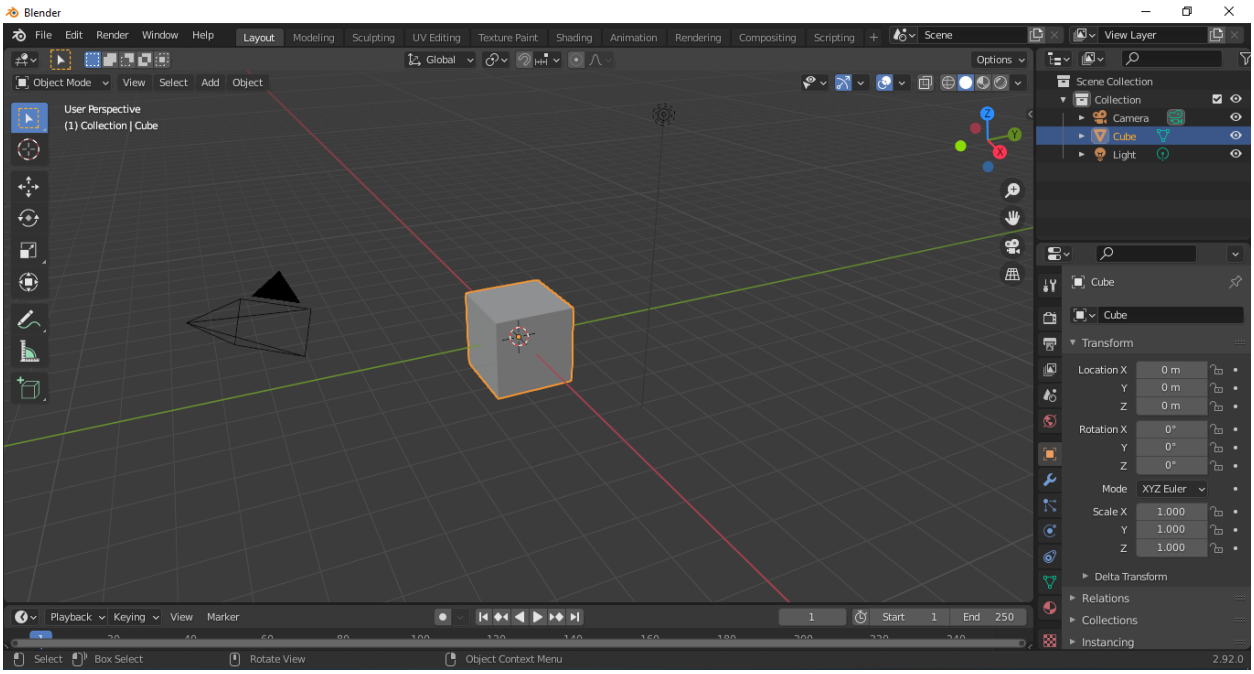
2.1. Blender

Jednu od sastavnih delova svake 3D igrice čine 3D modeli. Postoje razne tehnologije koje omugućavaju kreiranje modela. Za potrebe ove igrice korišćen je Blender.

Blender je besplatan grafički softver otvorenog koda, koji se koristi za kreiranje 3D modela, izradu animacija, vizuelnih efekata i tekstura. Svi korišćeni modeli, kao i teksture ove igrice urađeni su u ovom programu.

Blender je počeo da se razvija 1994. godine, a 1998. godine je prvi put pušten u javnost. Danas je najnovija verzija Blendera 2.93. Pre svega ovaj program omogućava korisnicima da besplatno koriste sve njegove alatke, kao i da bez ikakvih ograničenja distribuiraju sve proizvode napravljene unutar njega. Pored dostupnih alata, korisnici imaju mogućnost da naprave i svoje specijalizovane alate, pomoću pisanja skripti, a može se naći i veliki broj već napravljenih skripti koje dodatno olakšavaju rad u ovom programu.

Ekvivalenti Blenderu su Autodesk 3ds Max i Maya, ali Blender predstavlja jedinu besplatnu opciju koja ne generiše dodatne troškove.

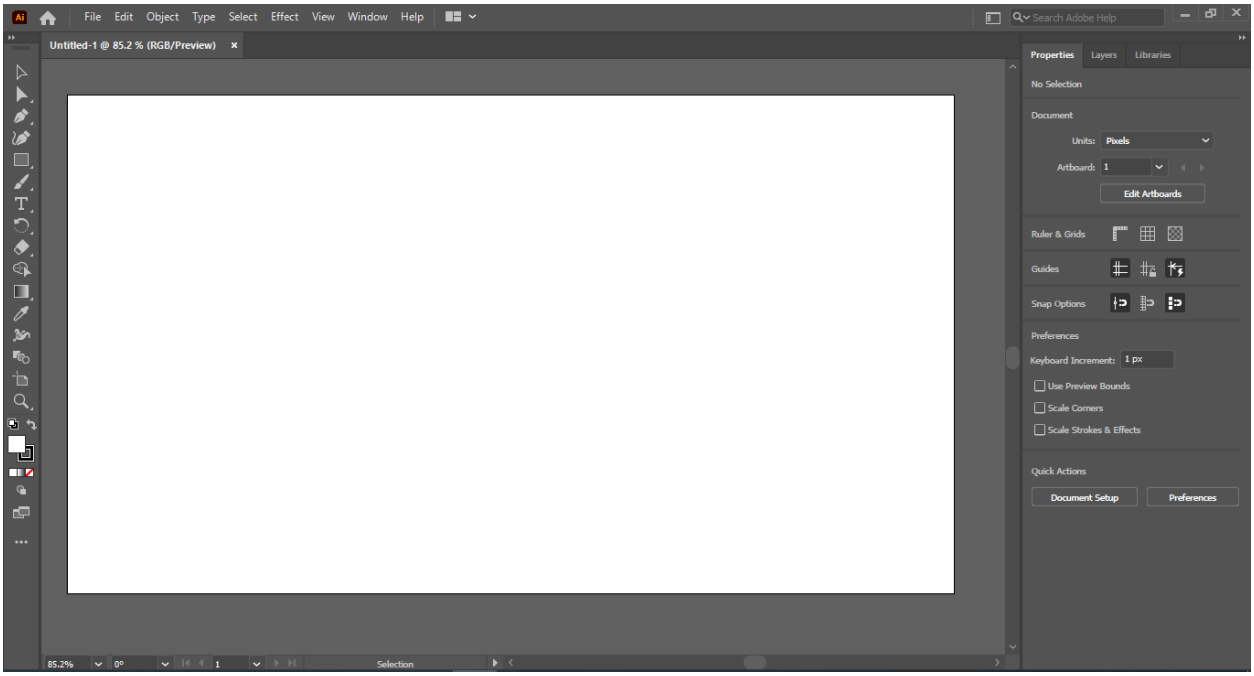


Slika 3 – Blender razvojno okruženje

2.2. Adobe Illustrator

Adobe Illustrator je vektorski grafički editor, koji je proizvod kompanije Adobe. Iako on ne predstavlja besplatnu opciju, smatra se jednim od najboljih programa za razvoj grafike. Svaka video igrica ima korisnički interfejs koji može biti kompleksan, i za potrebe ove igrice korišćen je baš Adobe Illustrator.

