The Heart Within - osnovni elementi implementacije

Autor: Jakov Begović

Mentor: doc. dr. sc. Mladen Konecki

Napomena: projekt se u nekim svojim dijelovima bazira na radovima [1] i [2]. Sličnosti u kodu i mehanikama će biti postojane. U slučajevima kada je kod ovih dokumenata uvelike izmijenjen oni neće biti navedeni kao izvor.

Sadržaj

1. Glavni lik

Izvor sprite-ova z lika igrača: [3]

1.1. Kretanje

Glavni lik se kreće pomoću fizike. Odnosno, koristi se komponenta Rigidbody2D.

```
private void FixedUpdate()
 {
     rb.MovePosition(rb.position + moveVector * Time.fixedDeltaTime);

 }
 private void HandleMovement()
 {

     inputVert = Input.GetAxisRaw("Vertical");
     inputHor = Input.GetAxisRaw("Horizontal");

     moveVector = new Vector2(inputHor, inputVert).normalized * movementSpeed;
```

```
    }
```

Metoda FixedUpdate se koristi kada god se pozivaju naredbe vezane uz fiziku. Zbog toga se unutar nje izvršava naredbe za pomicanje glavnog lika. Vektor pomicanja (moveVector) računa unutar HandleMovement metode.s

HandleMovement metoda računa vrijednost moveVector varijable prema unosu sa tipkovnice. Taj se unos normalizira kako se igrač ne bi brže micao ako hoda dijagonalno naspram samo po X ili samo po Y osi. Taj se normalizirani vektor množi sa brzinom kretanja kako bi se dobila željena brzina.

1.2. Sudarači igrača

Igraču su dodijeljena 2 sudarača. Jedan je trigger, a drugi nije. Trigger Collider služi za detektiranje sudaranja sa neprijateljima i njihovim projektilima izbjegavajući međudjelovanje njihovih sila. Drugi sudarač je ograničene veličine. On služi boljoj vizualizaciji i računanju kolizije sa okolinom.

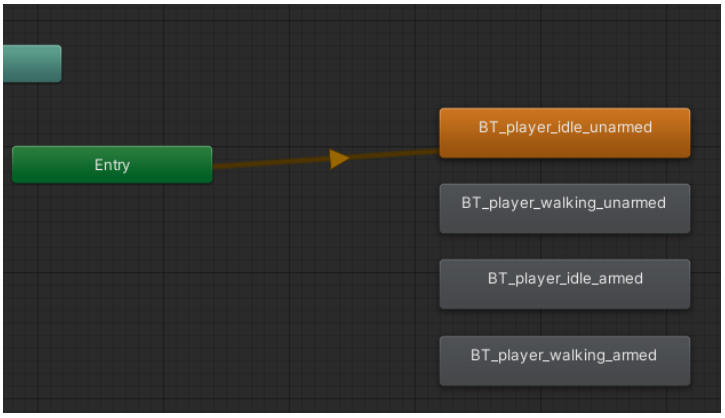
Slika 1: Obrub sudarača koji nije okidač

Slika 2: Obrub sudarača koji je okidač

1.3. Animacije glavnog lika

Izvor informacija: [4]

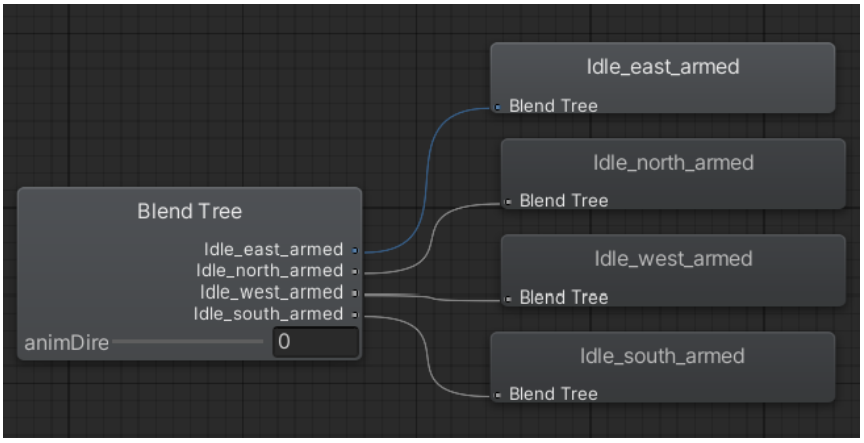
Činjenica da 4 Blend Tree-a na priloženoj slici nisu povezana možda izgleda začuđujuće. No, njihovim izvađanjem se upravlja u kodu. Kada želimo pokrenuti neki drugi Blend Tree jednostavno pozovemo metodu za promjenu.



Slika 3: Animator komponenta igrača

1.3.1. Rotacija igrača

Pod rotacija igrača ne odnosim se na rotaciju igrača oko Z osi, nego na promjene usmjerenja igrača. Animacije lika se dijele na animacije u slučaju da lik nosi oružje i da lik ne nosi oružje. Oba slučaja su pokrivena i omogućena, no zbog trenutnog stanja igrice igrač je uvijek naoružan.



Animiranje rotira lika u smjerove sjever, jug, istok ili zapad ovisno o poziciji kursora u odnosu na igrača kada je naoružan. Rotira ga ovisno o smjeru kretanja kada nije, no u trenutnoj implementaciji video igre igrač je uvijek naoružan. Na slici je vidljivo kako funkcionira Blend Tree. Ovisno o parametru animDir mijenja se animacija koja se trenutno izvodi. Svaki Blend Tree ima 4 mogućnosti vezano uz rotaciju igrača. Ti Blend Tree-evi s jednodimenzionalni. Odnosno, mijenjaju animaciju na temelju 1 parametra.

