VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

KRÁTKÉ MINIHRY PRO VÍCE HRÁČŮ

QUICK MULTIPLAYER MINIGAMES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **ADAM WOSKA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. CHLUBNA TOMÁŠ**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2021/2022

Zadání bakalářské práce

Student: **Woska Adam**
Program: Informační technologie
Název: **Krátké minihry pro více hráčů**
Short Multiplayer Minigames
Kategorie: Počítačová grafika

Zadání:

1. Seznamte se s vývojovým prostředím herních enginů (Unity, Unreal Engine...).
2. Vyberte a nastudujte zajímavé herní mechaniky vhodné pro hry s více hráči.
3. Navrhněte aplikaci demonstrující vybrané mechaniky jako sérii miniher.
4. Demo aplikaci implementujte.
5. Proveďte měření, případně uživatelskou studii hodnotící implementované výsledky.
6. Vytvořte video reprezentující výsledky vaší práce.

Literatura:

- Gregory, Jason. *Game engine architecture*. crc Press, 2018. ISBN 1351974289, 9781351974288
- Bishop, Lars, et al. "Designing a PC game engine." IEEE Computer Graphics and Applications 18.1 (1998): 46-53.
- Adams, Ernest, and Joris Dormans. Game mechanics: advanced game design. New Riders, 2012. ISBN 0321820274, 9780321820273

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3, experimenty vedoucí k bodu 4.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Chlubna Tomáš, Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 11. května 2022
Datum schválení: 1. listopadu 2021

Abstrakt

Cílem této práce je návrh a implementace herního dema, jehož obsahem jsou krátké mini hry pro více hráčů. K implementaci dema bylo použito Unity 3D a Photon Engine. Demo se skládá z třech samostatných mini her. Dalším obsahem této práce je průzkum a analýza her podobného žánru. V práci je dále popsán návrh aplikace a její implementace. Výsledek práce byl poté testován reálnými uživateli.

Abstract

The goal of this thesis is to design and implement a game demo that contains short mini multiplayer games. Unity 3D and Photon Engine were used to implement the demo. The demo consists of three separate mini games. Another content of this work is the research and analysis of games of similar genre. The design of the application and its implementation is also described in this thesis. The result of the work was then tested by real users.

Klíčová slova

hra, herní demo, počítačová hra, multiplayer, návrh her, Unity, C#, Photon

Keywords

game, game demo, computer game, multiplayer, game design, Unity, C#, Photon

Citace

WOSKA, Adam. *Krátké minihry pro více hráčů.* Brno, 2022. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Chlubna Tomáš

Krátké minihry pro více hráčů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Tomáše Chlubny. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Adam Woska
11. května 2022

Poděkování

Chtěl bych poděkovat vedoucímu mé bakalářské práce panu Ing. Tomáši Chlubnovi za odborné vedení práce a za čas, který mi věnoval.

Obsah

Kapitola 1

Úvod

Hry jsou součástí lidských životů již od jejich počátku. Malé děti milují hraní všelijakých her, ať to je hra na schovávanou, hraní si s hračkami či hraní karetních nebo deskových her. Postupem času, některé z těchto dětí objeví video hry. Někteří rodiče neradi vidí, když jejich dítě hraje počítačové či mobilní hry. Video hry jsou ale ideální nástroj k učení. Při jejich hraní totiž děti ani neví že se učí [5]. Hry ale nejsou určeny jen k učení ale i k zábavě a vytvořit takovou hru je cílem této práce.

Herní demo, které je výsledkem této práce umožňuje společně hrát dvěma až čtyřem hráčům, a to přes internet. Demo se skládá z menu a tří mini her. K tvorbě aplikace se bylo nutné seznámit s herním enginem Unity a také prozkoumat dostupná řešení k tvorbě online her pro více hráčů. Dále bylo za potřební vyhledat hry podobného charakteru a zanalyzovat je. Poté přišel na řadu návrh. Bylo navrženo grafické rozhraní aplikace a celkem 5 mini her. Ty se od sebe odlišují herními mechanikami a perspektivou, ze které je hra hrána. Změna perspektivy ve hrách ovlivňuje mechaniky, které u ní lze použít.