Pored korisničkog interfejsa, ovaj program koristi se za izradu svih vrsta grafičkih elemenata, od pozadina i ikonica, pa do početnog dizajna igrice. Različiti alati ovog programa omogućavaju korisniku da dobije željene rezultate, koji se zatim šalju u Photoshop na dalje editovanje.



Slika 4 – Izgled Adobe Illustrator programa

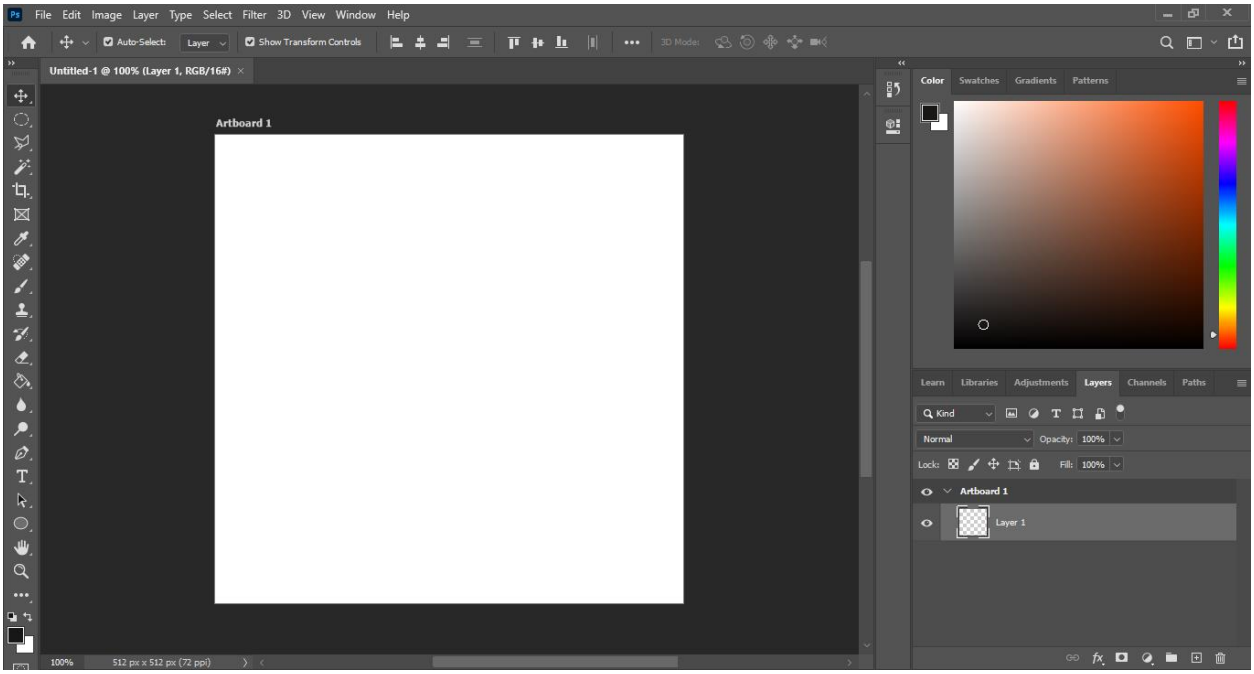
2.3. Adobe Photoshop

Adobe Photoshop predstavlja još jedan proizvod kompanije Adobe. Ovaj program se koristi za razvoj rasterske grafike i vodi se kao jedan od najpopularnijih softvera za editovanje koji ima primenu u različitim apsektima razvoja jedne mobilne igrice.

Nakon kreiranja vektorske grafike u Adobe Illustratoru, ona se učitava u Photoshop kako bi bila pripremljena za korišćenje u Unity-ju. Naravno photoshop se može direktno koristiti za razvoj grafike koja će se dalje koristiti, bez prethodnog kreiranja u drugim programima. Postoji više opcija eksportovanja iz Photoshopa, a za potrebe Unity-ja uglavnom se koriste .jpg i .png fajlovi, a postoji i opcija direktnog importovanja Photoshop fajla u Unity.

Potrebno je voditi računa o veličinama kreiranih fajlova, zato što to može nepotrebno povećati veličinu same igrice i smanjiti performanse. Preporučena veličina za elemente korisničkog interfejsa je 512x512 piksela, a za ikonice je 1024x1024 piksela.

U Unity-ju se mogu dodatno ograničiti veličine svih importovanih grafičkih elemenata.



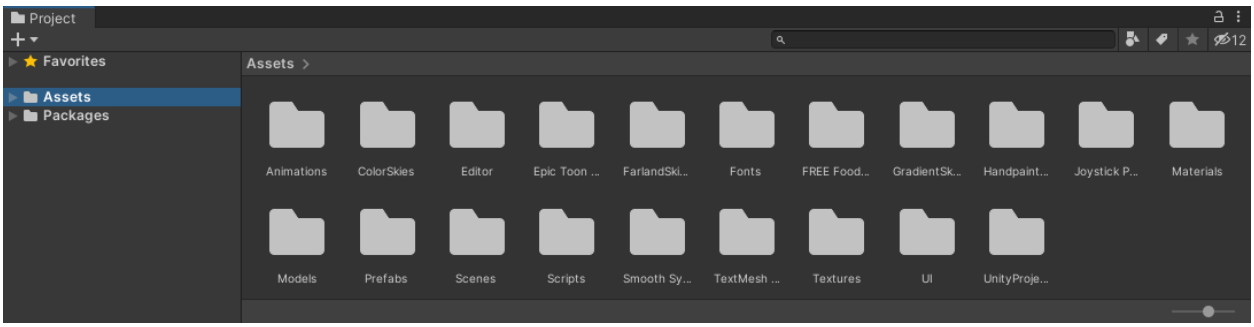
Slika 5 – Izgled Adobe Photoshop programa

