Hra z pohledu třetí osoby

Ročníkový projekt

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 30. dubna 2021

Anotace

Mým ročníkovým projektem je programování hry v herním engine Unity, který podporuje programovací jazyk C#. Hra je cílena na platformu Android. Hráč ovládá svoji postavu z pohledu třetí osoby. Postava se dokáže pohybovat, útočit a vyhýbat se útokům. Hráč také může spravovat inventář s předměty, které si zakoupil v obchodě.

Mapy, na kterých hráč hraje, jsou generovány náhodně s různými úkoly ke splnění. Na poslední mapě se vyskytuje boss, kterého hráč potřebuje porazit. Nepřátelé jsou implementováni jako konečné automaty. Nepřátelé vyhledávají cestu k hráči pomocí A \* algoritmu.

Zadání projektu

Mým ročníkovým projektem je vytvoření mobilní 3D hry pro platformu Android v herním engine Unity, který podporuje programovací jazyk C#.

Hráč hraje za postavu, kterou ovládá z pohledu třetí osoby. Postava se například dokáže pohybovat, útočit nebo sbírat předměty na mapě. Součástí hry je také hráčův inventář, ve kterém si může nasadit již získané předměty.

Hráč bojuje proti nepřátelským postavám ovládaným počítačem v náhodně generovaných mapách. Nepřátelé jsou implementováni jako konečné automaty a vyhledávají cestu pomocí A \* algoritmu.

Cílem hry je porazit finálního bosse ve hře.

Obsah

[Anotace 3](#_Toc61729096)

[Zadání projektu 3](#_Toc61729097)

[Obsah 4](#_Toc61729098)

[1. Úvod 6](#_Toc61729099)

[2. Struktura projektu 6](#_Toc61729100)

[3. Struktura 7](#_Toc61729101)

[3.1 Game manager 7](#_Toc61729102)

[3.2 Menu manager 7](#_Toc61729103)

[3.3 Hub manager 7](#_Toc61729104)

[3.4 Maze manager 7](#_Toc61729105)

[3.5 Game physics 7](#_Toc61729106)

[3.6 Input manager 7](#_Toc61729107)

[3.7 Player controller 7](#_Toc61729108)

[3.8 Camera controller 7](#_Toc61729109)

[3.9 Maze generator 7](#_Toc61729110)

[3.10 Cell generator 8](#_Toc61729111)

[3.11 Subcell generator 8](#_Toc61729112)

[3.12 Pathfinding node generator 8](#_Toc61729113)

[3.13 Tile generator 8](#_Toc61729114)

[3.14 Spawner 8](#_Toc61729115)

[3.15 Enemy controller 8](#_Toc61729116)

[3.16 State Controller 8](#_Toc61729117)

[3.17 Boss 8](#_Toc61729118)

[3.18 Pathfinding 8](#_Toc61729119)

[3.19 Game Physics 8](#_Toc61729120)

[4. Scény 8](#_Toc61729121)

[4.1 Menu 9](#_Toc61729122)

[4.2 Hráč 9](#_Toc61729123)

[4.3 Výběr úrovní 9](#_Toc61729124)

[4.4 Mapa 9](#_Toc61729125)

[5. Hráč 9](#_Toc61729126)

[6. Nepřátelé 10](#_Toc61729127)

[7. Generace mapy 10](#_Toc61729128)

[8. Hledání cesty 10](#_Toc61729129)

[9. Prostředí 11](#_Toc61729130)

[10. Použité knihovny 11](#_Toc61729131)

[11. Testování 11](#_Toc61729132)

[12. Závěr 12](#_Toc61729133)

[13. Seznam obrázků: 12](#_Toc61729134)

[14. Zdroje 12](#_Toc61729135)

1. Úvod

Rád hraji počítačové hry, a proto jsem se rozhodl vytvořit vlastní hru. Platforma Android byla zvolena, protože se mi zdá, že na ní není mnoho zajímavých her. Vybral jsem si herní engine Unity, protože jsem už v něm pár menších projektů dělal a také umožňuje vytváření mobilních her.

Hráč ovládá postavu rytíře z pohledu třetí osoby. Postava se dokáže pohybovat, vyhýbat útokům, skákat, útočit a sbírat předměty. Hráč může v inventáři měnit vybavení, které mění nejen hodnoty postavy ale i její vzhled.

Nepřátelé neútočí, dokud se nevšimnou hráče a poté ho následují. K vyhledání cesty používají A \* algoritmus. Když se k hráči přiblíží na dostatečnou vzdálenost, tak na něj zaútočí. Boss má několik různých útoků a chodí za hráčem.