1.3.2. Kod koji upravlja animacijom igrača

```
void HandleAnimation()
 {
     if (playerHasGun)
     {
         mouseDirection = Camera.main.ScreenToWorldPoint(Input.mousePosition)
- transform.position;
         // vector of direction from camera to mouse
         // - transform.transition alters it so it is not from the camera but
from the object
```

Slika 4: Blend Tree za stojećeg oružanog igrača

```
         // that this script belongs to

         mouseAngle = Mathf.Atan2(mouseDirection.y, mouseDirection.x) *
Mathf.Rad2Deg;
         // angle between vector origin and destination

         animDir = CalculateAnimDir();

     }
     else
     {
         if(inputVert == 1)
         {
             animDir = 2; // NORTH
```

```
        }
        else if(inputVert == -1)
        {
            animDir = 4; // SOUTH
        }
        else if(inputHor == 1)
        {
            animDir = 1; // EAST
        }
        else if(inputHor == -1)
        {
            animDir = 3; // WEST
        }

    }

    anim.SetFloat("animDirection", animDir);

    ChangeAnim(); // change animation from idle to walking
}

int CalculateAnimDir()
{

    if (mouseAngle > -45 && mouseAngle < 45)
    {
        GunPosition.transform.localPosition = gunPositionEast;

        return 1; // EAST
    }
    else if (mouseAngle >= 45 && mouseAngle <= 135)
    {
        GunPosition.transform.localPosition = gunPositionNorth;

        return 2; // NORTH
    }
    else if (mouseAngle < -135 || mouseAngle > 135)
    {
        GunPosition.transform.localPosition = gunPositionWest;

        return 3; // WEST
    }
    else if (mouseAngle >= -135 && mouseAngle <= -45)
    {
        GunPosition.transform.localPosition = gunPositionSouth;

        return 4; // SOUTH
    }
    else
    {
        return 0;
    }
}

void ChangeAnim()
{
    if (playerHasGun)
    {
        if (inputHor != 0 || inputVert != 0)
        {
            anim.Play("BT_player_walking_armed");
        }
        else
```

```
            {
                anim.Play("BT_player_idle_armed");
            }
        } else
        {
            if (inputHor != 0 || inputVert != 0)
            {
                anim.Play("BT_player_walking_unarmed");
            }
            else
            {
                anim.Play("BT_player_idle_unarmed");
            }
        }

    }
```

HandleAnimation metoda računa kut između igrača i kursora ili smjera kretanja ako igrač nije naoružan. Parametar animDirection se postavlja ovisno o potrebnom smjeru rotacije.

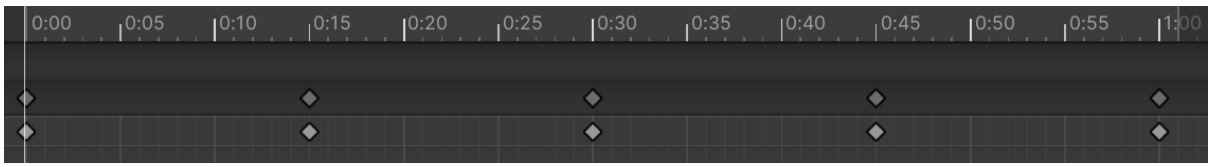
CalculateAnimDir metoda pretvara kut između igrača i kursora. Zbog inherentnog načina računanja kuta, njegove vrijednosti se nalaze u rasponu <-180, 180]. Zbog toga, malo je škakljivo preračunati kut između igrača i kursora i stoga treba koristiti dosta if/else if blokova. Vrijednosti se ovako postavljaju:

- Sjever: [45, 135], vrijednost 2
- Jug: [-135, -45], vrijednost 4
- Istok: <-45, 0> U <0, 45>, vrijednost 1
- Zapad: <-180, -135 U <135, 180>, vrijednost 3

ChangeAnim metoda mijenja Blend Tree koji se trenutno izvodi. Ovaj način mi je osobno bio pristupačniji od promjene tekućeg Blend Tree-a pomoću Unity Animator-a.

1.3.3. Animacije igrača

Animacije igrača kada se on kreće se sastoje od niza sličica koje se u određenom intervalu mijenjaju. Kod animacija gdje igrač stoji samo se neprestano prikazuje ista slika.



Slika 5: Vrijeme promjene sličica igrača

Kao što je vidljivo, jedna animacija traje 1 sekundu. Svakih 15 sličica se promijeni sličica igrača. Uz to, mijenja se i pozicija objekta GunPosition koja definira poziciju oružja u rukama igrača.

1.4. Primanje štete

Igrač primi štetu kada se sudari sa neprijateljem ili projektilom kojeg je neprijatelj lansirao. Neprijatelji i projektili pozivaju funkciju TakeDamage koja se nalazi u skripti igrača. Oni joj prosljeđuju parametar damage koji se odnosi na količinu štete koji igrač treba primiti.

Kod metode je:

```
public void TakeDamage(int damage)
 {
     //Debug.Log("took damage");
     if(timeOfNextHit <= Time.time) // Immunity frames
     {
         audioSource.Play();
         flashEffect.Flash();

         //Debug.Log("took " + damage);
         timeOfNextHit = Time.time + timeBetweenHits;

         lifePoints -= damage;
         if(lifePoints <= 0)
         {
             SceneManager.LoadScene(3); // YouLoseMenu
             Destroy(gameObject);
         }
     }
     else
     {
         //Debug.Log("Hit evaded");
     }

 }
```

Igraču su dodijeljeni trenuci imuniteta kako u slučaju preplavljenosti neprijateljima ne bi pre brzo gubio životne bodove. Kada igrač primi štetu počne svirati zvuk za primanje štete. To je odnosi na pozivanje metode Play komponente igrača audioSource. Osim toga, pozove se metoda Flash komponente flashEffect. Nadalje, igraču se oduzmu životni bodovi. U slučaju da je količina životnih bodova jednaka ili manja od 0, učita se scena gubitka.