3. Struktura igrice u Unity-ju

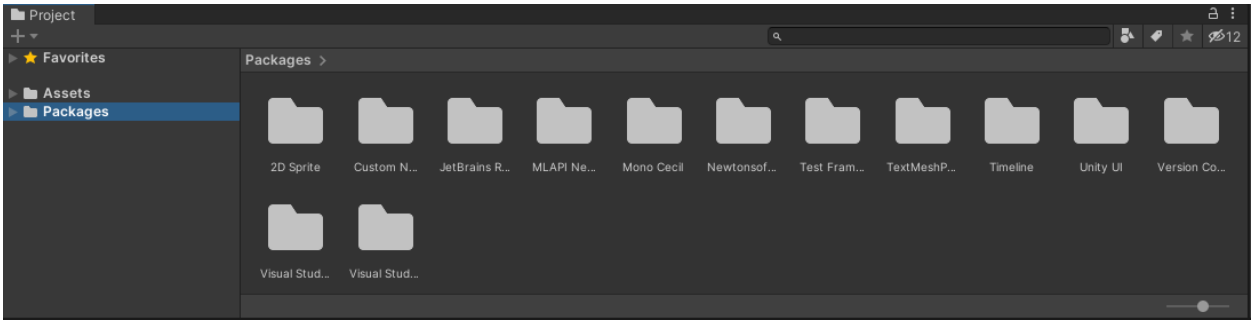
U ovom poglavlju prikazana je struktura kreirane igrice unutar Unity razvojnog okruženja, sa opisom različitih delova i prozora ovog programa.

3.1. Project

U project prozoru imamo uvid u dva glavna foldera naseg projekta. To su package folder, koji sadrži sve instalirane pakete projekta, i assets folder u kome se nalazi sve ono što smo mi importovali u Unity, i što će se dalje koristiti za razvoj. Ovde se mogu kreirati različiti folderi kako bi se razdvojili aseti[6], i kako bi se napravila urednija struktura projekta.



Slika 6 – Korišćeni aseti projekta

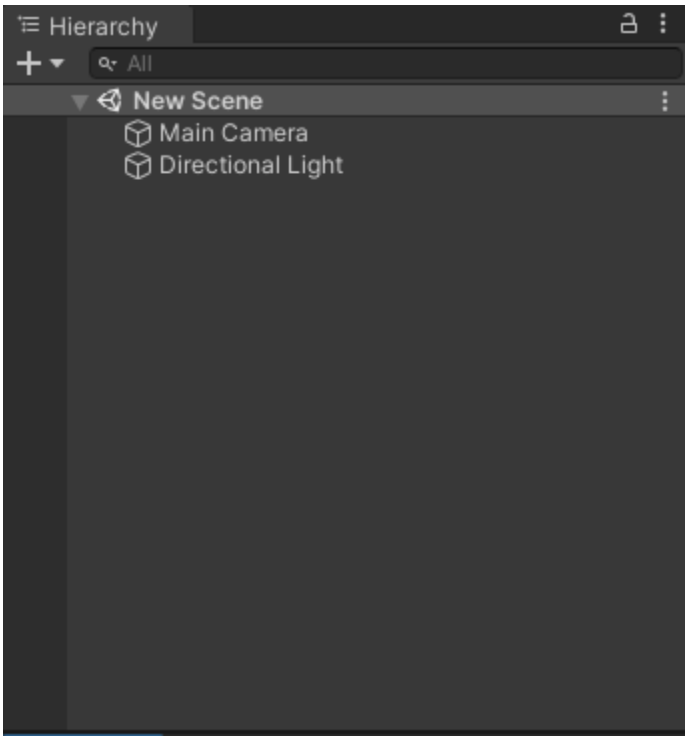


Slika 7 – Instalirani paketi projekta

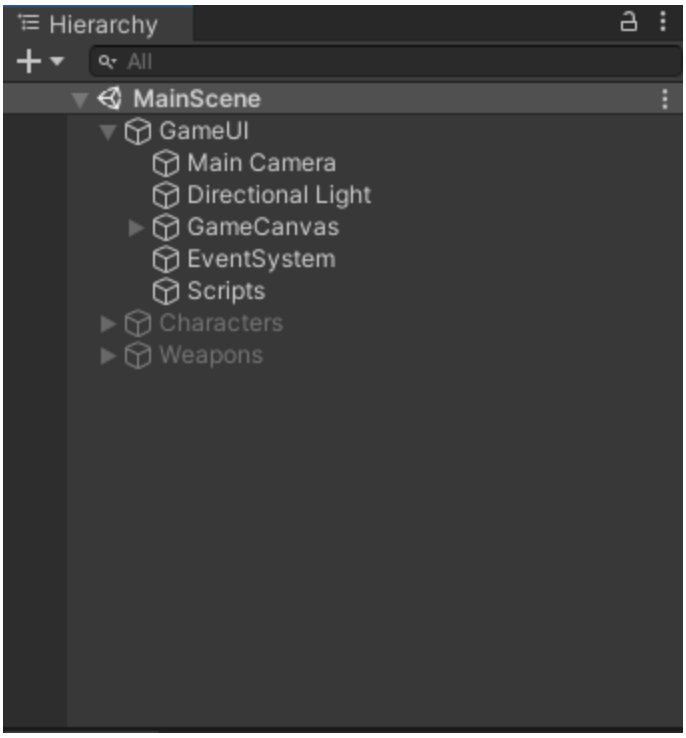
[6] Asset

3.2. Hierarchy

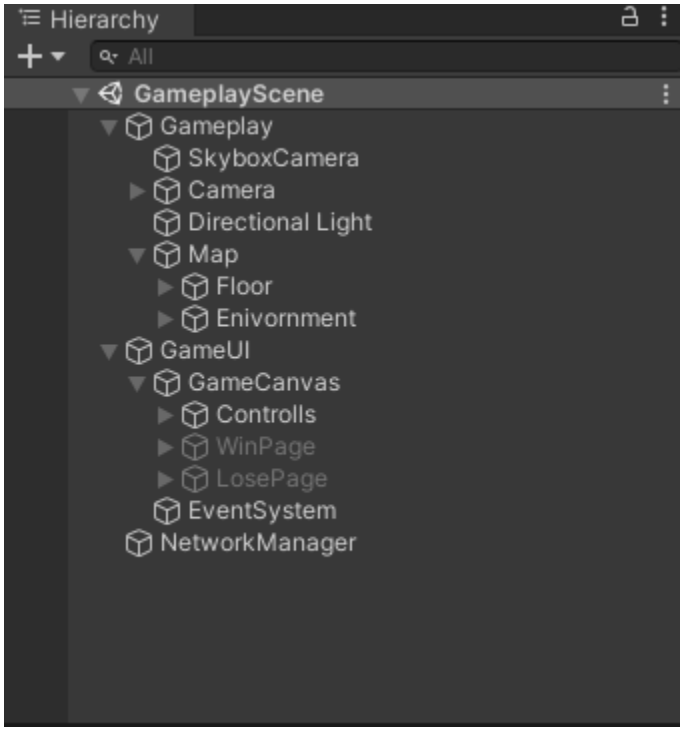
Hijerarhija u Unity-ju predstavlja mesto gde se nalaze svi elementi igrice. Tu kreiramo osnovnu strukturu i veze između naših objekata. Svaka različita scena u projektu ima svoju odvojenu hijerarhiju. Ovde možemo direktno pristupiti svakom objektu, kao i svakoj njegovoj komponenti. Na korisniku je da odluči kakva će struktura hijerarhije biti, ali je dobra praksa da se svaka celina odvoji *"praznim"* objektom, koji će imati indikaciju šta se u njemu nalazi. Svaka nova scena u hijerarhiji na početku sadrži dva osnovna objekta, glavnu kameru i svetlo.