Mapy jsou náhodně generovány v několika krocích a skládají se z předem vytvořených částí místností. Ve hře jsou mapy, které obsahují různé úkoly ke splnění.

1. Vlastnosti Unity

Projekt se skládá ze scén, což je v podstatě prostředí, do kterého se umisťují předměty typu Game Object. Game Object je objekt ve scéně, na který lze umístit komponenty a skripty. Většina skriptů je potomkem třídy Mono Behaviour, která nabízí funkce pracovního cyklu herního engine. Dále skripty mohou být potomky třídy Scriptable Object. Ta umožňuje ukládání dat, která mají stálou hodnotu. V projektu také používám particle systémy, které zjednodušují vytváření vizuálních efektů. Další zvláštností v Unity jsou metody typu Coroutine, které umožňují pozastavit v nich prováděný kód.

1. Struktura projektu

Veškerý kód, modely a textury jsou ve složce Assets, a proto popíšu pouze strukturu v ní. Soubory s koncovkou .meta jsou nutné ke správnému fungování hry v prostředí editoru a jejímu sestavení.

* Animation controllers – zde se nachází ovladače animací hráče a nepřátel. Ty určují, které animace u postav se mají přehrávat.
* Editor – obsahuje skripty, které nebudou obsaženy ve hře a slouží k ulehčení práce ve vývojovém prostředí.
* Joystick pack – obsahuje joystick, který používám a je převzatý z Unity Asset Store. (Fenerax Studios, 2021)
* Materials – zahrnuje materiály.
* Models – tento adresář obsahuje 3D modely, textury jejich normálové mapy.
* Prefabs – složka obsahuje objekty, kterým se říká „prefab“. Jsou to v podstatě vzory objektů, podle kterých se během hry vytváří instance objektů.
* Resources – do této složky se dávají objekty, které nelze přiřadit ke konkrétnímu objektu, např. protože ten objekt ještě nebyl vytvořen. Jinými slovy jsou načítány pomocí relativní cesty ve složce Resources.
* Scenes – obsahuje všechny scény v projektu.
* Scriptable Objects – obsahuje skripty scriptable objektů
* Scripts – v této složce se nachází většinu kódu až na scriptable objekty.
* Sprites – obsahuje 2D ikony.
* TextMesh Pro – obsahuje převzaté zdroje sloužící k vylepšení grafického rozhraní.

1. Přehled skriptů

Stručný přehled nejdůležitějších skriptů, které jsem pro hru vytvořil:

* 1. Game manager

Je to singleton, který má na starosti načítání scén a uchovávání odkazů k důležitým objektům jako jsou správce nebo databáze předmětů.

* 1. Menu manager

Spravuje grafické rozhraní, které uživatel uvidí, když spustí hru. V menu může uživatel spravovat uložené hry a nastavení.

* 1. Hub manager

Aktivuje portály a obchod a posílá jim nutné odkazy. Také otevírá portály podle postupu hráče.

* 1. Maze manager

Poté co hráč vstoupí do portálu a je načtena scéna s mapou, začne vytvářet mapu a taky je vytvořena podmínka ke splnění mapy podle nastavení, které je skriptu předáno.

* 1. Game physics

Řeší většinu pohybů, kolizí a gravitační sílu.

* 1. Input manager

Sbírá hráčovy vstupy spojené s ovládáním postavy. Také předává vstupy kameře potřebné k jejímu pohybu a rotaci.

* 1. Player controller

Skript odpovídající za pohyb a akce postavy. Přetváří vstup hráče na akce postavy. Během pravidelných intervalů, daných herním enginem, počítá pozici a rotaci postavy. Také ovládá její animace. Player controller dostavá vstup z Input manageru.

* 1. Camera Controller

Ovládá kameru. Skript je odpovědný za pohyb a rotaci kamery. Kamera vždy zachycuje hráčovu postavu a je nasměrovávána podle okolností. Hráč může kameru ovládat dotykem, zaměřit kameru na nepřátele, nebo se sama otáčí podle pohybu postavy. Camera Controller dostavá vstup ze skriptu Input Manager.

* 1. Maze Generator

Řídí vytváření mapy. Volá jiné skripty v daném pořadí, aby vznikla mapa a předává skriptům odkazy na objekty vytvořené během předchozích kroků.

* 1. Cell Generator

Vygeneruje v obdélníkovém (případně čtvercovém) poli buňky, tak aby všechny byly mezi sebou propojené.