Nakonec byly implantovány tři mini hry z pěti. Implementace všech pěti mini her se ukázala být těžší, než se na první pohled zdálo. Pokud má vývoj hry na starost pouze jeden člověk, tak musí dělat spoustu věci. Hry je nutné v celé fázi vývoje testovat, a to hraním, které zabírá spoustu času. Testování her pro více hráčů je ještě víc časově náročnější.

V následující kapitole 2, jsou rozebrány podobné herní tituly a také je zde uvedena definice slova hra. Popis použitého game enginu a jeho alternativ je uveden v kapitole 3. V kapitole 3 je také popsána síťová technologie, použitá k implementaci hry pro více hráčů. V kapitole 4 se nachází návrh grafické částí a návrh pěti mini her. Samotná implementace celkové aplikace a jednotlivých mini her je popsána v kapitole 5. V poslední kapitole 6 je zmíněno testování hry na cílových uživatelích.

Kapitola 2

Počítačové hry

Kapitola vysvětluje, co to je hra a také se zaměřuje na analýzu herních titulů, které jsou stejného žánru jako herní dema. Tímto žánrem je „párty hra“. Herních titulů, které by spadaly do podobného žánru není příliš mnoho. Tedy takových her, co obsahují různé krátké mini hry a lze je hrát online se svými přáteli. Tři vybrané hry z této kategorie jsou v téhle kapitole popsány.

2.1 Obecná definice hry

Definice toho, co vlastně je „hra“ není ustálená. Známí herní designeři uvádějí různé a někdy si i protiřečící definice hry. Například Sid Meier, designér série Civilizace uvádí následující definici: „Hra je série smysluplný rozhodnutí“. Katie Salen a Eric Zimmerman uvedli v jejich knize Rules of Play[1] definici následující: „Systém, kde jsou hráči zapojeni v umělém konfliktu, jež je definován pravidly, která ústí v měřitelné výsledky“. Většina definicí popisuje hru jako systém, simulaci, něco, co je odtrženo od reality. Mají však něco společné, a to, jak jsou ve hrách důležitá pravidla, možnost volby a konflikt. Hry jsou velmi silným nástrojem k učení. Stimulují mozek. Při hraní her má mozek za úkol rozpoznat opakující se vzor chování hry. To dělá až do té doby, dokud se je nenaučí tak dobře až ho hra začne nudit. Typickým příkladem je hra piškvorky. Pokud mozek odhalí vzor, podle kterého hru lze vyhrát, tak již její hraní nemá žádný smysl. Zároveň ale hra nesmí být příliš těžká to pak vede k tomu, že hráč hru nebude vůbec chtít hrát [6].

2.2 Pummle Party

Pummle Party je multiplayerová party hra pro až 8 hráčů. Obsahuje dva herní módy. Prvním z nich je mód, ve které hráč hraje náhodné mini hry a podle umístění získává body, které se mu sčítají. Na konci série mini her se ukáže celkové pořadí – kdo zvítězil a kdo prohrál.

V dalším módu se hraje desková hra. Na herních políčkách čekají různé postihy či bonusy, kterými lze například obrat protihráče o klíče. Některé políčka spouští mini hry. Za výhru v mini hře dostanete klíče. A jakmile má hráč určitý počet klíčů, tak může otevřít truhlu. Truhla se objevuje náhodně na hrací ploše. Z truhly hráč získá trofej a po získání určitého počtu trofejí vyhrává. Velkým nedostatkem hry je nekonzistentní ovládání. Čímž je myšleno to, že akce stejného charakteru jsou ve dvou odlišných mini hrách kontrolovány jinými tlačítky. Obrázek 2.1 ukazuje, jak vypadá jedna z mini her.

[1] https://mitpress.mit.edu/books/rules-play

Obrázek 2.1: Ukázka mini hry - Memory Menu ze hry Pummle Party. V této mini hře si má hráč zapamatovat zobrazené ingredience, nashromáždit je a přidat je do kotle. Hra je omezena časovým limitem, který lze vidět v pravém horním rohu. Za správně přidané ingredience, obdrží hráč bod. Počet získaných bodů lze vidět na spodní straně obrazovky.