Slika 8 – Početna hijerarhija scene



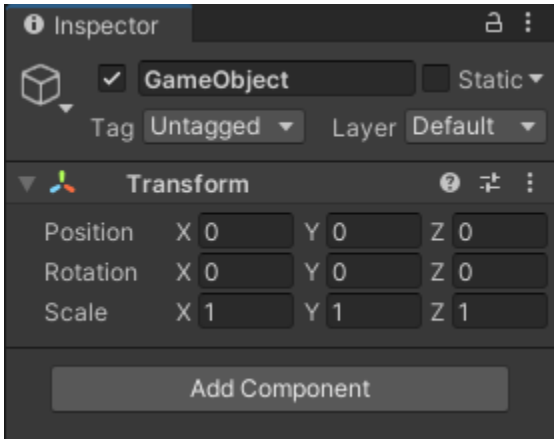
Slika 9 – Hijerarhija Main Menu scene igrice



Slika 10 – Hijerarhija Gameplay scene igrice

3.3. Inspector

Preko inspector prozora vrši se modifikacija svih komponenti jednog objekta. To podrazumeva dodavanje i uklanjanje komponenti, editovanje elemenata svake komponente, kao i dodavanje skripti. Osnovna komponenta svakog kreiranog objekta je Transform, i ona predstavlja jedinu obaveznu komponentu koja se ne može obirsati i bez koje objekat ne može postojati. Ona služi kao indikator pozicije, rotacije i velicine objekta, i predstavljena je x, y i z koordinatama.



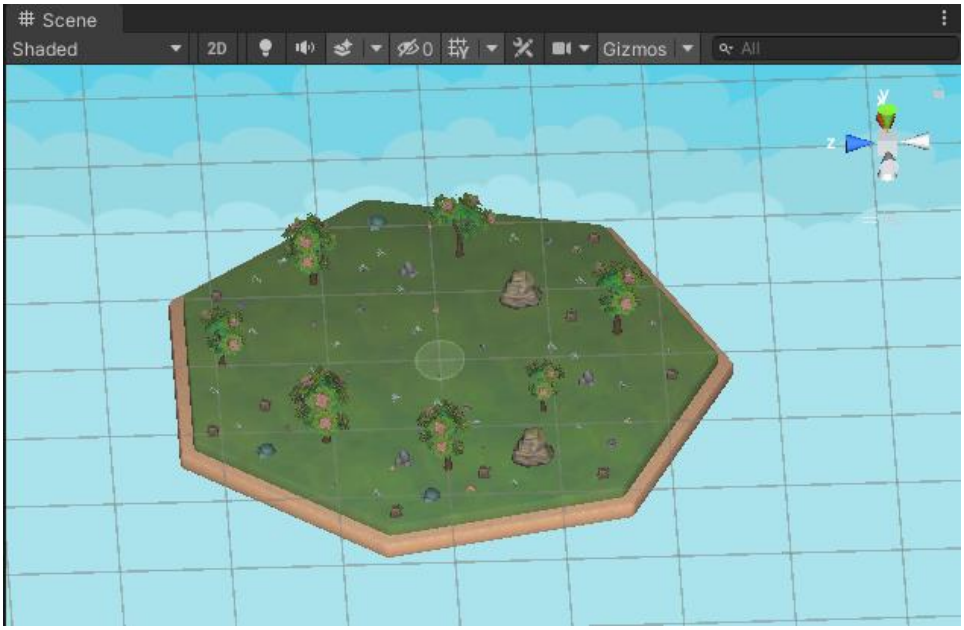
Slika 11 – Prikaz osnovne komponente objekta u inspector prozoru

3.4. Scene

Na sceni se nalazi vizuelna reprezentacija svih objekata iz hijerarhije. Ovde korisnik može direktno menjati i pozicionirati objekte. Svaki objekat dodat na scenu se automatski dodaje i u hijerarhiju i obrnuto. Neke od osnovnih dostupnih alatki za manipulisanje objekata u sceni su:

- Hand tool - služi za navigaciju korisnika
- Move tool - služi za pozicioniranje objekta
- Rotate tool - služi za rotaciju objekta
- Scale tool - služi za menjanje veličine objekta

Na sceni se jedino neće prikazati *"prazni"* objekti, ali se klikom na njih u hijerarhiji omogućava i njihovo modifikovanje na sceni.



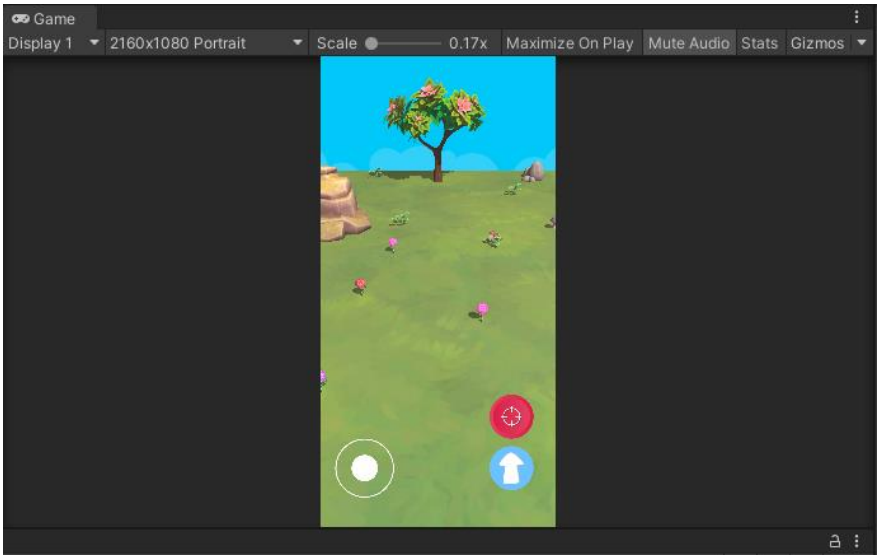
Slika 11 – Prikaz glavne scene igrice

3.5. Game

Game prozor služi da prikaže sve elemente scene, viđene kroz kameru koju korisnik sam podešava, i prikazuje kako će igrica izgledati na samom uređaju. Ovde se može pokrenuti igrica i izvršiti testiranje kao da je u pitanju mobilni telefon. Sve promene na sceni viđene od strane kamere se direktno ažuriraju i u game prozoru. Različiti telefoni imaju različite dimenzije, i uz izabranu Android platformu korisniku su dostupne sve postojeće dimenzije preko kojih korisnik može da vidi kako će njegova igrica izgledati na različitim uređajima. Postoji i mogućnost manuelnog dodavanja određenih dimenzija.