* 1. Subcell Generator

Rozdělí buňky na menší celky. Tyto menší celky buď jsou uspřádány ve tvaru místnosti nebo tvoří chodbu.

* 1. Pathfinding Node Generator

Vytvoří v každé pod-buňce mřížku s vrcholy sloužícími k vyhledávání cesty.

* 1. Tile Generator

Je zodpovědný za většinu vizuální stránky mapy. Poskládá dílky tak, aby dohromady tvořily uzavřenou oblast.

* 1. Spawner

Vytváří instance objektů jako například nepřátele nebo předměty na zemi.

* 1. Enemy Controller

Nepřítel se v každém momentu vyskytuje pouze v jednom stavu, jinými slovy je to konečný automat. Mezi jeho stavy patří pronásledování hráče, útočení, rozhlížení se po hráči atd. Nepřítel je ovlivňován gravitací a řeší kolize.

* 1. Enemy FSM

Skript je zodpovědný za stav protivníka. Obsahuje metody, které jej mění, ukončují starý stav a začínají nový stav. Také volá pravidelně se opakující metody v jednotlivých stavech.

* 1. Boss

Boss funguje podobně jako nepřátelé, to znamená, že je také implementovaný jako konečný automat. Na rozdíl od nepřátel má více druhů útoků a lze vytvářet jejich další kombinace v editoru. Boss také řeší kolize a působí na něj gravitace.

* 1. Pathfinding

Umí nalézt nejkratší cestu z bodu A do bodu B pomocí A \* algoritmu. Také dokáže vygenerovat částečně náhodnou cestu.

* 1. Game Physics

Tato třída obsahuje metody, které řeší pohyb a rotaci objektů, kolize mezi nimi a také to, jestli jsou na zemi. Výše uvedené věci jsou řešeny více způsoby, podle toho, jaký se hodí v dané situaci.

1. Scény

Následující podkapitoly popisují, co vše je možné ve hře dělat. Každá podkapitola popisuje samostatnou scénu, ze kterých se hra skládá. Hra se skládá ze čtyř scén: menu, výběr úrovní, mapa a scéna s hráčem. Při otevření aplikace je spuštěna scéna s menu. V Unity je možné mít více načtených scén najednou.

* 1. Menu

Po zapnutí hry, se hráč ocitne v menu, ze kterého může přejít na výběr uložených her, vstoupit do nastavení nebo vypnout hru. V nastavení si lze upravit rychlost otáčení obrazovkou pro oba směry zvlášť. Výběr úrovní obsahuje textové pole pro zadání názvu nové uložené hry a tlačítko pro její vytvoření. Když jsou na zařízení už uložené postupy ve hře, tak se zobrazí pro každé tlačítko s jeho názvem, které načte tento uložený postup, a také tlačítko, které postup smaže ze zařízení. Během změny úrovní se zobrazuje vždy načítací obrazovka.

* 1. Player

Tato scéna obsahuje hráčovou postavou a také grafické rozhraní s tlačítky, joystickem k ovládání postavy, hráčovým inventářem. V scéně ale není žádné prostředí, a tedy nemůže být načtena jen ona.

* 1. Hub

Ve hře není výběr úrovní ztvárněn jako nudné klikání tlačítek s čísly, ale jako portály, do kterých hráč vstupuje. Další portál se odemkne, když hráč splní úkol v předchozí lokaci. Aktivovaný portál se odlišuje tím, že se v něm točí barevný kruh, který je tvořen particle systémem. V editoru lze ke každému portálu přiřadit nastavení, které popisuje vlastnosti mapy k vygenerování. Ve výběru úrovní se kromě portálu nachází obchodník, který hráči prodá předměty za určitou cenu.