1.4.1. Efekt izbjeljivanja igrača

Efekt izbjeljivanja igrača nije ništa više nego zabjeljivanje sličice igrača u trenutku primanja štete. On koristi skriptu SimpleFlash koja je dodana objektu igrača kao komponenta.

Izvor koda: [5]

Materijal koji se dodijeli sličici unutar SpriteRenderer komponentae igrača u trenutku izbjeljivanja je GUI/Text Shader bijele boje. U mom je projektu složeno da taj efekt traje 0,15 sekundi.

2. Kamera

Kamera prati igrača. Igrica se temelji na ovoj funkcionalnost. Ona je ostvarena pomoću sljedećeg koda, sa izvorom [2]:

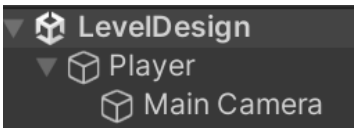
```
public class CameraFollowsPlayer : MonoBehaviour
{
    [SerializeField] GameObject playerObject;
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void LateUpdate()
    {
        transform.position =
        playerObject.transform.position +
        new Vector3(0f, 0f, -1f);
    }

}
```

LateUpdate je metoda koja se pozove nakon poziva svih drugih Update metoda. LateUpdate se koristi kod ove skripte kako bi praćenje igrača bilo glatko. Glavna kamera je postavljena kao dijete objekta Player, kao što je pokazano na slici.



Slika 6: Kamera kao dijete objekta igrača

3. Oružja

Izvori informacija: [1], [6], [7].

Izvori sprite-ova: [8]

Igrač može posjedovati 3 različita oružja. AK-47, sačmaricu i Glock. Svako od tih oružja definirano je kao predložak.

Slika 7: Predložak puške AK-47

3.1. GunPosition objekt

GunPosition je dijete objekta igrača. Kao takav objekt ono ima lokalnu poziciju. Pozicija mu se mijenja pri kretanju igrača. Osim toga, pozicija mu se mijenja s obzirom na rotaciju igrača, s tim da su mu pozicije s obzirom na rotaciju:

- Istok: X: 0.142, Y: 0.11
- Sjever: X: 0.096, Y: 0.179
- Zapad: X: -0.142 Y: 0.11
- Jug: X: -0.128, Y: -0.083

Ta je pozicija definirana u animacijama igrača. Predlošci za sve puške dodani su kao djeca objekta GunPosition.

Skripta MultipleGunsController dodjeljena je ovom objektu. Njezino je zaduženje upravljanju nad više oružja, mijenjajući ih na desni klik miša. Kod ove skripte je sljedeći:

```
public class MultipleGunsController : MonoBehaviour
{
    int totalGuns;
    public int currentGunIndex;

    public GameObject[] guns;
    public GameObject gunHolder;
    public GameObject currentGun;

    private float timeOfNextSwap;
    private float timeBetweenSwaps;

    // Start is called before the first frame update
    void Start()
    {
        totalGuns = gunHolder.transform.childCount;
        guns = new GameObject[totalGuns];

        for (int i = 0; i < totalGuns; i++)
        {
            guns[i] = gunHolder.transform.GetChild(i).gameObject;
            guns[i].SetActive(false);
        }

        guns[0].SetActive(true);

        currentGun = guns[0];
        currentGunIndex = 0;


        timeOfNextSwap = 0;
        timeBetweenSwaps = 0.2f;

    }

    // Update is called once per frame
    void Update()
    {
        if(Input.GetKey(KeyCode.Mouse1) && timeOfNextSwap <= Time.time)
        {
            timeOfNextSwap = Time.time + timeBetweenSwaps;

            SetCurrentGunInactive();
```

```
            currentGunIndex++;
            currentGunIndex %= totalGuns;
            SetCurrentGunActive();
            currentGun = guns[currentGunIndex];
        }

    }

    public void SetCurrentGunActive()
    {
        guns[currentGunIndex].SetActive(true);
    }


    public void SetCurrentGunInactive()
    {
        guns[currentGunIndex].SetActive(false);
    }
}
```

Pri pokretanju skripte ona izračuna broj oružja koje igrač trenutno posjeduje, sve ih deaktivira osim prve u nizu. Postoji minimalno vrijeme između promjene oružja. Oružje se mijenja pomoću pritiska miša. Zbog toga se može dogoditi više nepoželjnih pritisaka i to se sprječava pomoću varijable timeBetweenSwaps.

Metoda Update mijenja trenutno korišteno oružje na sljedeće u nizu.

Metode SetCurrentGunActive i SetCurrentGunInactive su referencirane 3 puta, svaka. Obje u ovoj metodi i obje kod pauziranja igrice. Kada se igrica pauzira trenutno aktivno oružje se onemogući kako igrač ne bi mogao pucati dok je igrica pauzirana.

3.2. Predlošci oružja

Predlošci definiraju oružja. Njihove komponente su skripta GunController, skripta ShootSoundController i Audio Source. Osim toga, kao djecu ima objekte SpriteSheet i ShootPoint.

Isto je za sva oružja, osim što su parametri skripte GunController različite ovisno o oružju. Osim toga, skripte su im drugačije. Nadalje, točke okreta (pivot) i pozicije ispaljivanja metaka su im na drugačiji pozicijama. Uz to, Audio Source sadrži različite zvukove ispaljivanja, ovisno o oružju.

