2D akcijska platformska igra

Šumiga, Borna

Undergraduate thesis / Završni rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:254957

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Preddiplomski studij računarstva

2D akcijska platformska igra

Završni rad

Borna Šumiga

Osijek, 2018.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 07.09.2018.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Borna Šumiga
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3706, 26.09.2017.
OIB studenta:	71026166449
Mentor:	Doc.dr.sc. Časlav Livada
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	2D akcijska platformska igra
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 2 razina
Datum prijedloga ocjene mentora:	07.09.2018.
Datum potvrde ocjene Odbora:	13.09.2018.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 13.09.2018.

Ime i prezime studenta:	Borna Šumiga
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3706, 26.09.2017.
Ephorus podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: 2D akcijska platformska igra**

izrađen pod vodstvom mentora Doc.dr.sc. Časlav Livada

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD 1

1.1. Zadatak završnog rada 1

2. PROGRAMSKO OKRUŽENJE ZA RAZVOJ I UPRAVLJANJE IGRAMA UNITY 2

3. RAZVOJ IGRE 4

3.1. Ideja igre 4

3.2. Razvoj likova 6

3.2.1. Glavni lik 6

3.2.2. Protivnici 11

3.3. Razvoj nivoa 15

3.4. Odabir glazbe i zvukova 20

3.5. Odabir vizualnih efekata 22

3.6. Razvoj korisničkog sučelja 24

3.7. Razvoj izbornika 25

4. Zaključak 29

Literatura 30

Sažetak 32

Abstract 32

Životopis 33

1. UVOD

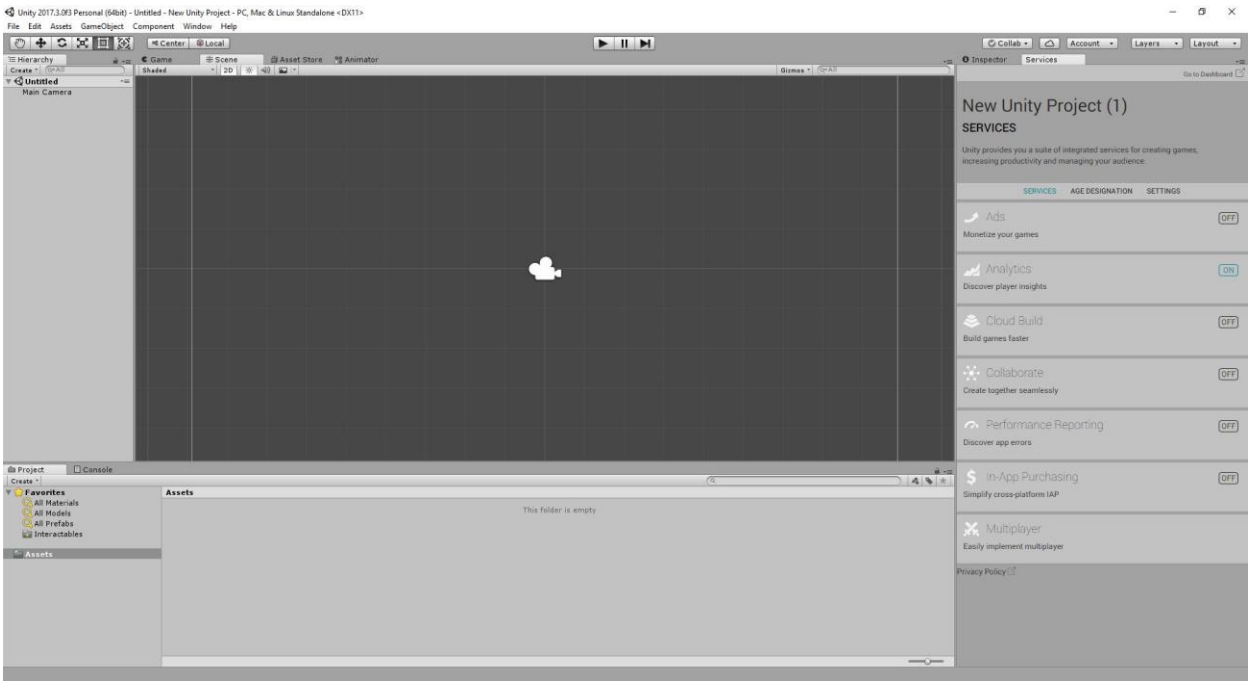
Ovaj završni rad je napravljen na osnovi igre pod nazivom Chernobyl Effect, koja je napravljena kao praktični dio završnog rada. Priča igre se odvija nakon strašnog događanja nuklearne nesreće u gradu Černobilu. Neimenovani robot kojeg je vojska pokušala ugasiti prije nesreće se budi i shvaća kako mu je jedina nada opstanka bijeg iz grada. Igra je razvijena za Windows operacijski sustav. Izrađena je u programskom okruženju za razvoj i upravljanje igrama Unity i sav programski kod koji igra sadrži napisan je u programskom jeziku C#. Animacije, slike, glazba su preuzete kao besplatni sadržaj s raznih internetskih stranica. Skripte koje se koriste kako bismo povezali različite dijelove igre u cjelinu su dijelom napisane vlastoručno, a dio preuzete kao rješenja iz raznih izvora. U ovom radu bit će opisano programsko okruženje Unity, razvoj ideje za igru i njezini dijelovi razvoja: likovi, nivo, glazba, efekti, korisničko sučelje i izbornik.

1.1. Zadatak završnog rada

Zadatak ovog završnog rada je bio proći i razumjeti osnove programa Unity te u njemu razviti 2D akcijsku platformsku igru s horizontalnim pomicanjem.

2. PROGRAMSKO OKRUŽENJE ZA RAZVOJ I UPRAVLJANJE IGRAMA UNITY

Programsko okruženje za razvoj i upravljanje igrama Unity je više-platformsko okruženje kojeg je razvila tvrtka Unity Technologies. Unity se koristi za razvoj dvodimenzionalnih i trodimenzionalnih igara te za simulacije koje se mogu pokretati na osobnim računalima, konzolama i mobilnim uređajima. Glavna obilježja su podrška za dvodimenzionalnu i trodimenzionalnu grafiku, povlačenje i ispuštanje i pisanje programskog koda u programskom jeziku C#. Unity pruža veliki spektar mogućnosti pri razvoju. Kao programsko okruženje pruža jednostavan uvoz grafike za objekte, napredno dvodimenzionalno iscrtavanje, sažimanje tekstura, razne postavke igre i simulacije te puno drugih naprednih tehnologija koje olakšavaju razvoj za početnike i dugogodišnje programere. Uz tehnologije za razvoj, Unity pruža razne usluge za analizu, certifikate, umreženo igranje, reklame i drugo. Unity također pruža razvoj na lokalnom računalu ili u oblaku kako bismo mogli pristupiti našem projektu u bilo koje vrijeme, s bilo kojeg računala.[1] Sučelje se sastoji od jedne glavne scene, scene je moguće dodavati i prilikom izvoza igre poslagati scene željenim redoslijedom.