* 1. Mapa

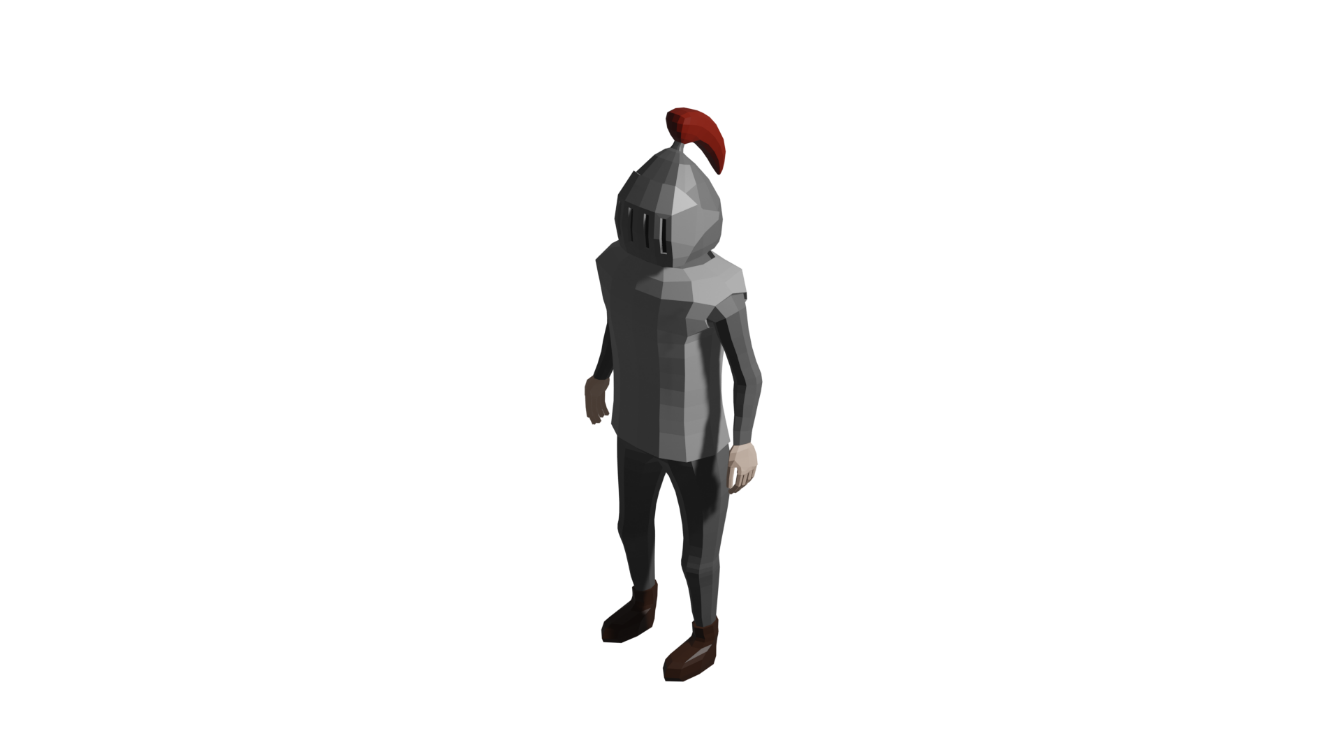
Zde je jediný objekt, na kterém je umístěno několik skriptů, které jsou zodpovědné za vygenerování mapy. Mapa se začne vytvářet až po zavolání příslušné metody v Game Manageru.

1. Hráč

Hráčova postava a s ní související objekty jsou ovládány třemi skripty. První skript (Input Manager) sbírá vstup z tlačítek ve hře a také z dotyku obrazovky. Zpracování kliknutí tlačítka je jednoduché, stačí pouze zavolat další metodu v jiném skriptu. Dotek obrazovky může znamenat jednu ze dvou věcí, buď jde o otáčení kamery, nebo o zaměření kamery na protivníka. Druhý případ se rozezná od prvního tak, že dotek trvá kratší dobu a je v podstatě stále na stejném místě. Input Manager poté předá vstup ovladači hráče a kamery (Player a Camera Controlleru).

Camera Controller udržuje svoji pozici relativně ke hráči. Přejíždění prstem přes obrazovku otáčí kamerou. Kamera se přiblíží k hráči, kdyby měl být nějaká překážka mezi hráčem a kamerou. Po kliknutí na protivníka kamera udržuje v uprostřed zorného pole nejen hráče ale i protivníka, dokud není s kamerou pohnuto.