3.3. GunController skripta

Svrha ove skripte je upravljanje ponašanjem oružja. To se odnosi na rotaciju u smjeru kursora igrača, sortiranje oružja ispred/iza igrača u svrhu pravilnog animiranja i pucanje. Formula za računanje te rotacije je dobivena sa Unity foruma [9].

Većina koda ove skripte je dijeljeno između oružja. No, projektili za svako oružje je drugačije. Osim toga, sačmarica ispucava 7 metaka u nasumičnom kutu u rasponu od [-9, 9] stupnjeva od rotacije točke ispucavanja.

Animiranje pušaka se odnosi na rotaciju prema kursoru.

Promjena sloja prikazivanja se radi prema parametru animDirection koji se nalazi u animatoru igrača.

Pucanje se odnosi na instanciranje metaka i brojanje koliko metaka je još dostupno. Instanciranje metaka se radi na poziciji objekta ShootPoint. Nakon pucanja postoji vrijeme kada je ono onemogućeno. To vrijeme je individualno za svaku pušku.

3.4. Metci

Postoji 3 drugačija metka za 3 drugačija oružja. Svaki metak je jedan predložak. Svaki predložak sadrži komponente Box Collider 2D i skriptu Projectile Controller. Uz to, kao dijete sadrži objekt sa Sprite Renderer komponentom.

Svaki metak definira svoje zasebne parametre putem ProjectileController skripte. Parametri su brzina, vrijeme leta (vrijeme prije nestajanja) i šteta koju radi neprijateljima.

Izvor koda za skripte: [1]

Taj metak se uništi kada se sudari sa neprijateljem ili kad mu istekne vrijeme leta.

4. Neprijatelji

Postoje 3 neprijatelja: Goat, Woodmother i Red Smile. Goat je neprijatelj koji trči prema i graču i zadaje mu štetu kada se njihovi sudarači dodiruju. Wodmother je neprijatelj koji generira Goat neprijatelje. Red Smile je neprijatelj koji je vrlo brz. On dotrči do igrača i na njega baca vatrene projektile. Svi su neprijatelji definirani predlošcima. Pri generiranju neprijatelja na sceni samo se pozove metoda Instantiate sa referencom na predložak neprijatelja.

Inspiracija i izvor dijela koda za skripte neprijatelja i projektila: [1]

4.1. Predlošci neprijetelja

Predlošci svih neprijatelja su vrlo slični. Oni sadrže Sprite Renderer, SimpleFlash skriptu, sudarač koji je okidač, sudarač koji nije okidač, skriptu upravljača neprijateljem, Animator, 3 skripte za pathfinding i Audio Source.

2 sudarača omogućuju pravilnu interakciju sa igračem i mecima dok također izbjegava preklapanje neprijatelja.

Sprite Renderer komponenta mijenja sličice pri animaciji neprijatelja.

SimpleFlash skripta radi kao i kod igrača – promijeni boju sličice kada neprijatelj primi štetu.

Audio Source omogućava puštanje zvukova, a kod neprijatelja je to zvuk na primanje štete.

4.2. EnemyController skripta

Ova skripta kontrolira više stvari kod neprijatelja: primanje i davanje štete, animaciju, dodjeljivanje bodova igraču kada ubije neprijatelja i zvukove. Ne kontrolira kretanje jer se time bave pathfinding skripte.

Svaka skripta neprijatelja implementira ovu skriptu. To će biti vrlo korisno kod generiranja valova neprijatelja.

Ova skripta definira InheretedStart metodu. To je zato jer je u C# programskom jeziku za Unity malo komplicirano pozvati Start metodu naslijeđene klase. Ovo je jednostavniji način. Ova metoda inicijalizira varijable.

Metoda TakeDamage pozvana je u skripti metaka igrača. Pri primanju štete neprijateljima se oduzmu životni bodovi, pusti se zvuk primanja štete, sličica neprijatelja pocrveni i u slučaju smrti neprijatelji igraču dodjeljuju bodove koje mogu koristiti za kupovanje metaka. Odmah pri pozivanju te metode zastavica isHit je postavi na true. Svrha toga je da svaki neprijatelj ne može primiti više od 1 metka u jednom trenutku. U stvarnosti, neprijatelji znaju primiti više od jednog metka u jednom trenutku iz sačmarice, što daje više bodova igraču za istog poraženog neprijatelja.

Metoda GiveSouls dodaje igraču bodove s kojima može kupovati metke.

OnTriggerStay2D je metoda koja se okida kada god se sudarači igrača i neprijatelja dodiruju [10]. Ona poziva metodu TakeDamage koja je dodijeljena neprijatelju.

4.3. GoatController skripta

Izvor sprite-ova: [11]

Ova skripta animira ovog neprijatelja. Prije ga je i pomicala, no to sada rade pathfinding skripte. Animacija ovog neprijatelja temeljena je na promjeni sličice pri promjeni smjera. To se radi u 2D prostoru i stoga su Blend Tree-evi tipa 2D. Parametri koji upravljaju njime se postavljaju u ovoj skripti.

4.4. WoodmotherController skripta

Izvor sprite-ova: [12]

Specifičnost ove skripte je što sadrži logiku za stvaranje Goat neprijatelja. On stvara te neprijatelje samo dok je na mjestu. Kada ih stvara odigra se animacija gdje ovaj neprijatelj podigne nogu i zabijeli.

Animacija ovog neprijatelja je jednodimenzionalna – temeljena samo na Y osi.