Sl. 2.1. Sučelje Unity programskog okruženja za 2D projekt

Prema slici 2.1. možemo vidjeti osnovne dijelove Unity sučelja. Glavni dio je scena (engl. Scene) u koju stavljamo naše objekte koje kreiramo ili stavljamo gotove koje je razvila neka druga osoba.[2]

[1] Unity programsko okruženje – 5.9.2018. https://en.wikipedia.org/wiki/Unity_(game_engine)
[2] Unity scena – 5.9.2018. https://docs.unity3d.com/Manual/CreatingScenes.html

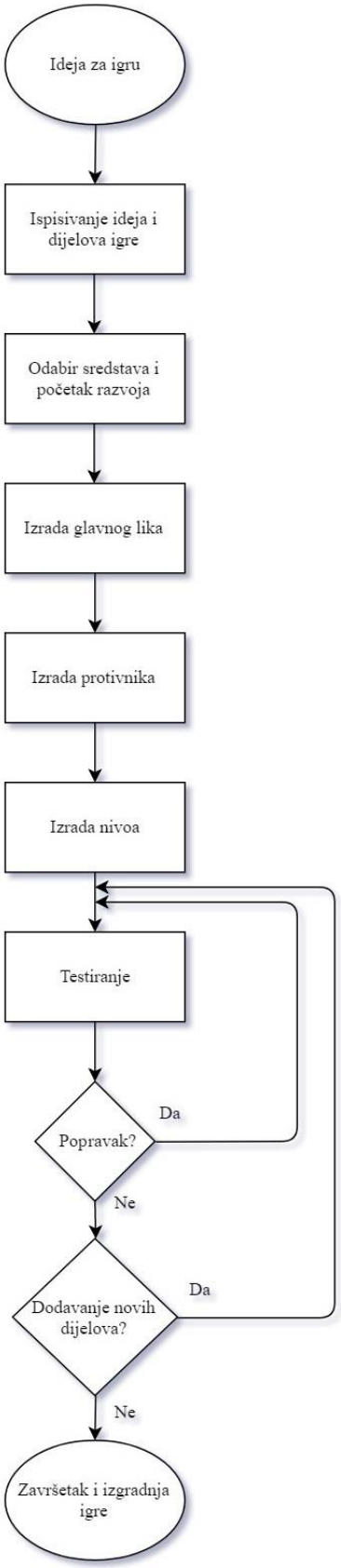
Ostali bitni dijelovi su nadglednik (engl. Inspector) u kojem namještamo ponašanje našeg objekta, njegove postavke i slično.[3] U donjem dijelu sučelja možemo vidjeti projektni prozor (engl. Project Window)[4] u kojem upravljamo našim sredstvima (engl. Assets)[5] i konzolu (engl. Console) u kojoj nam se ispisuju pogreške, tekst, upozorenja i slično.[6] Prednost sučelja Unity-a je njegova mogućnost mijenjanja rasporeda po želji korisnika te dodavanje mnogih drugih prozora za razvoj. Na lijevoj strani sučelja vidimo hijerarhiju u kojoj se nalaze objekti koje stavljamo u našu scenu. Objekte je moguće slagati redoslijedom kako bismo imali veću preglednost i lakše upravljali našim objektima. Na desnoj strani s nadglednikom se nalaze usluge (engl. Services) koje Unity pruža korisniku za lakše i bolje upravljanje projektom. Neke od tih usluga su besplatne, dok je druge potrebno kupiti kroz profesionalnu verziju Unity-a.[7] Unity svojim objektima daje život pomoću skripti koje su najčešće napisane u C# programskom jeziku. Podržavani su bili programski jezici Boo i JavaScript koji se nastoje zamijeniti u novijim verzijama.[8] Osnovne funkcije koje se koriste pri razvoju su Start() koja služi za postavljanje raznih varijabli, vrijednosti i drugog pri pokretanju skripte[9] te funkcija Update() koja se poziva svaki puta kada se osvježi slika (engl. Frame)[10].

[3] Unity nadglednik – 5.9.2018. https://docs.unity3d.com/Manual/UsingTheInspector.html
[4] Unity projektni prozor – 5.9.2018. https://docs.unity3d.com/Manual/ProjectView.html
[5] Unity sredstva – 5.9.2018. https://docs.unity3d.com/Manual/AssetWorkflow.html
[6] Unity konzola – 5.9.2018. https://docs.unity3d.com/Manual/Console.html
[7] Unity usluge – 5.9.2018. https://unity3d.com/services
[8] Unity podržavani jezici – 5.9.2018. https://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/
[9] Unity Start funkcija – 5.9.2018. https://docs.unity3d.com/ScriptReference/MonoBehaviour.Start.html
[10] Unity Update funkcija – 5.9.2018. https://docs.unity3d.com/ScriptReference/MonoBehaviour.Update.html

3. RAZVOJ IGRE

3.1. Ideja igre

Ideja igre je proizašla iz želje za izradom vlastite igre kao početni korak u svijet razvoja igara. Prvotno je ideja došla prilikom igranja brojnih dvodimenzionalnih igara i s vremenom se samo razvijala. Priča koja se razvija nakon nesreće u gradu Černobilu, potaknuta je vlastitim interesom prema Černobilu i svemu što se događalo na tom području. Glavni lik je mali bezimeni robot koji pokušava pobjeći iz grada, nakon što ga vojska, koja je preplavila grad pokušava ugasiti i uništiti. Vojnici koji su glavni protivnik našem liku, potaknuti su iz raznih filmova. Sama ideja za prolazak kroz nivoe su zamišljeni u tri dijela. Prvi nivo kao učenje kontrola i učenje mehanika igre. Drugi nivo kao malo bolje upoznavanje u protivnike i krajnji završava priču i predstavlja izazov igraču sa zadnjim borbama i općenito prolaskom kroz nivo. Mehanike igre su zamišljene što jednostavnije i zabavnije. Kretanje je napravljeno kao i u većini igara pomoću tipkovnice. Pucanje je vrlo jednostavno i radi na način gdje je usmjeren miš i igrač stisne lijevu tipku miša, tamo naš glavni lik puca. Igra je zamišljena kao jednostavan oblik zabave s mehanikama kojima je igrač upoznat igrajući druge igre ili jednostavnim mehanikama za svladavanje. Prema slici 3.1. možemo vidjeti kako se razvoj igre odvijao. Dijagram ne pokazuje općeniti proces razvoja igre, on je prikaz vlastitog procesa koji se može primjenjivati na druge igre uz određene promjene. Također ne sadrži potpun proces izrade igre. U procesu se još pojavljuje izrada sučelja, izbornika i dr. koji nisu stavljeni na dijagram kako mu se ne bi povećavala kompleksnost.



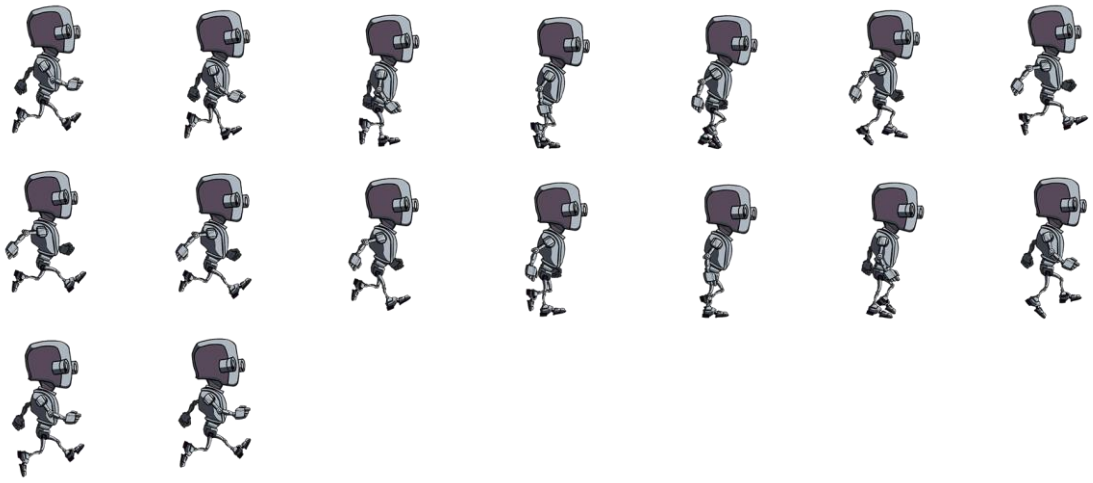
Sl. 3.1. Dijagrama toka razvoja igre

3.2. Razvoj likova

Razvoj likova je jedan od težih dijelova razvoja igre. Postoje dva glavna oblika likova: glavni lik (igrač) i protivnici. Oboje imaju različite funkcije i svi se razlikuju po svojim svojstvima.