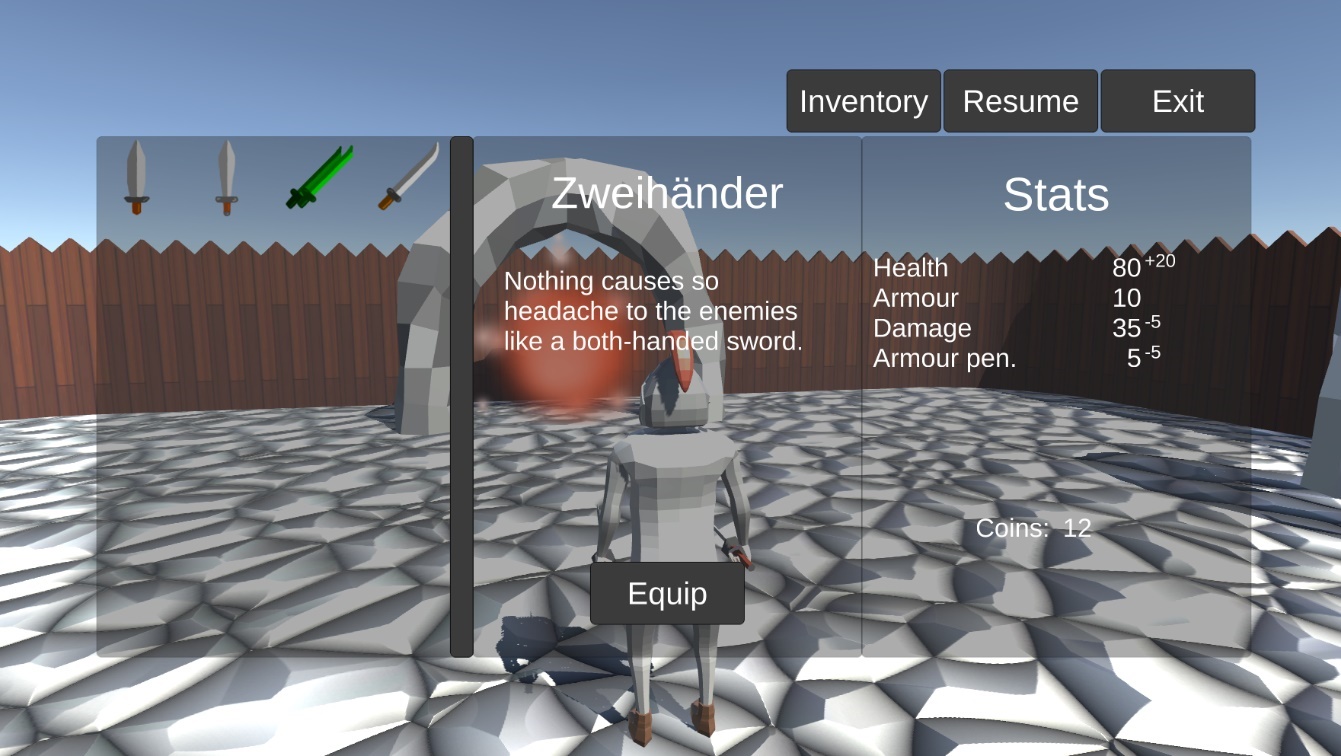
Player Controller ovládá hráčovu postavu. Nejdůležitější funkcí je reagování na hráčův vstup. Postava dokáže měnit pozici a rotaci a vykonávat akce podle hráčova vstupu. V pravidelných intervalech se hráč pohne a otočí podle pozice joysticku a vždy řeší kolize. Dále hráč dostává od Input Manageru informace stisknutí tlačítek, a když je to možné, tak příslušnou akci vykoná. Proveditelnými akcemi je zaútočení, provedení úhybného manévru a vyskočení. V neposlední řadě obsahuje Player Controller také odkaz na hráčovy atributy jako jsou jeho životy a poškození.



Obrázek - Postava ovládaná hráčem

1. Inventář

Skriptem, který ovládá inventář je Inventory Mono Behaviour. Grafické rozhraní se ve hře zobrazí interakcí s obchodem nebo otevřením menu. Hráč i obchod sdílí kód pro zobrazování inventáře, protože fungují velmi podobně. V inventáři jsou vidět atributy hráčovy postavy a předměty v inventáři, které lze vybavit, nebo použít. Struktura předmětů a inventáře byla převzata z následujících zdrojů: (Coding With Unity, 2021) (Coding With Unity, 2021).



Obrázek - Hráčův inventář

1. Nepřátelé

Nepřátele včetně bosse jsou implementováni jako konečné automaty, ale jsou mezi nimi drobné rozdíly ve způsobu implementace. Pro zjednodušení je zde popsán způsob fungování obyčejného protivníka. Jeho implementace se skládá z několika skriptů. Hlavními jsou Enemy Controller, který obsahuje metody, které přímo ovlivňují protivníka, a Enemy FSM, jenž mění stav, ve kterém se nepřítel nachází, a volá metody, které vykonávají akce patřící k danému stavu. Zbylé skripty týkající se protivníka jsou implementace jednotlivých stavů.

