DELTA – Střední škola informatiky a ekonomie, s.r.o.
Ke Kamenci 151, PARDUBICE

MATURITNÍ PROJEKT

KOMPLEXNÍ HORROROVÁ HRA

Příjmení, jméno: PETERA Martin
Třída: 4. A
Studijní obor: Informační technologie 18-20-M/01
Školní rok: 2022/2023

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení: *Martin Petera*

Pro školní rok: *2022/2023*

Třída: *4. A*

Obor: *Informační technologie 18-20-M/01*

Téma práce: *Komplexní hororová hra*

Vedoucí práce: *Mgr. Jan Mottl*

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Cílem projektu bude tvorba 3D hry s hororovým žánrem. Hra bude mít vytvořen příběh, který bude hráče provázet a zajišťovat jeho postup. Ve hře budou implementované herní prvky jako animovaná videa (cutscény), dialogy a další. Hra bude mít vytvořenou jednu komplexní úroveň s tím, že bude navržený storyboard pro případné rozšíření. Projekt bude tvořen v herním enginu Unity, skripty budou psané v jazyce C#.

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

Září – rešerše kolem her s podobnou tématikou, základní návrh příběhu

Říjen – naprogramování základních mechanik hry (pohyb, interakce s předměty atd.), dokončení storyboardu pro jednu úroveň

Listopad – práce na úrovni hry a implementace dalších rozšiřujících mechanik (dialogy, cutscény, speciální mechaniky)

Prosinec – dokončení hrubé verze hry, dokončení storyboardu pro zbytek hry

Leden – dodělání detailů, případné rozšíření stávající verze hry

Únor – testování hry a případné opravy chyb, začátek práce na dokumentaci projektu

Březen – tvorba dokumentace projektu

Prohlašuji, že jsem maturitní projekt vypracoval samostatně, výhradně s použitím uvedené literatury.

V Pardubicích 31.3.2023

Poděkování

Děkuji Mgr. Janu Mottlovi za studijní materiál a odborné vedení při zpracovávání maturitního projektu. Byl jste skvělým mentorem, vaše rady a nápady mi pomohly rozvinout projekt správným směrem. Dále děkuji Matějovi Beranovi za poskytnutí audio nahrávky použité ve vítězné obrazovce hry. Nakonec bych chtěl poděkovat rodině a přátelům za morální podporu během celého projektu. Bez vaší podpory by to nebylo možné. Děkuji Vám všem!

Anotace

Tato dokumentace přináší náhled do vývoje 3D her v herním enginu Unity. Dokumentace přináší vhled do jednotlivých kapitol tvorby, kde jsou popsány doporučené kroky, při takové tvorbě. Jako výstup této práce je 3D hororová hra s názvem „It Hunts“.

Klíčová slova

C#, Unity Engine, 3D, hororová hra, survival

Annotation

This documentation provides insight into the development of 3D games in the Unity game engine. The documentation provides an insight into the different chapters of development, describing all the steps involved in such development. The output of this work is a 3D horror game called "It Hunts".

Keywords

C#, Unity Engine, 3D, horror game, survival

Obsah

1. Úvod

Mnoho hráčů počítačových her se jistě někdy zamyslelo nad tím, jak taková hra vzniká. Jedná se o mnohem složitější proces, než se na první pohled může zdát. Je nutné zvážit mnoho faktorů. Mezi ty nejdůležitější patří originalita konceptu, atraktivita pro cílovou skupinu, použité zařízení a způsob ovládání. Pouze s těmito informacemi můžeme přistoupit k tvorbě grafiky a programování. Sám jsem hráčem počítačových her, proto jsem si tuto otázku položil a navzdory všem nástrahám jsem se do tvorby hry pustil. Rozhodl jsem se v rámci svého projektu vytvořit 3D komplexní hru s návrhem vlastní grafiky.

Cílem projektu je vytvořit 3D hru v herním enginu Unity a teoretický vhled do tématiky takové tvorby, který může složit jako návod pro další tvůrce. Praktickým výsledkem práce je 3D hra s hororovou tématikou, na které budu demonstrovat své teoretické znalosti. Od původního zadání vytvořit příběhovou hru bylo odchýleno, a nakonec bylo navrženo a schváleno vytvoření hry s arkádovým stylem.

Inspiraci svého projektu jsem vzal z populární hry Slender: The Eight Pages [1], kde musíte procházet herní svět, ve kterém hledáte a sbíráte papírky, abyste vyhráli. Ve hře se mi líbí volný průchod mapou a moment překvapení, když narazíte na nepřítele, proti kterému, bohužel nemáte šanci bojovat. Proto jsem se rozhodl do svojí hry implementovat mechaniku, která by to umožnila.

Do hry jsem zapracoval menu s výběrem tutoriálu nebo hry samotné. V menu je dále nastavení pro hlasitost, protože zvuk je důležitou součástí hry samotné. Cílem hry je sesbírat karty a dostat se do laboratoře. Po mapě létají nepřátelé, před kterými se dá utéct nebo se s nimi může bojovat pomocí svítilny. Hráč si musí hlídat počet baterek pro svítilnu.

2. Využité technologie

2.1 Blender

Blender [2] je moderní a výkonný 3D grafický software vyvinutý společností Blender Foundation. Patří mezi nejvyužívanější programy pro tvorbu animací, vizualizací, interaktivních aplikací a dalších digitálních projektů. Blender je navržen pro práci s objekty, což programátorům umožňuje snadno vytvářet a upravovat 3D modely, které mohou být následně použity v různých částech projektu. Tento přístup usnadňuje tvorbu komplexních scén a zlepšuje orientaci v kódu. Blender také poskytuje širokou škálu nástrojů pro tvorbu materiálů, textur a osvětlení, což umožňuje vytvářet realistické vizualizace a animace.

Tento software jsem využil na vytvoření prvotních 3D modelů nepřátel a na modely kamenů ve hře.

2.2 Adobe Photoshop

Adobe Photoshop [3] je grafický editor, který slouží k úpravám a vytváření digitálních obrázků a fotografií. Patří mezi nejpoužívanější a nejoblíbenější grafické programy a je oblíbený mezi profesionály i amatéry. Umožňuje uživatelům editovat obrázky, upravovat barvy a světlost, vytvářet nové vrstvy, upravovat pozadí a mnoho dalšího. Díky mnoha funkcím, jako jsou například masky, vrstvy, efekty a filtry.

Adobe Photoshop jsem využil na vytvoření informačních cedulek v tutoriálu a na vytvoření a úpravu loga hry.

2.3 Microsoft Visual Studio

Microsoft Visual Studio [4] je integrované vývojové prostředí (IDE) používané především pro tvorbu aplikací a programování v jazycích jako C++, C#, Visual Basic nebo Java. Jeho funkce zahrnují nástroje pro ladění, testování, vytváření uživatelského rozhraní, správu verzí a další. Je to oblíbené IDE mezi programátory díky jeho vyspělosti a uživatelskému prostředí.

Jako programovací prostředí je využito Microsoft Visual Studio. Všechny skripty jsou napsány v jazyce C#.