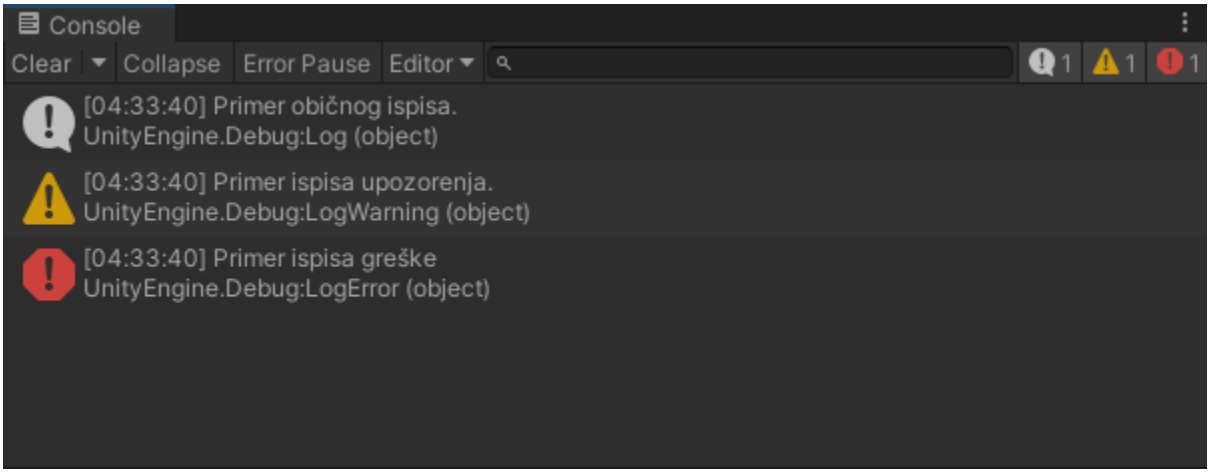
Slika 13 – Prikaz game prozora glavnog menija igrice



Slika 14 – Prikaz game prozora gameplay-a igrice

3.6. Console

Konzola služi za ispis grešaka, upozorenja i debug komandi iz napisanih skripti. Ovo predstavlja jako bitan korak razvoja svake igrice zato što nam omogućava da pronađemo eventualne greške u kodu, bile one sintaksne ili semantičke. Takođe se može vršiti testiranje funkcionalnosti preko konzole ispisom odgovarajućih poruka koje korisnik sam osmišlja u kodu. Korisnik može odabrati da sakrije svaki tip ispisa, a ukoliko greška nije uklonjena iz konzole, ne može se pokrenuti aplikacija.



Slika 15 – Prikaz ispisa u konzoli

4. Multiplayer u Unity-ju

Za razvoj multiplayer igrica mogu se koristiti različiti tehnologije. Najzastupljenije tehnologije koje se koriste u Unity-ju su:

- Photon
- Mirror
- MLAPI

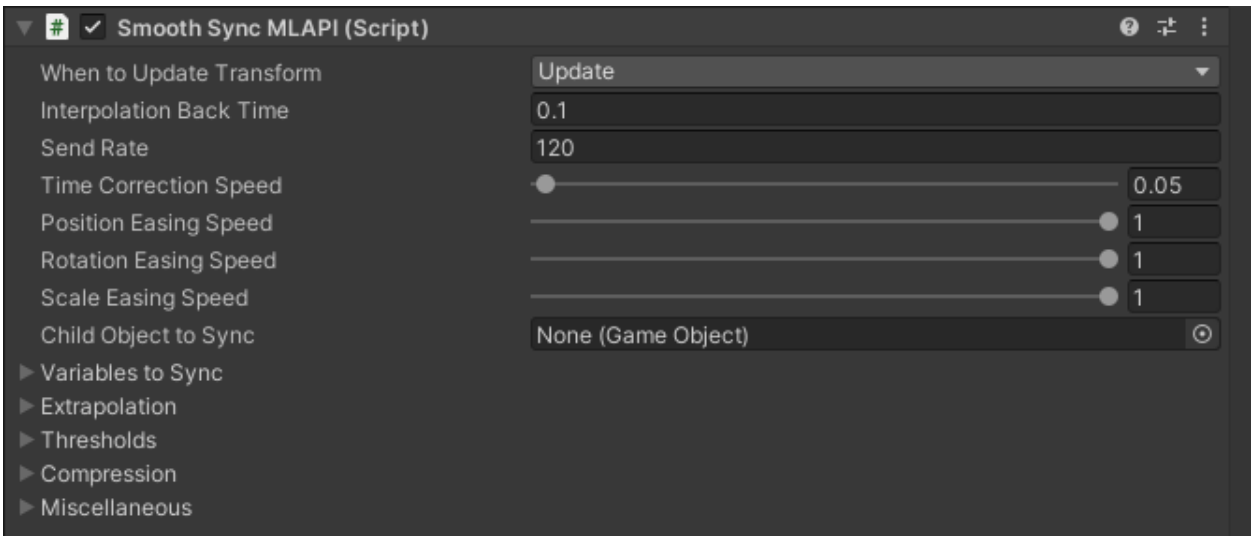
Dok su prve dve opcije pogodne za online igrice, poslednja opcija se uglavnom koristi za razvoj lokalnih multiplayer igrica, ali se i ona naravno može primeniti za razvoj online igrica.

MLAPI je najnovija oficijalna Unity tehnologija, koja je i dalje u fazi razvoja. Iz tog razloga ne postoji temeljna dokumentacija o njenom korišćenju, ali se iz osnovnih definicija može zaključiti primena ove tehnologije.