4.5. RedSmileController skripta

Izvor sprite-ova: [13]

Specifičnost ove skripte je što omogućuje da ovaj neprijatelj puca projektile. Smjer projektila je razlika između trenutne pozicije neprijatelja i igrača. Animacije ovog lika također su jednodimenzionalne i temeljene na Y osi. Uz to, pri micanju ovan neprijatelj ima posebnu animaciju, kao i kod pucanja.

4.6. RedSmileProjectileController skripta

Ova skripta vrlo je slična skripti metka za igračeva oružja. No, ova skripta definira nanos štete igraču, a ne neprijateljima.

4.7. Kolizija neprijatelja i igrača

Layer Collision Matrix nisam mijenjao. Umjesto toga, dodavao sam ignorirane Layer-e kolizije direktno na sudarače objekata. To sam napravio kako bih još uvijek mogao detektirati međuodnos između sudarača okidača, dok bih ignorirao koliziju sudarača koji nisu okidači.

Sudarač koji nije okidač objekta neprijatelja Goat ignorira slojeve Player i Woodmother. To znači da sudarač okidač neprijatelj tipa Goat još uvijek može biti u međuodnosu sa sudaračem igrača. Dodatno, to znači da se više neprijatelja tipa Goat neće međusobno preklapati na ekranu.

5. Valovi neprijatelja

Izvor inspiracije i koda: [1]

Svaki val ima određene karakteristike. Oni svi dijele točke stvaranja i vrijeme između valova. Točke stvaranja su objekti sa transform komponentom koji definiraju poziciju stvaranja neprijatelja. Nadalje, svaki val ima vrste neprijatelja koji se mogu stvoriti. Nadalje, postoji vrijeme između stvaranja neprijatelja. Uz to, postoji broj neprijatelja koji se stvara svaki val. Može se definirati koliko god valova želimo.

Skripta EnemyWaves sadrži logiku stvaranja valova. Ona je dodana kao komponenta istoimenog objekta. Taj objekt kao svoju djecu sadrži pozicije stvaranja neprijatelja u svrhu bolje organizacije hijerarhije objekata.

6. Pathfinding algoritam

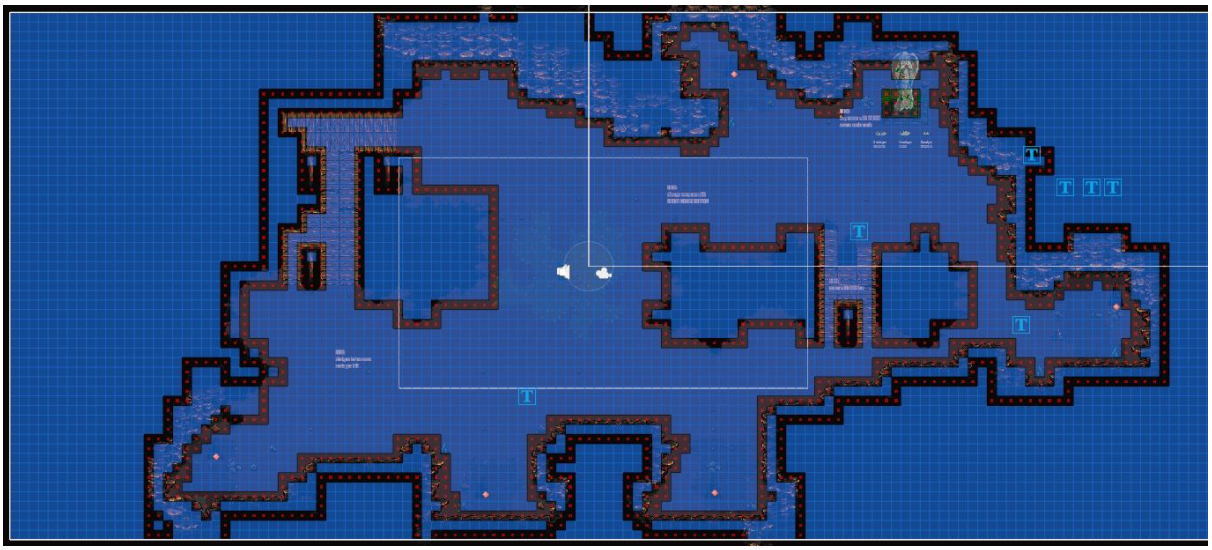
U svrhu ovog projekta koristio sam A* pathfinding algoritam. To je algoritam koji je opće priznat kao najbolji u svrhu upravljanja kretanjem objekata u 2D Unity igrici. Unity Navmesh nije dostupan u 2D prostoru.

Izvor informacija: [14]

Izvor koda: [15]

Ovaj izvor koda je općepoznati paket za implementaciju A* pathfinding algoritma za Unity. Ova implementacija pathfinding algoritma zahtjeva definiranje slojeva (Layer) s kojima se objekti koje pomičemo mogu sudariti. Nadalje, treba definirati kockastu mrežu koja će predstavljati čvorove za kretanje. Nakon toga, na tu mrežu se izračunaju čvorovi koji su dostupni (ne dodiruju se sa sudaračima).

Za svaki objekt pojedinačno treba definirati točku prema kojoj se trebaju micati (najčešće igrač), brzinu kojom se kreću, udaljenost od koje krenu usporavati, veličinu njihovog sudarača, udaljenost od točke na kojoj staju, koliko čvorova mogu preskočiti pri povezivanju i slično.



Slika 8: Izgled moje finalne mreže čvorova

Na slici gore je prikazana ta mreža čvorova. Crvene točkice prikazuju čvorove koji su nedostupni, a plavi kvadratići dostupne. Ako je tako namješteno, neprijatelj može preskočiti jedan ili više nedostupan čvor kako bi došao do dostupnog. U mom slučaju takvo bi kretanje stvaralo greške.

