



# LOGICKÁ HRA S EDITOREM A SDÍLENÍM MAP

Maturitní práce

Autor	<b>Petr Machačka</b>
Obor	<b>Informační technologie</b>
Vedoucí práce	<b>Michal Stehlík</b>
Školní rok	<b>2024/2025</b>
Počet stran	<b>29</b>
Počet slov	<b>4820</b>

## Přihláška k maturitní práci

**Jméno a příjmení studenta**

Machačka, Petr

**Název práce**

Logická hra s editorem a sdílením map

**Třída**

P4A

**Školní rok**

MP2024/25

**Přidělené role**

Vedoucí práce

Oponent

Podpis



Stehlík, Michal

<b>Obecná ustanovení</b>	Vypracování a odevzdání práce proběhne v souladu s platnými normami (vyhláška 177/2009 Sb.) a aktuálním dokumentem "Pokyny k vypracování prací" vydaným školou.
	Práce bude hodnocena z hlediska jejího praktického využití, zvládnutí dokumentace po věcné i formální stránce a obhajoby celé práce. Student byl seznámen s kritérii hodnocení maturitní práce.
	Práce bude odevzdána ve dvou stejnopisech vázaných pevnou nebo kroužkovou vazbou.
	Veškeré náklady na MP včetně vyhotovení obou tištěných kopií si student hradí sám.
<b>Licenční ujednání</b>	Ve smyslu § 60 (Školní dílo) autorského zákona č. 121/2000 Sb. poskytují SPŠSE a VOŠ Liberec výhradní a neomezená práva k využití této mé maturitní práce.
	Bez svolení školy se zdržím jakéhokoliv komerčního využití mé práce.
	Pro výukové účely a prezentaci školy se vzdávám nároku na odměnu za užití díla.

**Finanční rozvaha - odhad celkových nákladů**

V Kč	Náklady celkem	Hrazené školou
<b>Výrobní</b>	0	0
<b>Na služby</b>	0	0

Jedná se o MP, jejíž vypracování si škola vyžádala? ☒ Ano – ☐ Ne

**Podpis studenta (vyjadřuje souhlas s uvedenými údaji a ujednáními)**

V Liberci 27.09.2024

**Konzultant**

Práci podporuji

**Předmětová komise**

Práci doporučuji

**Třídní učitel**

Práci doporučuji

**Garant oboru**

Práci doporučuji

**Ředitel školy**

Práci doporučuji

Podpis



Podpis



Podpis



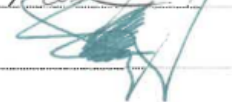
Podpis



Podpis



Podpis



## Zadání maturitní práce

**Název**

Logická hra s editorem a sdílením map

**Předmět**

PRG

**Téma**

Vytvoření local multiplayer hry v Unity Engine. Hra je hlavně založená na level editoru díky kterému pomocí předurčených prvků budou moci hráči tvořit své levely. Následně je sdílet pomocí Steam workshopu.

**Použité prostředky**

Visual Studio, Unity Engine, C#

**Cíle práce**

1	Základ logické local coop hry s několika gamemody
2	Komplexní level editor
3	Vydání na steam
4	Použití Steam workshopu pro sdílení výtvorů

**Osnova práce**

1	Nastudovat Unity dokumentaci
2	Najít nejlepší způsob vytvoření rošiřitelného level editoru
3	Splnění podmínek pro vložení hry na steam
4	Prostudování implementace steam workshopu

---

## Anotace

Tato práce se zaměřuje na vývoj kooperativní 3D hry v Unity, přičemž klíčovou součástí je editor úrovní, systém ukládání a načítání dat a integrace se službou Steam. Cílem bylo vytvořit intuitivní herní prostředí, které umožní hráčům nejen hrát, ale také tvořit a sdílet vlastní obsah. Důraz byl kladen na modularitu a efektivní správu herních dat, což zajišťuje snadnou rozšiřitelnost projektu. Výsledkem je plně funkční prototyp s podporou lokálního multiplayeru a komunitního sdílení prostřednictvím Steam Workshopu.

## Summary

This thesis focuses on the development of a cooperative 3D game in Unity, with a key emphasis on the level editor, data storage and retrieval system, and integration with the Steam service. The goal was to create an intuitive gaming environment that allows players not only to play but also to create and share their own content. Emphasis was placed on modularity and efficient data management, ensuring easy project expandability. The result is a fully functional prototype with support for local multiplayer and community content sharing via Steam Workshop.

## Čestné prohlášení

Prohlašuji, že jsem předkládanou maturitní práci vypracoval sám a uvedl jsem veškerou použitou literaturu a bibliografické citace.

V Liberci dne 11.03.2025

.....  
Petr Machačka

# Obsah

Úvod.....	1
1 Použité technologie.....	2
1.1 Unity .....	2
1.2 Steam.....	2
1.2.1 Steam Workshop .....	3
1.2.2 Steamworks .....	3
2 Praktická část .....	4
2.1 Steam implemetace .....	4
2.1.1 Propojení se Steamem.....	4
2.1.2 Správa obsahu Workshopu.....	4
2.2 Editor.....	5
2.2.1 Inventory.....	6
2.2.2 Stavící systém.....	7
2.2.3 Propojování.....	9
2.3 Systém ukládání a načítání .....	10
2.4 Herní scéna.....	12
2.5 UI .....	13
2.5.1 Hlavní menu .....	13
2.5.2 Upload menu.....	15
3 Chyby a problémy .....	17
3.1 Editor.....	17
3.1.1 Nekonzistence bloků.....	17
3.1.2 nemazáním propojení .....	17
3.2 Steam.....	18
3.2.1 Editování prvků ve workshopu .....	18
3.2.2 18	
Závěr .....	19
Seznam zkratk a odborných výrazů.....	19
Seznam obrázků.....	21
Použité zdroje .....	22
A. Seznam přiložených souborů .....	I



# Úvod

Tato práce se zaměřuje na vývoj kooperativní 3D hry, která hráčům umožňuje nejen hraní, ale také tvorbu a sdílení vlastních úrovní prostřednictvím Steam Workshopu. Při vývoji herního systému bylo nutné řešit různé aspekty, včetně návrhu editoru úrovní, implementace multiplayeru, ukládání a načítání dat či integrace s platformou Steam.

Hlavním cílem této práce je popsat proces vývoje hry, od základního návrhu až po její realizaci a propojení s komunitními funkcemi. Důraz byl kladen na modularitu a rozšiřitelnost, aby bylo možné hru dále upravovat a rozvíjet podle potřeb hráčů. Významnou součástí projektu byla také optimalizace herního prostředí a návrh intuitivního uživatelského rozhraní pro editor úrovní.