3.2.1. Glavni lik

Glavnim likom upravlja igrač, ima mogućnost kretanja lijevo, desno, skakanja, saginjanja i pucanja.

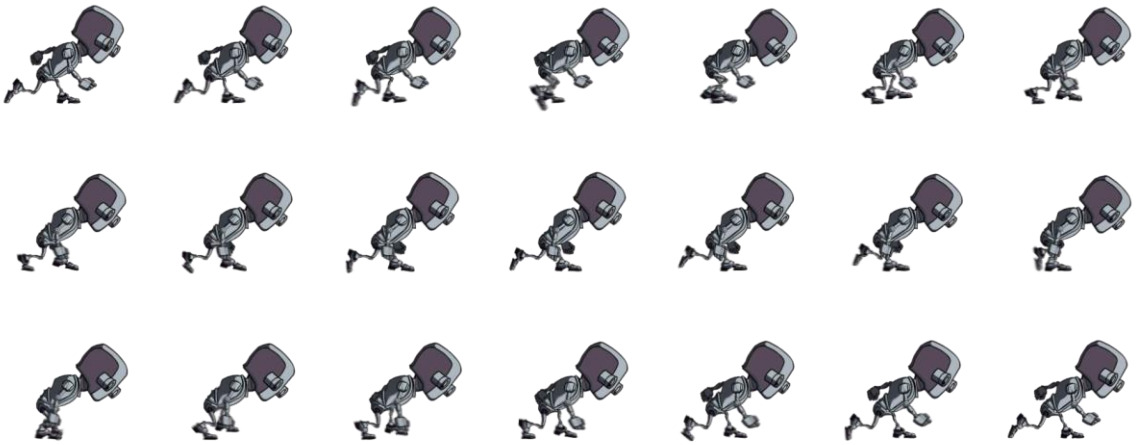


Sl. 3.2. Niz slika (engl. *sprite sheet)* trčanja glavnog lika[11]

[11] Niz slika trčanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064



Sl. 3.3. Niz slika (engl. *sprite sheet)* skakanja glavnog lika[12]



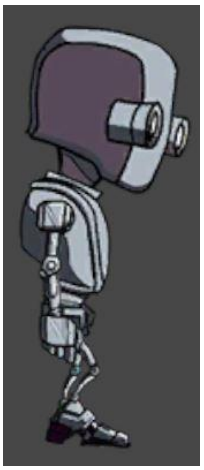
Sl. 3.4. Niz slika (engl. *sprite sheet)* saginjanja glavnog lika[13]

[12] Niz slika skakanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064

[13] Niz slika saginjanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064



Sl. 3.5. Izgled glavnog lika kako puca



Sl. 3.6. Izgled glavnog lika[14]

Upravljanje glavnim likom je vrlo jednostavno, provjeravaju se tipke koje su podešene za kretanja, ako su pritisnute, glavni lik se kreće u tom smjeru. Skakanje se odvija na sličan način kao i kretanje, provjerava se ako je tipka pritisnuta i glavni lik već nije u zraku, glavni lik zatim skoči ako su oba uvjeta ispunjena. Pucanje koje glavni lik ima kao svoju osobinu napravljena je tako da se na glavnom liku kao objektu, nalazi „ShootingPoint“ objekt kao dijete, iz kojeg izlaze laseri. Također se provjerava ako je igrač pritisnuo tipku i mjesto, odnosno smjer u kojemu treba ispucati laser.

[14] Izgled glavnog lika slika – 5.9.2018. https://k30.kn3.net/taringa/F/2/9/1/4/C/Shioo4Play/83A.png

```
void Shoot()
{
    Vector2 mousePosition = new Vector2(Camera.main.ScreenToWorldPoint(Input.mousePosition).x, Camera.main.ScreenToWorldPoint(Input.mousePosition).y);
    Vector2 firePointPosition = new Vector2(firePoint.position.x, firePoint.position.y);
    RaycastHit2D hit = Physics2D.Raycast(firePointPosition, mousePosition - firePointPosition, 100, whatToHit);
    if (Time.time >= timeToSpawnEffect)
    {
        Effect();
        timeToSpawnEffect = Time.time + 1 / effectSpawnRate;
    }
}

1 reference
void Effect()
{
    PlayerLaser.Play();

    Instantiate(BulletTrailPrefab, firePoint.position, firePoint.rotation);
    Transform clone = Instantiate(MuzzleFlashPrefab, firePoint.position, firePoint.rotation) as Transform;
    clone.parent = firePoint;
    float size = Random.Range(0.6f, 0.9f);
    clone.localScale = new Vector3(size, size, size);
    Destroy(clone.gameObject, 0.0005f);
}
```

Sl. 3.7. Kod iz skripte za pucanje

Prema slici 3.7. možemo vidjeti kod koji se koristi za pucanje. Pronalazimo poziciju gdje se miš nalazi i poziciju gdje se nalazi mjesto odakle će laser biti ispucan. Provjeravamo ako je vrijeme koje je prošlo od zadnjeg ispucanog lasera jednako ili veće od onog kojeg smo postavili i pozivamo funkciju Effect(). U toj funkciji pozivamo prvo zvuk koji koristimo za laser, zatim pozivamo rutinu Instantiate() koja postavi laser na mjesto iz kojeg treba biti ispucan i uputi ga u smjeru u kojem se treba kretati.

```
void Start () {
    DestroyTimeStart = Time.time;
}
0 references
void Update () {
    if ((Time.time - DestroyTimeStart) >= 0.45f)
    {
        Destroy(this.gameObject);

    }
}
0 references
void OnTriggerEnter2D(Collider2D trig)
{

    if (trig.gameObject.tag == "Enemy1" || trig.gameObject.tag == "Enemy3" || trig.gameObject.tag == "PlantAndRock"
        || trig.gameObject.tag == "EnemyBoss" || trig.gameObject.tag == "Ground")
    {

        Destroy(GameObject.Find("Laser(Clone)"));
        Destroy(GameObject.Find("LaserLeft(Clone)"));
    }
}
```