Korisnik ima više opcija za instaliranje MLAPI tehnologije, od kojih je najbolja dodavanje samog paketa preko linka sa github-a. Nakon uspešnog instaliranja paketa, može se krenuti sa razvojem video igrice.

Osnovni elementi MLAPI tehnologije su ServerRpc i ClientRpc metode. Preko njih se programu daje do znanja da li se određena metoda izvršava na serverskoj ili klijenstkoj strani.

Kako bi sinhronizovanje podataka preko internet mreže bilo obavljeno, mora se obratiti posebna paznja na pisanje skripti koje se izvršavaju i na serverskoj i na klijentskoj strani. Postoji veliki broj alata na oficijalnoj Unity prodavnici koji služe da korisniku olakšaju ovaj posao. Primer jednog ovakvog alata je Smooth Sync koji bez pisanja dodatnog koda, na jednostavan način vrši sinhronizaciju varijabli preko interneta.



Slika 16 – Prikaz Smooth Sync komponente

5. Korisnički interfejs

U ovom poglavlju prikazan je izgled korisničkog interfejsa kao i opis njegovih funkcionalnosti.

5.1. Početna stranica igrice

Na početnoj stranici nalazi se ime igrice kao i nekoliko dugmića, od kojih je glavno play dugme čiji klik pokreće igricu. Pored njega dva sporedna dugmeta služe za ulaz u podešavanja i izlaz iz igrice.



Slika 17 – Prikaz početne stranice

U okviru početne stranice nalazi se i stranica sa podešavanjima, gde korisnici mogu da isključe i uključe zvučne efekte i muziku, kao i da izađu iz podešavanja.



Slika 18 – Prikaz podešavanja

5.2. Unos imena

Korisnici klikom na input polje upisuju svoje ime koje će biti prikazano drugim igračima u toku igranja. Ime je moguće promeniti pri svakoj novoj započetoj igrici. Klikom na dugme confirm korisnik potvrđuje unete podatke.



Slika 19 – Prikaz unosa imena

5.3. Odabir hostovanja ili priključivanja serveru

Na ovoj stranici korisnik dobija opciju da hostuje server, ili da se priključi već postojećem serveru.



Slika 20 – Prikaz opcije hostovanja ili priključivanja serveru

5.4. Unos IP adrese

Korisnik se može priključiti serveru samo preko privatne IP adrese tog servera. Na ovoj stranici vrši se unos IP adrese a klikom na dugme confirm se potvrđuje unos.



Slika 21 – Prikaz stranice za unos IP adrese

5.5. Odabir karaktera

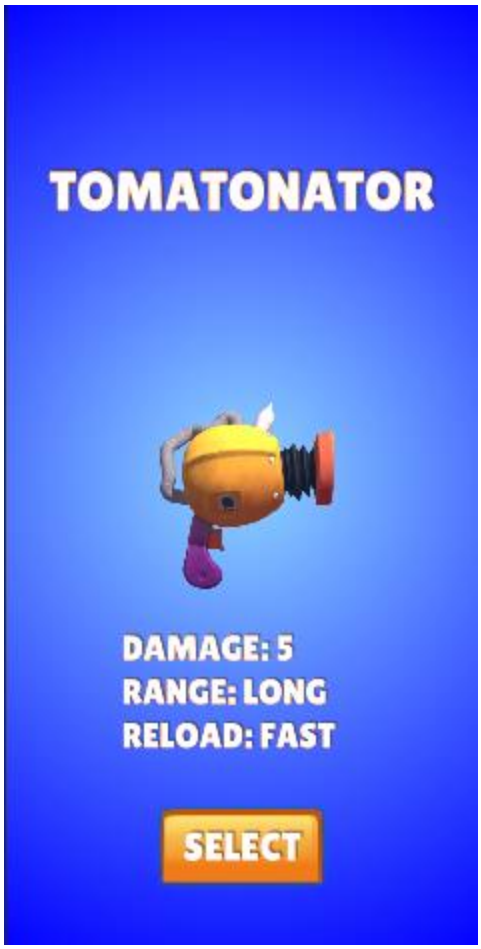
U igrici su dostpuna četiri karatkera između kojih korisnik može da bira. Svaki karakter razlikuje se po izgledu i imenu. Klikom na dugme select korisnik vrši odabir karaktera koji se nalazio na stranici.



Slika 22 – Prikaz odabira karaktera

5.6. Odabir oružja

Isto kao što se vršio odabir karaktera, vrši se i odabir oružja. Postoje tri dostpuna oružja sa različitim karakteristikama koje korisnik može da vidi na ekranu i na osnovu toga da se odluči za ono koje mu najviše odgovara. Nakon što se izvrši klik na dugme select, završava se interakcija korisnika sa korisničkim interfejsom i započinje se igrica.



Slika 23 – Prikaz odabira oružja

5.7. Kontrole

Korisnik na raspolaganju ima tri kontrole. Sa leve strane nalazi se joystick pomoću kojeg se igrač pomera, dok se sa desne strane nalaze dugmići za skakanje i pucanje.



Slika 24 – Prikaz kontrola

5.8. Pobeda i poraz

Nakon što jedan od igrača pobedi, prikazuju se i odgovarajuće stranice. Pobednik će videti stranicu koja ga izveštava da je pobedio, dok će igrač koji je izgubio dobiti stranicu sa obaveštenjem o porazu. Sa obe stranice igrač ima mogućnost da se klikom na dugme kućice vrati na početnu stranicu.



Slika 25 – Prikaz stranice pobede



Slika 26 – Prikaz stranice poraza

6. Funkcionalnosti igrice

U ovom poglavlju biće opisane osnovne funkcionalnosti igrice, a takođe će biti prikazani i delovi koda.

6.1. Main menu

Iz početnog menija kreće sekvenca aktivnosti koje korisnik treba da ispuni kako bi nastavio sa uspešnim početkom igrice.

