**STŘEDNÍ PRŮMYSLOVÁ ŠKOLA BRNO, PURKYŇOVA, PŘÍSPĚVKOVÁ ORGANIZACE**

Obsah obrázku Písmo, text, Grafika, grafický design

Popis byl vytvořen automaticky

**ASTRA**

**ŠIMON ZELINKA**

**V4C**

**PROFILOVÁ ČÁST MATURITNÍ ZKOUŠKY**

**MATURITNÍ PRÁCE**

**BRNO 2024**

# Prohlášení

Prohlašuji, že jsem maturitní práci na téma Astra vypracoval samostatně a použil jen zdroje uvedené v seznamu literatury.

Prohlašuji, že:

Beru na vědomí, že zpráva o řešení maturitní práce a základní dokumentace k aplikaci bude uložena v elektronické podobě na intranetu Střední průmyslové školy Brno, Purkyňova.

Beru na vědomí, že bude má maturitní práce včetně zdrojových kódů uložena v knihovně SPŠ Brno, Purkyňova, dostupná k prezenčnímu nahlédnutí.

Beru na vědomí, že SPŠ Brno, Purkyňova má právo celou moji práci použít k výukovým účelům a po mém souhlasu nevýdělečně moji práci užít ke své vnitřní potřebě.

Beru na vědomí, že pokud je součástí mojí práce jakýkoliv softwarový produkt, považující se za součást práce i zdrojové kódy, které nejsou předmětem maturitní práce, případně soubory, ze kterých se práce skládá. Součástí práce není cizí ani vlastní software, který je pouze využíván za přesně definovaných podmínek, a není podstatou maturitní práce.

Šimon, Zelinka

Žarošice 386, 696 34

Dne: Podpis:

# Poděkování

Rád bych poděkoval všem testerům, kteří mi pomohli vychytat všechny chyby v programu, jmenovitě Ondřeji Buchlovskému a Ondřeji Němcovi, který pro tento projekt také vytvořil zvukovou stopu.

# Anotace

Cílem této maturitní práce je vytvoření dvojrozměrné počítačové videohry v prostředí vesmíru. Hráč se pohybuje v otevřeném světě a není nijak omezen. Má možnost prozkoumávat vesmír nebo může plnit úkoly, a tak za ně získat energetické jednotky k postavení a rozšíření své lodi.

# Obsah

[Seznam použitých zkratek 7](#_Toc164247759)

[Úvod 8](#_Toc164247760)

[1 Rozbor řešení 9](#_Toc164247761)

[1.1 Použité technologie 9](#_Toc164247762)

[1.1.1 Godot 2D 9](#_Toc164247763)

[1.1.2 Aseprite 9](#_Toc164247764)

[1.1.3 Programovací jazyk GDScript 9](#_Toc164247765)

[1.1.4 Programovací jazyk C# 9](#_Toc164247766)

[1.2 Hlavní body hry 10](#_Toc164247767)

[1.2.1 Smysl hry 10](#_Toc164247768)

[1.2.2 Ovládání 10](#_Toc164247769)

[1.2.3 Zorné pole 11](#_Toc164247770)

[1.2.4 Herní úkoly 11](#_Toc164247771)

[1.2.5 Logika NPC 12](#_Toc164247772)

[1.2.6 Jedinečnost NPC 13](#_Toc164247773)

[1.2.7 Tvorba grafiky 13](#_Toc164247774)

[1.2.8 UI 14](#_Toc164247775)

[1.2.9 Zvuk 14](#_Toc164247776)

[1.2.10 Editor 15](#_Toc164247777)

[1.2.11 Menu 16](#_Toc164247778)

[2 Experimentální část 17](#_Toc164247779)

[2.1 Úvod do pracovního postupu v Godotu 17](#_Toc164247780)

[2.1.1 Herní objekty a komponenty 17](#_Toc164247781)

[2.1.2 Programování v Godotu 17](#_Toc164247782)

[2.2 Realizace systému lodě 18](#_Toc164247783)

[2.3 Části lodě 20](#_Toc164247784)

[2.3.1 Stěna 20](#_Toc164247785)

[2.3.2 Dveře 20](#_Toc164247786)

[2.3.3 Trysky 20](#_Toc164247787)

[2.3.4 Konektor 21](#_Toc164247788)

[2.3.5 Jádro 21](#_Toc164247789)

[2.3.6 Stanoviště pro stavbu lodi 21](#_Toc164247790)

[2.3.7 Kormidlo 21](#_Toc164247791)

[2.4 Schvalovač lodí 22](#_Toc164247792)

[2.5 Ukládání a načítání 23](#_Toc164247793)

[2.6 Testování 23](#_Toc164247794)

[3 Manuál 25](#_Toc164247795)

[3.1 Minimální požadavky 25](#_Toc164247796)

[3.2 Instalace 25](#_Toc164247797)

[3.3 Ovládání 26](#_Toc164247798)

[3.3.1 Ve hře 26](#_Toc164247799)

[3.3.2 V editoru 26](#_Toc164247800)

[Závěr 27](#_Toc164247801)

[Seznam ilustrací 29](#_Toc164247802)

[Odkazy 30](#_Toc164247803)

# Seznam použitých zkratek

NPC – Non-Player Character (nehráčská postava)

UI – User Interface (uživatelské rozhraní)

2D – Two Dimensional (dvojdimenzionální)

OST – Original SoundTrack (originální hudba)

# Úvod

Videohry, jako forma interaktivního softwaru, jsou stále populárnější nejen pro svou zábavnost, ale také pro schopnost přenášet určité hodnoty a příběhy. Tato digitální forma umění je v současnosti oblíbeným způsobem trávení volného času u mnoha lidí po celém světě. S rostoucím počtem vytvářených her vzniká také výzva vytvořit unikátní a odlišné projekty, které upoutají pozornost hráčů.