Sl. 3.8. Kod iz skripte za laser

Kako bismo bolje razumjeli kako će se laser ponašati, prema slici 3.8. možemo vidjeti u Start() funkciji koja se poziva samo na početku skripte, postavljamo vrijeme uništavanje objekta lasera

na trenutno vrijeme. Zatim, svaku novu sliku (engl. Frame) provjeravamo ako je prošlo vrijeme bez da je laser išta pogodio i ako mu je prošlo vrijeme trajanja, uništavamo taj objekt. Ako laser pogodi jedan od objekta koji su navedeni u skripti, to se registrira u funkciji OnTriggerEnter2D() i objekt laser se uništava. Ozljeđivanje protivnika ili glavnog lika se izvršava u posebnim skriptama.

```
void Update () {
    if(Input.GetButtonDown("Use")){
        if (ExtraHealthCount == 1)
        {

            if (Player_Health.health <= 75)
            {
                HealingSound.Play();

                Instantiate(HealthEffect, transform.position, transform.rotation);
                Player_Health.health += ExtraHealth;
                ExtraHealthCount = 0;
                if (GameObject.Find("HealingEffect(Clone)") != null)
                {
                    Destroy(GameObject.Find("HealingEffect(Clone)"), 2f);
                }
            }
        }
    }

    if (scoreTemp >= 100)
    {
        ExtraHealth += 5;
        scoreTemp -= 100;
    }

    if (gameObject.transform.position.y < -110) {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name);
    }
    if(health<1){
        SceneManager.LoadScene (SceneManager.GetActiveScene().name);
    }
    HealthUI.gameObject.GetComponent<Text> ().text = ("Health: " + (int)health);
    ScoreUI.gameObject.GetComponent<Text>().text = ("Score: " + score);
```

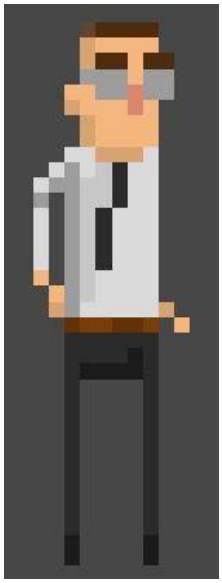
Sl. 3.9. Dio koda iz skripte glavnog lika

Prema slici 3.9. možemo vidjeti kod koji se izvršava za posebnu moć, koju glavni lik ima te za vrijednosni sustav i energiju glavnog lika. Posebnu moć koju glavni lik ima je dodavanje količine energije glavnom liku prema vrijednosnom sustavu. Ako glavni lik ima manje od ili 75 energije i igrač je pritisnuo tipku za aktivaciju posebne moći, glavnom liku se dodaje određena količina energije. Dodatna količina energije raste s porastom „score“ vrijednosti. Ona je varijabla koja se dodaje igraču kao nagrada za uspješno svladavanje protivnika. Također, možemo vidjeti da ako igrač padne ispod određene razine na nivou da ga vraća na početak i ako mu energija padne ispod vrijednosti 1. Na kraju ove slike možemo vidjeti dvije linije koda koje su odgovorne za

pokazivanje trenutne vrijednosti energije glavnog lika i vrijednosti „score“ varijable, koje će biti objašnjene u kasnijem poglavlju.

3.2.2. Protivnici

U ovoj igri postoje 4 vrste protivnika. Agent koji napada šakama, agent koji može pucati na glavnog lika, plaćenici šefa i glavni šef. Svaki od protivnika ima svoju energiju, svoj iznos koji se doprinosi varijabli „score“ te udaljenost s koje može primijetiti glavnog lika i napasti ga. Protivnici koriste gotove skripte „AstarPathFindingProject“[15] koje su dostupne svima iz razloga kompleksnosti rješavanja prepoznavanja terena i mjesta gdje se protivnici mogu kretati.



Sl. 3.10. Izgled agenta koji napada šakama[16]

Agent koji napada šakama je vrlo jednostavan lik. Njegova udaljenost s koje može vidjeti glavnog lika je najmanje i najmanje može oštetiti glavnog lika. Pojavljuje se kroz sve nivoe kao jednostavna nagrada igraču.

[15] A* Pathfinding Project – 2.6.2018. https://arongranberg.com/astar/
[16] Agent koji napada šakama – 5.9.2018. https://opengameart.org/content/business-of-rage-characeter-beatemup-2d

```
//ANIMATIONS
if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) > attackDistance)
{
    GetComponent<Animator>().SetBool("IsRunningEnemy1", false);
    GetComponent<Animator>().SetBool("PunchPlayerEnemy1", false);
}
else if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) <= attackDistance && Mathf.Abs(distanceFromEnemy1ToPlayer.y) <= 90f)
{
    if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) <= 20 && Mathf.Abs(distanceFromEnemy1ToPlayer.y) <= 90f)
    {
        GetComponent<Animator>().SetBool("PunchPlayerEnemy1", true);
    }
    else if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) > 20)
    {
        GetComponent<Animator>().SetBool("PunchPlayerEnemy1", false);

    }
    GetComponent<Animator>().SetBool("IsRunningEnemy1", true);

    //ENEMY ATTACK
    if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) > attackDistance)
    {
    }
    else if (Mathf.Abs(distanceFromEnemy1ToPlayer.x) <= attackDistance && Mathf.Abs(distanceFromEnemy1ToPlayer.y) <= 90f)
    {
        if (distanceFromEnemy1ToPlayer.x < 0f )
        {
            transform.Translate(-1 * enemySpeed * Time.deltaTime, 0, 0);
        }
        else if (distanceFromEnemy1ToPlayer.x > 0f)
        {
            transform.Translate(1 * enemySpeed * Time.deltaTime, 0, 0);
        }
```

Sl. 3.11. Dio koda za agenta koji udara šakama

Prema slici 3.11. možemo vidjeti na koji način se ovaj agent ponaša. Animacije su bitan dio igre, pridonose doživljaju igre. Prema kodu možemo vidjeti tek kada agent ugleda glavnog lika njegova animacija trčanja se aktivira i kada dođe dovoljno blizu glavnom liku se aktivira animacija udaranja. Što se tiče samog napadanja agenta, vrlo je jednostavno. Ako se nalazi na udaljenosti na kojoj može vidjeti glavnog lika, objekt agent se počinje pomicati prema glavnom liku.



Sl. 3.12. Izgled agenta koji može pucati

Prema slici 3.12. vidimo izgled agenta koji može pucati i koji je na idućoj težini od prijašnjeg agenta. Osobine koje definiraju ovog agenta su njegova brzina, veća količina energije i pucanje na glavnog lika. Kako igra ne bi bila vrlo jednostavna, ovaj agent se pobrine kako bi se igrač prilagodio i pronašao način izbjegavanja lasera koje ispucava agent. Kretanje i animacije su slične agentu koji napada šakama. Pucanje je odrađeno na sličan način kao i kod glavnog lika. Na objektu agenta se nalazi objekt kao dijete „ShootingPoint“ koji služi kao mjesto iz kojeg se ispucavaju laseri.

```
void Update()
{
    distanceFromEnemy3ToPlayer = Player.transform.position.x - transform.position.x;
    LaserEnemy.transform.position = transform.position + Vector3.zero;
    LaserLeftEnemy.transform.position = transform.position + Vector3.zero;
    if (fireCountdown <= 0f)
    {
        Shoot();
        fireCountdown = 1f / fireRate;
    }
    fireCountdown -= Time.deltaTime;
}
1 reference
void Shoot()
{
    if (distanceFromEnemy3ToPlayer < 150 && distanceFromEnemy3ToPlayer >= 0)
    {
        Instantiate(LaserEnemy);
    }
    else if (distanceFromEnemy3ToPlayer > -150 && distanceFromEnemy3ToPlayer < 0)
    {
        Instantiate(LaserLeftEnemy);
    }
}
```

Sl. 3.13. Dio koda za agentovo pucanje

Prema slici 3.13. možemo vidjeti način na koji se izvršava pucanje. Svake nove slike (engl. Frame) računa se udaljenost od glavnog lika do agenta i pozicija s koje se ispucava laser se podešava. Provjerava se ako je prošlo vrijeme otkad je zadnji laser ispucan, ako je prošlo ili je jednako zadanom, poziva se funkcija Shoot(). U funkciji se provjerava udaljenost od glavnog lika do agenta i s koje strane mu se nalazi. Ako se glavni lik nalazi lijevo od agenta, laser se ispucava na lijevo i obrnuto ako se glavni lik nalazi desno.