Potrebno je izvršiti unos traženih podataka i klikom na dugmiće prelaziti iz jedne stranice na drugu.

```
using UnityEngine;

public class LoadNextPageHandler : MonoBehaviour
{
    [SerializeField]
    private GameObject m_PageToDisable;
    [SerializeField]
    private GameObject m_PageToEnable;

    public void LoadNextPage()
    {
        m_PageToDisable.SetActive(false);
        m_PageToEnable.SetActive(true);
    }
}
```

Listing 1 - Prikaz skripte za navigaciju između stranica

```
using UnityEngine;
using TMPro;

public class SetNameHandler : MonoBehaviour
{
    [SerializeField]
    private TMP_InputField m_NameInput;

    public void SetName()
    {
        PlayerPrefs.SetString("Name", m_NameInput.text);
    }
}
```

Listing 2 - Prikaz skripte za evidentiranje imena korisnika

```
using UnityEngine;
using TMPro;

public class SetIPAddress : MonoBehaviour
{
    [SerializeField]
    private TMP_InputField m_IPAddress;

    public void SetIP()
    {
        PlayerPrefs.SetString("IP", m_IPAddress.text);
    }
}
```

Listing 3 - Prikaz skripte za evidentiranje IP adrese

```
using UnityEngine;
using UnityEngine.SceneManagement;

public class StartGameHandler : MonoBehaviour
{
    public void StartGame()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
    }
}
```

Listing 4 - Prikaz skripte za početak igrice

6.2. Osnovni gameplay

Osnovni gameplay igrice zasniva se na borbi sa hranom. Ovde je to predstavljeno u vidu oružja koja koriste različitu hranu kao municiju i cilj igrice je da jedan igrač sto više puta uspešno pogodi drugog igrača. Svaki igrač ima početni health bar koji drugi igrač može da vidi. Igrači na mapi imaju različite prepreke i zaklone, što igricu čini zanimljivijom.

Pritiskom i prevlačenjem joystick-a korisnik može da kontroliše svog karatkera, a prevlačenjem prstom po ekranu korisnik može da kontroliše poziciju kamere. Klikom na dugmiće za skok i pucanje, igrač izvršava odgovarajuće akcije.

```
private void MovePlayer(Vector3 i_Direction)
{
        if (i_Direction.magnitude > 0.1f)
        {
            float m_TargetAngle = Mathf.Atan2(i_Direction.x, i_Direction.z) *
            Mathf.Rad2Deg + Camera.main.transform.parent.transform.eulerAngles.y;

            float m_Angle = Mathf.SmoothDampAngle(transform.eulerAngles.y, m_TargetAngle,
            ref m_TurnSmoothVelocity, m_TurnSmoothTime);

            Vector3 m_MoveDir = Quaternion.Euler(0f, m_Angle, 0f) * Vector3.forward;

            transform.rotation = Quaternion.Euler(0, m_Angle, 0);
            m_Controller.Move(m_MoveDir.normalized * m_Speed * Time.deltaTime);

            m_Anim.SetBool("isRunning", true);
        }
 }
```

Listing 5 - Prikaz dela koda za pomeranje karaktera

```
private void PlayerJump(bool i_StandJump)
{
       if (m_IsGrounded)
        {
            if (!i_StandJump)
            {
                m_Anim.SetTrigger("jump");
            }

            else
                m_Anim.SetTrigger("standJump");


            m_Velocity.y = m_JumpHeight;
        }

}
```

Listing 6 - Prikaz dela koda koji kontroliše skakanje karaktera

```
IEnumerator ShootSequence(string i_Name, Vector3 i_Pos)
    {
        m_CanShoot = false;
        t = 1;
        m_LerpValue = 0;
        m_Anim.SetTrigger("shoot");
        m_Anim.SetLayerWeight(1, 1);

        yield return new WaitForSeconds(0.35f);

        Rigidbody m_Bullet = Instantiate(m_BulletPrefab);
        m_Bullet.transform.position = m_SpawnTrans.position;
        m_Bullet.isKinematic = false;
        m_Bullet.AddForce(i_Pos * m_Force + Vector3.up, ForceMode.Impulse);
        m_Bullet.name = i_Name;

        yield return new WaitForSeconds(0.5f);

        while (t >= 0)
        {
            t -= Time.deltaTime;
            m_LerpValue += Time.deltaTime;
            m_Anim.SetLayerWeight(1, Mathf.Lerp(m_Anim.GetLayerWeight(1),
            0, m_LerpValue));
            yield return null;
        }
        StartCoroutine(EnableShooting());
    }

    IEnumerator EnableShooting()
    {
        yield return new WaitForSeconds(m_WaitTime);
        m_CanShoot = true;
    }
```

Listing 7 - Prikaz dela koda koji kontroliše pucanje

```
    private void ProgressBar()
    {
        float m_HealthLeft = m_Health - m_CurrentHealth;
        float fillAmmount = 1 - (m_HealthLeft / m_Health);

        m_TempFIllAmmount -= 0.5f * Time.deltaTime;

        if (m_TempFIllAmmount < fillAmmount)
            m_TempFIllAmmount = fillAmmount;

        m_HealthBar.fillAmount = m_TempFIllAmmount;
    }
```

Listing 8 - Prikaz dela koda koji kontroliše health bar

6.3. Upravljanje hostovanjem i priključivanjem serveru

Svaka multiplayer igrica, bilo da je u pitanju lokalna ili online, mora da ima server i klijente. Kada je reč o lokalnoj igrici, pomoću IP adrese klijenti se mogu priključiti na server. Kako bi se videla IP adresa na računaru, potrebno je da se u konzoli iskuca naredba ipconfig, koja daje listing potrebnih informacija, dok se za mobilne telefone IP adresa može naći u podešavanjima.

Na početku igrice korisnik bira opciju da li će da hostuje server ili će da se priključi već postojećoj igrici. Važno je napomenuti da server istovremeno ima i ulogu klijenta.

```
using UnityEngine;
using MLAPI;
using MLAPI.Transports.UNET;

public class GameLoadHandler : MonoBehaviour
{
    private int m_OptionIndex;

    private void Start()
    {
        m_OptionIndex = PlayerPrefs.GetInt("HostJoin");

        SetGameOption();
    }

    private void SetGameOption()
    {
        switch (m_OptionIndex)
        {
            case 0:
                NetworkManager.Singleton.StartHost(new Vector3(-2f, 1f, 0f));
                break;
            case 1:
                UNetTransport mTransport = MLAPI.NetworkManager.Singleton.
                GetComponent<UNetTransport>();
                mTransport.ConnectAddress = PlayerPrefs.GetString("IP");
                NetworkManager.Singleton.StartClient();
                break;
        }
    }
}
```

Listing 9 - Prikaz dela koda koji pokreće server i dodaje klijente

7. Testiranje

Testiranje je završni, a takođe i jedan od najbitnijih koraka razvoja jedne video igrice. Bez testiranja korisnik ne može biti siguran da su sve funkcionalnosti igrice ispravne i rizikuje da snosi velike posledice ako dođe do otkrivanja anomalija nakon što je igrica već distribuirana.

Testiranje unutar Unity-ja ne nudi stopostotnu sigurnost da će sve proći kako treba i na platformi za koju je igrica pravljenja. Iz tog razloga vrši se direktno testiranje na odabranoj platformi, ili korišćenje posrednih programa ukoliko korisnik nije u mogućnosti da nabavi neophodni uređaj.