Moje rozhodnutí zaměřit maturitní práci na tento fenomén vyplývá z mé hluboké fascinace digitálním světem a tvorbou her, která mě baví více než samotné hraní her. Celý proces vývoje hry mi umožňuje nejen předat hráčům určité poselství skrze hratelný obsah, ale také se snažím vytvořit prostředí, které je inovativní a esteticky přitažlivé, což mě osobně obohacuje a poskytuje mi nové perspektivy v oblasti digitálního designu a programování.

Rozhodnutí zaměřit se na tvorbu videoher mě umožňuje propojit technické dovednosti s kreativním vyjádřením, což je pro mě jako pro vývojáře a člověka, který se vždy nejen o digitální tvorbu zajímal, velmi důležité. Tento projekt mě nejen vyzývá k technické preciznosti, ale také mi dává prostor pro vyprávění příběhů a tvorbu jedinečných herních světů, které mohou hráče emocionálně oslovit. S každou novou hrou, kterou vytvořím, se snažím překročit běžné hranice herního designu a nabídnout hráčům nejen zábavu, ale i zážitky, které mohou vést k hlubšímu přemýšlení nebo diskusi o daných tématech. Tento přístup k tvorbě her mi umožňuje neustále se rozvíjet a udržovat si vášeň pro tento dynamicky se vyvíjející průmysl.

# Rozbor řešení

## Použité technologie

### Godot 2D

V rámci této maturitní práce jsem se rozhodl využít herní engine Godot, který se vyznačuje širokým spektrem vestavěných nástrojů. Vyžaduje minimální místo na disku pro instalaci, což z něj činí efektivní a nenáročnou volbu pro tvorbu videoher. Je open-source a naprosto zdarma i pro komerční účely. Jeho komunita v posledních letech rapidně vzrostla, pokud si tedy tvůrce neví s nějakým problémem rady, nic mu nebrání požádat na oficiálních fórech o pomoc od ostatních nadšenců.

Za dobu deseti let, po kterou se tento engine neustále vyvíjí, bylo také vytvořeno mnoho rozšířeni, které může vytvářet kdokoliv, kdo chce podpořit tvorbu ostatních. Já jsem tyto rozšíření nepoužil a všechno jsem naprogramoval od základů.

### Aseprite

Pro tvorbu grafických prvků jsem zvolil Aseprite. Tento bitmapový editor, určený především pro tvorbu ve stylu pixel art, je naprosto zdarma a pro mé účely ideální. Umožňuje export v mnoha používaných formátech, z jejichž jsem využil formát *.png*, který umožňuje průhlednost, jenž je v tomto projektu nezbytná. Pomocí tohoto nástroje jsem vytvořil veškeré 2D textury, UI prvky i ikony.

### Programovací jazyk GDScript

K vytváření skriptů v enginu Godot je možno si vybrat ze tří programovacích jazyků: C#, C++ a GDScript.

### Programovací jazyk C#

Tento jazyk jsem využil čistě ze zvědavosti a za účelem zjistit, jestli je pro mě výhodnější psát všechny skripty ve složitějším jazyce – C# nebo v jazyce, který je jednodušší – GDScript.

Využívání více programovacích jazyků při skriptování se může v mnoha herních enginech stát překážkou, toto ovšem zde není pravdou. Kombinování jazyků je zde podporováno a já jsem této možnosti krajně využil. Je zde možno využívat funkcí z jednoho jazyku ze souborů jazyku druhého.

Nakonec jsem ale vše přepsal do jednoho jazyku, pro přehlednost a konzistentnost celého projektu.

## Hlavní body hry

### Smysl hry

Když jsem přemýšlel, jaké téma bude mít tato hra, okamžitě mě napadl vesmír, nekonečná a neprozkoumaná plocha záhad, plátno, na které mohu vytvořit cokoliv, co mě napadne. Vybranému tématu již stačilo vytvořit smysl, kterým se staly herní úkoly.

K tomuto tématu jsem byl z velké části inspirován hrou z roku 2003 názvem Space Station 13, která se podobně jako tento projekt odehrává na vesmírné stanici, pohled kamery je shora a loď je celá vytvořena pomocí dlaždic.

2D styl a pixelartovou grafiku jsem vybral hlavně kvůli své lehké implementaci a jednoduchosti. K vytvoření tomuto stylu grafiky není potřeba veliké nadání, aby byl finální produkt alespoň trochu prezentovatelný. Dvojdimenzionální formát jsem zvolil kvůli nenáročnosti tvorby a zobrazení.

K ovládání lodi mě inspirovala horrorová videohra s názvem Barotrauma, ve které hráč prozkoumává oceán Europy, měsíce Jupiteru, za pomocí své posádky a plní různé úkoly, které mu zadávají NPC, se kterými na různých stanovištích může komunikovat.

Hra je vytvořena ve stylu *sandbox* (v překladu „pískoviště”) a *exploration* (v překladu „průzkum”). Samotná hra se skládá ze dvou částí, každá z nich plní jeden styl. Stavění lodi a plnění úkolů.

První část, se kterou se setkáme, když zapneme hru je průzkumná část, kdy se hráč objeví na hlavní stanici a potkává první NPC, které mu zadá jeho první úkol. Pomocí těchto úkolů hráč získává energetické jednotky, mnou zvolenou měnu.

Stavění lodi představuje sandboxovou část této práce, kdy hráč po prozkoumání postupně vylepšuje a rozšiřuje svou původně malou vesmírnou loď. K pořizování jednotlivých dílů utrácí měnu získanou v první části. Čím lepší loď, tím jednodušší je pro hráče úkoly plnit. Tímto se spojují tyto dvě části.