2.3 Party Panic

Hra Party Panic je velmi podobná jako výše zmíněná Pummle Party 2.2. Jeden z hlavních rozdílů je absence herního módu s deskovou hrou. To, co je podobné je složení jednotlivých mini her. A opět se zde nachází problém s nekonzistentním ovládání. V jedné z her se ovládá skok pomocí mezerníku, ale v jiné je to tlačítko W. Ovládání lze přenastavit v nastavení hry. Uživatelé by jistě ocenili, aby ovládání bylo nastaveno správně již od začátku. Obrázek 2.2 ukazuje, jak vypadá jedna z mini her.

Obrázek 2.2: Ukázka mini hry – Fizzle Floors ze hry Party Panic. V horní části obrazovky lze vidět aktuální skóre hráčů a časomíru se zbylým časem. Ve hře mají hráči za úkol vydržet co nejdéle na hrací ploše. Ta je tvořena z kostek, které se po dotyku hráče začnou hroutit. Tím vznikne prázdný prostor, kterým může hráč propadnout do lávy.

2.4 Mario Party Superstars

Hra Mario Party Superstars obsahuje 100 mini her. Nachází se zde více kategorií mini her, jako třeba *všichni proti všem*, *duel* nebo *jeden hráč proti všem* či *nejvyšší skóre*. Je to nejnovější díl z herní série Mario Party. Hra je dostupná pouze pro herní konzoli Nintendo Switch. Vydavatelem série je společnost Nintendo[2], která byla z prvních průkopníků video herního průmyslu. Obrázek 2.3 ukazuje, jak vypadá jedna z mini her.

[2] https://www.nintendo.com/

Seznam vybraných miniher:

- Spin Doctor – mini hra z kategorie duel. Hráči se musí dostat z bodu A do bodu B. K cíli vedou propojené mosty, ale na rozcestí se nachází točící se platforma, na kterou musí hráč ve správný moment vstoupit.
- Handcar Havoc – mini hra z kategorie nejvyšší skóre. Hráči proti sobě závodí ve vozících. Pokud však naberou velkou rychlost, tak můžou z tratě vypadnout a prohrát závod.
- Tug o' War – mini hra z kategorie 1 vs. 3. Hráč, který je sám má větší sílu a snaží se lanem stáhnout protihráče do propasti. Protihráči musí správně spolupracovat, aby ho do propasti přetáhli oni.
- Etch 'n' Catch – mini hra z kategorie 2 vs. 2. Hráči musí se svým partnerem zakroužkovat co nejvíce razítek na jejich hrací ploše. Vyhrává ten tým, který jich zakroužkuje víc.
- Snowball Summit – mini hra z kategorie všichni proti všem. Hráči jsou na zasněženém vrcholku hory. Jejich úkolem je uválet velkou sněhovou kouli a odstrčit s ní protivníka z hory.
- Cheep Cheep Chase – mini hra z kategorie všichni proti všem. Hráči musí uplavat velké rybě, která se je snaží dohnat. Ve vodě jsou bomby. K vyhnutí se bombě je nutné se ve správný moment potopit.

Obrázek 2.3: Ukázka mini hry – Messy Memory[3] ze hry Mario Party Superstars. Hráč má za úkol vrátit všechny předměty na jejich původní místo. Jakmile je hotov, tak má možnost jeho obrazovku zastřít, aby ho ostatní hráči nekopírovali. Uprostřed obrazovky lze vidět časomíru se zbylým časem.

[3]Převzato z https://www.nintendo.com/ - hra Mario Party Superstars (2021)

Kapitola 3

Použité technologie

Tahle kapitola rozebírá technologie, které byly použity při vývoji aplikace. Dále vysvětluje, co je to herní engine, a proč je pro herní vývoj důležitý. Dále uvádí herní engine, který byl k implementaci použit a také vysvětluje jeho základní funkce a schopnosti. Na konci kapitoly jsou zmíněny alternativní herní enginy.