Každý protivníka stav je potomkem třídy Enemy State. Když se stav změní, tak je nejprve zavolána metoda On Exit starého stavu, která zakončí probíhající aktivity starého stavu. Poté je stav změněn a je zavolána jeho metoda On Entered, která nastaví proměnné na počáteční hodnoty a např. ještě najdu cestu k hráči. Jednotlivé stavy ještě obashují metody Frame Update a Physics Update. Frame Update je volán při renderování každého snímku a otáčí ukazatelem životů protivníka směrem ke kameře. Metoda Physics Update je volána v pravidelných intervalech daných herním engine a u ní se už implementace liší u jednotlivých stavů. Někdy pohybuje hráčem k cíli a jindy třeba jenom čeká nepřítel na místě a vyhlíží protivníka.



Obrázek – Nepřítel



Obrázek - Boss

1. Generace mapy

Mapa se generuje náhodně v několika krocích. Vytváření mapy je kontrolováno parametry ze Scriptable Object Maze Settings. V něm lze nastavit velikost mapy, počet nepřátel na ní, různé pravděpodobnosti generace a také úkol, který je potřeba splnit k odemčení další mapy. Každá úroveň má vlastní Maze Settings, které lze upravit v editoru.

Nejprve se z Maze Settings převede Enum s vítěznou podmínkou na objekt implementující interface I Win Condition, která může ovlivnit generaci skrz jeho speciální pravidla.

Poté Cell Generator vytvoří pole s buňkami různých šířek a délek. Šířka buňky je dána náhodným číslem v rozmezí, které je specifikováno v Maze Settings, a je uložena v poli. Totéž platí pro délku. Buňky jsou vytvářeny tak, že se do zásobníku vloží pozice okolo první buňky, která je ze všech stran otevřená a dokud není zásobník prázdný, tak se opakuje následující cyklus. Vytvoří se nová buňka na pozici vyndané ze zásobníku a zkopíruje zdi okolních buněk. U stran, kde ještě neexistuje buňka, se podle pravděpodobností uvedených v Maze Settings rozhodne, jestli bude z té strany buňka otevřená. A nakonec se do zásobníku vloží sousední pozice s ještě nevygenerovanými buňkami. Cell Generator vrací třídu Cell Data, která obsahuje vše, co bylo v něm vygenerováno.

V dalším kroku pracuje Subcell Generator s Cell Data a podle nich a opět Maze Settings rozdělí buňky na menší celky, které nazvu pod-buňky. U každé buňky, kromě první, se rozhoduje podle pravděpodobnosti v Maze Settings, jestli budou pod-buňky v uspořádání do místnosti nebo do chodby. Chodby na rozdíl od místností se skládají pouze z pod-buňek, které propojují sousední buňky. I zde se uplatní zásobník, do kterého se vloží první buňka a dokud není zásobník prázdný, tak se vyjme první buňka ze zásobníku, ta se rozdělí na pod-buňky a vloží do zásobníku své sousedy. Maze Generator obdží zpět třídu Subcell Data, která obsahuje informace o vytvořených pod-buňkách. Poté se případně ještě vytvoří instance místnosti s bossem, když je to v pravidlech generace u aktuální I Win Condition.

Na řadu přijde Tile Generator, který k práci potřebuje data z Subcell Data z předchozího kroku. Tile Generator vytvoří na místě každé pod-buňky instanci části místnosti podle propojených okolních pod-buňek. Části místnosti můžou být náhodně zvoleny, když je na výběr z více setů.

Předposledním krokem je vytvoření mřížky k hledání cesty pro nepřátele. V Maze Settings je uvedeno kolik vrcholů pro hledání cesty by mělo být v jedné pod-buňce. Ve hře je pod-buňka rozdělena na 25 vrcholů ve čtvercové síti. Výsledné vrcholy jsou předány Pathfinder skriptu, který slouží k vyhledávání cest.

Nakonec je mapa zaplněna protivníky. Minimální a maximální počet protivníku je nastavitelný v Maze Settings, ale je také možné jejich počet zadat jako pravděpodobnost, u které se pokouší vytvořit instanci nepřítele na každém vrcholu k hledání cesty. I Win Condition, také může ovlivnit to, že se místo protivníka vytvoří instance předmětu.