2.4 Unity Engine

Unity Engine [5] je herní engine používaný pro tvorbu 2D a 3D her, simulací, virtuální reality a dalších interaktivních aplikací. Nabízí intuitivní uživatelské rozhraní, možnost vytvářet scény, animace, efekty a skripty. Unity Engine je dostupný na mnoha platformách, včetně Windows, MacOS, Linux, iOS, Android a dalších. Jeho rozmanité

funkce, široká komunita a podpora virtuální reality ho činí populárním nástrojem pro tvorbu her a interaktivních aplikací.

Nejdůležitější software, který jsem použil jako základ tvorby. Mohl jsem využít i jiný herní engine například Unreal Engine[6], ale kvůli jeho nárokům na hardware a kvůli tomu, že už jsem v Unity Engine dříve pracoval, jsem si vybral Unity Engine.

3. Herní design [7]

Herní design je proces tvorby her, který zahrnuje vytváření pravidel, herních mechanik, herních prostředí, uměleckého stylu a dalších prvků, které tvoří celkový zážitek z hraní hry. Klíčové prvky při tvorbě herního designu jsou motivace hráče k dokončení hry, získání nových dovedností, nebo k zábavě. Hratelnost se týká pravidel a interakce mezi hráčem a hrou. Hry RPG se často soustředí na vývoj postav a příběhové linie, zatímco hry akčního žánru se zaměřují na bojový systém a pohyblivost postav. Gameplay-loop (herní smyčka) by měl hráče udržovat zapojeného tím, že poskytne něco, co mohou dělat stále dokola. Občas může nastat u hráče stav – ponoření se do hry, kdy hráč není ochoten hru pozastavit nebo vypnout. Celkově lze říci, že herní design je klíčovým prvkem při tvorbě her a ovlivňuje jak úspěch, tak i kvalitu hry.

3.1 Hráčské základní mechaniky

V této podkapitole budu vysvětlovat základní mechaniky hráče.

3.1.1 Pohyb hráče

Pohyb hráče je asi nejdůležitější mechanika, nad kterou je nutné se zamyslet. Hráčský pohyb lze zkoumat z více perspektiv, v rámci, kterých hráč do hry může vstupovat. Ve většině her hráč ovládá herní postavu (svého avatara), naopak v některých hrách není hráčská postava přítomna (např. ve hře Tetris). Hráčský pohyb je ovlivněn také perspektivou, ze které hráč svého herního avatara vidí. Základní dělení je podle rozměru, ve kterém hra je. V 2D hře, hráč postavu většinou vidí pohledem ze shora, či z boku. V obou případech, se pro pohyb používá buď klikání myší, nebo pohyb realizuje pomocí stisknutí kláves. Zatímco u her v 3D se jedná o kombinaci těchto periferních zařízení, kdy hráč ovládá pohyb postavy pomocí kláves, ale naklání kameru pomocí myši, či naopak.

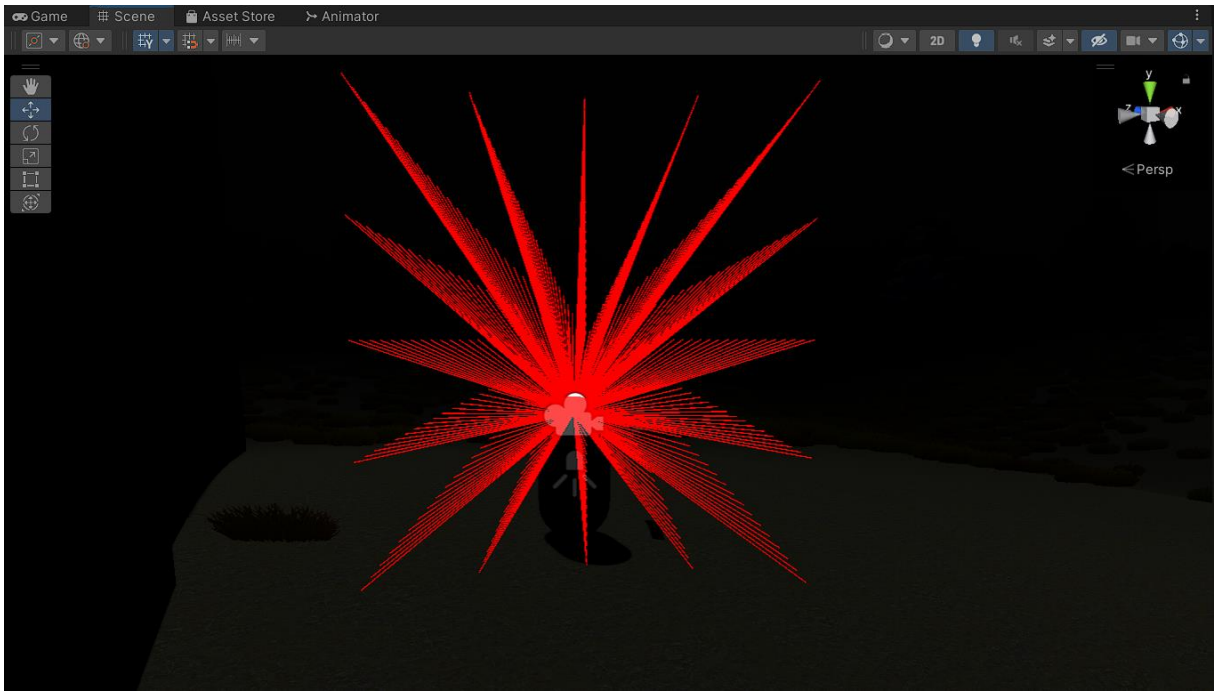
Já jsem si ve svém projektu vybral pohyb pomocí klávesnice a pohyb kamery pomocí myši, protože mi to přijde nejpřirozenější. Ve skriptu FirstPersonController.cs se nachází funkce FixedUpdate(), která je konstantně volána v určitém časovém intervalu. Na začátku proběhne kontrola, jestli je hodnota playerCanMove nastavena na true. Pokud je hodnota true, začne se provádět výpočet rychlosti pohybu hráče v závislosti na vstupu z klávesnice. Konkrétně se vypočítává vektor cílové rychlosti "targetVelocity" na základě vstupů z klávesnice. Pokud jsou tyto hodnoty nenulové, znamená to, že hráč se snaží pohybovat. Následující podmínky se zaměřují na pohyb hráče v závislosti na tom, zda hráč drží klávesu pro sprint nebo ne. Pokud hráč zmáčkne tlačítko pro sprint a má dostatek SprintBaru (sprintRemaining > 0), a pokud nemá cooldown na sprintu, vypočítá se cílová rychlost hráče a pomocí fyzikální simulace se aplikuje síla na hráče (rb.AddForce) tak, aby hráč dosáhl této rychlosti. Pokud hráč nevyužívá sprint a chodí, cílová rychlost se nastaví na hodnotu walkSpeed a aplikuje se podobná síla pro pohyb hráče (rb.AddForce). Během tohoto pohybu se také provádí některé akce, jako je skrčení hráče, nebo změna průhlednosti SprintBaru.

Stejné použití Rigidbody.AddForce() jako v mé ukázce je například použito ve hře Portal 2[8], kde hra využívá fyzikální simulace pro interakce s portály a objekty v prostředí, díky kterým hráč může plnit hádanky. Jiná možnost pohybu postav je použití CharacterController.Move(). CharacterController.Move() je metoda, která pohybuje objekty s CharacterController komponentou. CharacterController je navržený speciálně pro pohyb hráče a poskytuje možnosti, jako jsou detekce kolizí s povrchem, snadná změna směru pohybu, a podobně. Metoda Move je volána každý frame a posouvá hráče na základě zadaného vektoru rychlosti. Použití Move je vhodné pro hry, kde se hráč pohybuje po rovině a kde není nutné simuloval fyzikální interakce, jako jsou třeba odraz od stěn nebo gravitace. Příklad takovéto hry je Inside [9].