3.1 Obecná definice enginu

Termín „herní engine“ se objevil v polovině 90. let 20. století v souvislosti s hrami z pohledu první osoby střílečeк, jako je například nesmírně populární Doom od společnosti id Software[1] [3]. Doom byl navržen, tak že jeho jádro – softwarové komponenty (vykreslovací jednotka, detekce kolizí nebo zvukový systém) bylo odděleno od modelů, herních úrovní a herních pravidel. To umožnilo vývojářům lehce vytvářet nové produkty, aniž by něco museli měnit v jádru enginu. Tímhle se také zrodila komunita lidí, kteří vytvářejí modifikace her. Což jsou jednotlivci či malá studia, kteří upravují stávající hry s využitím sad nástrojů od vývojářů, a to často bez jakéhokoliv nároku na zisk.

Herní enginy jsou často uzpůsobeny určitému žánru her. Potřeby karetní hry pro jednoho hráče se budou dost lišit od potřeb masivně multiplayerové hry. Unreal Engine, rozebíraný v sekci 3.3, byl například prvně navržen pro střílečky z pohledu první osoby, byl úspěšně použit k tvorbě her řady dalších žánrů [4].

Herním Enginem je tedy myšleno vývojové prostředí, které je uzpůsobeno ke tvorbě her. Od jejich prototypování až po následnou distribuci. To, že jsou již klíčové systémy vytvořeny, šetří vývojářům spoustu času a herním studiím spoustu peněz. Mnoho společností si vytváří a spravuje vlastní herní enginy, takzvané proprietární interní enginy. Společnost Electronic Arts postavila mnoho svých her na vlastním enginu s názvem SAGE. Spousta enginů, které prvně byly proprietární interní enginy, se nakonec staly veřejně dostupnými. Jsou to například Unreal Engine či CRYENGINE[2] [4].

