**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

[Prohlášení 2](#_Toc163918215)

[Anotace 3](#_Toc163918216)

[Obsah 4](#_Toc163918217)

[Úvod 7](#_Toc163918218)

[1 Rozbor řešení 8](#_Toc163918219)

[1.1 Použité technologie 8](#_Toc163918220)

[1.1.1 Godot 2D 8](#_Toc163918221)

[1.1.2 Aseprite 8](#_Toc163918222)

[1.1.3 Programovací jazyk GDScript 8](#_Toc163918223)

[1.1.4 Programovací jazyk C# 8](#_Toc163918224)

[1.2 Hlavní body hry 9](#_Toc163918225)

[1.2.1 Smysl hry 9](#_Toc163918226)

[1.2.2 Ovládání 9](#_Toc163918227)

[1.2.3 Herní úkoly 10](#_Toc163918228)

[1.2.4 Logika NPC 11](#_Toc163918229)

[1.2.5 UI 11](#_Toc163918230)

[1.2.6 Editor 12](#_Toc163918231)

[1.2.7 Hlavní menu 12](#_Toc163918232)

[2 Experimentální část 14](#_Toc163918233)

[2.1 Úvod do pracovního postupu v Godotu 14](#_Toc163918234)

[2.1.1 Herní objekty a komponenty 14](#_Toc163918235)

[2.1.2 Programování v Godotu 14](#_Toc163918236)

[2.2 Realizace systému lodě 16](#_Toc163918237)

[2.3 Schvalovač lodí 17](#_Toc163918238)

[2.4 Ukládání a načítání 17](#_Toc163918239)

[2.5 Testování 18](#_Toc163918240)

[4 Manuál 19](#_Toc163918241)

[4.1 Minimální požadavky 19](#_Toc163918242)

[4.2 Instalace 19](#_Toc163918243)

[Závěr 20](#_Toc163918244)

[Seznam použitých zdrojů a literatury 21](#_Toc163918245)

Seznam použitých zkratek

NPC – Non-Player Character (nehráčská postava)

UI – User Interface (uživatelské rozhraní)

# Úvod

Videohry jsou interaktivní software, jehož cílem je nejen zábava hráče, ale v mnoha případech také předání určitých hodnot. Tato forma digitálního umění se stala v posledních letech mimořádně oblíbeným způsobem trávení volného času. S průběhem času vzniká stále více her, což vytváří výzvu při vytváření projektů, které se odlišují od ostatních.

Rozhodl jsem se zaměřit svou maturitní práci právě na toto téma, protože mám zájem o tuto část digitálního světa a vytváření těchto projektů mě zabavilo více než samotné trávení času u onen videoher.

# 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.

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.

### 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 automaticky

Mise 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.

Obrázek 2 - Složka s misemi

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 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.

### UI

Obsah obrázku pixel, Grafika, symbol, snímek obrazovky

Popis byl vytvořen automatickyObsah obrázku Grafika, grafický design, Písmo, logo

Popis byl vytvořen automatickyUživatelské rozhraní musí hráče informovat o nejdůležitějších prvcích videohry, v tomto případě je to měna, kterou má a u sebe, aktivní mise a samozřejmě životy, které hráč momentálně má.

Obrázek 4 - Měna

Obrázek 5 - Životy

### 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

### Hlavní menu

Tvorba hlavního menu byla z těch jednodušších částí této práce. Tlačítka, které se zde nachází vás uvedou do jiných scén projektu. „Hrát“ a „Nová hra“ spustí stejnou scénu, při výběru nové hry se ale nejprve vymaže uložená hra.

Pozadí 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.

Obsah obrázku text, Písmo, snímek obrazovky

Popis byl vytvořen automatickyOba mají různou rychlost a tím zde vzniká hloubka pozadí.

Obrázek 7 - Skript k posouvání pozadí

# Experimentální část

## Úvod do pracovního postupu v Godotu

### sfhfghHerní objekty a komponenty

Obrázek 8 - 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.

### 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 9 - Proměnná

Obrázek 10 - Exportovaná proměnná v editoru

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 11 - 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 12 - 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 13 - Seznamy objektů

Obrázek 14 - 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 15 - Scéna dveří s viditelným hitboxem

Obrázek 16 - Scéna dveří

## 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 17 - 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.

## Ukládání a načítání

Obsah obrázku text, Písmo, snímek obrazovky

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

Obrázek 18 - 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 19 - Seznamy instancí ukládaných objektů

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

## Testování

Při prvních částech testování jsem zjistil, že hra s první lodí, kterou hráč dostane, je poměrně obtížná vzhledem k nemotornosti ovládání lodi jen s jednou tryskou. První jsem to bral jako problém, ale postupem času jsem si řekl, že je to pro hráče dobrá výzva, která se postupem času vytratí vzhledem k tomu, že po první splněné misi si hráč svou loď může vylepšit dalším motorem.

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

Popis byl vytvořen automatickyDalší z problémů, který nastal byl ten, že když se hráč vydal splnit misi na jiné stanici, měl potom problém se vrátit na původní hlavní stanici, kde se nachází většina NPC. Tento problém jsem vyřešil přidáním tlačítka navigace k mateřské stanici do tabulky s misemi.

Obrázek 20 - Tlačítko k navigaci k hlavní stanici

V testovací fázi také nastal problém s pohybem stanic. Pokud hráč neúmyslně kolidoval s nějakou ze stanic, přijmul zde nějakou misi a poté ji opustil, postupem času tato stanice odplouvala dál a dál, až bylo téměř nemožné přijatou misi splnit. Tento problém jsem opravil přidáním tření ke všem lodím kromě lodě hráče.

Opravení také vyžadovalo pohybování s hlavní stanicí. Pokud hráč měl dostatečně silnou loď, mohl se připojit k mateřské stanici a odtáhnout si ji kamkoliv se mu zachtělo. Jelikož toto by velice ulehčovalo hru, jelikož by měl tuto loď hráč vždy po ruce, musel jsem zmrazit pozici této hlavní lodě, aby s ní hráč nemohl nijak manipulovat.

# 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>

Stažením souboru Astra.exe je instalace dokončena. Poté stačí jen soubor otevřít.

## 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

# Seznam použitých zdrojů a literatury

[https://cs.wikipedia.org/wiki/Videohra#](https://cs.wikipedia.org/wiki/Videohra)

[https://cs.wikipedia.org/wiki/Godot#](https://cs.wikipedia.org/wiki/Godot)

[https://cs.wikipedia.org/wiki/Aseprite#](https://cs.wikipedia.org/wiki/Aseprite)