Sl. 3.14. Izgled plaćenika šefa[17]

Plaćenici šefa su drugi najteži protivnici, zamišljeni kako bi uzrokovali štetu glavnom liku i probleme igraču. Prema slici 3.14. možemo vidjeti njihov izgled. Njihove osobine su: visina, brzina, jačina udarca i količina energije koju imaju. Pojavljuju se puno rjeđe od ostalih protivnika, zbog njihove težine. Kao i prijašnja dva agenta, imaju isti način kretanja i korištenja animacija. Ponašanje je gotovo jednako agentu koji napada šakama, ali ovaj plaćenik ima više energije, agresivniji je i udarci nanose više štete glavnom liku nego ostali.

[17] Izgled plaćenika šefa – 5.9.2018. https://opengameart.org/content/business-of-rage-characeter-beatemup-2d



Sl. 3.15. Izgled glavnog šefa

Prema slici 3.15. vidimo izgled glavnog šefa koji se nalazi kao posljednji protivnik igre. Njegove osobine su: Visina, energija, brzina i jačina udarca. Zamišljen je kao zadnja prepreka koju igrač mora riješiti prije nego završi priču igre. Kao i njegovi plaćenici ima jednake kretnje i animacije. Razlikuje ih količina energije, s time da glavni šef ima više.

3.3.Razvoj nivoa

U ovo projektu se nalaze 3 glavna nivoa i izbornik. Sve je podijeljeno po scenama, postoji šest scena sveukupno. Prve dvije scene se odnose na izbornik, koje ćemo proći više na jednom od idućih poglavlja. U ovom poglavlju ćemo pričati više od samom dizajniranju i načinu izrade samih nivoa. Nulti nivo je zamišljen kao uvod u igri i objašnjavanje kontrola igraču kako bih razumio način igranja. Prvi nivo je namijenjen upoznavanju s prve dvije vrste protivnika. Nivoi su osmišljeni tako da se njihova težina povećava i potiče igrača da se prilagodi i pronađe način kojim će riješiti problem.



Sl. 3.16. Slika nultog nivoa[18]

Nulti nivo, prema slici 3.16., napravljen je na način kako glavni igrač prolazi po ravnoj podlozi, da tekst iskače i pokazuje mu određene dijelove igre. Svaki od idućih nivoa, ima svoju priču koja se prepričava na isti način. Cilj dizajniranja ovog nivoa nije toliko bitna, fokus je postavljen na objašnjavanje mehanike igre igraču.



Sl. 3.17. Slika prvog nivoa[19]

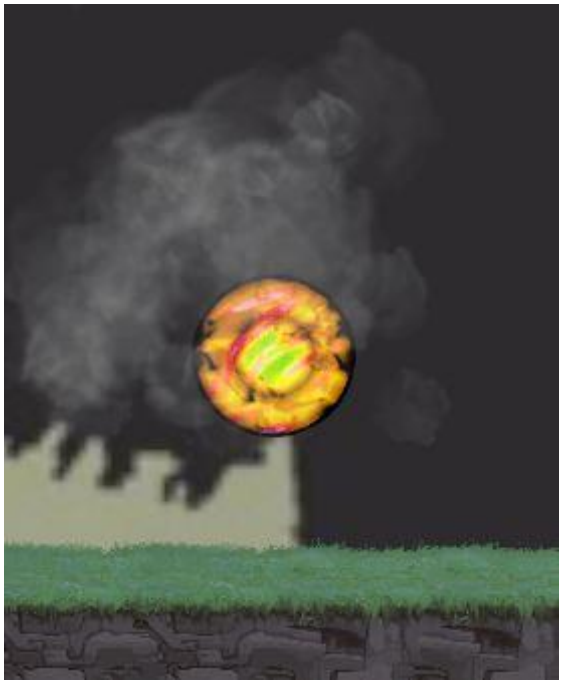
[18] Pozadina nultog nivoa – 5.9.2018. https://goo.gl/NG3AVt
[19] Pozadina prvog nivoa – 5.9.2018. https://goo.gl/RXzv9x

Na slici 3.17. vidimo izgled prvog nivoa. Dizajniran je na način kako bi se pokazala kretanja protivnika i uvježbale tehnike skakanja za daljnje izazove. Nivo se odvija u šumi van samog grada gdje se robot prvi puta probudi. Tu se igrač upoznaje s agentima koji ga napadaju.



Sl. 3.18. Radioaktivne biljke koje se nalaze u prvom nivou

Jedan od dodatnih problema na koje igrač može naići možemo vidjeti na slici 3.18.. Radioaktivne biljke koje glavnom liku oduzimaju energiju ako prolazi kroz njih. Razlog dodavanja ovakvih elemenata igri je dodavanje kompleksnosti samom nivou i poticanje igrača da prilagodi svoju putanju kroz isti.



Sl. 3.19. Izgled kugle kao dodatni element

Na slici 3.19. možemo vidjeti dodatne elemente koji se nalaze na svakom nivou. Razlog dodavanja takvih elemenata je kako igrač ne bi samo pretrčao svaki od nivoa i tako prešao igru. U svakom nivou se nalaze dvije takve kugle, osim u nultom.



Sl. 3.20. Izgled drugog nivoa[20]

Na slici 3.20. vidimo izgled drugog nivoa, glavni lik prvi puta ulazi u grad pomoću kugli koje je skupio u prijašnjem nivou. Igrač se po prvi puta upoznaje s plaćenikom na kraju nivoa. Zamišljen je više vertikalno kako se igra ne bi osjećala ustaljenom i kako bi se dodala dinamičnost.



Sl. 3.21. Izgled bodlji

U ovom nivou je dodan drugi oblik bodlji koje imaju isti utjecaj kao radioaktivne biljke iz prvog nivoa. Također se pojavljuju u idućem.

[20] Pozadina drugog nivoa – 5.9.2018. https://goo.gl/anW4S7



Sl. 3.22. Izgled trećeg (posljednjeg) nivoa[21]



Sl. 3.23. Izgled platforme koja se kreće

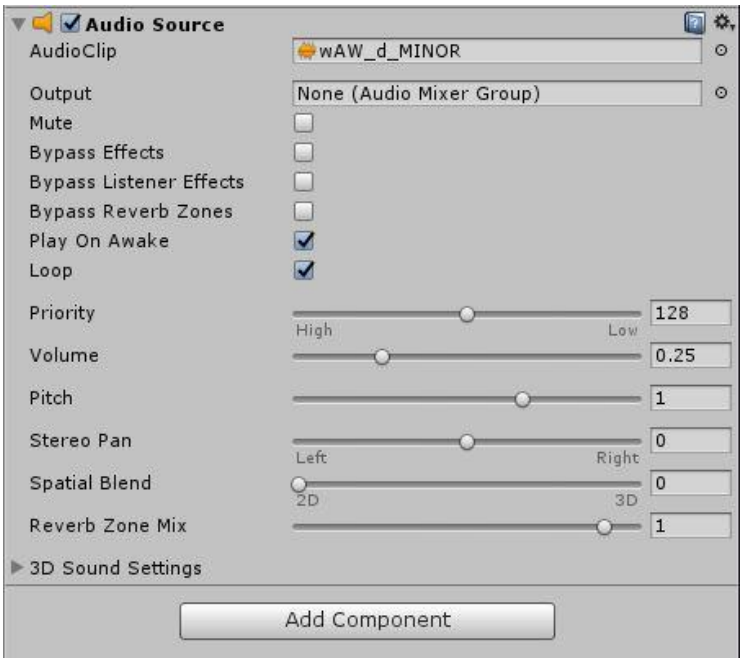
Na slici 3.22. i 3.23. možemo vidjeti treći i posljednji nivo, ujedno i najteži. U nivo su dodane platforme koje se kreću kako bi se dodala dinamičnost i kompleksnost igri. Platforme čekaju na svakom kraju kako bi igrač stigao stati na njih ili ako se odluči vratiti na prošle dijelove nivoa. Rade tako da uhvate igrača i dok se kreću igrač se ne može kretati kako bi spriječili padanje s nje.