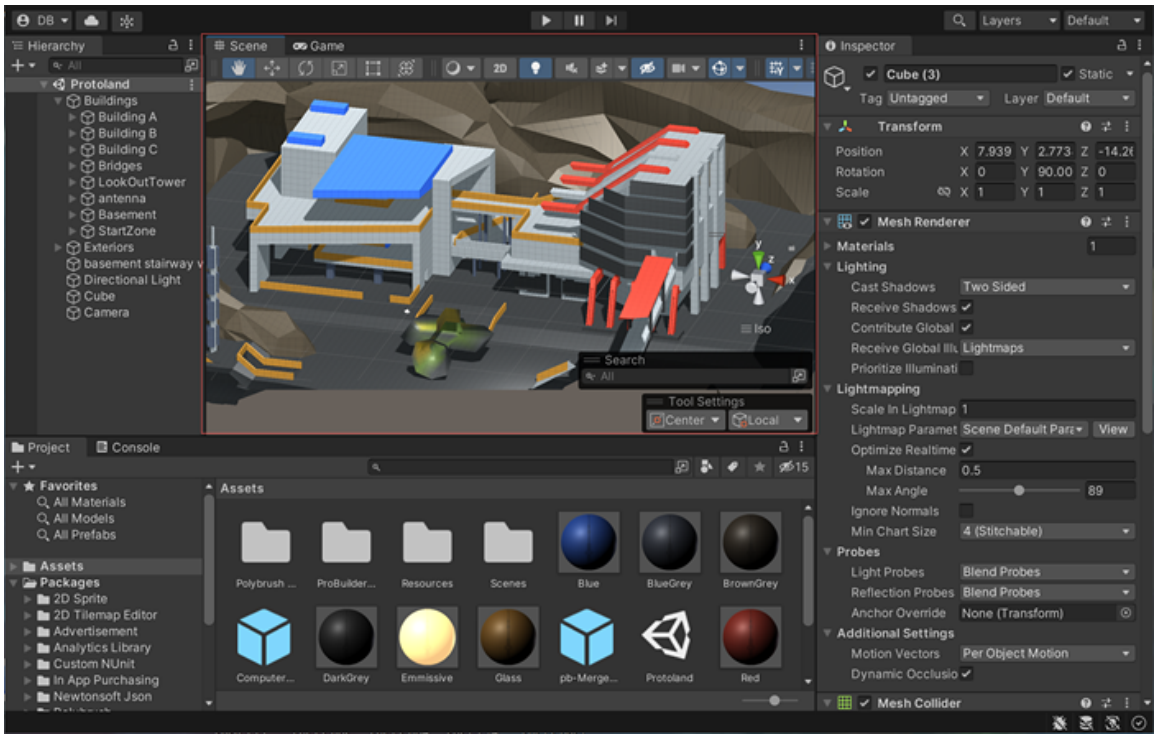
3.2 Unity

Unity[3] je vývojové prostředí sloužící hlavně pro tvorbu her. Pomocí Unity mohou vývojáři své hry vydat na mobilních platformách (např. Apple iOS, Google Android), konzolích (Microsoft Xbox, Sony PlayStation), stolních počítačích (Windows, MacOs, Linxu), systé-

[1] https://www.idsoftware.com/
[2] https://www.cryengine.com/

mech virtuální reality (VR) (např. Oculus Rift, Steam VR, Gear VR). Hlavními cíli návrhu Unity jsou snadný vývoj a multiplatformní nasazení her. Unity taky poskytuje snadno použitelný integrovaný editor, ve kterém můžete vytvářet a manipulovat s prostředky a objekty, které tvoří váš herní svět, a rychle si prohlédnout hru přímo v akci v editoru anebo na cílovém hardwaru. Na obrázku 3.1 lze vidět rozložení editoru.

Unity bylo použito k vytvoření spousty známých her (například Deus Ex: The Fal, Cuphead, Hearthstone). Ale je využíváno i v automobilovém a filmovém průmyslu. Krátký film Adam: The Mirror, který vyhrál cenu Webby Award[4], byl vyrendrován v Unity. V automobilovém průmyslu se zase využívá k trénování nových pracovníků na montážních linkách a prohlížení interiérů aut, a to za pomocí virtuální reality [4]. Jako programovací jazyk Unity využívá C# 8.0. Dříve byl podporován i UnityScript, což byl jazyk založený na JavaScriptu, ten ale není od verze 2017.2 oficiálně podporován. Unity nabízí spousty nástrojů, které ulehčují vývoj her. Nástroje, které zde nenajdete je možné získat na Unity Asset Store[5]. Je to obchod, kde se nabízí nástroje, shadery, modely a mnoho dalších užitečných věcí.



Obrázek 3.1: Ukázka herního enginu Unity. Na levé straně je hierarchie scény. Uprostřed lze vidět, jak obsah scény vypadá. Napravo je inspektor game objektů s jednotlivými komponentami. V dolní části pak hierarchie projektu[6].

Herní objekt

Herní objekt, dále game object, je základní objekt Unity. Game object reprezentuje vše, co může existovat ve scéně (definováno v sekci níže 3.2). Tím jsou myšleny objekty jako

[3] https://www.unity.com/
[4] https://www.webbyawards.com/
[5] Obchod, kde uživatelé nabízejí nástroje či grafiku ke tvorbě her.
[6] Převzato z https://www.unity.com/

například hráč, světla, prostředí či zdroj zvuku. Samy o sobě ale skoro nic nedělají. Tvoří „kontejnery“ pro komponenty (definováno v sekci 3.2), které implementují požadovanou funkcionalitu [9].

Komponenta

Komponenta je dílčí část objektu, která mu přidává funkci. Game object musí obsahovat nejméně jednu komponentu a to *Transform* konečný počet ale omezen není. Unity nabízí mnoho vestavěných komponent a lze si také vytvářet vlastní pomocí psaní skriptů. Ty ale musí dědit ze třídy *MonoBehaviour* [9].