Kada je u pitanju Android platforma, testiranje direktno na telefonu vrši se instaliranjem APK[7] fajla, i pokretanjem igrice.

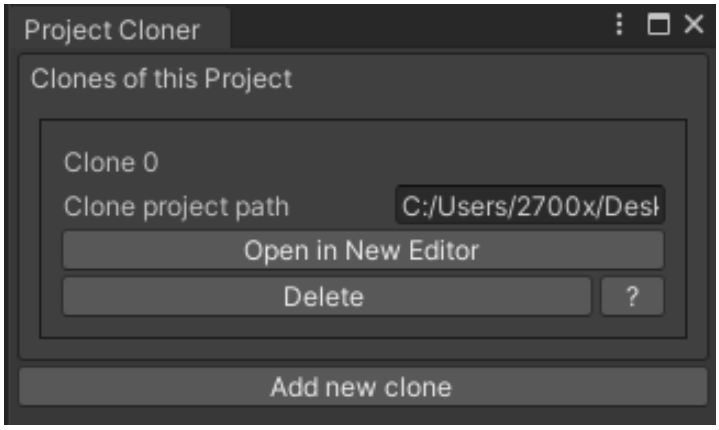
Korisniku su takođe dostupni i Android emulatori, preko kojih se direktno na računaru može izvršiti testiranje na različitim uređajima, što pruža dodatnu sigurnost. Neki od najpopularnijih Android emulatora su:

- BlueStacks
- LDPlayer
- Android Studio
- GameLoop
- Xamarin

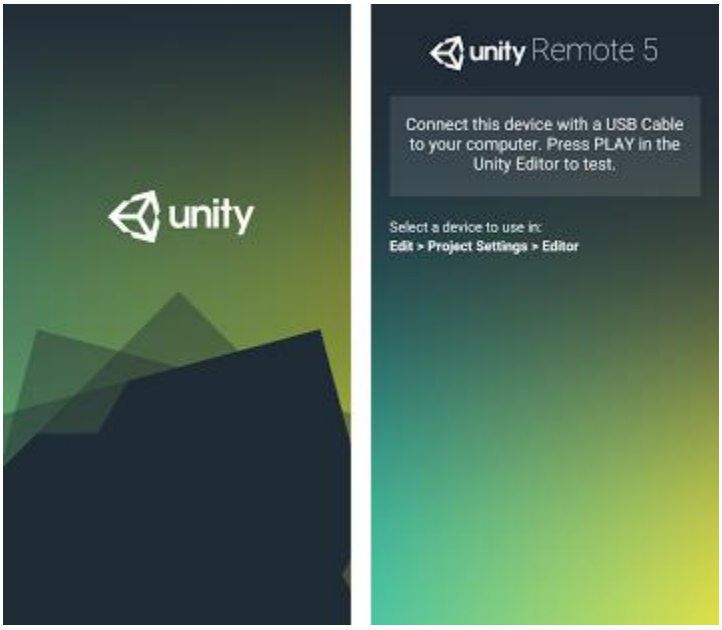
Pošto uspešno testiranje treba izvršiti u fazama i testirati svaku funkcionalnost pojedinačno, kako korisnici ne bi trošili previše vremena na instaliranje igrice ponovo, Unity je razvio posebnu aplikaciju za mobilne telefone. Priključivanjem preko USB[8] kabla korisnik direktno može na svom uređaju da vidi sve što se nalazi unutar game prozora u samom Unity-ju.

Kada je reč o testiranju multiplayer igrica, bilo bi potrebno duplo više vremena da se izvrši instalacija na više uređaja, i zato je razvijena alatka koji omogućava kloniranje i sinhronizaciju kloniranog projekta sa originalom. Na ovaj način korisnik može sa jednog računara da testira i server i klijente. Naravno nakon uspešnog testiranja na ovaj način, potrebno je i izvršiti testiranje izvan Unity-ja.

[7] Android application package
[8] Universal serial bus



Slika 27 – Prikaz alatke za kloniranje projekta



Slika 28 – Prikaz Unity Remote aplikacije

8. Poboljšanje igrice u budućnosti

Svaka potencijalno uspešna igrica u toku svog životnog veka treba da ima nekoliko unapređenih verzija. Bilo da se radi o ispravljanju postojećih grešaka, dodavanju novih funkcionalnosti ili o menjanju samog izgleda igrice važno je da se korisnik oseća kao da se njegovo korisničko iskustvo unapređuje.

Jedan od načina poboljšanja ove igrice bio bi dodavanje novih karaktera, oružja i mapa. Kako bi se to omogućilo potrebno je uložiti dosta vremena i truda u sam dizajn i kreiranje modela. Takođe dodavanje opisa i karakteristika svakog karaktera učinilo bi samo igranje igrice zanimljivijim.

Sa aspekta multiplayer-a, sledeći korak bio bi migracija na online režim koji bi prvo obuhvatao povezivanje sa korisnicima iz bilo kog dela sveta preko IP adrese, a zatim i nasumično povezivanje korisnika i korišćenje odvojenih servera.

Još jedan mali detalj koji bi znatno poboljšao korisničko iskustvo jeste dodavanje većeg broja vizuelnih i zvučnih efekata.

9. Zaključna razmatranja

U ovom radu prikazana je realizacija lokalne multiplayer igrice koja je zasnovana na ideji sedmogodišnjeg dečaka. Igrica je razvijena za Android platformu i mobilne telefone što je čini pristupačnom i mlađoj populaciji.

Sa tim u vidu, struktura ove igrice napravljena je tako da bude razumljiva i dostupna svim korisnicima.

Igrica osim aktivne internet konekcije, ne zahteva dodatne dozvole. Pošto je igrica napravljena sa ciljem da bude besplatna i ne koristi sistem plaćanja, ne postoji opasnost od dodatnih troškova.

Tehnologije i programi korišćeni u procesu razvoja, uz detaljno razmatranje izabrani su kao najbolje opcije. Svaki program korišćen je za realizaciju posebnih delova igrice, koji kada se povežu u jednu veliku celinu predstavljaju završni proizvod.

Link ka projektu:

https://drive.google.com/file/d/1WOGqwVmIqeSDjBkteeOpIjjjtZ5v59ga/view?usp=sharing

10. Literatura

1. https://docs.unity3d.com/Manual/index.html
2. https://docs.blender.org
3. https://helpx.adobe.com/illustrator/user-guide.html
4. https://helpx.adobe.com/photoshop/user-guide.html
5. https://docs-multiplayer.unity3d.com
6. https://stackoverflow.com
7. https://www.history.com/topics/inventions/history-of-video-games
8. https://techcrunch.com/2015/10/31/the-history-of-gaming-an-evolving-community/