3.1.2 Záblesk

Záblesk je jednou z hlavních mechanik hry. Hráč má díky záblesku šanci bojovat s nepřáteli, ale musí mít zapnutou svítilnu, musí mít baterku v inventáři a v předešlých pěti sekundách nesměl použít záblesk.

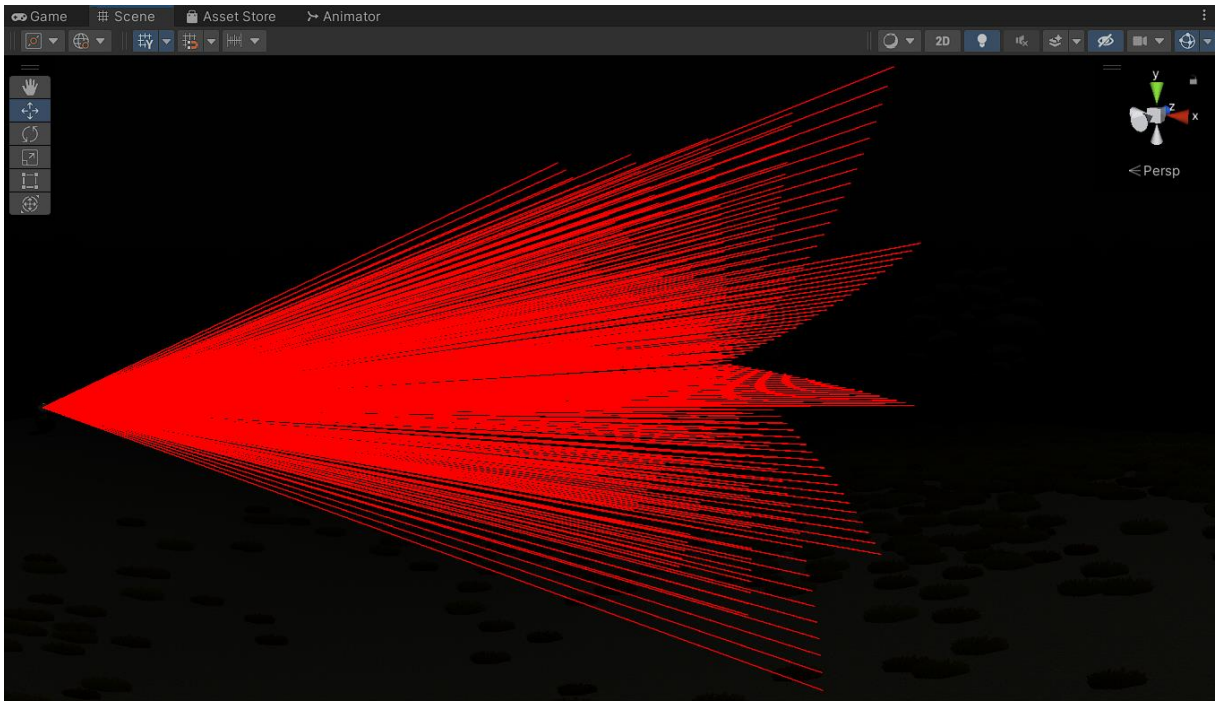
Pokud tedy hráč může použít záblesk a stiskne tlačítko pro záblesk, tak se zavolá ve skriptu FlashlightAdvanced funkce Shoot(). Tato funkce má nastavenou vzdálenost pro záblesk, která se později předává RayCastům, a volá další funkce. Jednou z nich je ShootSpreadRays(), které se předávají dvě hodnoty, první je currentRange(jak daleko paprsky dosahují), další je angleStep(úhlový krok). Poté se v cyklu definuje pole 13 paprsků, které určují směr paprsků.



Ilustrace 1. Ukázka 13 paprsků

Každý prvek pole je vypočten pomocí funkce GetSpreadDirection(), která bere úhel s osou a vrací vektor, který odpovídá danému směru rozptylu. Nakonec se ve for cyklu každý

vektor použije pro vystřelení paprsku pomocí druhé funkce RayHit().Funkce RayHit() bere jako vstupní parametr paprsek a currentRange, díky kterým zjišťuje, zda paprsek zasáhl nějaký objekt v herním světě.



Ilustrace 2. Ukázka detekce kolize Rayů

Pokud byl objekt zasažen a má tag „FakeEnemy“ nebo „Enemy“, zavolá se na nepříteli funkce TakeDamage().

3.2 Nepřátelé a boj s nimi

Ve většině 3D her jsou nepřátelé, kteří se snaží zabránit hráči dosáhnout určitý cíl hry. Nepřátelé se dělí na dvě hlavní skupiny. Jsou to nepřátelé řízení počítačem, nebo nepřátelé ovládaní ostatními hráči. Hráči s nimi musí bojovat a používat různé taktiky, samozřejmě záleží na žánru hry.

Já jsem použil nepřátele řízené počítačem. Hráč s nimi může bojovat pomocí záblesku ze svítilny, nebo se jim může pokusit utéct. V mém projektu jsou 3 druhy nepřátel. Dva z nich mají podobný vzhled, ale liší se barvou. Dále zvukem, který vydávají, a stylem boje proti nim. Poslední má úplně odlišný vzhled a také jinou mechaniku.