### Ovládání

Jako hlavní prvky ovládání, co se týče směru chůze a letu lodi, jsem vybral klasický způsob pomocí kláves W, A, S, D, který je v moderních videohrách velmi rozšířený. Jelikož jsem do své hry v zhruba polovině vývoje přidal další aspekt – otáčení, potřeboval jsem přiřadit nějaké klávesy k této nové funkci Vybral jsem klávesy Q a E, jelikož intuitivně určují směr otáčení ovládaného objektu.

Ovládání různých objektů, například kormidla lodě, je nastaveno na klávesu F. K tomuto jsem se opět inspiroval, tentokrát ze známé hry Grand Theft Auto V, která má na tomto tlačítku nastavené nastupování a vystupování z vozidel. Loď funguje podobným způsobem jako ovládání hráče s tím rozdílem, že pokud na loď působíme nějakým směrovým vektorem a následně na ni přestaneme působit, zůstane její rychlost konstantní, jelikož ve vesmíru není žádný odpor. Toto ovšem neplatí pro vektor otáčení, kdy se loď po ukončení rotace sama ustálí do relativního klidu.

Hráč má možnost se pohybovat po lodi chůzí, kde je ovšem rotace zablokována, aby nedošlo k zbytečným chybám typu NPC je natočeno do jiného směru než hráč nebo k rozladění hitboxů (v překladu „krychle zásahu“).

Když ovšem opustí oblast vesmírné lodi, přepne se hráč do módu *floating* (v překladu „plovoucí“), ve kterém se povolí otáčení a pohyb funguje stejně jako pro vesmírné plavidlo. Pokud je ale mimo loď, rapidně mu začnou klesat životy. Když hráč dosáhne nuly životů, zemře. Poté se znovuzrodí u jádra načtením nejnovějšího uložení hry.

### Zorné pole

Jelikož by nebylo ideální, aby hráč viděl všechno ve svém dosahu, přidal jsem do hry světla. Hráč má k sobě připevněný zdroj světla a všechny stěny mají mezi svými komponenty prvek, který vytváří stín. Poté na objekty, které nemají být přes stěny vidět, připevňuji shader, který ho při neosvětlení zneviditelní.

### Herní úkoly

Herní úkoly jsou jedním z hlavních aspektů tohoto projektu. Jsou realizovány přes NPC, které se mohou objevit v lodích napříč vesmírem.

Většina úkolů je realizována způsobem *go and fetch* (v překladu „běž a přines”), který je jeden ze tří typů mise ve hře. Další z těchto typů je muset si promluvit s nějakým dalším NPC.

Cíle úkolů mohou být buďto vytvořeny specificky pro misi na náhodném místě vzdáleném od hráče přibližnou vzdálenost, která závisí na obtížnosti úkolu, nebo jako cíl úkolu může být vybrán objekt, který již ve světě existuje.

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyVeškeré konverzace jsou uloženy ve slovníku. Poté jsou rozděleny do 3 kategorií: pozdravy, mise a odpovědi na splněné mise. Když se NPC načte, má 25% šanci na to, že bude mít nachystanou misi pro hráče. Všechny NPC obnovují své konverzace v intervalu pěti minut. Pokud tedy hráč začne konverzaci s NPC, které má již nachystanou misi, vybere se náhodný úkol z dané kategorie. Každá mise má svůj index a je spuštěna, když se v seznamu řetězců naskytne číslo.

Obrázek 1 - Konverzace

Obsah obrázku text, snímek obrazovky, Písmo

Popis byl vytvořen automatickyMise jsou uloženy ve složce v souboru pojmenovaném po indexu mise pro přehlednost. Navazující mise jsou uloženy po indexu mise, na kterou mají navazovat s tisícinou mezerou určující index navazující mise.

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyKdyž je úkol splněn promluvou s NPC, kterou ji zadalo, vyhledá se v kategorii odpovědí na mise index, pokud se nenalezne, spustí se výchozí odpověď, jinak se spustí odpověď se stejným indexem.

Obrázek 2 - Složka s misemi

Obrázek 3 - Navazující konverzace

Díky tomuto způsobu lze řetězit více úkolů za sebou, mohou takto vznikat příběhové prvky.

### Logika NPC

Hlavním úkolem NPC je zadávat hráči úkoly. K tomuto ale potřebujeme první projít několik kroků. K prvnímu z těchto kroků nastává při načtení hry, kdy se načítají i jednotlivé vybrané úkoly u každého NPC. Pro každou postavu ovládanou počítačem se spustí funkce *random\_mission\_id* s parametrem *can\_return\_empty\_quest* na *true*.S tímto parametrem je šance 25%, že funkce vrátí prázdný úkol, tudíž se uloží do vybraného úkolu *-1.* To znamená, že když se hráč přiblíží, spustí se jenom pozdrav a žádný úkol se nenabídne.

Pokud ovšem nastane druhá možnost, tedy zbylých 75%, spustí se generátor náhodného indexu mise. Tento generátor vezme seznam rolí, které má dané NPC a cyklem projede všechny existující mise. Pokud mise je vhodná pro nějakou roli, která je v seznamu a zároveň není již zařazena do statického seznamu zablokovaných misí, přidá se do seznamu dostupných misí, ze kterého se po ukončení cyklu vybere náhodný index.

Pokud se hráč začne konverzaci, mohou nastat tři typy dialogů. Pokud není dostupná žádná mise, spustí se pozdrav. Pokud nějaká mise dostupná naopak je, spustí se dialog určený k misi. Třetí možnost je ta, že hráč splnil úkol vrací se k NPC pro odměnu, v tuto chvíli se spustí konverzace ze seznamu dokončené mise.