Práce se zaměřuje na využití Unity Engine a Steam API, přičemž se zabývá jak technickými, tak designovými aspekty vývoje. Součástí procesu bylo i testování a odstraňování chyb, zejména v souvislosti s ukládáním a synchronizací herních dat mezi lokálním úložištěm a Steam Workshopem.

Tímto způsobem práce poskytuje ucelený pohled na problematiku vývoje her s důrazem na interaktivní obsah, online spolupráci a uživatelskou přívětivost.

---

# 1 Použité technologie

## 1.1 Unity

Unity umožňuje vývoj 2D i 3D aplikací a podporuje širokou škálu platforem, včetně PC, herních konzolí (PlayStation, Xbox, Nintendo Switch), mobilních zařízení (Android, iOS), webu a také zařízení pro virtuální a rozšířenou realitu, jako jsou Oculus Quest nebo HoloLens. Tato široká škála možností dělá z Unity jedno z nejrozšířenějších vývojových prostředí na světě.

Jedním z hlavních důvodů popularity Unity je jeho přístupnost. Samotný engine je k dispozici ve verzi zdarma, což umožňuje jednotlivcům a malým týmům začít vyvíjet hry bez nutnosti velkých počátečních investic. Pro větší projekty nabízí Unity placené verze s rozšířenými funkcemi, jako jsou pokročilé analytické nástroje, cloudová synchronizace a technická podpora. (1)

### 1.1.1 Line Renderer

**Line Renderer** v Unity je komponenta pro vykreslování čar ve 3D prostoru, které lze využít například pro trajektorie, laserové paprsky nebo propojení objektů. Umožňuje nastavit šířku, barvu a barevný přechod podél délky čáry, případně ji uzavřít do smyčky. Lze také upravovat počet bodů, ze kterých se čára skládá, a vylepšit její interakci se světlem. Mezi základní funkce patří nastavení pozic jednotlivých bodů čáry, jejich úprava a možnost redukovat jejich počet pro optimalizaci vykreslování. Line Renderer se často využívá pro vizuální efekty a propojení prvků ve hře. (2)

#### 1.1.1.1 RayCasting

Raycast v Unity vysílá neviditelný paprsek a detekuje, zda narazí na objekt. Používá se pro interakce, kolize nebo střelbu. Vrací informace o zásahu, jako je pozice, povrchová normála a vzdálenost. Lze jej omezit na určitou vzdálenost nebo vrstvy objektů. (3)

## 1.2 Steam

Steam je digitální distribuční platforma vytvořená společností Valve Corporation. Slouží k nákupu, stahování a správě her, softwaru a dalších digitálních produktů. Kromě obchodní funkce nabízí i sociální prvky, jako jsou přátelé, chat, diskusní fóra a achievementy. Steam podporuje různé platformy, včetně Windows, macOS a Linuxu, a je jedním z největších a nejpopulárnějších ekosystémů pro hráče na světě. (4)



### 1.2.1 Steam Workshop

Workshop služby Steam je středobodem veškerého uživateli vytvářeného obsahu a nástrojů pro jeho zveřejnění, organizaci a stažení do příslušných her. Každý titul navíc workshop využívá jinak, takže například pro Team Fortress 2 můžete vytvářet a publikovat nové položky (klobouky, zbraně, odznaky, boty apod.), u nichž je posléze rozhodnuto o oficiálním přidání do hry. Workshopy jiných her, třeba The Elder Scrolls V: Skyrim, zase umožňují uživatelům odebírat zveřejněné modifikace a snadno je tak instalovat přímo do hry. (5)

### 1.2.2 Steamworks

Steamworks je sada nástrojů od Valve pro integraci her se službou Steam. Umožňuje správu licencí, multiplayer funkcí, achievementů, cloudového ukládání, workshopu a dalších služeb. Vývojáři mohou využít API pro propojení hry se Steamem a jeho komunitními funkcemi. (6)

#### 1.2.2.1 Facepunch Steamworks

Facepunch.Steamworks je C# wrapper nad knihovnou Steamworks, navržený pro usnadnění integrace funkcí Steamu do her vyvíjených v C#. Na rozdíl od Steamworks.NET, který je věrným přepisem původní C++ knihovny, Facepunch.Steamworks nabízí reinterpetaci API přizpůsobenou pro C# prostředí. Tím zjednodušuje implementaci funkcí, jako je například získávání seznamu přátel. Projekt je dostupný na GitHubu a obsahuje podrobnou dokumentaci pro nastavení a použití. (7)

---

## 2 Praktická část

### 2.1 Steam implemetace

Workshop ve službě Steam je systém pro správu uživatelského obsahu, který umožňuje hráčům sdílet, stahovat a organizovat vlastní levely přímo ve hře. Implementace Workshopu v této hře se zaměřuje na automatizované propojení s platformou Steam, které hráčům poskytuje přístup k uživatelským levelům bez nutnosti manuální správy souborů.

Funkcionalita je rozdělena do dvou hlavních částí:

1. **Propojení se Steamem**, které umožňuje hře komunikovat s platformou, synchronizovat data a spravovat uživatelské úložiště.
2. **Správa obsahu Workshopu**, která zahrnuje **nahrávání, filtrování, stahování a mazání levelů**, včetně přiřazování kategorií pro snadnější orientaci.

Tento systém umožňuje **rychlou distribuci obsahu mezi hráči** a zajišťuje, že komunitní levely jsou snadno dostupné přímo ve hře.

#### 2.1.1 Propojení se Steamem

Každá hra, která využívá funkce Steamu, musí mít **unikátní App ID**, které umožňuje její identifikaci v rámci platformy. Při spuštění hry se inicializuje **SteamManager**, což je skript zajišťující spojení mezi hrou a Steamem. Díky tomuto propojení hra získává přístup ke službám, jako je **Steam Overlay, ukládání do cloudu a správa Workshopu**.

Součástí propojení je také **kontinuální zpracování odpovědí ze Steamu**, které umožňuje **asynchronní načítání nových dat** bez zbytečného zpomalování hry.

#### 2.1.2 Správa obsahu Workshopu

Steam Workshop umožňuje **hráčům nahrávat vlastní levely** a sdílet je s ostatními uživateli. Každý level je identifikován unikátním **GUID**, což zajišťuje, že nedochází k duplicitám a konfliktům při ukládání souborů.