Nakon što se na objekte doda funkcionalnost pathfinding-a objektima on odrađuje cijelo kretanje. Odnosno, ako je do tada već implementirana funkcionalnost kretanja sada se treba oduzeti iz koda.

7. Izrada igraćeg područja

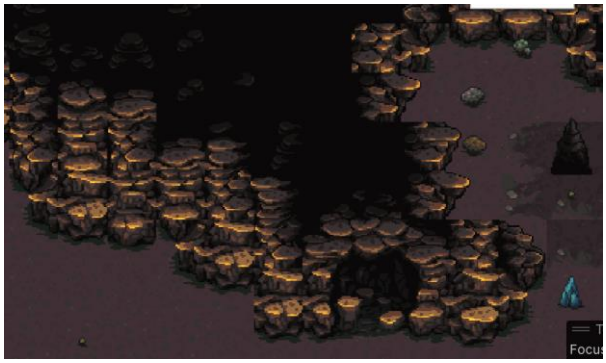
Izvor sprite-ova: [16]

Izvori informacija: [17], [18], [19]

Zamislio sam da je igraće područje ove igrice nalik na pakao. Uspio sam pronaći asset pack koji liči na špilju, što nije daleko u tematici. Igraće područje sadrži 3 nepremostiva ponora. Oko tih ponora igrač hoda/trči i bori se sa neprijateljima. Neprijatelji se stvaraju na područjima gdje su crni kameni.

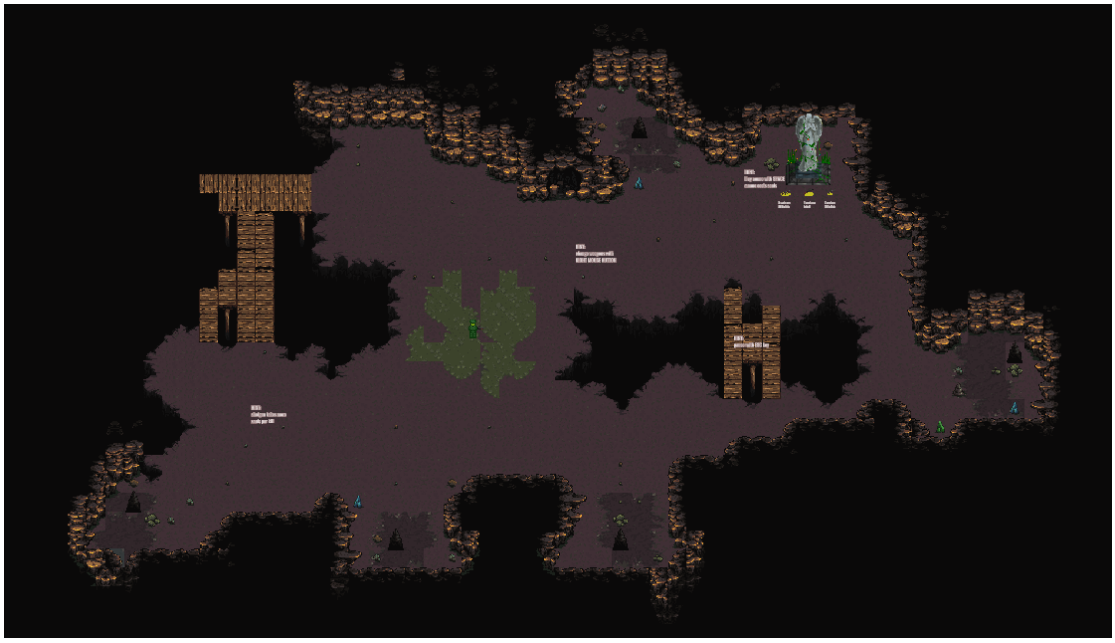
U gornjem desnom kutu na igraćem području nalazi se statua anđela. Na njezinom dnu igrač može kupovati metke za svoja oružja koristeći prikupljene bodove (duše) od neprijatelja.

Pri izgradnji igraćeg područja koristio sam Tile Pallete prozor. U njemu sam definirao novu paletu. Tu sam paletu koristio na 3 drugačija Tilemap-a. Jedan je za područja na kojima se može hodati, jedan kao pozadina područjima na kojima se može hodati i jedan za područja s kojima se igrač i neprijatelji sudaraju. Kasnije mi je taj zadnji sloj bio koristan kod definiranja sloja za koliziju kod A* algoritma.



Slika 11: Detalj igraćeg područja 1

Slika 10: Detalj igraćeg područja 2



Slika 9: Pregled igraćeg područja

8. Dodatne scene

Izvori slika: [20], [21], [22]

Izvor informacija: [23]

Osim igraće scene, bilo je potrebno dodati 3 scene. Prva za početni izbornik pri pokretanju video igre. Drugi za scenu pri gubitku igrice. Treći za scenu pri pobjedi igrice.