### Jedinečnost NPC

Pro vytvoření jedinečnosti mezi NPC ve své hře jsem implementoval systém, kde každé NPC je při prvním načtení hry unikátně generováno s různými vizuálními atributy. Pro každého NPC je náhodně stanovena barva jednotlivých částí těla, včetně vlasů, těla, nohou, kůže a očí, což zajistí vizuální rozmanitost. Dále je každému NPC přiřazen jeden ze sedmi možných účesů, čímž se zvyšuje jejich rozpoznatelnost a diverzita. Jména NPC jsou také vybírána náhodně z předem připraveného seznamu, což každé postavě dodává další vrstvu individuality. Všechny tyto charakteristiky jsou uloženy při ukládání a jsou obnovovány pokaždé, když je hra načtena.

### Tvorba grafiky

Jako jediný výtvarník na projektu jsem byl odpovědný za vytvoření veškerých vizuálních prvků hry, což zahrnovalo i design a produkci všech textur. Jako grafický styl jsem si zvolil pixel art, nejen kvůli své estetické oblíbenosti a retro kouzlu, ale také kvůli jeho schopnosti vyvolat nostalgičnost, a přitom poskytnout čistotu a jednoduchost, která se skvěle hodí k hernímu designu.

S cílem zachovat jednotný vzhled a pocit celé hry, jsem se rozhodl vytvořit tři hlavní *tilemapy* (mřížka textur dlaždic):

První tilemapa se skládá z různých variant zdí a segmentů lodi, jako jsou trysky a dveře, což hráči umožňuje nejen prozkoumat rozmanité prostředí, ale také postupně budovat a modifikovat vlastní vesmírnou loď.

Druhá tilemapa je vytvořena pro objekty na lodi, které slouží nejen k estetickým účelům, ale také mohou mít funkční význam, například místo pro načtení NPC a předmětů.

Třetí tilemapa obsahuje herní předměty, od surovin až po nástroje a zařízení, které hráči využijí k plnění misí během svého dobrodružství.

### UI

Uživatelské rozhraní v jakékoliv videohře hraje zásadní roli v tom, jak jsou hráči schopni vnímat a interagovat se světem hry. V rámci mého projektu jsem kladl zvláštní důraz na to, aby bylo uživatelské rozhraní intuitivní a zároveň informativní, poskytující hráčům nezbytné informace bez zbytečného přetížení nebo zmatení.

Obsah obrázku Grafika, grafický design, Písmo, logo

Popis byl vytvořen automatickyObsah obrázku pixel, Grafika, symbol, snímek obrazovky

Popis byl vytvořen automatickyJedním z nejvýznamnějších prvků, které UI zobrazuje, je měna ve hře. Je nezbytné, aby hráči měli neustále přehled o množství finančních prostředků, které mají k dispozici, jelikož to umožňuje strategické rozhodování, například při nákupu vylepšení. Měna je zobrazena na snadno viditelném místě rozhraní a je aktualizována v reálném čase, aby odrážela současný stav zdrojů hráče.

Obrázek 4 - Životy

Obrázek 5 - Měna

Dalším klíčovým aspektem, který UI musí efektivně komunikovat, jsou aktivní mise. Ukazatel mise poskytuje hráčům přehled o jejich cílech, úkolech a postupu. Tato část rozhraní je navržena tak, aby byla snadno přístupná, a přitom nenarušovala základní hratelnost nebo vizuální prezentaci hry. Umožňuje hráčům rychle zkontrolovat své cíle, aniž by museli opustit hlavní herní obrazovku.

### Zvuk

Součástí mé práce byla i implementace zvuku. Veškeré zvukové záznamy pro mě vytvořil Ondřej Němec, na mně zbývala již jen logika zvuku a její design v samotné hře. Hudbu v pozadí tvoří necelý dvaceti pěti minutový OST puštěných ve smyčce bez žádných dalších úprav.

Jelikož jeden z důležitých prvků tohoto projektu je atmosféra, bylo zřejmé použít poziční zvuk pro zvukové efekty. Jelikož teď hráč ví z kudy se zvuk ozývá, zbývá jen přidat další důležitý efekt a tím je nepatrná ozvěna. Pokud toto všechno dáme dohromady, vyjde nám z toho až trochu znepokojující, ale záhadný zvukový design, který se pro tuto hru podle mě velice hodí.

### Editor

Důležitou částí tohoto projektu je editor lodí, který umožňuje hráči si vytvářet a modifikovat loď podle svých představ.

První částí tohoto editoru je pole, kde hráč pokládá jednotlivé stavební bloky. Každý stavební blok má své vlastnosti a podmínky pro položení. Některé se dají položit jen na určitý materiál, některé mají jen omezený počet instancí v lodi.

Druhou částí je inventář, který se hráči zobrazí při zmáčknutí klíčového tlačítka tabulátor. Zde se nachází seznam veškerých použitelných bloků s popisem, který se skládá ze jména komponentu a z ceny, která je nutná k pořízení a položení bloku.

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyEditor také musí dokázat ukládat lodě do souboru, specificky do souboru s přílohou *.dat*, do které se postupně ukládá každá část lodi. Loď můžeme poté načíst zpátky.

Obrázek 6 - Skript k ukládání lodě do souboru

### Menu

Tvorba menu byla relativně jednoduchou částí projektu. Úvodní menu obsahuje několik tlačítek, které slouží jako vstupní body do různých částí hry. Tlačítko „Hrát“ (1) umožňuje hráčům pokračovat v již existující hře, zatímco tlačítko „Nová hra“ (2) nejdříve vymaže veškeré uložené údaje před spuštěním stejné herní scény, což hráčům poskytuje možnost začít od začátku.

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automatickyPozadí (6) v hlavním menu jsou dva obrázky hvězd přeložené přes sebe, bližší obrázek je z většiny průhledný. Oba k sobě mají připojený shader, který postupem času posouvá jejich texturu. Oba mají různou rychlost a tím zde vzniká hloubka pozadí.

Obrázek 7 - Skript k posouvání pozadí

Další důležitou součástí hlavního menu je sekce nastavení (3), kde mohou hráči upravit hlasitost zvukových efektů a hudby podle svých preferencí. Tato možnost umožňuje uživatelům lepší kontrolu nad svým audiovizuálním zážitkem ve hře, což je klíčové pro pohodlí a osobní přizpůsobení herního prostředí.