[21] Pozadina trećeg nivoa – 5.9.2018. https://goo.gl/NG3AVt

Na kraju ovog nivoa, igrač se susreće s glavnim šefom i u slučaju da ga pobjedi, igrač završava igru.

3.4. Odabir glazbe i zvukova

Glazba, kao i glazbeni efekti koje dodajemo igri donose dubinu i poboljšavaju igračevo iskustvo. Glazba koju sam odabrao za ovu igru je osam bitne tematike i oponaša poslije apokaliptičnu temu. Svaki od nivoa ima svoju glazbu koja doprinosi scenama. Također, dodani su glazbeni efekti čestim radnjama, kao što je pucanje, skakanje, udarac i drugo. Kako bismo dodali glazbu nekom nivou, potrebno je napraviti prazni objekt koji je izvor zvuka (engl. Audio Source)[22] i na njega postaviti našu željenu pjesmu (engl. Audio clip)[23].



Sl. 3.24. Izgled postavki objekta izvora zvuka

Na slici 3.24. možemo vidjeti izgled postavki koje možemo izmijeniti za neki izvor zvuka. Ovaj dio je bitan ako se radi o glazbi za cijeli nivo. Dvije ključne postavke su kretanje na buđenje (engl. Play On Awake)[24] koja pokreće glazbu kada se objekt učita i petlja (engl. Loop)[25] koja je iznimno korisna ako ne želimo da pjesma završi prije nego igrač završi nivo. U slučaju ove igre imamo dvije scene za glavni izbornik. Kako ne želimo da pjesma krene iz početka kada promijenimo scenu, napravljena je skripta koja nastavlja pjesmu kroz scene.

[22] Unity izvor zvuka – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioSource.html
[23] Unity audio isječak – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioClip.html
[24] Unity kretanje na buđenje – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioSource-playOnAwake.html
[25] Unity petlje – 5.9.2018. https://unity3d.com/learn/tutorials/topics/scripting/loops

```
0 references
void Start()
{
    DontDestroyOnLoad(this.gameObject);

}

0 references
void Update()
{
    if (SceneManager.GetActiveScene().name != "MainMenu" && SceneManager.GetActiveScene().name != "MainMenu 1")
    {
        Destroy(this.gameObject);
    }
}
```

Sl. 3.25. Dio koda odgovoran za nastavak glazbe

Prema slici 3.25. možemo vidjeti kod odgovoran za nastavak glazbe s prelaska jedne scene u drugu. U Start() funkciji koristimo funkciju koja ne uništava objekt pri učitavanju nove scene. U funkciji Update() koristimo uvjete koji osiguravaju da glazba svira na samo dvije scene kojima želimo, inače uništava objekt izvora zvuka i prekida se pjesma. Zvučni efekti koje koristimo u igri rade na sličan način kao i glazba nivoa.

```
0 references
void OnTriggerEnter2D(Collider2D trig)
{

    if (trig.gameObject.tag == "Enemy1" || trig.gameObject.tag == "EnemyBoss" || trig.gameObject.tag == "Enemy3")
    {
        PlayerGotHit.Play();
    }
    if (trig.gameObject.tag == "PlantAndRock")
    {
        PlayerGotHit.Play();
    }
    if (trig.gameObject.tag == "GunEnemy")
    {
        PlayerGotHit.Play();
    }

}
```

Sl. 3.26. Dio koda odgovoran za puštanje zvučnih efekata

Prema slici 3.26. možemo vidjeti kako vrlo jednostavno možemo pokrenuti nekakav zvuk pomoću skripte. Ako glavni lik uđe u neki od objekata ili ga neki objekt označi, efekt će se pokrenuti i jednom će svirati. Potrebno je u skripti definirati izvor zvuka kao tip podatka izvor zvuka i u nadgledniku (engl. Inspector) pridodati zvuk skripti. Drugi zvučni efekti rade na isti ili vrlo sličan način, razlikuju se samo po uvjetima koji ih pokreću. Glazba korištena u igri je besplatna i postoji veliki izbor na internetskoj stranici[26].

[26] OpenGameArt stranica besplatnog sadržaja – 4.6.2018. https://opengameart.org/

3.5. Odabir vizualnih efekata

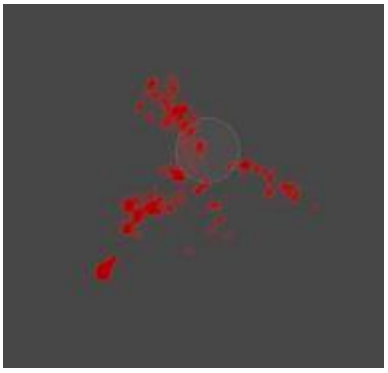
Vizualni efekti se pojavljuju ako je glavni lik oštećen, koristi posebnu moć dodavanja energije, ako su protivnici pogođeni i na kuglama koji su dodatni elementi igre.



Sl. 3.27. Izgled efekta kada je glavni lik oštećen



Sl. 3.28. Izgled efekta kada igrač koristi posebnu moć



Sl. 3.29. Izgled efekta kada je protivnik pogođen



Sl. 3.30. Izgled efekta kugle kao dodatnog elementa igri

Na slikama 3.27.,3.28.,3.29.,3.30. možemo vidjeti izglede vizualnih efekata u igri. Vizualni efekti se izvode pomoću skripte koju smo dodali objektu na kojem želimo vidjeti i prikazati efekt. Pri samom početku skripte definiramo objekt igre (engl. Game Object)[27] kojemu ćemo dodati gotovi efekt.

```
0 references
private void Start()
{

    Collectable = 0;
        if (transform.name == "CollectableOrb")
        {
            Instantiate(pickupEffect, CollectableObject1.transform.position, CollectableObject1.transform.rotation);
        }
        if (transform.name == "CollectableOrb (1)")
        {
            Instantiate(pickupEffect, CollectableObject2.transform.position, CollectableObject2.transform.rotation);
        }

}
```

Sl. 3.31. Dio koda za vizualne efekte

[27] Unity objekt igre – 5.9.2018. https://docs.unity3d.com/ScriptReference/GameObject.html

Na slici 3.31. možemo vidjeti kod koji pokreće vizualni efekt koji smo dodali objektu. Pomoću rutine Instantiate() pokrećemo efekt na određenoj poziciji i rotaciji.[28] Efekt možemo uništiti kao i svaki drugi objekt pošto smo ga tako definirali u bilo koje vrijeme, pod bilo kojim uvjetima. Vizualni efekti korišteni u ovoj igri su besplatni i preuzeti sa stranice besplatnog sadržaja[29] koju podržava tvrtka Unity.