8.1. Početni zaslon

Slika 12: Slika početnog zaslona

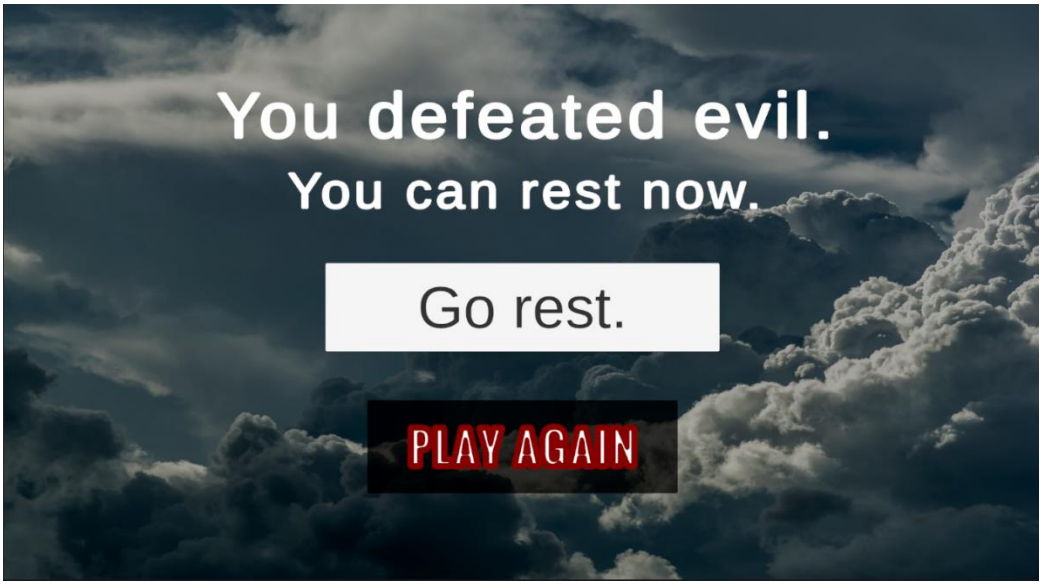
Početni zaslon ima 3 opcije: pokreni igricu, postavi opcije i izađi iz igrice. To su sve elementi tipa Button – TextMeshPro. U njima se nalaze objekti tipa Text – TextMeshPro. TextMeshPro je zapravo ekstenzija za dekoriranje teksta u Unity-u koja je nedavno stopljena sa programom.

Pritisak na prvu tipku predstavlja pokretanje igrice. To je ostvareno pomoću naredbe definirane u StartMenuController skripti:
`SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);`

Pritisak na drugu tipku otvara opcije. U opcijama je moguće postavljati glasnoću. Glasnoća se mijenja tako da se mijenja glasnoća glavnog Audio Mixera. On je pridodan svakoj Audio Source komponenti koja se pojavljuje u igrici.

Pritisak na treću tipku zatvara igricu pomoću naredbe: `Application.Quit();`

8.2. Zaslon pri pobjedi



Slika 13: Zaslon pri pobjedi

Zaslon pri pobjedi pruža 2 opcije: zatvori igricu i igraj ponovo.

8.3. Zaslon pri gubitku

Slika 14: Zaslon pri gubitku

Ovaj zaslon sadrži samo 1 opciju: igraj ponovno.

9. Elementi korisničkog sučelja

9.1. Pauziranje igrice

Izvor informacija: [24]

Pauziranje igrice daje mogućnosti nastavljanja sa igrom, opcije i povratan ka početni zaslon. U opcijama je moguće namjestiti glasnoću. Uz to, pozadina se potamni kada se pauzira igrica. Skripte koje time upravljaju su PauseMenuController i PauseOptionsMenu.

Kada se igrica igra i pritisne se gumb Esc aktivira se ekran pri pauziranju. Parametar klase Time timeScale se postavi na 0. Skripta igrača se onemogući, kao i trenutna puška koju drži u rukama.

Nakon ponovnog pritiska gumba Esc ili pritiska na RESUME igrica se ponovno nastavlja izvoditi.

Options menu omogućuje pojačavanje i stišavanje glazbe.

9.2. Pomoćni tekst

Na igraćoj površini se nalazi svega 7 tekstualnih objekata na podu. Oni pomažu igraću i vode ga kroz mehaniku igranja.

9.3. Metci dostupni za oružje

Metci dostupni za oružje su navedeni u donjem lijevom kutu ekrana. Pri pokretanju igrice te se vrijednosti odmah postavljaju. Pri igračevom pucanju te se vrijednosti prilagođavaju.

Slika 15: Prikaz koliko je metaka ostalo igraču

Logika tog sustava sadržaja ne u skripti AmmoCountController. GunController skripta poziva metode za oduzimanje metaka, a WeaponsOnGround skripta poziva metode za dodavanje metaka.

9.4. Količina životnih bodova

Izvor informacije: [25], [26]

Količina životnih bodova prikazana je kao krug oko plave vatre, koja simbolizira dušu. Kada se krug isprazni igrač gubi.

Skripta HealthBarCntroller također mijenja boju iz plave u sivu dok igrač prima štetu. Uz to, pri primanju štete taj se krug polako smanjuje, umjesto da skoči na potrebnu vrijednost.

9.5. Broj prikupljenih bodova

Ovi bodovi se mogu koristiti u trgovini metaka. Oni se prikupljaju poražavanjem neprijatelja. Svaki neprijatelj daje određenu količinu ovakvih bodova.

Slika 16: Broj prikupljenih bodova od neprijatelja

9.6. Trgovina mecima

U ovoj trgovini je moguće kupovati metke u zamjenu za prikupljene bodove. Kada igrač pritisne tipku Enter on kupi neke metke ako se njegov sudarač sudara sa sudaračima pušaka na podu.

Slika 17: Sudarači trgovine

Logikom kupovanja nosi se skripta WaponsOnGround koja je komponenta svakog od ovih objekata. Ona osigurava da se municija ne može pre brzo kupovati. Anđeo stoji kao ukras.

10. Zvučni efekti i muzika

Izvori muzike: [27], [28], [29], [30], [31]

Izvor zvučnih efekata: [32], [33], [34], [35]

Izvor informacija: [36]

10.1. Muzika

Izvor muzike dodan je kao objekt na sve scene. Parametar komponente Audio Source Spatial Blend postavljen je na potpuni 2D – što e značiti da će muzika uvijek svirati, bez obzira na poziciju igrača.