Poslední částí menu je sekce *Credits* (význam slova – poděkování) (4), kde se nachází seznam lidí, kteří mi s tvorbou projektu pomáhali v různých částech jako například při tvorbě zvuku, nebo co se testování a ladění chyb týče. Ukončení hry provedeme tlačítkem „Odejít“ (5).



Obrázek 8 - Menu

# Experimentální část

## sfhfghÚvod do pracovního postupu v Godotu

### Herní objekty a komponenty

Herní engine Godot funguje na principu takzvaných *nodes* (v překladu „uzlů”), tyto nodes jsou základní stavební jednotkou celého projektu. Z této základní jednotky dědí skoro všechny stavební bloky a rozšiřují jejich funkčnost.

Každá node má své vlastnosti a čím více jsou rozšířené, tím více upravitelných vlastností nabízí. Na jakýkoliv z nich můžeme vložit svůj skript.

Scény v Godotu fungují jako kontejnery pro skupiny nodes, uspořádané do hierarchické struktury. Tato struktura umožňuje vývojářům organizovat a spravovat složité herní světy s vysokou úrovní přehlednosti. Například, scénu lze rozdělit na podscény, jako jsou menu, editor nebo interaktivní objekty, jako jsou dveře nebo NPC. Scény a podscény lze snadno znovu použít nebo kombinovat, což umožňuje efektivní vývoj a snižuje potřebu opakovaného kódování.

Obrázek 9 - Komponenty

### Programování v Godotu

Godot Engine používá GDScript, jazyk inspirovaný Pythonem, který je speciálně navržen pro vysokou integraci s Godotovou architekturou a zjednodušení herního vývoje. Skripty v GDScriptu se připojují k nodes a rozšiřují jejich funkčnost. Například, přidáním skriptu k node reprezentujícímu hráče můžeme přidat pohybové schopnosti interakce s objekt.

Díky tomuto dědění nyní můžeme využívat všechny funkce, které nám tento node umožňuje. Jednou z těchto funkcí je například *move\_and\_slide*, díky které můžeme plynule pohybovat s hráčem.

GDScript je dynamicky typovaný jazyk, proměnné se zde vytvářejí pomocí klíčového slova „var“, při vytvoření proměnné však můžeme staticky nastavit její typ pomocí „:“. Proměnnou můžeme také označkovat například klíčovým slovem Obsah obrázku text, snímek obrazovky, Písmo, řada/pruh

Popis byl vytvořen automaticky„@export” a zobrazit ji tak v samotném Godotu.

Obrázek 10 - Exportovaná proměnná v editoru

Obrázek 11 - Proměnná

Obsah obrázku text, snímek obrazovky, Písmo, řada/pruh

Popis byl vytvořen automatickyPokud nechceme skript připevnit k žádnému nodu, můžeme ho vytvořit bez zdědění ze třídy node, toto se ovšem používá jen ve krajních případech, kdy chcete zjednodušit a zpřehlednit soubory. Já jsem to například použil při tvorbě třídy *ShipValidator* (v překladu „schvalovač lodí“), který slouží v editoru ke zkontrolování, zda loď splňuje podmínky k uložení a případně se je pokouší opravit.

Obrázek 12 - Identifikátor třídy

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyZda je tedy potřeba vytvořit nějaký typ skriptu, který nevyžaduje scénu, je mnohem spolehlivější a účinnější využít nativní třídu *Resource* (v překladu „zdroj“). Tento typ skriptu si můžete nastavit, jak potřebujete, uložit do souboru i načíst z uloženého souboru, vytvořit jeho instanci a poté s ním pracovat, jak jen potřebujete. Já tento typ skriptu využívám jako uložení typu misí a cílů:

Obrázek 13 - Skript pro ukládání mise

## Realizace systému lodě

Základním stavebník blokem lodi je takzvaná dlaždice. Tyti dlaždice jsou skládány do dvou nodů typu TileMap (v překladu „dlaždicová mapa”), *WallTileMap* a *ObjectTileMap*.

WallTileMap se stará o všechny zdi, stavby a pevné části lodi, zatímco ObjectTileMap obsahuje nepevné objekty jako například místo na vytvoření předmětů nebo NPC.

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyObsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyTileMap je sice velice užitečný node a dá se díky němu velice jednoduše loď vytvářet a upravovat, proto je použit v editoru, ale z funkčního hlediska nám moc neposlouží, nelze s ním do velkého rozsahu interagovat. Proto se při načtení světa zmíněná TileMapa převede do objektů. Převádí se postupně každá dlaždice na svoji vlastní scénu. Dlaždice s označením *wall* (v překladu „stěna“) se převede na instanci scény wall na stejném místě. Dlaždice je následně smazána a zůstává nahrazena.

Obrázek 14 - Seznamy objektů

Obrázek 15 - Ukázka naplněného seznamu

Obsah obrázku snímek obrazovky, software, text, Počítačová ikona

Popis byl vytvořen automatickyObsah obrázku snímek obrazovky, text, software, Obdélník

Popis byl vytvořen automatickyTímto způsobem se jen z dlaždic, s kterými se nedá nijak interagovat, stanou například otevíratelné dveře:

Obrázek 16 - Scéna dveří s viditelným hitboxem

Obrázek 17 - Scéna dveří

Další věc, která se musí při tvorbě lodě vyřešit je hitbox. Kontrolovat hitbox každé části lodi zvlášť by bylo neefektivní, proto se před načtením lodě vezmou všechny dlaždice a zkombinují se do jednoho obrazce, který se poté vezme a nastaví se jako hitbox. Tento hitbox ale funguje pouze mezi loděmi, tudíž hráče ignoruje.