##### 2.1.2.1 Nahrávání levelu na Steam Workshop

Před nahráním se kontroluje, zda level **již existuje na Workshopu**. Pokud ano, aktualizuje se jeho obsah. Pokud ne, vytvoří se **nový záznam** v komunitních souborech. Během nahrávání se k levelu přidávají **kategorie (tagy)**, které umožňují **lepší organizaci a filtrování obsahu**.

Používané kategorie zahrnují:

- Coop – level je určený pro více hráčů.
- Singleplayer – level je navržen pro jednoho hráče.
- Map – označuje level jako hratelnou mapu.

Jakmile je upload úspěšně dokončen, hra aktualizuje uložená data a přiřadí levelu **Steam ID**, které umožňuje jeho správu v rámci Workshopu.

#### 2.1.2.2 Stahování levelů z Workshopu

Hráči si mohou stahovat levely podle **různých filtrů**, které umožňují snadné vyhledávání v komunitním obsahu. Hra podporuje řazení podle:

- **Data nahrání**
- **Hodnocení hráčů**
- **Počtu odehraných hodin**
- **Trendu a popularity**

Po výběru filtru hra **odesílá dotaz na Steam API**, které vrátí seznam dostupných levelů odpovídajících daným kritériím. Hráč si pak může vybrat konkrétní level a stáhnout jej do své knihovny.

#### 2.1.2.3 Stažení a aktivace levelu

Jakmile si hráč zvolí level ke stažení, hra ověří jeho dostupnost v Workshopu a stáhne odpovídající soubor. Stahované levely jsou **automaticky umístěny do složky Workshop**, odkud je hra může **načíst a zobrazit v editoru nebo herní scéně**.

Po stažení je level připraven k **hraní, úpravě nebo dalšímu sdílení**.

#### 2.1.2.4 Mazání levelu z Workshopu

Hráči mají možnost své levely nejen nahrávat a spravovat, ale také **odstraňovat** z Workshopu. Pokud hráč smaže svůj level, hra odešle **žádost na Steam API**, které soubor odstraní z databáze.

Tento systém zajišťuje, že hráč má **plnou kontrolu nad svým obsahem**, a umožňuje snadnou správu nahraných souborů.

## 2.2 Editor

Editor je jeden ze základních systému této hry. Dají se lze vytvářet hratelné levely pomocí několika různých nástrojů.

---

## 2.2.1 Inventory

Inventář ve hře slouží jako rozhraní pro výběr nástrojů a objektů, které může hráč využít při stavbě. Objekty a nástroje lze pomocí systému drag and drop přesouvat do spodního panelu, kde jsou následně dostupné pro rychlé vyvolání pomocí kláves 1 až 9.



Obrázek 1 - Inventory

### 2.2.1.1 Naplnění inventáře

Inventář se automaticky generuje na základě **obsahu složek v Resources**, konkrétně **složky Build pro stavební objekty a složky Tools pro nástroje**. Skript **Inventory** prochází tyto složky, načítá názvy dostupných objektů a následně vyhledává jejich **obrázky ve složkách Building Pictures a Tools Pictures**. Po nalezení odpovídajícího obrázku se vytvoří **objekt se skriptem DraggableItem**, který umožňuje hráči manipulovat s objekty v inventáři.

### 2.2.1.2 Generování náhledů objektů

Jelikož Unity Previews slouží primárně pro editor a nelze je použít přímo ve hře, bylo nutné vytvořit speciální skript **PreviewGenerator**. Tento skript při spuštění vygeneruje náhledy všech objektů, uloží je do správných složek v Resources, a tím umožní jejich načítání i ve finální sestavené verzi hry.

### 2.2.1.3 Systém Drag and Drop

Manipulace s objekty v inventáři je řešena pomocí drag and drop systému, který využívá tři hlavní události:

1. OnBeginDrag (zahájení přetahování) – Vytvoří nový vizuální obrázek objektu, zatímco původní ikona v inventáři zůstává na místě, protože počet objektů není omezen.
2. OnDrag (během přetahování) – Zajišťuje, že přetahovaný obrázek sleduje pozici kurzoru myši.
3. OnEndDrag (položení objektu) – Pomocí RayCast paprsku se detekuje, kde byl objekt puštěn:
  - I. Pokud je upuštěn do volného prostoru, zmizí a hráč si může vzít nový.
  - II. Pokud je upuštěn nad políčkem spodního panelu, zjistí se konkrétní slot a objekt se k němu přiřadí jako podržený prvek. Získávání obrázků

#### 2.2.1.4 Výběr aktivního objektu

Hráč si může vybírat objekty ve spodním panelu pomocí Unity Input systému, který detekuje stisk kláves 1–9. Po stisku se ve složkách Build nebo Tools vyhledá odpovídající objekt podle jména, a ten se následně uloží do proměnné buildingPrefab, která slouží k uchování aktuálně vybraného objektu pro stavění.

#### 2.2.2 Systém Stavění

Pokládání bloků v této hře funguje dost jednoduše, ale zároveň musí dodržovat určité zásady, aby všechno fungovalo tak, jak má. V editor režimu mají všechny položené bloky **deaktivované collidery**, protože kdyby zůstaly aktivní, mohly by způsobovat problémy při pokládání dalších objektů. Místo toho je nad bloky umístěna **pomocná síť neviditelných kostek**, která zajišťuje, že hráč může stavět pouze tam, kde je to povoleno. Díky tomu se bloky nepokládají na sebe chaoticky a **drží se v gridovém systému**. Jakmile je blok úspěšně položen, přesune se do **složky Build**, kde se s ním dál pracuje, například při kontrole kolizí nebo pro další úpravy.

Jenže ne všechny objekty jsou stejně jednoduché. **Dveře, žebříky nebo startovní pozice hráče** jsou větší než běžné bloky a musí se hlídat, jestli se náhodou nepokládají někam, kde by kolidovaly s jinými objekty. Například kdyby hráč chtěl umístit dveře na blok, nad kterým je jiný objekt, mohlo by se stát, že se **část dveří zasekne uvnitř jiného bloku**, což rozhodně není žádoucí. Aby k tomu nedošlo, skript před umístěním kontroluje složku **Build**, kde hledá ostatní objekty a porovnává jejich souřadnice s horní částí právě pokládaného objektu. Pokud se zjistí, že umístění není možné, **náhled objektu se začne zobrazovat červeně**, čímž hráči jasně ukazuje, že tam blok položit nejde.