Seznam vybraných vestavěných komponent:

- Transform – je jediná povinná komponenta na herním objektu. Nelze oddělat. Určuje pozici objektu v herním světě, jeho rotaci a měřítko.
- Sprite Renderer – je komponenta, která určuje vzhled 2D objektu.
- Animator Controller - je komponenta, která slouží k ovládání animací objektu.
- Rigidbody – je komponenta, která objekt udělá ovlivnitelný fyzikou. Jsou dva druhy, a to pro 2D a 3D fyziku.
- Collider – je komponenta, která vymezuje kolizní tvar objektu. Existuje více druhů (box, edge, polygon, cube, mesh). Lze ji přepnou do trigger módu. Ve kterém nebude vytvářet kolizi, ale bude ji detekovat.
- Rect Transform – GUI verze klasické komponenty `Transform`. Určuje pozici grafického prvku na obrazovce.
- Canvas – Komponenta, která určuje prostor, do kterého se mají všechny UI prvky vykreslovat. Všechny UI elementy musí být potomky objektu, který má na sobě Canvas. UI elementy, které jsou potomky Canvas se vykreslují podle pořadí v hierarchii (pokud není specifikováno jinak). Tedy pokud se dva UI prvky překrývají, tak ten, který je v hierarchii dřív bude překryt. Canvas má nastavení `Render Mode`, které určuje způsob renderování. Ty jsou celkem tři. V projektu je použit mód `Screen Space - Overlay`, který renderuje UI elementy na obrazovku nad scénou (z pohledu kamery). Dalším módem je `Screen Space - Camera`, při tomhle nastavení se Canvas přizpůsobuje velikosti kamery. A posledním typem je `World Space`, který umožňuje přímo určit souřadnice Canvasu a používá se k takovým grafickým rozhraním, které mají být součástí herního prostoru.
- Canvas Scaler – Komponenta, která mění měřítko `Canvas` a přizpůsobuje ho různým rozlišením.

Scéna

Scéna je kontejnerem pro game objekty. Umožňuje, aby se s objekty dalo manipulovat a aby byly vidět. Jednoduchou hru jde vytvořit v jedné scéně, zatímco u složitější her nebo u her na kterých pracuje více lidí je vhodné použít scén víc. Projekt může obsahovat jakýkoliv počet scén a lze mít aktivních více scén najednou [9].

Coroutine

Coroutine je metoda s návratovým typem `IEnumerator`. Ve většině situací se při zavolání normální metody, metoda vykoná během jednoho snímku. Coroutine umožňuje vykonání metody rozložit do více snímků. V Unity je coroutine metoda, která může pozastavit své provádění a vrátit řízení Unity. Poté pokračuje tam, kde skončila. Coroutine je možné využít například u procedurální animace, či vykonání sérii akcí během nějaké doby nebo u asynchronních operací (čekaní na HTTP přenos, práce se soubory). Coroutine ale nepracují na více vláknech. Běží stále na hlavním vláknu [8].