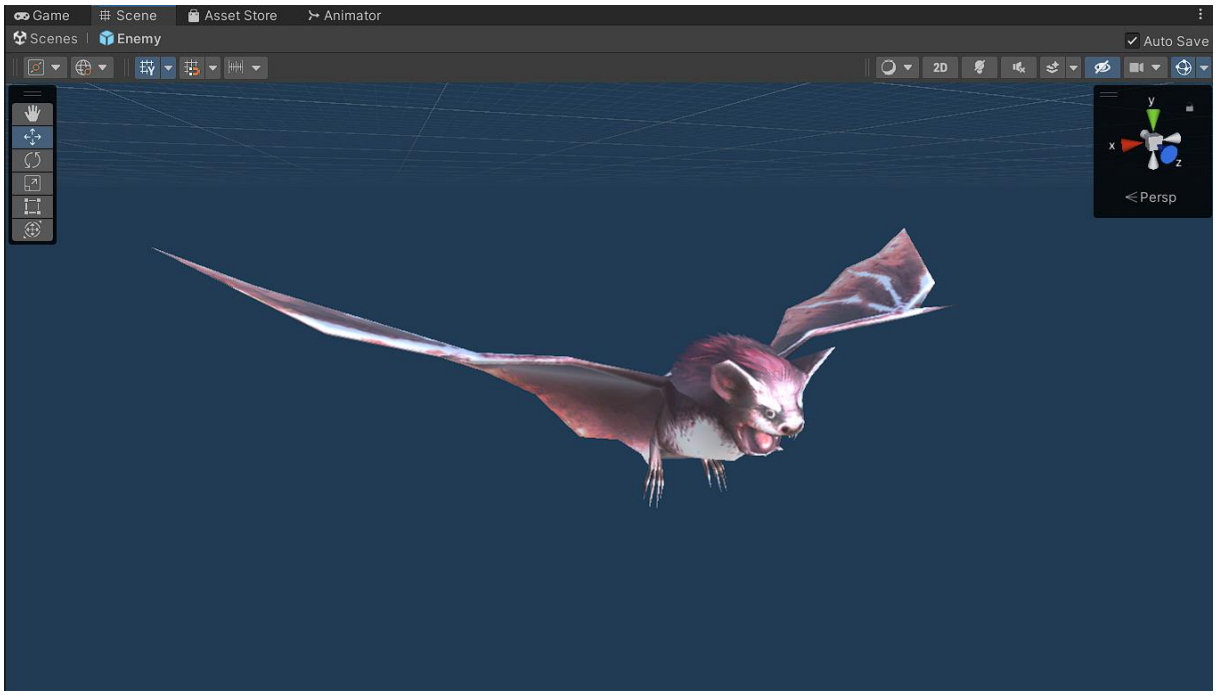
3.2.1 Mechaniky nepřátel

Nepřátelé mohou být navrženi s různými mechanikami, aby vyvolávali strach a nejistotu. Například mohou mít nepředvídatelné pohyby, schopnost jumpscare (vystrašit nečekaně

hráče), nebo je téměř nemožné je zabít. Některé hry se zaměřují na souboje, zatímco jiné kladou větší důraz na schovávání a útěk.

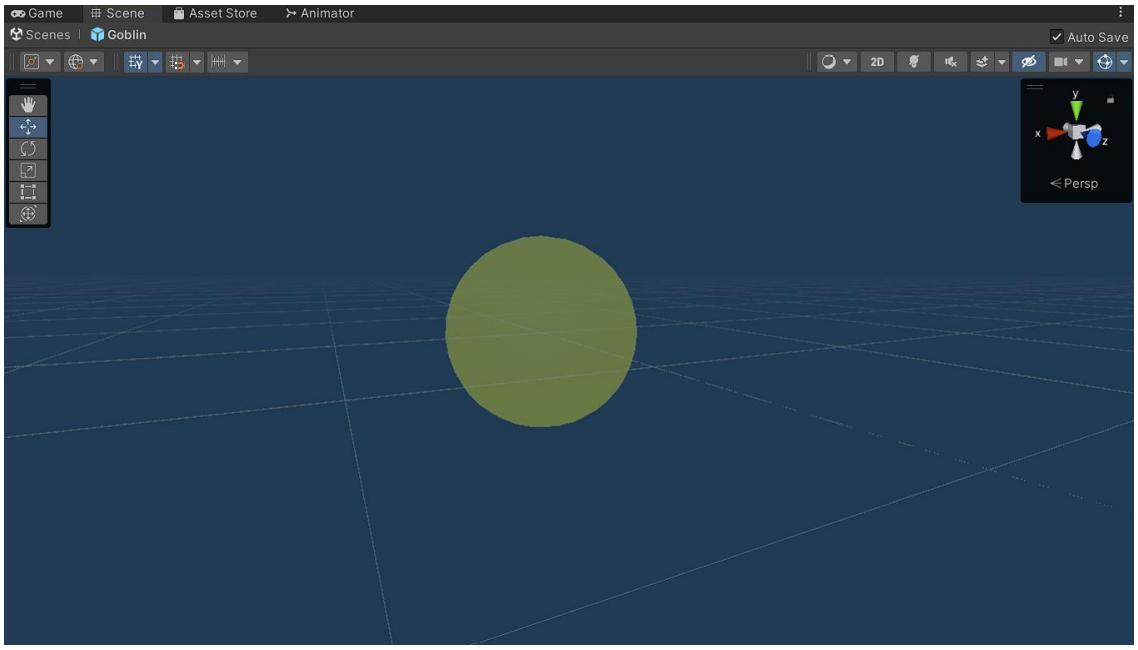
Moji nepřátelé mají společné chování. Když vidí hráče, chytají ho, aby mu ztížili sbírání karet, které jsou důležité pro vítězství ve hře. Jak už jsem zmiňoval, tak jsem do hry zakomponoval 3 druhy nepřátel. Všichni tři nepřátelé sledují hráče díky skriptu EnemyAI, ale o tom více povím v další kapitole. Další společné mechaniky jsou, že se spustí funkce TakeDamage(), když jsou zasaženi zábleskem. Dva z nepřátel jsou hlavní a jsou na mapě jenom jednou. Třetí nepřítel se objevuje v každé lokaci.

První nepřítel se nazývá Enemy. Skripty tohoto nepřítele jsou Target a KillPlayer. Skript Target obsahuje funkci TakeDamage() a Respawn(). Funkce TakeDamage() se volá, když je nepřítel zasažen paprskem vytvořeném ve skriptu AdvancedFlashlight. Tato funkce nejprve vypočítá podle pozice hráče waypoint, který má od hráče největší vzdálenost a nastaví si furthestWaypointIndex na index vypočítaného waypointu. Poté se SetActive objektu nepřítele nastaví na false. To zneviditelní objekt nepřítele a znemožní mu reagovat na vstupy od hráče a od herního enginu. Na konci funkce TakeDamage() se po 10 sekundách zavolá metoda Respawn(). Metoda Respawn() pouze nastaví pozici nepřítele na požadovaný index z uložené proměnné furthestWaypointIndex a nastaví objektu SetActive na hodnotu true. Jestliže se nepřítel dostane na pozici hráče a aktivuje se trigger ve skriptu KillPlayer, tak dojde nejdříve k porovnání tagu, jestli je to opravdu hráč a poté se spustí animace a skript pošle hráče do posmrtného menu.



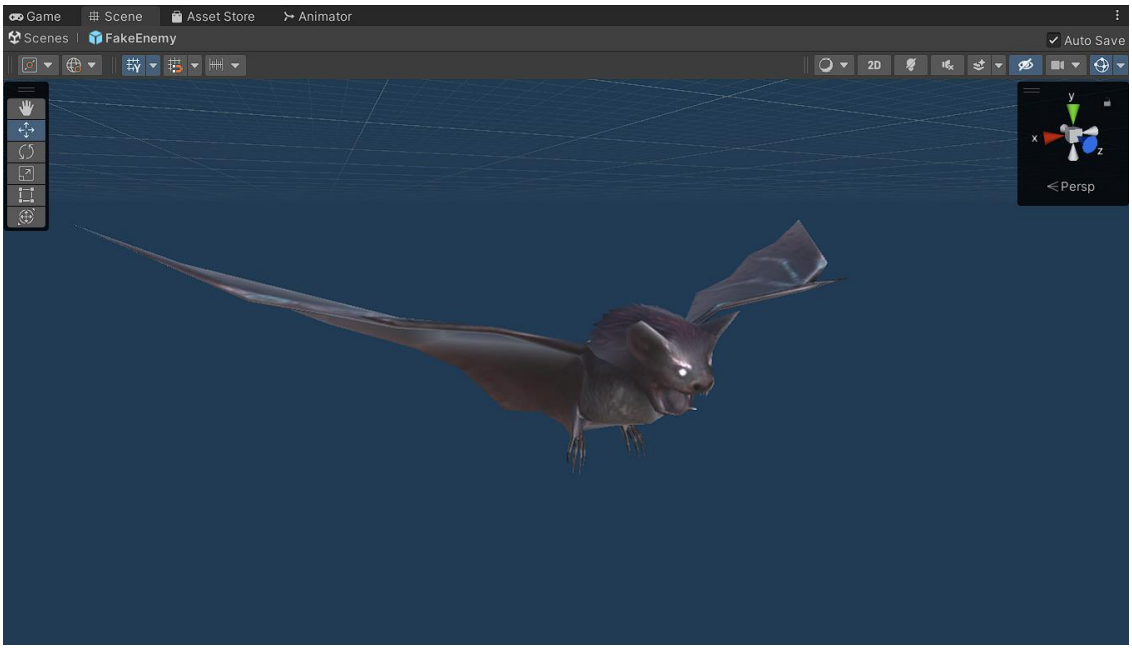
Ilustrace 3. Vzhled Enemy

Druhý se jmenuje Goblin. Tento nepřítel má opět skript Target a vlastní skript StealBattery. StealBattery je jednoduchý skript, který při doteku Goblina s hráčem odebere hráči jednu baterku, která není ve svítilně.