Na svaku scenu je dodana drugačija pjesma. Na scenu glavne razine dodan je objekt sa skriptom PlaySongs. Ona pokrene izvađanje pjesme i odgodi pokretanje izvađanja sljedeće za duljinu izvađanja tekuće. Ona sadrži listu pjesama koja može biti proizvoljno velika. U mojoj igrici sadrži 2 zvučne datoteke.

10.2. Zvučni efekti

Igrač, neprijatelji i oružja sadrže komponentu Audio Source. Igračev Audio Source se izvede svaki put kada primi štetu. Audio Source neprijatelja se izvede svaki put kada primi štetu. Audio Source oružja se izvede svaki put kada ono opali.

To se ostvaruje tako da se ta komponenta referencira u skripti i nad njom se pozove metoda Play().

Popis literature

[1] M. Konecki, „Tamnica“.

[2] M. Konecki, „Transporter“.

[3] „RPG Asset Character ‚Soldier‘ SMS by chasersgaming“, itch.io. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://chasersgaming.itch.io/rpg-asset-character-soldier-sms

[4] *How to Setup Blend Trees for Easy 4 Directional Movement in Unity 2022*, (2022.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=iFe_jCkMEV0

[5] *Sprite Blink/Flash on Impact in 3 Minutes - Unity Tutorial*, (2021.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=9rZkiEyS66I

[6] *Equip and Switch Multiple Weapons in Unity 2D | Weapon System Unity 2D*, (2020.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=-YISSX16NwE

[7] *How to make 2D Shooting in UNITY with Gun Recoil - Unity 2D*, (2020.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=NKF-FkDzE-s

[8] „Pixel art military weapon set |“. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://www.gamedevmarket.net/asset/pixel-art-military-weapon-set

[9] „Getting the angle between 2 objects“, Unity Forum. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://forum.unity.com/threads/getting-the-angle-between-2-objects.13785/

[10] *Unity3D Tutorial: How to Move a Rigidbody Towards a Target*, (2023.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=UqgVVB-sg9o

[11] Jason, „Time Fantasy“, Time Fantasy. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://finalbossblues.com/timefantasy/

[12] „Woodcrawler |“. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://www.gamedevmarket.net/asset/woodcrawler

[13] „Red Smile by yggDrazzil“, itch.io. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://yggdrazzil.itch.io/red-smile

[14] *How to setup PATHFINDING AI in UNITY (Topdown Shooter Game)*, (2020.). Pristupljeno: 27. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=63zWZ2rJZ68

[15] „A* Pathfinding Project“. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://arongranberg.com/astar/

[16] „RPG Worlds Caves |“. Pristupljeno: 27. siječanj 2024. [Na internetu]. Dostupno na: https://www.gamedevmarket.net/asset/rpg-worlds-caves

[17] *Creating Tilemaps For Your 2D Game in Unity 2021 - Tutorial*, (2021.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=DTp5zi8_u1U

[18] *Adding Collisions For Your Tilemaps and 2D Character - Unity Tutorial*, (2022.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=2obtqmBHUW4

[19] *Unity 2D - RPG Tutorial 2023 - Part 01 Adding Background*, (2022.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=Ts9JzLo6zII

[20] „4.3 million+ Stunning Free Images to Use Anywhere - Pixabay - Pixabay“. Pristupljeno: 28. siječanj 2024. [Na internetu]. Dostupno na: https://pixabay.com/

[21] „Hell Burning Fire - Free photo on Pixabay“. Pristupljeno: 28. siječanj 2024. [Na internetu]. Dostupno na: https://pixabay.com/photos/hell-burning-fire-path-digital-21079/
[22] „Photo by Pixabay on Pexels“, Pexels. Pristupljeno: 28. siječanj 2024. [Na internetu]. Dostupno na: https://www.pexels.com/photo/bonfire-wallpaper-266388/
[23] *START MENU in Unity*, (2017.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=zc8ac_qUXQY
[24] *PAUSE MENU in Unity*, (2017.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=JivuXdrIHK0
[25] *Three Cool Health Bars in Unity (2022/2020)*, (2020.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=ZzkIn41DFFo
[26] „Change an Sprite Instance Color Palette.“, Unity Forum. Pristupljeno: 28. siječanj 2024. [Na internetu]. Dostupno na: https://forum.unity.com/threads/change-an-sprite-instance-color-palette.212134/
[27] *Mick Gordon - Doom 2016: Menu theme (HQ, file rip)*, (2016.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=bI5l43cljBM
[28] *Doom OST - Victory Music*, (2007.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=kX3IA3Nop3o
[29] *Left 4 dead (left 4 death) death music*, (2019.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=soKrGy0bhHI
[30] *Mick Gordon - 02. Rip & Tear*, (2016.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=zZMg9ryeWOw
[31] *Mick Gordon - 08. Flesh & Metal*, (2016.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=VYVFXYEm_Mw
[32] „90,000+ Free Sound Effects for Download - Pixabay - Pixabay“. Pristupljeno: 28. siječanj 2024. [Na internetu]. Dostupno na: https://pixabay.com/sound-effects/
[33] *Shotgun shot sound effect*, (2012.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=US_Mr0xrKU4
[34] *Glock 19 Sound Effect [FREE]*, (2021.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=ENvln5SG3ro
[35] *AK 47 Assault Rifle- Sound Effect Pack (HQ)*, (2020.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=D2TG6kNEkDE
[36] *Sound Effects & Background Music | Build a 2D Platformer Game in Unity #11*, (2021.). Pristupljeno: 28. siječanj 2024. [Na internetu Video]. Dostupno na: https://www.youtube.com/watch?v=J77CMuAwVDY