Celý proces pokládání bloků závisí na **RayCast paprsku**, který se vysílá z pohledu hráče a **narazí na collider jednoho z pomocných objektů**. Jakmile dojde ke kolizi, vypočítají se souřadnice a zjistí se, **z jaké strany se na objekt míří**. Poté se na odpovídající místo **umístí náhled bloku**, který hráč právě drží. Aby to ale neběhalo

---

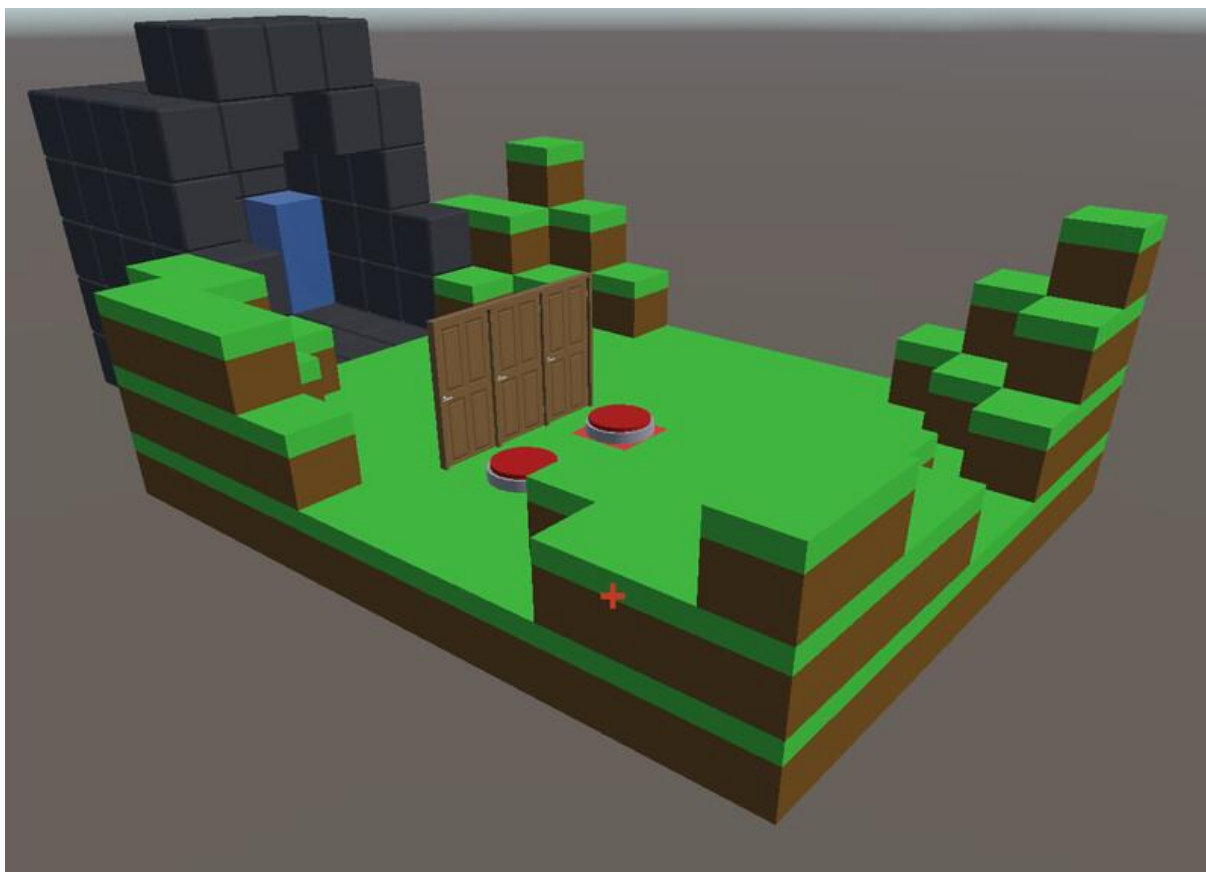
zbytečně pomalu, náhled **nepřepočítává svou pozici pokaždé**, ale jen když paprsek narazí na jiný objekt nebo změni stranu stávajícího bloku. Tím se **šetří výkon**, protože kdyby se všechno přepočítávalo neustále, hra by se pravděpodobně zpomalila.

Dále se dá taky rotovat jednotlivé objekty. Rotování objektů je u různých objektů jiné. Většina objektů se rotuje podle středu samotného objektu. Například dveře, strom, kámen. Na druhou stranu jsou bloky které musí mít posunutý střed otáčení protože musí být vždy na kraji bloku. Například žebříky nemůžou být uprostřed bloku, protože by nemohly být postaveny na stěně a jenom by levitovaly půl bloku od stěny. Dále se také ukládá poslední rotace, takže například když se třeba pokládá větší množství žebříků nad sebe tak stačí jeden otočit správným směrem ke stěně, potom i další budu otočený stejným směrem.

Ve hře je také možné rotovat jednotlivé objekty, přičemž způsob otáčení se liší podle typu objektu. Většina objektů se otáčí kolem svého středu, což platí například pro dveře, stromy nebo kameny. Tyto objekty nemají pevně danou orientaci vůči jiným prvkům a jejich umístění není závislé na konkrétní straně bloku.

Na druhou stranu existují objekty, které musí mít posunutý střed otáčení, protože jejich umístění vyžaduje specifickou polohu vůči jiným prvkům. Typickým příkladem jsou žebříky, které nemohou být umístěny uprostřed bloku, protože by levitovaly ve vzduchu a nebylo by možné je připevnit ke stěně. Z tohoto důvodu se jejich osa otáčení nachází na kraji bloku, aby při otáčení zůstaly správně zarovnané ke stěně.

Systém také ukládá poslední rotaci objektu, což znamená, že pokud hráč například staví více žebříků nad sebou, první otočí správným směrem a všechny další se budou automaticky otáčet stejným způsobem. Tato funkce zjednodušuje práci se stavebními prvky a eliminuje nutnost opakovaného nastavování správné orientace.



Obrázek 2 - Building example

### 2.2.3 Propojování

Systém propojování objektů ve hře funguje na principu signálních spojení, která umožňují interakci mezi herními prvky. Každý interaktivní objekt, například tlačítko, dveře nebo přepínač, může být propojen s jinými objekty prostřednictvím signálních paprsků, které určují směr a funkci propojení.