3.6. Razvoj korisničkog sučelja

Korisničko sučelje je bitan dio igre jer igrač dobiva povratne informacije o događanjima u igri. U ovoj igri korisničko sučelje je napravljeno što jednostavnije jer nema puno informacija o kojima igrač treba brinuti.



Sl. 3.32. Izgled korisničkog sučelja

Kao što možemo vidjeti prema slici 3.32. korisničko sučelje je vrlo jednostavno. Kako bismo izradili korisničko sučelje ovog tipa, potrebno je izraditi platno (engl. Canvas)[30] na koji ćemo dodavati naše elemente. U postavkama platna, mijenjamo na koji će način ono raditi s kamerom. U slučaju korisničkog sučelja potrebno je postaviti da prati kameru. Nakon što postavimo naše platno, potrebno je izraditi objekte koje želimo prikazati. U ovoj igri postoji energija i bodovi (engl. Health and Score) koje izrađujemo kao tekst. Kako bismo osvježili podatke koji se prikazuju potrebno ih je u pripadajućim skriptama pratiti. U skripti je izrađen objekt igre (engl. Game Object) kojemu pridodajemo izrađeni objekt.

```
HealthUI.gameObject.GetComponent<Text> ().text = ("Health: " + (int)health);
ScoreUI.gameObject.GetComponent<Text>().text = ("Score: " + score);
```

Sl. 3.33. Dio koda za korisničko sučelje

[28] Unity rutina Instantiate – 5.9.2018. https://docs.unity3d.com/ScriptReference/Object.Instantiate.html
[29] Unity asset store – 4.6.2018. https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-73777
[30] Unity platno – 5.9.2018. https://docs.unity3d.com/Manual/UICanvas.html

Prema slici 3.33. možemo vidjeti na koji način osvježavamo iznose pomoću funkcije Update() koje prikazujemo igraču.

3.7. Razvoj izbornika

Izbornik je dio s kojim se igrač susreće pri pokretanju igre. U ovoj igri su napravljene dvije scene koje predstavljaju izbornik.

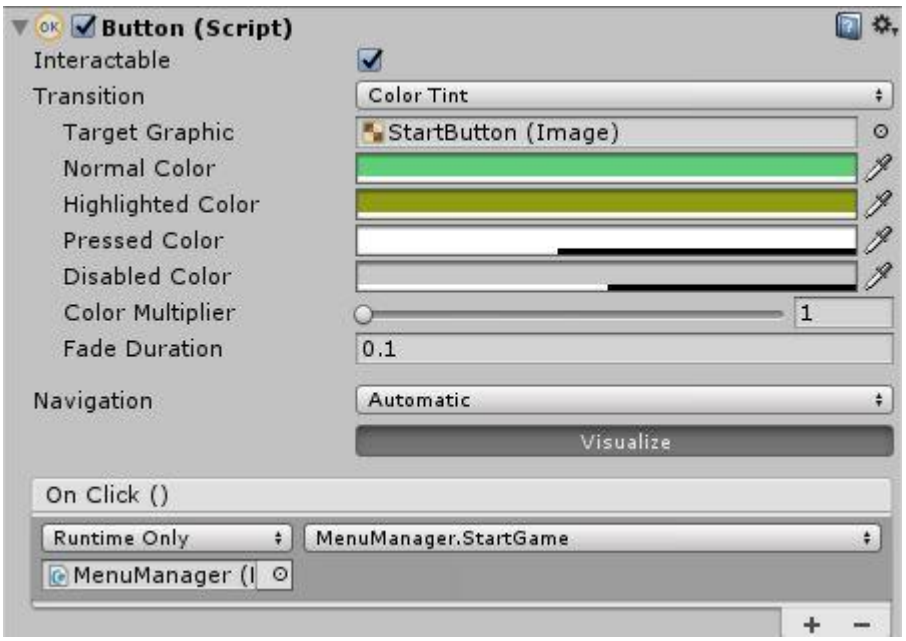


Sl. 3.34. Izgled glavnog izbornika



Sl. 3.35. Izgled izbornika nivoa

Prema slikama 3.34. i 3.35. možemo vidjeti izgled glavnog izbornika i izbornika nivoa. Način na koji su izrađeni su vrlo slični. Prvo su postavljene pozadine, zatim je napravljeno platno (engl. Canvas) na kojeg izrađujemo objekte koji će predstavljati dugmad (engl. Buttons).



Sl. 3.36. Postavke dugmeta

Na slici 3.36. vidimo postavke jednog od dugmeta. Moguće je izmijeniti više postavki boje, ali nama najbitniji dio je OnClick()[31] funkcija pomoću koje namještamo što će to dugme napraviti. U slučaju ove igre potrebno je bilo napraviti posebnu skriptu koja će odrađivati sve funkcije koje su nam potrebne.

[31] Unity funkcija OnClick – 5.9.2018. https://docs.unity3d.com/ScriptReference/UI.Button-onClick.html

```
0 references
public void StartGame()
{
    SceneManager.LoadScene("Level0");
}
0 references
public void QuitGame()
{
    Application.Quit();
}

0 references
public void SelectLevel()
{
    SceneManager.LoadScene("MainMenu 1");
}
```

Sl. 3.37. Dio koda iz skripte za funkcije dugmeta

Na slici 3.37. vidimo dio koda skripte koja nam omogućuje da dugmad koju smo postavili napravi željenu radnju. Jedan od zanimljivijih dijelova izrade izbornika je izrada ako igrač želi pauzirati igru. Na svaki nivo, na platnu na kojem se nalazi korisničko sučelje potrebno je izraditi prazan objekt kojemu ćemo dodati dugme i njemu dodati skriptu koja će pauzirati igru i promijeniti grafiku kako bi se igraču dalo do znanja da je igra pauzirana.

```
void Update () {
    if (Input.GetKeyDown(KeyCode.Escape))
    {
        if (GameIsPaused == true)
        {
            Resume();
        }
        else
        {
            Pause();
        }
    }
}
1 reference
void Resume()
{
    pauseMenuUI.SetActive(false);
    Time.timeScale = 1f;
    GameIsPaused = false;
}
1 reference
void Pause()
{
    pauseMenuUI.SetActive(true);
    Time.timeScale = 0f;
    GameIsPaused = true;
}

0 references
public void LoadMainMenu()
{
    SceneManager.LoadScene("MainMenu");
}
```

Sl. 3.38. Dio koda za pauziranje

Na slici 3.38. možemo vidjeti kod odgovoran za pauziranje igre. Kako bismo mogli pritisnuti tipku za pauzu bilo kada u igri, u funkciji Update() potrebno je provjeravati ako je tipka pritisnuta i provjeravati *boolean*[32] vrijednost kojom pratimo ako je tipka pritisnuta. Zatim su izrađene dvije funkcije za nastavak (engl. Resume) i pauzu (engl. Pause). U funkciji za nastavak, gasimo izbornik za pauzu i postavljamo vrijeme da normalno prolazi. U funkciji za pauzu radimo obrnuto.