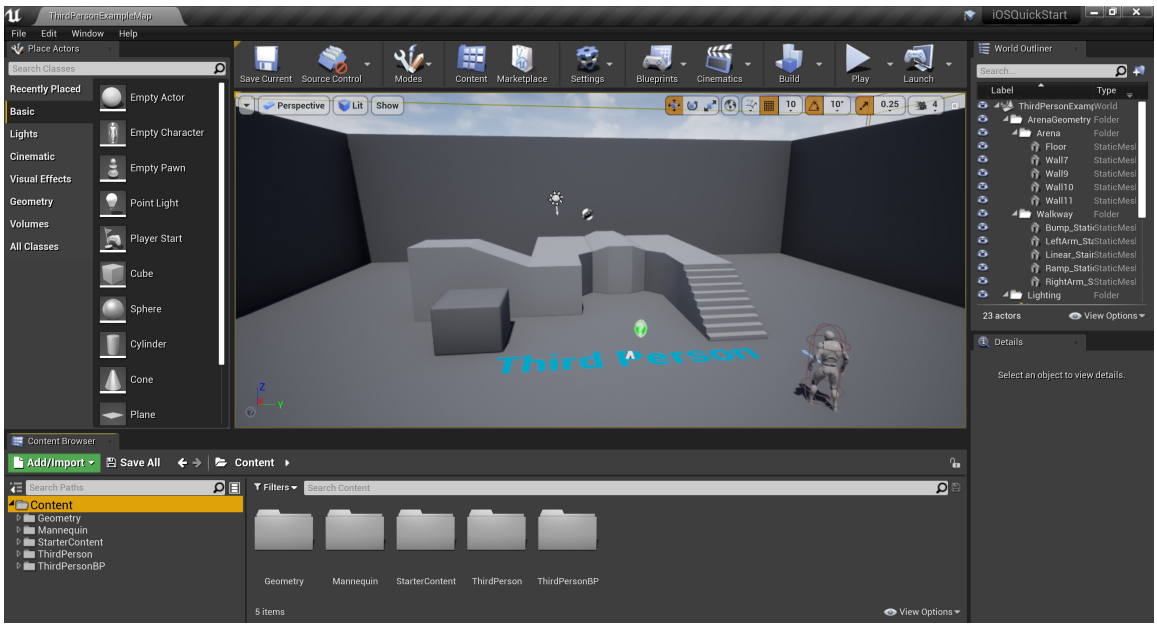
3.3 Alternativní herní enginy

Existuje samozřejmě spousta dalších herních enginů, které se od sebe liší zaměřením, licencí, cenou a nativním programovacím jazykem. Následující sekce některé vybrané z nich popisuje.

Unreal Engine

Unreal Engine[7] byl vytvořen společností Epic Games, Inc. v roce 1998. Unreal Engine a Unity jsou si v mnoha ohledech hodně podobné. Slouží pro tvorbu 2D i 3D her. Nejvýznamnější rozdíl mezi nimi je to, jaký používají nativní programovací jazyk. Unreal Engine používá C++, kdežto Unity používá C#. Na obrázku 3.2 lze vidět grafické rozložení editoru.

Samotné používání není nijak zpoplatněno. Pokud by však hra vydělala více než jeden milión amerických dolarů, pak je nutné odvádět poplatky, které jsou 5% z vydělané částky.



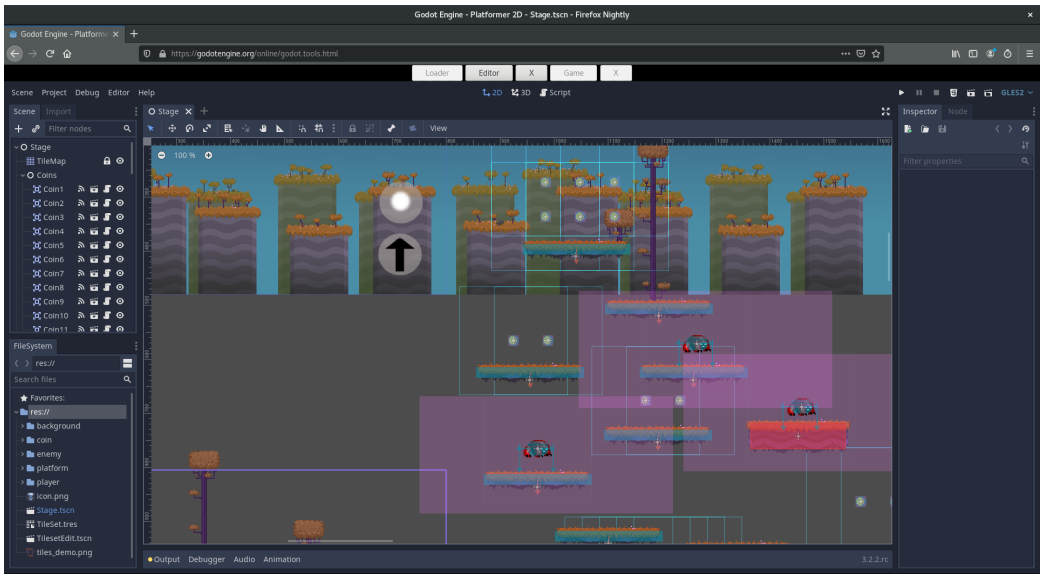
Obrázek 3.2: Ukázka herního enginu Unreal Engine[8].