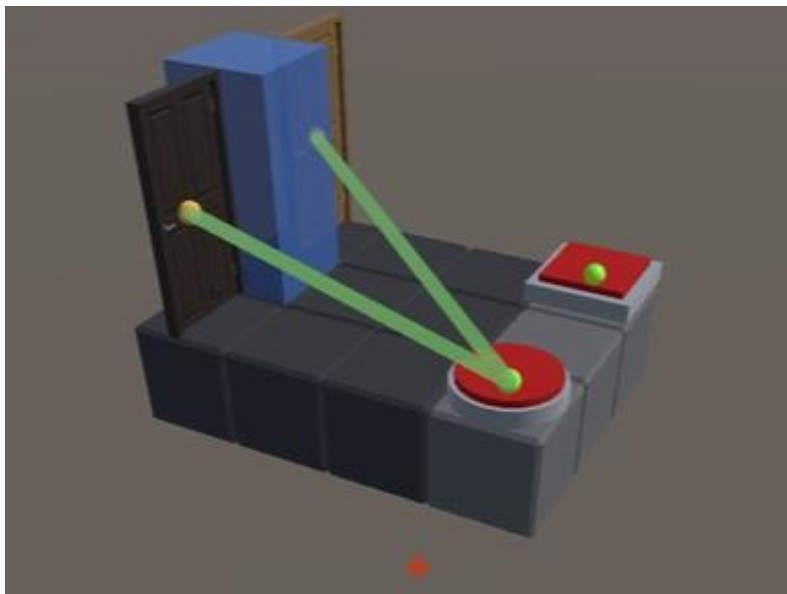
Při aktivaci vstupního prvku, například stisknutím tlačítka, se vysílá signál k propojeným objektům, které na základě této informace změní svůj stav (například se dveře otevřou). Systém umožňuje vícenásobné propojení, což znamená, že jeden prvek může ovládat více objektů současně.

Systém propojování objektů je založen na komponentě Line Renderer, která je dostupná v herním engine Unity. Nejprve si hráč musí v inventáři vybrat speciální náradí, které mu umožní zobrazit všechny dostupné body pro propojení. Jakmile je náradí aktivní, hráč může zamířit na konkrétní bod, který chce propojit. Pro lepší vizuální zpětnou vazbu se bod po zamíření zvětší, což signalizuje, že na něj hráč skutečně míří. Pokud hráč klikne, vytvoří se propojení mezi náradím v ruce a vybraným bodem pomocí Line Rendereru.

Důležité je, zda hráč kliknul na input nebo output bod, protože propojení funguje pouze mezi input a output body – zelená barva označuje input body a oranžová output body. Nelze propojit dva output body nebo dva input body mezi sebou. Pokud hráč následně označí druhý bod a klikne na něj, oba body se propojí a vytvoří funkční spojení.

---

Propojení lze také zrušit, a to opětovným propojením stejných bodů. V samotných input a output objektech jsou uloženy informace o objektech, ke kterým jsou připojeny, a také reference na Line Renderer komponentu. Díky tomu, pokud se jeden z propojených objektů smaže, automaticky se odstraní i propojení. U input bodů je navíc uložen údaj o tom, který output bod je aktivuje, což umožňuje správné fungování logických mechanismů ve hře.



Obrázek 3 - Connections example

## 2.3 Systém ukládání a načítání

Systém ukládání a načítání je implementován pro lokální uložení levelů pro případné další úpravy, nebo hraní.

### 2.3.1 Ukládání

Při ukládání levelu je prvním krokem ověření, zda již daný level existuje. To se provádí procházením všech lokálně uložených levelů a porovnáváním jejich ID. Pokud se ve stávajících souborech nenajde žádné shodné ID, vygeneruje se nové ID, pod kterým se level uloží. Pro pojmenování souborů se používá GUID (Globally Unique Identifier), což zajišťuje, že každý level má unikátní identifikátor a nedochází ke konfliktům způsobeným stejnými názvy souborů.

Dalším krokem je získání aktuálního stavu levelu, což se provádí procházením složky Build, kde jsou uloženy všechny herní objekty. Pro každý objekt se zaznamenají pozice, rotace a název objektu. Kromě těchto základních informací obsahuje každý objekt také Item skript, ve kterém lze zjistit, zda je objekt součástí systému propojování. Pokud objekt podporuje propojování, uloží se navíc i informace o připojených objektech, aby se zachovala logika propojení mezi interaktivními prvky.



Poslední fází ukládání je kontrola základních herních prvků. Skript ověří, zda se v uloženém levelu vyskytuje startovní bod, přičemž v případě kooperativního režimu (Coop) musí být startovní body dva. Dále se kontroluje, zda je v levelu umístěn cíl, bez kterého by level nebyl kompletní.

Tímto způsobem se zajišťuje, že každý level je správně strukturován, obsahuje všechny potřebné herní prvky a je možné jej bez problémů načíst a použít v dalším průběhu hry.

```
public class ObjectData
{
    public string name;
    public Vector3 position;
    public Quaternion rotation;
    public List<Vector3> connectionData;
}
```

Jakmile jsou všechna potřebná data levelu sesbírána a zpracována, probíhá jejich uložení do souboru. Veškeré informace o levelu, včetně pozic a rotací objektů, jejich propojení a základních herních prvků, se serializují do formátu JSON. Tento formát je zvolen kvůli své jednoduchosti, čitelnosti a snadné manipulaci při načítání.

Výsledný soubor se uloží do složky určenou pro tuto hru která se nachází v adresáři appData.

### 2.3.2 Načítání

Při načítání levelu hra vyhledá uložená data na základě GUID a poté z nich rekonstruuje všechny objekty, propojení i nastavení levelu. Tento proces se skládá z několika hlavních kroků:

1. Získání GUID uloženého levelu - Na začátku skript získá GUID levelu, který má být načten. Tato hodnota se uchovává v PlayerPrefs, což umožňuje hře zapamatovat si, který level má být načten i po restartu.
2. Vyhledání souboru s uloženým levellem - Hra poté prochází adresář Levels v AppData a hledá odpovídající složku s uloženými daty. Pokud hráč načítá level stažený přes Steam, přepne se cesta do SteamManager.steamDownloadPath. Pokud složka neexistuje, načítání se zastaví a zobrazí se chybová zpráva.
3. Načtení JSON souboru - Po nalezení odpovídající složky se načte JSON soubor, který obsahuje uložená data levelu. Tento soubor se následně deserializuje do objektu LevelData, což umožní manipulaci s uloženými informacemi.
4. Zpracování režimu načítání (Build / Play)