## Části lodě

### Stěna

Stěna je základní stavební jednotkou každé lodě, jelikož loď nemůže bez svého rámu existovat. Sama o sobě jen ztvárňuje hitbox lodi a omezení pohybu hráče. Implementoval jsem zde i funkci zničení po vyčerpání její výdrže, ale zatím jsem tuto funkci v samotné hře nevyužil.

### Dveře

Stěny hráče oddělují od okolního světa a dveře mu umožňují se s ním opět propojit. Dveře jsem z grafického hlediska udělal jako brány, které vypadají vzduchotěsně. Samotná funkčnost je jednoduchá, dveře mají dva stavy a to zavřeny/otevřeny. Tito dva stavy se mění po kliknutí myši, pokud jsou splněny dvě podmínky. První podmínka je taková, že hráč musí být v dosahu. Druhá podmínka se u obou stavů liší. Pokud jsou zavřeny, musí být odemčeny, aby se dali otevřít. Tuto funkčnost zatím opět ve hře nevyužívám. Pokud jsou dveře otevřeny a mají být zavřeny, druhou podmínkou je to, že nesmí být zatarasené žádnou postavou. Jelikož NPC nemají implementovaný pohyb, mohou být dveře zablokovány jenom hráčem.

Dveře otevřené hráčem spustí časovač, který po stanovené době dveře automaticky zavře. Pokud ovšem jsou dveře zablokované, časovač se obnoví.

### Trysky

Bez pohonu, který lodi umožňuje pohyb, by tato hra nedávala smysl. Tuto roli plní v mém projektu trysky, které jsou hlavní komponentou, kterou si hráč vylepšuje svou loď.

Pro správnou funkčnost trysky musí být splněna jedna podmínka – nesmí být nijak zablokována. Při prvním načtení lodě každá tryska zkontroluje tiles směrem, kterým je natočena, pokud zde je volné místo, nezablokuje se. Potřebné volné místo je měnitelné a je momentálně nastaveno na čtyři tiles.

Jelikož loď s pohybem do všech čtyř stran by nebyla velice realistická, v polovině tvorby produktu jsem přidal rotaci. Rotaci umožňují krajní trysky, které jsou připevněny ke každé hlavní trysce. Tyto malé pohony mají také kontrolu blokace. Rozdíl je v tom, že pokud je z boku tryska zablokovaná, neztrácí rotační sílu. Blokuje se pouze vizuální efekt při otáčení na danou stranu.

### Konektor

Pokud chceme svou loď na chvíli opustit, bylo by ideální, kdyby zůstala tam, kde jsme ji nechali. K tomuto slouží konektor. Má vedle sebe ve svém směru malý hitbox, který při kolizi se stejným typem dokáže rozeznat, zda se dva konektory dotýkají. Pokud ano, hráč při ovládání může svoji loď připevnit k jiné a nepřijít tak o svou loď.

### Jádro

Každá loď potřebuje zdroj své energie, proto jsem přidal tuto součástku, která musí být přesně jednou na každé funkční lodi. Jelikož se na lodi nachází pouze z logiky hry, neměla z počátku žádnou jinou speciální funkci, přiřadil jsem hráči při přiblížení k jádru uložit hru.

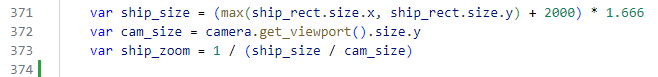
### Stanoviště pro stavbu lodi

Abychom svou loď mohli vylepšit, potřebujeme nějaké místo, kde můžeme svou loď připevnit a poté s ní volně v editoru manipulovat. Touto součástkou umožňuji hráči přivolat svou ztracenou loď, vylepšit ji nebo ji úplně předělat. Toto stanoviště nalezneme pouze v hlavní stanici.

Při načtení je *builder* (v překladu „stavěč“) nepoužitelná součástka. Builder vytvoří při načtení malý hitbox, který se následující cykly zvětšuje. Když tento hitbox zachytí kolizi s konektorem, ukončí se zvětšování. Následně se builder propojí s daným konektorem. Díky tomuto se nově postavená loď automaticky připojí na nejbližší konektor, aby do ní následně hráč mohl pohodlně nastoupit.

### Kormidlo

Pro ovládání lodi je klíčové kormidlo, umožňuje hráči se přepnout do režimu plavby s lodí. V tomto režimu se vypne pohyb hráče, oddálí se kamera a zmenší se zorné pole.

Jelikož přiblížení kamery v godotu se nastavuje proměnnou, která určuje přiblížení s tím, že hodnota 1 je spojena s výchozím přiblížením, dělalo mi ze začátku správně vyřešit oddálení u lodí různých velikostí. Nakonec jsem tento problém zvolil následujícím vzorcem:

Obrázek 18 – Výpočet přiblížení kamery

Vezme se zde větší ze dvou rozměrů lodi (šířka/výška), k tomuto rozměru se přičtou 2000, výsledek se vynásobí další konstantou 1,666. K uvedení do kontextu této výsledné hodnoty se musí výsledek vydělit šířkou obrazovky. Poté se hodnota umocní na -1. Tímto získáme potřebné přiblížení kamery, aby při ovládání lodě byla celá loď na obrazovce.

Poté se kamera ještě musí přesunout do středu lodě a pohled v režimu ovládání lodě je vyřešen.

## Schvalovač lodí

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyPřed uložením lodi do souboru se musí schválit, aby nedošlo k uložení neplatného plavidla. Pravidla pro uložení jsou mít v lodi potřebné části, jako jsou například jádro a konektor. Další pravidla jsou, že v lodi nesmí být díry a že loď musí být jedna, respektive zde nemůžou být dvě oddělené části. Schválení se určuje podle algoritmu, který jsem nazval *bucket* (v překladu „kyblík“), kvůli podobné funkčnosti jako stejnojmenný nástroj v grafických editorech.