1. Hledání cesty

K vyhledávání cesty je použit A \* algoritmus, který se podobá Dijkstrovu algoritmu. Oproti němu obsahuje heuristický prvek. Popis algoritmu:

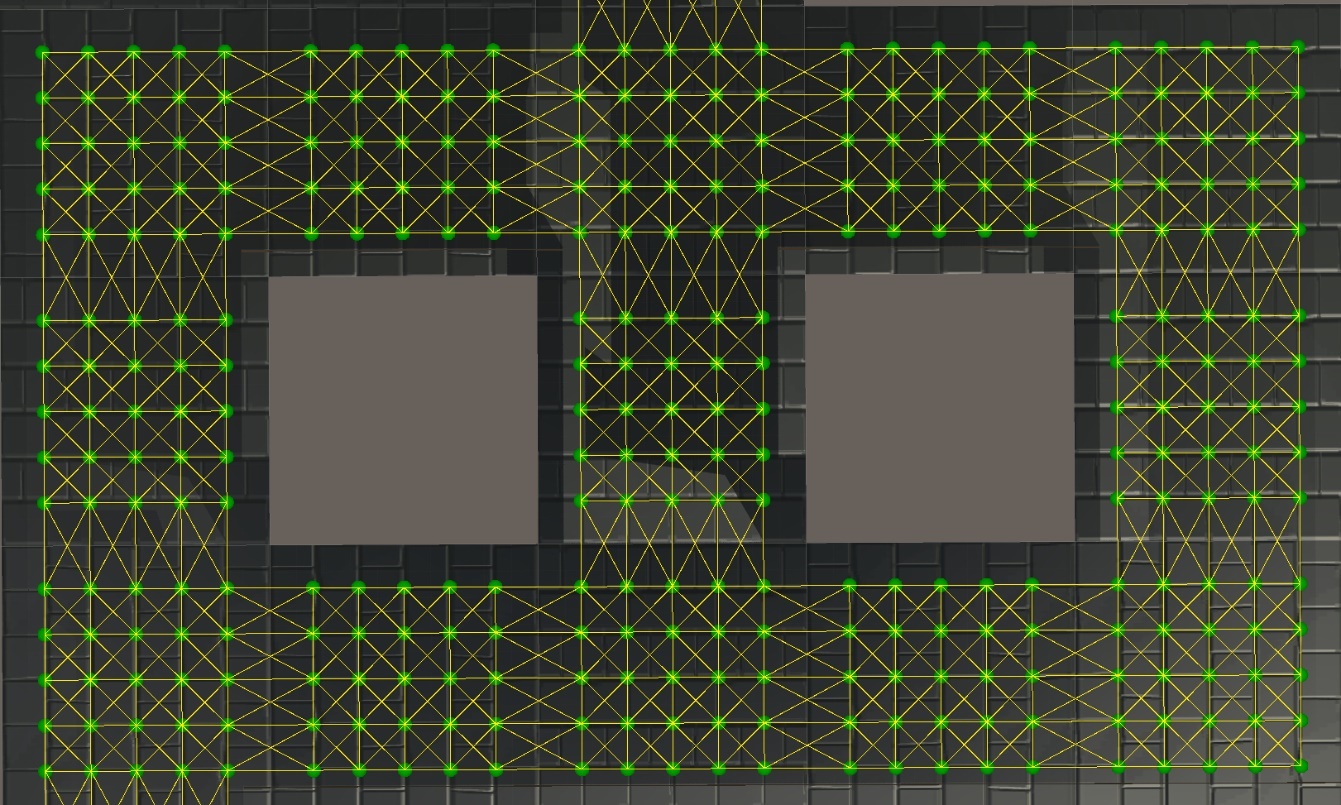
Na počátku jsou dvě množiny. První lze nazvat otevřenou množinou, protože obsahuje zpracovávané prvky. Ta obsahuje zprvu počáteční uzel. Druhá množina se nazývá uzavřená a jsou do ní vkládány prvky, které nabyly konečnou hodnotu. Dokud není otevřená množina prázdná, tak se opakuje následující posloupnost příkazů:

1. Z otevřené množiny se vyjme prvek X s nejmenší hodnotou F
2. Když je X cílovým prvkem, tak je cesta nalezena
3. X se z otevřené množiny přesune do množiny uzavřené
4. Pro všechny sousedy prvku X proběhne následující cyklus:
   1. Když je soused v uzavřené množině, tak se přeskočí
   2. Když není v otevřené množině, tak je do ní přidán a spočítá se jeho F hodnota
   3. Když je v otevřené množině, tak je případně jeho hodnota F nahrazena novou nižší hodnotou

Kde hodnota F je součet vzdálenosti od počátku (G) a odhadované vzdálenosti od konce (H).

Implementace haldy ve hře je dále optimalizovaná tím, že je otevřená množina tvořena haldou, takže vyhledání vrcholu s nejmenší hodnotou je velmi rychlé.