- 
- I. Level se může načíst ve dvou režimech:
  - II. Build mód – Slouží pro editaci levelu v editoru.
  - III. Play mód – Používá se pro hraní levelu. Pokud je level určený pro jednoho hráče, druhá postava (CharacterB) se deaktivuje a kamera se přizpůsobí na fullscreen.
5. Rekonstrukce objektů - Každý uložený objekt v levelu se postupně znovu instancuje.
- I. Skript načte objekt podle jeho názvu a vyhledá odpovídající prefab ve složce Prefabs/Building.
  - II. Pokud objekt existuje, vytvoří se jeho instance, nastaví se pozice, rotace a umístí se do herní scény.
  - III. Pokud objekt podporuje editaci bloků, skript ověří, zda je objekt dvoudílný (TwoBlocks), a v takovém případě přidá další editační kostky.
6. Obnova propojení mezi objekty
- I. Pokud měl objekt při ukládání aktivní propojení, tato propojení se znovu vytvoří.
  - II. Skript si uchovává seznam propojení a po načtení všech objektů vygeneruje nové spojovací čáry (Line Renderer).
  - III. Pokud je načítání v Build módu, spojovací čáry jsou uloženy do složky Lines pro další editaci.
  - IV. Skript vyhledá odpovídající objekt v Build složce a propojí jej se správným vstupním/výstupním bodem.
7. Nastavení pozic hráče
- I. Pokud je načítaný level kooperativní (Coop), hra umístí dvě postavy na odpovídající startovní pozice.
  - II. V opačném případě se pouze nastaví pozice hlavní postavy.

Výsledek načítání - Po dokončení těchto kroků je level úspěšně načten a obsahuje všechny objekty, propojení i startovní pozice hráče. Systém načítání tak umožňuje rychlé a přesné obnovení uložených levelů, přičemž zajišťuje správné propojení všech interaktivních prvků ve scéně.

## 2.4 Herní scéna

V **herním režimu** nejsou kolize mezi hráči vypnuty, protože umožňují **zajímavé interakce**, které mohou ovlivnit průběh hry. Díky tomu si hráči mohou **vzájemně pomáhat**, například **lezením na sebe**, což může být užitečné při překonávání překážek nebo při hledání alternativních cest k cíli. Tento prvek **přidává do hry další vrstvu spolupráce**, aniž by rušil plynulost pohybu nebo způsoboval výrazné problémy v mechanikách hry.

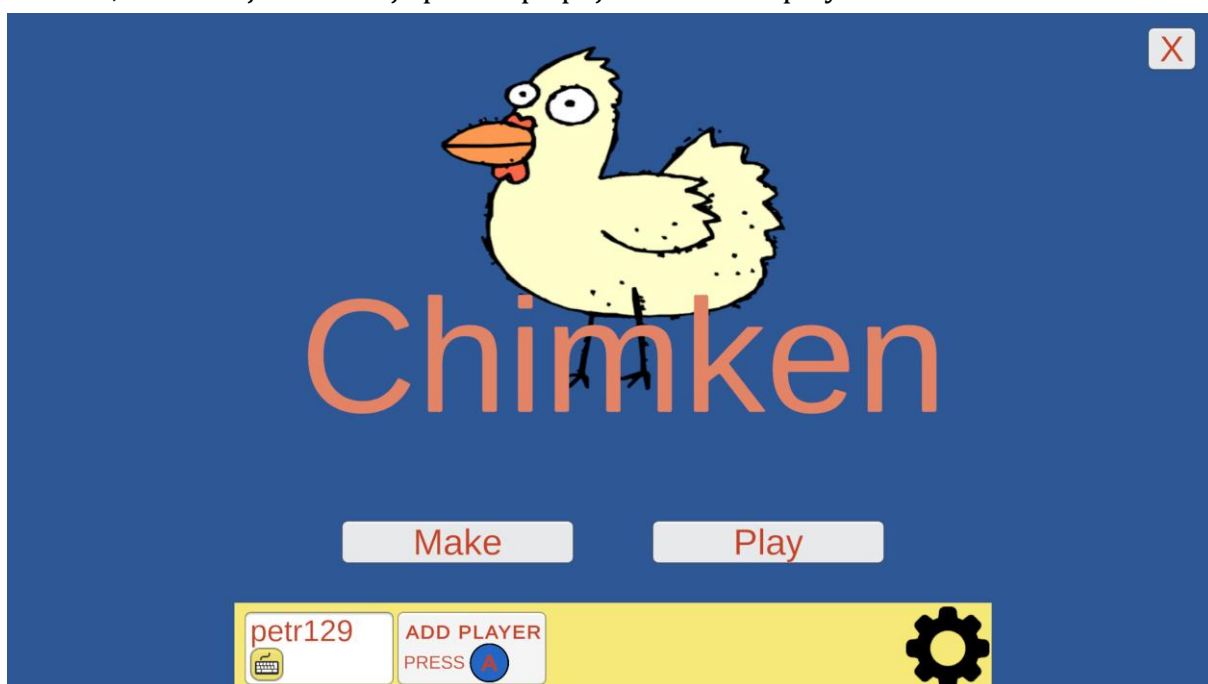
Každý level obsahuje **dva startovní body a dva cílové body**, které jsou **nezávislé na konkrétním hráči**. To znamená, že **není důležité, do kterého cíle se který hráč dostane** – klíčové je, aby **oba hráči současně stáli na cílových pozicích**. Tento systém zajišťuje, že **hráči musí spolupracovat a koordinovat své pohyby**, aby mohli úspěšně dokončit level.

## 2.5 UI

### 2.5.1 Hlavní menu

Hlavní menu slouží jako **centrální rozcestník** pro navigaci v uživatelském rozhraní hry. Umožňuje hráči **rychlý přístup ke všem důležitým funkcím**, včetně výběru herního režimu, editoru levelů a dalších nastavení.

Jednou z klíčových funkcí hlavního menu je **možnost připojení druhého hráče**. Pokud hráč používá **herní ovladač**, může se druhý hráč připojit **stisknutím tlačítka A**. Tento mechanismus je navržen tak, aby umožnil **okamžitý vstup do kooperativního režimu**, čímž se zjednodušuje proces připojování v multiplayeru.



Obrázek 4 - Main menu

#### 2.5.1.1 Vytvářecí menu

Toto menu slouží ke správě lokálně uložených levelů a umožňuje hráči jejich vytváření, úpravu, testování a mazání. Umožňuje tak jednoduchý přístup ke všem vytvořeným úrovním bez nutnosti manuální správy souborů.