Ilustrace 4. Vzhled Goblin

Poslední je FakeEnemy, který má také skript KillPlayer. Místo skriptu Target má skript FakeTarget, který se volá při použití záblesku na něj. Tento skript je identický skriptu KillPlayer, protože se nemá na nepřítele používat záblesk.



Ilustrace 5. Vzhled FakeEnemy

3.2.2 AI Nepřátel

Kód EnemyAI popisuje chování nepřátelského AI ve hře. Nepřítel je schopen patrolovat mezi různými body (waypoints) a při hledání hráče se chová jinak podle toho, zda hráče vidí, nebo ho má již na dosah. Metoda EnviromentView() zjišťuje, zda se hráč nachází v poli dohledu nepřítele. Pokud ano, nastaví se proměnná m_playerInRange na true. Metoda Chasing() je volána, když se nepřítel rozhodne pronásledovat hráče. Nepřítel se pohybuje rychleji a snaží se dostat co nejblíže k hráči. Pokud se mu to podaří, hráč je chycen. Pokud je hráč mimo dosah, nepřítel se vrací k patrolování. Metoda Patroling() se volá, když se nepřítel vrací k patrolování mezi jednotlivými body. Pokud zjistí, že je hráč poblíž, snaží se dostat k němu.

3.3 Uživatelské rozhraní

3.3.1 Hlavní menu

Hlavní menu v hrách je první věcí, kterou hráči vidí a umožňuje jim přístup k různým funkcím hry, jako jsou začátek nové hry, načtení uložené pozice nebo změna herních nastavení. Je důležité, aby hlavní menu bylo přehledné, intuitivní a esteticky příjemné. Design a umístění tlačítek a navigačních prvků by měly být dobře promyšleny, aby hráči mohli snadno najít to, co potřebují, a rychle se dostat do hry. Hlavní menu také může odrážet téma nebo styl hry a být doprovázeno odpovídající hudbou nebo zvukovými efekty.

Hlavní menu obsahuje kromě názvu hry 4 tlačítka, *START*, *TUTORIÁL*, *NASTAVENÍ* a *ODEJÍT*. Každé tlačítko provádí svoji předurčenou akci. Tlačítko *START* a *TUTORIÁL* má za úkol přepínat scény v Unity, tlačítko *NASTAVENÍ* načte v menu možnosti nastavení zvuku a myši a *ODEJÍT* vypne hru. Pozadí menu je les s cestou, na kterou svítí svítilna.

Ilustrace 6. Hlavní Menu

3.3.2 Uživatelské rozhraní ve hře

Uživatelské rozhraní ve hře zahrnuje různé prvky, jako jsou menu, ovládací prvky, mapy, inventáře a dialogová okna. Důležitou součástí dobrého designu UI je jeho přehlednost, intuitivnost a estetika. Dobře navržené UI umožňuje hráčům snadno a plynule interagovat s herním světem a poskytuje potřebné informace, aniž by hráče rušilo při hraní.

Při návrhu uživatelského rozhraní pro hru jsem se zaměřil na co nejefektivnější zobrazení důležitých informací pro hráče. Konkrétně jsem se snažil zobrazovat procentuální množství zbytku baterie, počet baterií, počet karet a množství zbývajícího sprintu. Tyto informace jsou zobrazeny na sedmé ilustraci. V levé horní části jsou informace o bateriích, v pravé horní části se nachází počítadlo karet a na střední části dolního okraje obrazovky se nachází ukazatel zbývajícího množství sprintu.

Ilustrace 7. Uživatelské rozhraní hráče ve hře samotné

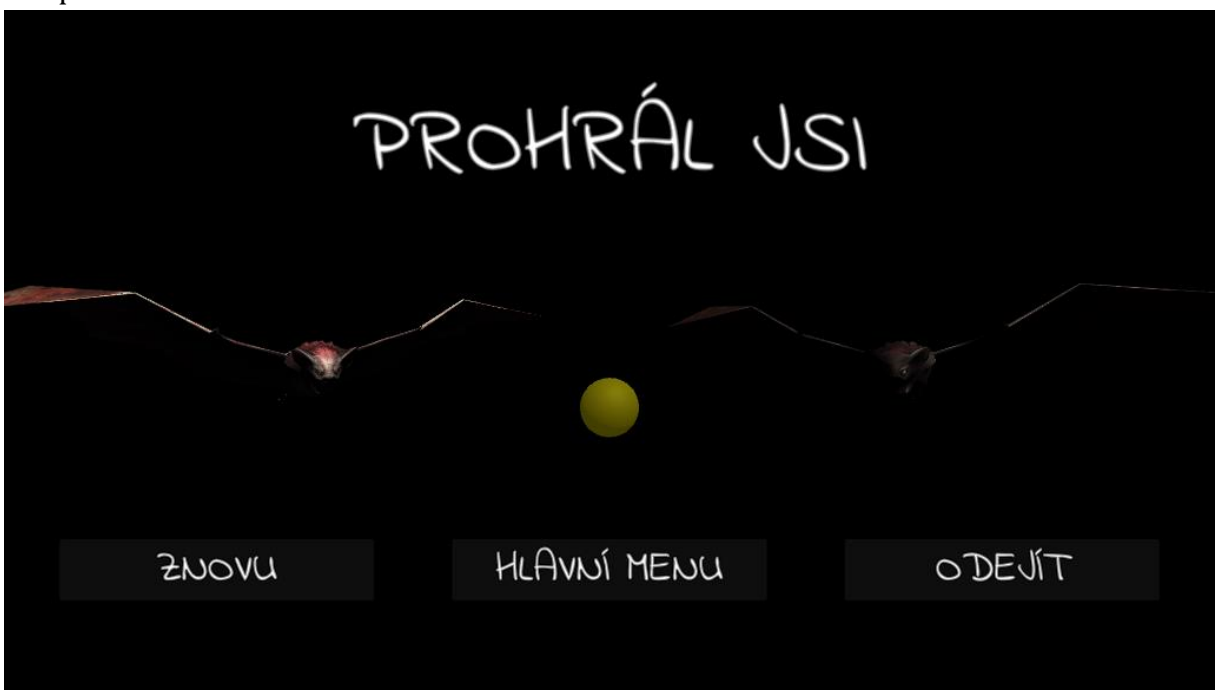
3.3.3 Menu po ukončení hry

Menu po ukončení hry je důležitou součástí celkového herního zážitku. Umožňuje hráčům ukončit hru, dát si pauzu, nebo se vrátit a znovu se pokusit zvítězit, pokud hráč prohrál. Vítězné menu, i menu porážky mohou obsahovat různé prvky, jako jsou statistiky nebo různé možnosti restartování, načtení uložené pozice nebo návratu do hlavního menu. Menu by mělo být dobře navrženo, aby hráči mohli snadno interagovat s funkcemi a přehledně vidět svůj postup v hře. Vítězné menu může být také doplněno o výsledkovou obrazovku a speciální efekty, aby se hráči cítili uspokojeni svým úspěchem.