Obrázek 19 - Skript bucket

Vyplňování podlahy, aby v lodi nebyly žádné díry, funguje tak, že se vezme pole dlaždic lodi a rozšíří se do každé strany o jeden bod. Následovně se vyplní pomocí kyblíku celý exteriér lodi, oddělený od interiéru zdí, speciálními dlaždicemi. Poté se celé pole projede cyklem a prázdná místa jsou nahrazena podlahou.

Zjišťování, jestli se loď skládá pouze z jedné části probíhá tak, že se tentokrát kyblík použije na jednu část lodi a označí všechny dlaždice, kterých se dotkne. Poté algoritmus projede všechny dlaždice podruhé a pokud narazí na neoznačenou dlaždici, je tu zde více než jedna část lodi, a tudíž loď není validní.

## Obsah obrázku text, Písmo, snímek obrazovky Popis byl vytvořen automatickyUkládání a načítání

K ukládání a načítání světa jsem využil již zmíněné resources. Pro všechny významné prvky jsem vytvořil vlastní resource, která ukládá jednotlivé vlastnosti objektu.

Obrázek 20 - Skripty k ukládání jednotlivých objektů

Obsah obrázku text, Písmo, snímek obrazovky, číslo

Popis byl vytvořen automatickySeznam těchto uložených objektů poté ukládám do seznamů objektu ve třídě SaveFile, kde řeším i načítání.

Obrázek 21 - Seznamy instancí ukládaných objektů

Při načítání vymažu všechny objekty, které se momentálně nacházejí ve světě a poté z uloženého souboru načtu zmíněné soubory.

## Testování

V raných fázích testování jsem narazil na několik překážek, které jsem původně považoval za problémy, ale které se nakonec staly důležitými milníky ve vývoji hry. Jedním takovým byla obtížnost hry při ovládání počáteční lodě, která je vybavena pouze jednou tryskou. Zpočátku se zdálo, že to hráčům zbytečně komplikuje život, ale postupem času jsem dospěl k názoru, že tento prvek představuje pro hráče vítanou výzvu. Tato obtížnost se navíc s první úspěšně splněnou misí stává méně výraznou, neboť hráč má možnost vylepšit svou loď přidáním dalšího motoru, což je motivuje ke hře a zároveň jim ukazuje, že jejich úsilí má přímý vliv na hratelnost.

Další problém, který jsem identifikoval během testování, souvisel s navigací. Hráči, kteří se vydali plnit mise na sekundárních stanicích, měli následně potíže s návratem na hlavní stanici, která slouží jako centrální uzel pro většinu NPC a herních aktivit. Abych hráčům usnadnil orientaci, rozhodl jsem se přidat do rozhraní mise tlačítko pro navigaci zpět na mateřskou stanici. Tímto způsobem se hráči mohou jednoduše a bez frustrace vrátit na hlavní stanici a pokračovat Obsah obrázku snímek obrazovky, Písmo, řada/pruh, design

Popis byl vytvořen automatickyve hře.

Obrázek 22 - Tlačítko k navigaci k hlavní stanici

Při testování také vyšel najevo problém s pohybem stanic. Stanice, se kterými hráči kolidovali, se začaly postupně oddalovat od své původní pozice, což znesnadňovalo splnění misí, které hráči na těchto stanicích přijali. Abych tento problém vyřešil, rozhodl jsem se implementovat tření pro všechny objekty kromě lodě hráče, což efektivně zabrání nechtěnému pohybu stanic.

Nečekanou komplikací bylo také to, že hráči s vylepšenými, silnějšími loděmi byli schopni manipulovat s pozicí mateřské stanice. Tento fakt by mohl vést k narušení herní rovnováhy, protože by hráči mohli táhnout hlavní stanici kdekoli a tím si výrazně ulehčit hru. Abych zajistil fér a vyvážený herní zážitek, bylo třeba pozici mateřské stanice "zmrazit", čímž jsem odstranil možnost jejího pohybu ze strany hráče.

.

# Manuál

## Minimální požadavky

OS: Windows 7 a novější

Procesor: x86\_64 CPU

Operační paměť: 4 GB

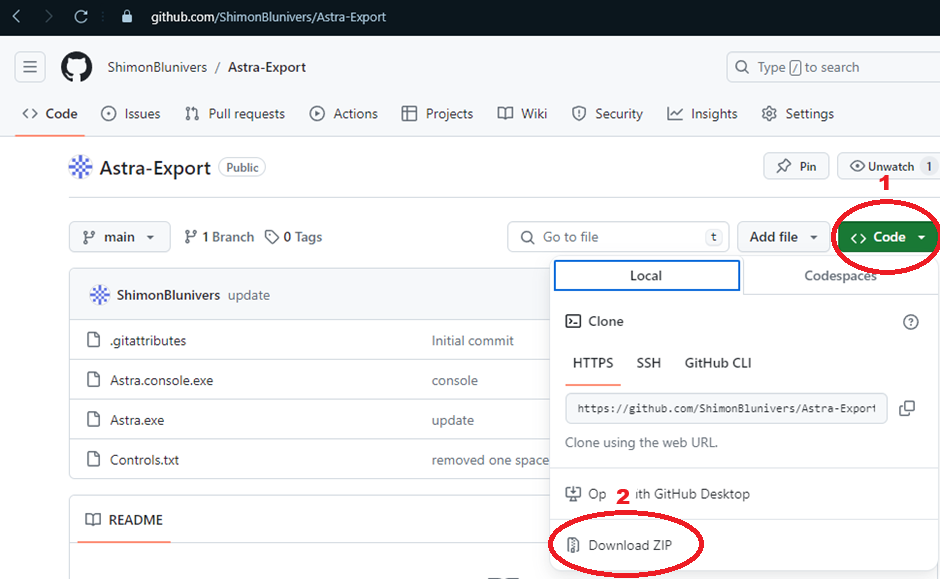
GPU: Integrovaná grafika s plnou podporou OpenGL 3.3

Úložný prostor: 100 MB

## Instalace

Hru můžeme stáhnout z tohoto odkazu: <https://github.com/ShimonBlunivers/Astra-Export.git>

Hru stáhneme kliknutím na „Code“ (1) a následovně na „Download ZIP“ (2). Poté obsah extrahujeme a instalace je dokončena. Hru spustíme otevřením souboru „Astra.exe“.