Funkce menu:

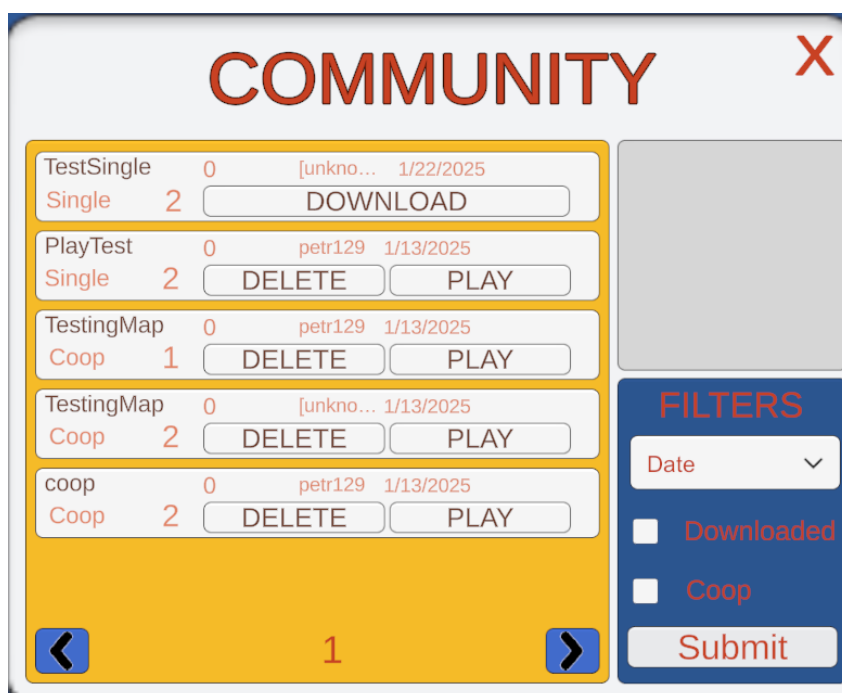
1. Seznam uložených levelů – V hlavní části menu se nachází seznam všech lokálně uložených map, přičemž každý level je doplněn jeho názvem a označením herního režimu (Single nebo Coop).
2. Editace levelů – Každý level má u sebe ikonu tužky, která umožňuje otevření editoru a úpravu dané úrovně.
3. Mazání levelů – Vedle každého levelu se nachází tlačítko „X“, které slouží k odstranění levelu z úložiště.
4. Vytvoření nového levelu – V pravé části menu je tlačítko „NEW MAP“, které umožňuje vytvoření zcela nové úrovně.



Obrázek 5 - Make menu

#### 2.5.1.2 Komunitní levely menu

Community menu slouží k prohlížení, stahování a správě levelů vytvořených komunitou. Umožňuje hráčům stahovat nové levely, organizovat již stažené mapy a aplikovat filtry pro snadnější orientaci v dostupném obsahu.



Obrázek 6 - Community menu

### 2.5.1.3 Nahrávací menu

Systém **uploadu levelu na Steam Workshop** je realizován prostřednictvím **upload menu**, které umožňuje hráči **pořídít náhled levelu a následně jej nahrát** na Steam. Toto menu je klíčovou součástí **editoru levelů**, protože poskytuje jednoduchý a intuitivní způsob, jak sdílet vytvořené úrovně s komunitou.



Obrázek 7 - Upload menu

V horní části okna je **sekce „PREVIEW IMAGE“**, kde se zobrazuje **aktuální náhled levelu**. Tento obrázek je důležitý, protože slouží jako **vizuální reprezentace**

---

**levelu na Steam Workshopu.** Hráč má možnost **pořídít nový náhled** stisknutím tlačítka „**SHOOT**“, což vygeneruje přepne hráče do fotografického módu ve kterém může létat okolo, zamířit na důležitou část levelu, nebo celý level a zmáčknout P které ho vezme zpátky do upload menu.

Pod náhledem se nachází **tlačítko UPLOAD**, které umožňuje **odeslat level na Steam Workshop**. Toto tlačítko je zmáčknutelné pouze tehdy, pokud je level **správně uložen a obsahuje všechny potřebné prvky**, jako jsou startovní, cílové body a je vyfocen obrázek. Po stisknutí tlačítka dojde k **asynchronnímu nahrání souboru**, přičemž hra zpracuje veškerá potřebná data, jako je **název levelu, jeho ID a přidružené soubory**.

## 3 Chyby a problémy

### 3.1 Editor

#### 3.1.1 Nekonzistence bloků

Na začátku vývoje byly všechny stavební prvky ve hře **pouze kostky**, což umožňovalo **jednoduchý systém pokládání** bez větších komplikací. Každý objekt byl obklopen **pomocnými kostkami**, které zajišťovaly, že se **všechny bloky umisťují přesně do gridové struktury**.

Během testování se však ukázalo, že **mezi některými bloky vznikají drobné odchylky**, zatímco jiné se **mírně překrývají**. Tento problém se prohluboval **s rostoucím počtem položených objektů**, což vedlo k narušení **rovnoměrnosti stavební mřížky**.

Příčinu problému se podařilo identifikovat **pomocí logování názvů objektů**, do kterých **RayCast paprsek** při pokládání narazil. Analýza ukázala, že **některé objekty měly mírně větší rozměry**, než bylo očekáváno. Kvůli tomu docházelo k tomu, že se **nové bloky umisťovaly přímo na povrch těchto objektů místo na pomocné kostky**, což narušovalo přesnost gridového systému.

Řešení spočívalo v **úpravě detekce kolizí** v editor módu. Bylo provedeno **deaktivování všech colliderů** u objektů a zároveň bylo nastaveno, aby **RayCast ignoroval všechny vrstvy kromě vrstvy pomocných kostek**. Tím se zajistilo, že se **všechny nové objekty pokládají výhradně na určené body gridu**, čímž se odstranily chyby v umisťování a zachovala se konzistence stavebního systému.

#### 3.1.2 Nemazáním propojení

Systém propojování bloků byl do hry implementován až v pozdější fázi vývoje, což vedlo k nekompatibilitě s některými existujícími funkcemi. Během testování se zjistilo, že při mazání objektů se jejich propojení neodstraňovala, což vedlo k tomu, že propojení zůstala v prostoru bez připojených objektů. Tento problém však neovlivňoval proces ukládání a načítání, protože propojení jsou uložena přímo v propojených objektech. Chyba se tedy vyskytovala pouze během jedné editační relace, nikoliv po restartu hry.