Výherní menu obsahuje tři tlačítka. Nalevo je tlačítko *ZNOVU* - pro opakování hry, uprostřed je *HLAVNÍ MENU* - vrátí hráče do hlavního menu, napravo se nachází tlačítko *ODEJÍT*, které vypne hru.

Ilustrace 8. Obrazovka při výhře

V posmrtném menu jsou všechny tlačítka stejné jako ve výherním. Pouze má jiné pozadí, s nepřáteli.



Ilustrace 9. Obrazovka při prohře

3.4 Mapa

Herní mapa je jedním z nejdůležitějších prvků herního designu, protože mapa je obvykle prvním kontaktem hráče s hrou, takže musí být přehledná a snadno čitelná. Správně navržená mapa může hráče vést k plnění úkolů, objevování nových oblastí a získávání odměn.

3.4.1 Návrh mapy

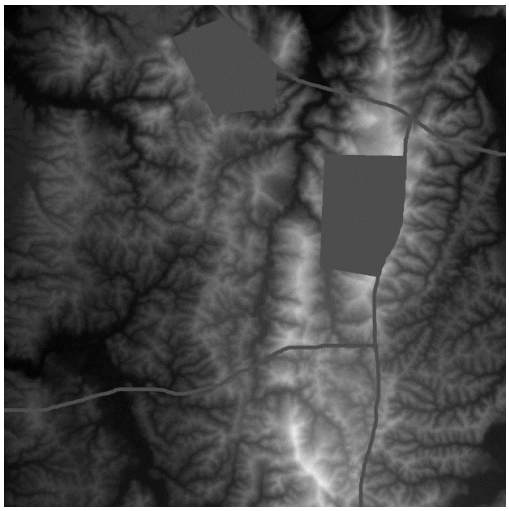
Navržení herního světa zahrnuje mnoho faktorů, jako jsou geografie, klima, topografie, architektura, historie a kultura. Mapa by mohla odrážet tuto komplexitu, podle žánru a záměrů hry. Návrh herního světa také zahrnuje rozmístění důležitých prvků, jako jsou budovy, postavy a předměty, aby hráči mohli hladce plnit úkoly a objevovat herní svět.

Design vycházel z hororového žánru hry. V hororové tématice se často objevuje motiv bloudění v lese, proto bylo rozhodnuto, že mapa bude z většiny tvořena právě lesem. Mapa byla navržena na papír a následně konzultována, než byla přenesena do grafického editoru.

3.4.2 Tvorba mapy

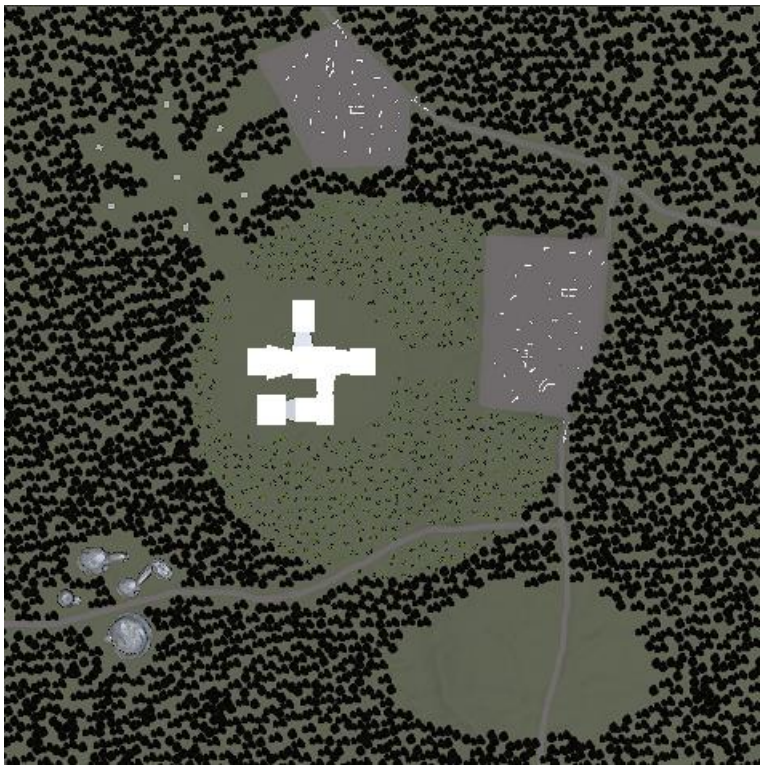
Tvorba herního světa může být velmi náročná a vyžaduje mnoho práce. Musíme zohlednit mnoho faktorů, jako jsou herní mechanismy, dějové linie, hráčovy cíle. Mapa je základním prvkem tohoto procesu, protože slouží jako nástroj pro plánování a organizaci herního světa. Je vhodné se zaměřit na detaily, jako jsou textury, osvětlení a zvuky, aby herní svět vypadal co nejvěrněji a autenticky.

Vzhledem k náročnosti procesu tvorby práce byl vybrán systém Unity, který umožňuje tvořit mapu z jejího výškového modelu. V projektu byl vybrán snímek výškové mapy, hor v Japonsku, do kterého byly zakresleny silnice.



Ilustrace 10. Výšková mapa hry

Obrázek výškové mapy byl naimportován do prostředí Unity, které vygenerovalo hrubý tvar, který bylo nutné ručním dokreslením dotvořit a upravit. Les byl vložen pomocí nástroje *štětec*, který pomohl vložit několik desítek objektů stromů naráz. Následně byly do mapy naimportovány další objekty z knihoven, které byly volně dostupné.



Ilustrace 11. Mapa hry z ptačí perspektivy

3.4.3 Generace klíčů/baterek

Generaci klíčů a baterek jsem vyřešil vytvořením třídy RandomSpawner, která umožňuje náhodné rozmístění herních objektů ze seznamu na zadané pozice.

Tato třída využívá několik tříd a funkcí z knihoven Unity, jako například třídu Transform pro určení pozice a rotace herního objektu, funkci Instantiate[10] pro vytvoření nové instance herního objektu a funkci Random.Range pro náhodné generování indexů objektů a pozic.

Kód této třídy umožňuje také nastavení parametru allowOverlap, který určuje, zda mohou být herní objekty umístěny na stejných pozicích nebo ne.

3.5 Hudba a zvuky

Zvuk a hudba jsou důležité pro herní design, protože přispívají k celkovému zážitku z hry. Zvukový design a hudba pomáhají hráčům ponořit se do herního světa a vytvářejí atmosféru. Informují hráče o důležitých událostech, ovlivňují jejich emoce a naladění. Také pomáhají vytvořit jedinečnou identitu hry. Celkově je zvukový design a hudba klíčovým prvkem herního designu a mohou výrazně ovlivnit zážitek hráče.