Na následujícím obrázku je znázorněna síť vrcholů pro vyhledávání cesty. Každý úsek chodby obsahuje čtvercovou síť vrcholů, které jsou mezi sebou propojeny.



Obrázek - Vrcholy k vyhledávání cesty a jejich propojení

1. Modely

Všechny modely ve hře jsem vytvářel já. Jsou vytvořeny v programu Blender. Blender je bezplatný open-source 3D nástroj určený pro vytváření kreslených filmů, vizuálních efektů a 3D modelů.

Modely byly vytvářeny v několika krocích. Nejprve byl vytvořen model skládající se z vrcholů a stěn. Poté byla vytvořena pro model specifická textura, nebo přiřazeny barvy stěn z barevné palety. Když se u modelu jednalo o postavu, která se ve hře pohybuje, bylo potřeba pro ni vytvořit kostru a animace. Modely byly nakonec exportovány do Unity ve formátu FBX.

1. Prostředí

Hra je napsána v jazyce C#, protože je to jeden z jazyků, které podporuje herní engine Unity (druhým je Unityscript). Herní engine Unity byl vytvořen Unity Technologies původně pro vytváření her výhradně pro operační systémy OS X, ale nyní je multiplatformní. IDE mého výběru je Microsoft Visual Studio.

1. Testování

Testování hry by se dalo rozdělit na dvě fáze. V první fázi byly odhalovány chyby a bugy v programu. Mezi ně například patří detekce objektů na špatné vrstvě nebo nekonečný cyklus při změně stavu protivníka. Druhá část testování obsahovala úpravu hodnot hráče a protivníků, aby hra byla hratelná. Hratelnost hry byla testována v Unity editor a na mobilním zařízení Samsung Galaxy S10 s verzí Android 10.

1. Instalace

Pro instalaci hry stáhněte .apk soubor z GitHub repozitáře ve složce Build. Poté připojte mobilní zařízení s operačním systémem Android. Ujistěte se, že je povolen přenos souborů na mobilním zařízení. Poté klikněte pravým tlačítkem myši na stáhnutý soubor typu apk, najeďte na možnost odeslat a vyberte připojené zařízení. Na mobilu povolte instalaci aplikací z neznámých zdrojů, najděte aplikaci v paměti na mobilním zařízení a nainstalujte ji.

1. Závěr

Podařilo se mi vytvořit funkční aplikaci, která běží na platformě Android. Hráč může ovládat svoji postavu podle od něj obdrženého vstupu. Hráč má také inventář, ve kterém si může vybavovat předměty, které lze si zakoupit v obchodě.

Nepřátelé jsou implementováni jako konečné automaty. Rozhodují se pro různé akce a ty také vykonávají. K hledání cesty používají optimalizovaný A \* algoritmus. Hráč se také utká s bossem, kterého potřebuje porazit k dokončení hry.

Po programovací stránce projektu jsem s ním spokojen. Kromě naprogramování kódu, jsem vytvořil i modely pro hráče, protivníky atd.

1. Seznam obrázků:

[Obrázek 1 - Postava ovládaná hráčem 10](#_Toc62410524)

[Obrázek 2 - Hráčův inventář 11](#_Toc62410525)

[Obrázek 3 – Nepřítel 12](#_Toc62410526)

[Obrázek 4 - Boss 12](#_Toc62410527)

[Obrázek 5 - Vrcholy k vyhledávání cesty a jejich propojení 14](#_Toc62410528)

1. Zdroje

*Blender*. (22. Leden 2021). Načteno z Blender: https://www.blender.org/

Coding With Unity. (22. Leden 2021). *Unity3D - Saving and Loading your Inventory with Scriptable Objects | Part 2*. Načteno z YouTube: https://www.youtube.com/watch?v=232EqU1k9yQ

Coding With Unity. (22. Leden 2021). *Unity3D - Scriptable Object Inventory System | Part 1*. Načteno z YouTube: https://www.youtube.com/watch?v=\_IqTeruf3-s

Fenerax Studios. (22. Leden 2021). *Joystick Pack*. Načteno z Unity Asset Store: https://assetstore.unity.com/packages/tools/input-management/joystick-pack-107631

glitchers. (22. Leden 2021). *Answers: Unity*. Načteno z Unity: https://answers.unity.com/questions/486545/getting-all-assets-of-the-specified-type.html

Imphenzia. (22. Leden 2021). Načteno z Dropbox: https://www.dropbox.com/s/c5olic38j8fopet/ImphenziaPalette01.png

*Unity*. (21. Leden 2021). Načteno z Unity: https://unity.com/