Obrázek 23 - Instalace

## Ovládání

* + 1. Ve hře

W Pohyb hráče / lodi směrem nahoru  
S Pohyb hráče / lodi směrem dolů  
A Pohyb hráče / lodi směrem doleva  
D Pohyb hráče / lodi směrem doprava  
Shift Běh  
Q Pohyb otočení v protisměru hodinových ručiček  
E Pohyb otočení po směru hodinových ručiček  
F Použití objektu  
M Otevření / Zavření inventáře  
Levé tlačítko myši Otevření / Zavření dveří, Dialog s NPC, Zvednutí

předmětu

* + 1. V editoru

W Pohyb kamery směrem nahoru  
S Pohyb kamery směrem dolů  
A Pohyb kamery směrem doleva  
D Pohyb kamery směrem doprava  
Shift Zrychlení pohybu kamery  
Tabulátor Otevření / Zavření inventáře  
R Otočení s vybraným nástrojem po směru hodinových

ručiček  
Esc Vrácení do hry  
Levé tlačítko myši Použití vybraného nástroje  
Pravé tlačítko myši Prodání vybrané části lodi  
Kolečko myši Změna přiblížení kamery

# Závěr

Výsledky mého maturitního projektu představuje úspěšné dokončení 2D sandboxové videohry pro desktop, vytvořené v herním enginu Godot. Hra je navržena s otevřeným světem, různorodými herními úkoly a editorem lodí, který umožňuje hráčům vylepšování svých plavidel. Celý projekt úspěšně splňuje všechny požadavky specifikované v zadání a nabízí i další funkce.

Hra je v tomto stádiu již hratelná, ale ještě postrádá příběh a možnost dohrání hry, které ovšem nebyly součástí zadání. Jelikož je tato videohra určena k nekonečnému zlepšování své lodě, absence tohoto aspektu není katastrofální, alespoň je zde prostor pro další vývoj.

Při vytváření této práce jsem nabral spoustu zkušeností, co se tvorby her i tvorby grafiky týče i přes to, že jsem se tvorbě her věnoval již nějakou dobu před tímto projektem. Díky této práci jsem konečně nějaký z těchto projektů dostal do fáze, kdy je možné jej vypustit alespoň mezi své přátele, kteří mezi sebou kompetitivně porovnávali dosažené lodě.

Získané dovednosti a zkušenosti bych rád nadále rozvíjel a využíval ve svém volném čase, pokračováním ve vývoji této a dalších her. Tento projekt mi ukázal, že i při samostatné práci lze dosáhnout významného pokroku a vytvořit něco, co může přinést radost a zábavu mnoha lidem.Seznam použitých zdrojů a literatury

Videohra [online]. 2024 [cit. 2024-04-15]. Dostupné z:  
[https://cs.wikipedia.org/wiki/Videohra#](https://cs.wikipedia.org/wiki/Videohra)

Godot [online]. 2024 [cit. 2024-04-15]. Dostupné z:  
[https://cs.wikipedia.org/wiki/Godot#](https://cs.wikipedia.org/wiki/Godot)

Aseprite [online]. 2024 [cit. 2024-04-15]. Dostupné z:  
[https://cs.wikipedia.org/wiki/Aseprite#](https://cs.wikipedia.org/wiki/Aseprite)

# Seznam ilustrací

[Obrázek 1 - Konverzace 12](#_Toc164250418)

[Obrázek 2 - Složka s misemi 12](#_Toc164250419)

[Obrázek 3 - Navazující konverzace 12](#_Toc164250420)

[Obrázek 4 - Životy 14](#_Toc164250421)

[Obrázek 5 - Měna 14](#_Toc164250422)

[Obrázek 6 - Skript k ukládání lodě do souboru 15](#_Toc164250423)

[Obrázek 7 - Skript k posouvání pozadí 16](#_Toc164250424)

[Obrázek 8 - Menu 16](#_Toc164250425)

[Obrázek 9 - Komponenty 17](#_Toc164250426)

[Obrázek 10 - Exportovaná proměnná v editoru 18](#_Toc164250427)

[Obrázek 11 - Proměnná 18](#_Toc164250428)

[Obrázek 12 - Identifikátor třídy 18](#_Toc164250429)

[Obrázek 13 - Skript pro ukládání mise 18](#_Toc164250430)

[Obrázek 14 - Seznamy objektů 19](#_Toc164250431)

[Obrázek 15 - Ukázka naplněného seznamu 19](#_Toc164250432)

[Obrázek 16 - Scéna dveří s viditelným hitboxem 19](#_Toc164250433)

[Obrázek 17 - Scéna dveří 19](#_Toc164250434)

[Obrázek 18 – Výpočet přiblížení kamery 21](#_Toc164250435)

[Obrázek 19 - Skript bucket 22](#_Toc164250436)

[Obrázek 20 - Skripty k ukládání jednotlivých objektů 23](#_Toc164250437)

[Obrázek 21 - Seznamy instancí ukládaných objektů 23](#_Toc164250438)

[Obrázek 22 - Tlačítko k navigaci k hlavní stanici 24](#_Toc164250439)

[Obrázek 23 - Instalace 25](#_Toc164250440)

# Odkazy

Github repositář: <https://github.com/ShimonBlunivers/Astra-Godot.git>  
Export: <https://github.com/ShimonBlunivers/Astra-Export.git>