V tomto projektu hrají zvuky důležitou část, při rozpoznávání nepřátel, protože každý nepřítel vydává rozdílný zvuk. V hlavním menu a ve hře samotné je hudba použita pro naladění strašidelné atmosféry hry. Dále je při výhře spuštěna nahrávka zvuku od Matěje Berana.

4. Popis skriptů

4.1 Skripty hráče

V této kapitole se zaměřím na popis skriptů, které spravují všechny prvky související s objekty hráče. Tyto skripty jsou uloženy v adresáři “Assets/Scripts”.

4.1.1 FirstPersonController

Tento skript je obsahem Unity Assetu, který má název „ModularFirstPlayerController“[11]. Jedná se o skript, který umožňuje hráči se pohybovat při stisknutí kláves a také může za pohyb kamery. Pomocí posuvníku v menu se nastaví citlivost kamery v tomto skriptu. Kontroluje, zda hráč může vyskočit či se skrčit. Také vytváří a zobrazuje Stamina Bar (ukazatel, jak dlouho hráč může sprintovat). V neposlední řadě skript umožňuje nastavení kývání při chůzi, nakonec umožňuje hráči při stisknutí daného tlačítka zoom (přiblížení).

4.1.2 FlashlightAdvanced

Druhý nejobsáhlejší skript, kromě zapínání a vypínání svítilny spravuje počet baterek hráče, který je vyobrazen pomocí spritů (grafická ikona baterky). Dále počítá vybíjení baterky a zobrazuje stav aktuální baterky ve svítilně. Také se stará o počet karet, které již hráč sesbíral. Nejdůležitější věc, o kterou se stará je jednou z hlavních mechanik hry, záblesk. Když je stisknuta klávesa pro záblesk, tak se spustí funkce Shoot(), která vyšle několik RayCastů (funkce Physics, která promítá paprsek do scény a vrací logickou hodnotu, pokud byl cíl úspěšně zasažen) z pozice kamery hráče. Pokud na něco narazí, tak se zkontroluje podle tagu (označující slovo pro herní objekty), jestli se jedná o nepřítele nebo ne. Jestli ano, tak se spustí funkce TakeDamage() na daném nepříteli. Poté vypočítává další směry paprsků pomocí rotačních operací a opakuje proces detekce kolizí pro každý z nich. Výsledkem je detekce zásahu cíle v širokém poli pohledu hráče.

4.1.3 BatteryPickUp

Skript BatteryPickUp umožňuje hráči sbírat baterky a karty opět pomocí RayCast. Kontroluje, zda má hráč sesbírané všechny karty, aby mohl interagovat s dveřmi laboratoře a vyhrát hru.

4.1.4 BatteryBar

Skript BatteryBar už podle názvu nastavuje zbývající procenta baterky ve svítilně.

4.1.5 OffsetFlashlight

Skript OffsetFlashlight je jednoduchý skript, který pomocí Quaternion (vyjádření orientace nebo rotace herního objektu) vypočítává opoždění pohybu svítilny, aby hráč měl realističtější pocit ze hry.

4.1.6 PauseMenu

Skript PauseMenu obstarává ve hře funkci pozastavení. Pokud hráč stiskne danou klávesu pro pozastavení hry, tak se zavolá funkce PauseGame() která nastaví Time.timeScale (měřítko ve kterém plyne čas) na 0, čímž způsobí zastavení všeho dění ve hře, i s pozastavením všech zvuků ve hře. Po opětovném stisknutí tlačítka pozastavení se Time.timeScale nastaví na výchozí hodnotu 1 a čas opět plyne, jak má. Opět se spustí zvuky. Když z pozastaveného menu přejdeme tlačítkem do hlavního menu, tak se ve skriptu nastaví animace přechodu mezi scénami, tato animace je jednoduché zčernání obrazovky.

4.1.7 Minimap

V tomto skriptu se děje všechno okolo minimapy. Po stisknutí tlačítka pro minimapu se minimapa otevře a hráč vidí svoji pozici na mapě. Pozice hráče se vypočítává pořád pomocí reálné pozice hráče na mapě a pomocí dělení a násobení vektorů se měřítko zmenší, aby se vešlo do minimapy. Díky tomu, pokud se pohybuje tak vidí svůj pohyb na minimapě.

4.2 Skripty nepřátel

Tato kapitola se zabývá funkčností skriptů pro nepřátele. Tyto skripty jsou uloženy v adresáři “Assets/Scripts”.

4.2.1 EnemyAI

Skript EnemyAI dává nepřátelům život. Dají se v něm nastavit různé hodnoty, jako například rychlost chození, běhání, otáčení, vzdálenost vidění nepřítele. Nepřítel díky skriptu pořád kontroluje, zda nevidí hráče a jestli ano, hráč nesmí být schovaný za jiným objektem, tak se zavolá funkce Chasing(). Tato funkce kontroluje, zda je hráč poblíž. Jestli ano, nepřítel se snaží dojít na pozici hráče a tím se ho dotknout. Pokud hráč v dohledu není, zavolá se funkce Patroling(). Pomocí této funkce se nepřítel podívá na poslední pozici hráče. Jestli se mu ho nepodaří nalézt, tak začne procházet mapou náhodně, pomocí checkpointů (kontrolní bod), které má předurčené. Náhodně si z nich vybírá a poté se dostane na jejich pozici, kde se otočí a pokračuje na nový checkpoint, dokud nenarazí na hráče.

4.2.2 Target

Jak už jsem popisoval v kapitole *4.1.2 FlashlightAdvanced,* při záblesku použitém na nepřítele se zkontroluje tag nepřítele a jestli se jedná o Target, tak se zavolá funkce TakeDamage(), která v tomto skriptu objekt nepřítele nastaví na disabled (neplatí na něj žádné skripty a zneviditelní se). Po funkci TakeDamage() se zavolá funkce Respawn(), která nepřítele přemístí na checkpoint nejdál od hráče, aby ho hráč hnedka nepotkal znovu.

4.2.3 FakeTarget

Podobně jako u *4.2.2 Target* se spustí funkce TakeDamage(). Tato funkce ale u skriptu vyvolá pouze načtení scény s obrazovkou porážky.

4.2.4 StealBattery

Skript StealBattery umožňuje nepříteli, aby se dotknul hráče a tím mu sebral jednu baterku z inventáře.

4.2.5 KillPlayer

Díky skriptu KillPlayer mohou nepřátelé hráče poslat do scény s obrazovkou porážky. Tento skript opět obsahuje přechod mezi scénami, jak jsem již zmiňoval v kapitole *4.1.6 PauseMenu.*

4.3 Skripty v menu

Tyto skripty jsou obsaženy v hlavním menu, aby fungovalo správně. Tyto skripty jsou uloženy v adresáři “Assets/Scripts”.

4.3.1 MainMenu

MainMenu skript zajišťuje, aby se při kliknutí na tlačítka v menu přepínaly scény. Opět nastavuje přechod mezi scénami.

4.3.2 AudioManager

Skript AudioManager nastavuje všechno okolo hudby a zvukových efektů, a navíc je rozděluje do AudioMixerGroup(mix zvuku, díky němuž se dá hlasitost hudby a zvuku nastavit zvlášť).