[32] Tip podatka u programiranju koji može biti postavljen na istinu ili laž

4. Zaključak

Cilj ovog završnog rada je bio izraditi dvodimenzionalnu akcijsku platformsku igru te prikazati i primijeniti osnove programskog okruženja za razvoj i upravljanje igrama Unity. Osim samog cilja pravljenja igre i pisanja rada, želja je bila upoznati se s procesom izrade igre. Razviti dovoljno znanja kako bih pisao vlastite skripte potrebne za pokretanje likova, nivoa, raznih efekata i sličnog. Uz pisanje skripti, primijenjen je objektno orijentirani programski jezik C#. Cilj same igre je upoznavanje igrača s pričom i mehanikama koje predstavljaju osnove dvodimenzionalnih igara. Od početka razvoja igre, gdje se osnovala ideja za igru, krenuo je razvoj likova koji se pokazao kao kompleksan zadatak zbog potrebe za izradom umjetne inteligencije protivnika te dodavanja raznih mogućnosti glavnom liku, kako bi se razlikovao od ostalih igara. Razvoj nivoa tematski se uklopio u naziv igre. Uz odabir glazbe za efekte i glazbe za nivoe dodana je dubina igri i njezina čar. Krajnji dio je bio razvoj korisničkog sučelja s kojim se igrač susreće tijekom cijele igre i izbornika s kojim igrač stupa u kontakt pri prvom pokretanju igre, pokazali su se kao dobar dodatak igri koji će pomoći s ukupnim doživljajem. Ciljevi postavljeni prilikom razvijanja ideje igre, su postignuti uz dovoljan prostor za razvoj novih ideja, provedbu novih mehanika i dodavanje novih dijelova igre. Probleme na koje se nailazilo tijekom razvoja igre, su riješeni s količinom sadržaja i znanja koji se pružaju na raznim internetskim stranicama i službenoj dokumentaciji koju pruža programsko okruženje Unity.

Literatura

[1] Unity programsko okruženje – 5.9.2018. https://en.wikipedia.org/wiki/Unity_(game_engine)

[2] Unity scena – 5.9.2018. https://docs.unity3d.com/Manual/CreatingScenes.html

[3] Unity nadglednik – 5.9.2018. https://docs.unity3d.com/Manual/UsingTheInspector.html

[4] Unity projektni prozor – 5.9.2018. https://docs.unity3d.com/Manual/ProjectView.html

[5] Unity sredstva – 5.9.2018. https://docs.unity3d.com/Manual/AssetWorkflow.html

[6] Unity konzola – 5.9.2018. https://docs.unity3d.com/Manual/Console.html

[7] Unity usluge – 5.9.2018. https://unity3d.com/services

[8] Unity podržavani jezici – 5.9.2018. https://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/

[9] Unity Start funkcija – 5.9.2018. https://docs.unity3d.com/ScriptReference/MonoBehaviour.Start.html

[10] Unity Update funkcija – 5.9.2018. https://docs.unity3d.com/ScriptReference/MonoBehaviour.Update.html

[11] Niz slika trčanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064

[12] Niz slika skakanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064

[13] Niz slika saginjanja glavnog lika – 5.9.2018. https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-4-6-21064

[14] Izgled glavnog lika slika – 5.9.2018. https://k30.kn3.net/taringa/F/2/9/1/4/C/Shioo4Play/83A.png

[15] A* Pathfinding Project – 2.6.2018. https://arongranberg.com/astar/

[16] Agent koji napada šakama – 5.9.2018. https://opengameart.org/content/business-of-rage-characeter-beatemup-2d

[17] Izgled plaćenika šefa – 5.9.2018. https://opengameart.org/content/business-of-rage-characeter-beatemup-2d

[18] Pozadina nultog nivoa – 5.9.2018. https://goo.gl/NG3AVt

[19] Pozadina prvog nivoa – 5.9.2018. https://goo.gl/RXzv9x

[20] Pozadina drugog nivoa – 5.9.2018. https://goo.gl/anW4S7

[21] Pozadina trećeg nivoa – 5.9.2018. https://goo.gl/NG3AVt

[22] Unity izvor zvuka – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioSource.html

[23] Unity audio isječak – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioClip.html

[24] Unity kretanje na buđenje – 5.9.2018. https://docs.unity3d.com/ScriptReference/AudioSource-playOnAwake.html

[25] Unity petlje – 5.9.2018. https://unity3d.com/learn/tutorials/topics/scripting/loops

[26] OpenGameArt stranica besplatnog sadržaja – 4.6.2018. https://opengameart.org/

[27] Unity objekt igre – 5.9.2018. https://docs.unity3d.com/ScriptReference/GameObject.html

[28] Unity rutina Instantiate – 5.9.2018. https://docs.unity3d.com/ScriptReference/Object.Instantiate.html

[29] Unity asset store – 4.6.2018. https://assetstore.unity.com/packages/essentials/asset-packs/unity-particle-pack-73777

[30] Unity platno – 5.9.2018. https://docs.unity3d.com/Manual/UICanvas.html

[31] Unity funkcija OnClick – 5.9.2018. https://docs.unity3d.com/ScriptReference/UI.Button-onClick.html

[32] Tip podatka u programiranju koji može biti postavljen na istinu ili laž

Sažetak

Ovaj se završni rad temelji na izradi dvodimenzionalne akcijske platformske igre s ciljem upoznavanja programskog okruženja za razvoj i upravljanje igrama Unity. Priča igre odvija se nakon tragične nesreće koja se dogodila u gradu Černobilu. Cilj je glavnog lika bijeg iz Černobila zbog vojske koja ga progoni kako bi ga ugasila. Igra koristi besplatne sadržaje pronađene na raznim internetskim stranicama i sadržaju koji pruža samo programsko okruženje Unity. Skripte koje pokreću igru napisane su u objektno-orijentiranom programskom jeziku C#. Mehanike igre protežu se od običnog kretanja do skakanja i pucanja. Elementi poput glazbe, zvučnih i vizualnih efekata dodaju igri kompleksnost i čine ju potpunom. Igra je napravljena kroz više različitih scena koje se povezuju kako bi načinile cijelu igru. Scene sadrže glavni i dodatni izbornik kojim možemo pristupiti svim nivoima igre. Cilj je igre postignut i postoji prostor za daljnji razvoj i dodavanje novih mehanika i sadržaja igri.

Ključne riječi: dvodimenzionalna akcijska platformska igra, Unity, objektno orijentirani programski jezik C#, Černobil

Abstract

2D action platformer game

This final paper deals with developing a two-dimensional action platform game with a purpose of learning about a Unity game engine. The story takes place after the tragic accident that took place in the town of Chernobyl. The main character's goal is to escape from Chernobyl from an army that is persecuting him to shut him down. The game uses free content found on different websites and the content provided by the program environment itself. Scripts that run the game are written in the object-oriented programming language C#. Mechanics of the game range from ordinary motion to jumping and shooting. Elements like music and visual effects add complexity to the game and make it complete. The game is built out of several different scenes which, when connecting, make the game a unit. Scenes contain the main and additional menu to access all levels of the game. The goal of the game is achieved and there is room for further development and adding new mechanics and content to the game.

Keywords: Two-dimensional action platformer game, Unity, object-oriented programming language C#, Chernobyl

Životopis

Borna Šumiga rođen je 22.03.1996. u gradu Osijeku. Pohađao je Osnovnu Školu Frana Krste Frankopana od 2002. do 2010. godine. Nakon toga upisao je srednju Elektrotehničku i prometnu školu Osijek smjer elektrotehnika, koju je završio u 2014. godini. Odmah nakon završetka srednje škole, upisao je preddiplomski studij računarstva na sadašnjem Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek. Od svoje 4 godine upoznat je s video igrama i od onda se samo razvija njegova ljubav prema igranju i želja za razvijanjem vlastitih igara. Na fakultetu se upoznao s osnovama programiranja i upoznat je s objektno orijentiranim programskim jezikom C# i raznim drugima. U osnovnoj i srednjoj školi je sudjelovao u natjecanjima iz fizike i engleskog. Od stranih jezika govori engleski jezik.