Identifikace problému byla relativně snadná, protože propojení ve hře existují jako samostatné objekty, a proto se při odstranění propojeného bloku nesmazala automaticky. Řešením bylo přidání kontroly při mazání objektu, která ověřuje, zda je objekt součástí propojovacího systému. Pokud ano, skript projde všechna aktivní propojení a odstraní je spolu s daným objektem. Tento přístup zajistil, že ve hře nezůstávají osamocená propojení, která by mohla vizuálně i funkčně narušovat herní prostředí.

---

## 3.2 Steam

### 3.2.1 Editování prvků ve workshopu

Při práci se Steam Workshopem nastal problém s aktualizací úrovní, protože neexistoval spolehlivý způsob, jak z lokálních souborů zjistit, zda již byl daný level na Steam nahrán. To vedlo k situaci, kdy bylo obtížné rozlišit mezi novým uploadem a aktualizací existujícího obsahu.

Řešení spočívalo v tom, že při prvním nahrání levelu na Steam se získalo unikátní Steam ID a to se následně uložilo do lokálního souboru levelu. Díky tomu bylo při dalším pokusu o upload možné ověřit, zda soubor obsahuje existující Steam ID. Pokud ID bylo přítomno, provedla se aktualizace daného levelu. Pokud ID chybělo, znamenalo to, že level dosud na Steamu neexistuje, a byl tedy vytvořen jako nový upload.

### 3.2.2 Získávání náhledů levelů

Při implementaci načítání náhledových obrázků levelů z Steam Workshopu nastal problém, kdy hodnota PreviewImageURL v objektu item nebyla správně vyplněna a vracela null, přestože tato položka existovala. To znemožňovalo automatické získání náhledu přímo z položky ve Workshopu.

Řešení spočívalo v tom, že při prvním vygenerování tlačítka pro spuštění levelu se ImageURL uloží do proměnné přímo v komponentě tlačítka. Když poté hráč klikne na tlačítko, URL je použito k načtení náhledu levelu pomocí UnityWebRequestTexture, který stáhne obrázek a vykreslí ho jako sprite v uživatelském rozhraní. Tento přístup umožnil spolehlivější zpracování náhledů a odstranil problém s chybějícími obrázky ve výběru levelů.



## Závěr

Tato maturitní práce se zabývá vývojem kooperativní 3D hry v prostředí Unity. Cílem bylo vytvořit komplexní herní systém zahrnující editor úrovní, inventář, stavební mechaniky a propojení se službou Steam. Na projektu bylo odpracováno 250 hodin, přičemž vývoj probíhal s důrazem na modularitu, uživatelskou přívětivost a efektivní správu herních dat.

Během práce bylo implementováno několik klíčových mechanik, jako je systém drag and drop v inventáři, ukládání a načítání levelů pomocí JSON formátu a pokročilé propojení objektů ve scéně. Významným prvkem bylo také zajištění podpory kooperativního hraní a integrace se Steam Workshopem pro sdílení obsahu mezi hráči.

Celkově tato práce poskytuje ucelený přehled o vývoji her v Unity a demonstruje využití moderních technologií pro tvorbu interaktivního prostředí. Přestože během vývoje bylo nutné řešit řadu výzev, výsledný projekt ukazuje praktické možnosti implementace komplexních herních systémů. Budoucí rozšíření by se mohlo zaměřit zejména na přidávání nových stavebních objektů, rozšíření interakcí mezi hráči a implementaci online multiplayeru, čímž by se dále zvýšil potenciál hry a její herní variabilita.

---

## Seznam zkratek a odborných výrazů

### **API**

Application Programming Interface – rozhraní pro programování aplikací.

### **COOP**

Cooperative Multiplayer (kooperativní hra)

### **FPS**

First-Person Shooter (střílečka z pohledu první osoby) **CORS**

### **GUID**

Globally Unique Identifier

### **JSON**

JavaScript Object Notation

### **UI**

User Interface (uživatelské rozhraní)

### **ID**

Identifikační číslo

### **COLLIDER**

Komponenta Unity určující kolizní plochy objektu

## Seznam obrázků

Obrázek 1 - Inventory .....	6
Obrázek 2 - Building example .....	9
Obrázek 3 - Connections example.....	10
Obrázek 4 - Main menu.....	13
Obrázek 5 - Make menu.....	14
Obrázek 6 - Community menu .....	15
Obrázek 7 - Upload menu.....	15

---

## Použité zdroje

1. **Staff, Coursera.** What Is Unity? *coursera*. [Online] Coursera, 25. listopad 2024. [Citace: 16. leden 2025.] <https://www.coursera.org/articles/what-is-unity>.
2. **Unity Technologies.** LineRenderer. *Unity Documentation*. [Online] Unity Technologies, 17. únor 2025. [Citace: 18. únor 2025.] <https://docs.unity3d.com/6000.0/Documentation/ScriptReference/LineRenderer.html>.
3. —. Physics.Raycast. *Unity Documentation*. [Online] Unity Technologies, 18. únor 2025. [Citace: 18. únor 2025.] <https://docs.unity3d.com/6000.0/Documentation/ScriptReference/Physics.Raycast.html>.
4. **Valve Corporation.** about. *Steam*. [Online] Valve Corporation. [Citace: 3. 1 2025.] <https://store.steampowered.com/about/?l=czech>.
5. —. Steam Workshop. *Steamworks*. [Online] Valve Corporation. [Citace: 2. 1 2025.] <https://partner.steamgames.com/doc/features/workshop>.
6. **Steamworks.** *Valve Developer Community*. [Online] 15. červen 2024. [Citace: 16. prosinec 2024.] <https://developer.valvesoftware.com/wiki/Steamworks>.
7. **What is Steamworks.** *Facepunch.Steamworks Wiki*. [Online] [Citace: 13. prosinec 2024.] <https://wiki.facepunch.com/steamworks/>.

## A. Seznam příložených souborů

Na přiloženém datovém nosiči se nacházejí následující soubory a složky:

- **MP2025-Machačka-Petr-P4A-Logicka\_hra\_s\_editorem\_a\_sdilenim\_map.docx** – editovatelná verze dokumentace maturitní práce
- **MP2025-Machačka-Petr-P4A-Logicka\_hra\_s\_editorem\_a\_sdilenim\_map.pdf** – tisknutelná verze dokumentace maturitní práce
- **MP2025-Machačka-Petr-P4A-Logicka\_hra\_s\_editorem\_a\_sdilenim\_map-Code.zip** – kompresovaný projekt
- **Steam odkaz** – <https://store.steampowered.com/app/3336140/Chimken/>