4.3.3 AudioOptionsManager

AudioOptionsManager, na rozdíl od předešlého skriptu, pouze nastavuje hlasitost hudby a zvukových efektů, když se změní hodnoty posuvníků v menu.

4.3.4 MouseSensitivity

MouseSensitivity ze třídy PlayerPrefs vybírá hodnotu s citlivostí myši a nastavuje ji pomocí posuvníku a ukládá zpět do PlayerPrefs.

4.3.5 BreathingCamera

Jednoduchý skript, který se stará o pohyb kamery v rámci menu, tento pohyb má připomínat pohled udýchaného člověka a je pro navození hororové atmosféry hry.

4.3.6 LightFlickerEffect

Skript náhodně generuje časové intervaly a aktualizuje sílu intenzity svícení svítilny v menu.

4.3.7 EnemyTeleport

EnemyTeleport slouží k přemisťování nepřítele v hlavním menu, jsou tu dány určité lokace s pravděpodobností, kde se může nepřítel na chvíli objevit.

4.4 Další skripty

Další skripty jsou skripty použité jinde v projektu, které se nehodí do předešlých kategorií. Tyto skripty jsou uloženy v adresáři “Assets/Scripts”.

4.4.1 ItemRespawn

Tento skript se používá pouze v tutoriálu hry u baterky. Každých pět sekund se kontroluje, zda je baterie na zemi, pokud tomu tak není, vygeneruje se baterie nová.

4.4.2 Sound

Primitivní skript pro nastavení hlasitosti zvuku a pro kontrolu, jestli je zvuk isLoop (jestli se po přehrání opakuje dokola) a onAwake (jestli se zvuk spouští na začátku scény).

4.4.3 RandomSpawner

RandomSpawner je jedním z nejdůležitějších skriptů, protože se díky němu do hry přidávají baterky a karty. Skript vybere lokaci ze seznamu, kde se může objevit baterie či karta a zkontroluje, zda už tam není jiná baterie či karta. Jestli ne tak ji to tam přidá.

4.4.4 DeathMenu

Tento skript zajišťuje funkčnost tlačítek v posmrtném menu, a dále se stará o spuštění animací, které probíhají při přechodu mezi scénami.

5. Testování hry

V průběhu vytváření hry, jsem publikoval testovací verze, které odzkoušeli někteří z mých přátel, včetně mého vedoucího projektu, pana učitele Mottla a opětovali mi zpětnou vazbu.

Jednou z častých proseb ve zpětné vazbě bylo přidání tutoriálu. Tento problém byl vyřešen přidání nové scény v Unity, kde se hráč dozví veškeré ovládání hry.

Další prosba byla přidat kompas nebo jiný ukazatel, který bude navádět hráče k lokaci karet. Problém byl eliminován tvorbou minimapy, na které je hráč a hlavní lokace, kde jsou karty umístěny.

Závěr

Podařilo se mi vytvořit hru, která částečně splňuje zadání. Cílem projektu bylo vytvořit příběhovou 3D hru s hororovou tématikou. Tento cíl nebyl zcela splněn, jelikož bylo odchýleno od příběhové tématiky. Celková hra tedy byla nakonec vytvořena jako arkádová, 3D hororová hra s prvky vlastních mechanik.

Do hry se povedlo vytvořit 3D modely. Byl vymyšlen a naprogramován skript pro hlavní mechaniku hry a tou je záblesk. Vznikly tři druhy nepřátel, kteří mají každý svoji mechaniku a byla použita umělá inteligence pro jejich schopnost pohybu a pro enviromentální pohled. Povedlo se vytvořit náhodný spawn baterek a karet při každém startu, díky kterému je každé kolo unikátní. Proběhlo zkompletování funkčnosti scén Unity. Hra byla testována a na základě zpětné vazby byly přidány nové prvky. Nakonec se podařilo ozvučit důležité části projektu.

Projekt je v tuto chvíli dokončený a funkční. Hra z mé strany obsahuje všechny mechaniky, které jsem chtěl implementovat. Do budoucna, pokud by se k projektu někdo vracel, či by ho chtěl někdo rozšířit, tak je možné implementovat vlastní modely nepřátel. Stejně jako by bylo možné rozšířit projekt o další mapy, či více unikátních druhů nepřátel.

Seznam literatury

[1] Slender: The Eight Pages. *Wikipedia* [online]. [cit. 2023-03-19]. Dostupné z: https://cs.wikipedia.org/wiki/Slender:_The_Eight_Pages

[2] Blender.org. *Blender* [online]. [cit. 2023-03-19]. Dostupné z: https://www.blender.org/

[3] Transformujte fotky a tvořte skvělé grafiky | Adobe Photoshop. *Adobe* [online]. [cit. 2023-03-19]. Dostupné z: https://www.adobe.com/cz/products/photoshop.html

[4] Visual Studio: Integrované vývojové prostředí (IDE) a editor kódu pro vývojáře softwaru a týmy. *Microsoft* [online]. [cit. 2023-03-19]. Dostupné z: https://visualstudio.microsoft.com/cs/

[5] Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine. *Unity* [online]. [cit. 2023-03-19]. Dostupné z: https://unity.com/

[6] The most powerful real-time 3D creation tool – Unreal Engine. *UnrealEngine* [online]. [cit. 2023-03-19]. Dostupné z: https://www.unrealengine.com/en-US

[7] DŘÍMAL, Jiří. *Vývoj videoher v herním enginu Unity* [online]. [cit. 2023-03-19]. Dostupné z: https://dk.upce.cz/bitstream/handle/10195/79583/DrimalJ_VyvojVideoher_AK_2022.pdf?sequence=1&isAllowed=y

[8] Portal 2. *GiantBomb* [online]. [cit. 2023-03-19]. Dostupné z: https://www.giantbomb.com/portal-2/3030-21662/

[9] Inside. *Playdead* [online]. [cit. 2023-03-19]. Dostupné z: https://playdead.com/games/inside/

[10] Object.Instantiate. *Unity Documentation* [online]. [cit. 2023-03-19]. Dostupné z: https://docs.unity3d.com/ScriptReference/Object.Instantiate.html

[11] *Unity Asset Store* [online]. [cit. 2023-03-19]. Dostupné z: https://assetstore.unity.com/packages/3d/characters/modular-first-person-controller-189884

Seznam využitých assetů

Outdoor Ground Textures. *Unity Asset Store* [online]. [cit. 2023-03-19]. Dostupné z: https://assetstore.unity.com/packages/2d/textures-materials/floors/outdoor-ground-textures-12555

Bat Monster Beast. *Unity Asset Free* [online]. [cit. 2023-03-19]. Dostupné z: https://unityassets4free.com/bat-monster-beast/

Low Poly Cars. *Unity Asset Store* [online]. [cit. 2023-03-19]. Dostupné z: https://assetstore.unity.com/packages/3d/vehicles/land/low-poly-cars-101798

Survival Game Tools. *Unity Asset Store* [online]. [cit. 2023-03-19]. Dostupné z: https://assetstore.unity.com/packages/3d/props/tools/survival-game-tools-139872

Free Trees. *Unity Asset Store* [online]. [cit. 2023-03-19]. Dostupné z: https://assetstore.unity.com/packages/3d/vegetation/trees/free-trees-103208

Seznam ilustrací