[7] https://www.unrealengine.com/

[8] Převzato z https://unrealengine.com/

Godot

Godot[9] je open source engine pro tvorbu 2D a 3D her. Jeho využití je zcela zdarma. Při vydání hry rovněž žádné poplatky platit nemusíte. Nativním jazykem Godot je GDScript, jeho syntaxe je velmi podobná jazyku Python. Godot taky podporuje psaní v jazyku C#. Na obrázku 3.3 lze vidět rozložení Godot.



Obrázek 3.3: Ukázka herního enginu Godot[10].

3.4 Photon Engine

Photon Engine[11] je nejpoužívanější služba pro tvorbu online her pro více hráčů. Využívá ji více než 600 000 herních studií a vývojářů s celkovým počtem hráčů dosahující jedné miliardy. Photon Engine nabízí několik multiplayerových[12] řešení (Fusion, Quantum, PUN 2). Hlavní rozdíl mezi nimi je ve službách, které nabízí, cenou a jak moc do hloubky síťování jdou. K účelům tohoto projektu bylo nejvhodnější řešení PUN2 [2]. Photon Engine byl vytvořen firmou `Exit Games` a není součástí herního enginu Unity.

Photon Unity Networking

Photon Unity Networking neboli PUN (verze 2). Je balíček pro Unity dostupný na Unity Asset Store[13]. Slouží k vytvoření multiplayerových her. Je úzce spjatý s Unity a díky tomu lze vytvořit online hru pro všechny platformy, které Unity podporuje. Hry, které jsou vytvořeny s pomocí PUN jsou hostovány na jejich serverech. Ty jsou umístěny v různých částech světa (Amsterdam, Tokyo, Washington a další), aby byla zaručena co nejmenší latence. Je zde i možnost hostování na externím serveru. Využívání tohoto produktu není zpoplatněno do dvaceti souběžně připojených uživatelů. Využívá klient-server architektury. Při potřebě

[9] https://godot.com/
[10] Převzato z https://godot.com/
[11] https://www.photonengine.com/
[12] Anglický výraz pro online hru pro více hráčů.

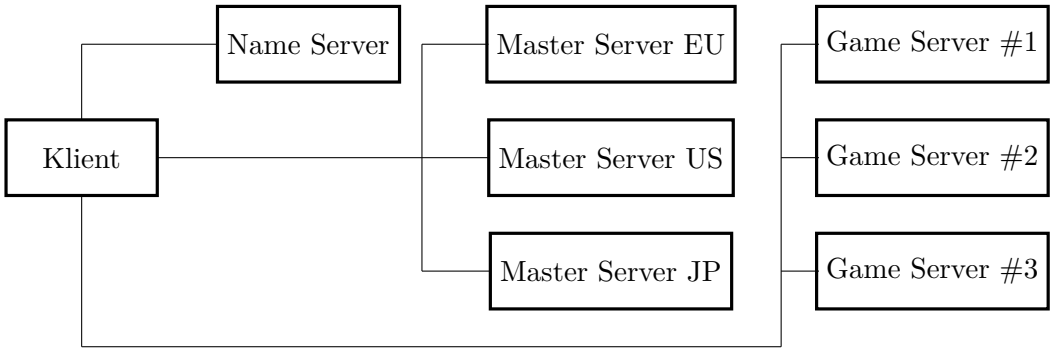
navýšení kapacity ji lze za poplatek kdykoliv navýšit. Rozšíření na 500 souběžně připojených uživatelů pak stojí 95 USD za měsíc [2].

Master server

Prvotní připojení klientů vede na Photon Name Server – ten klientovi vrátí list dostupných regionů. Region se skládá z Master serveru a Game serverů (definováno v sekci 3.4). Master server se stará o match making[14]. Photon nabízí několik Master serverů v různých regionech na světě. Hráči, kteří se nacházejí na různých regionech mezi sebou nemůžou nijak interagovat.

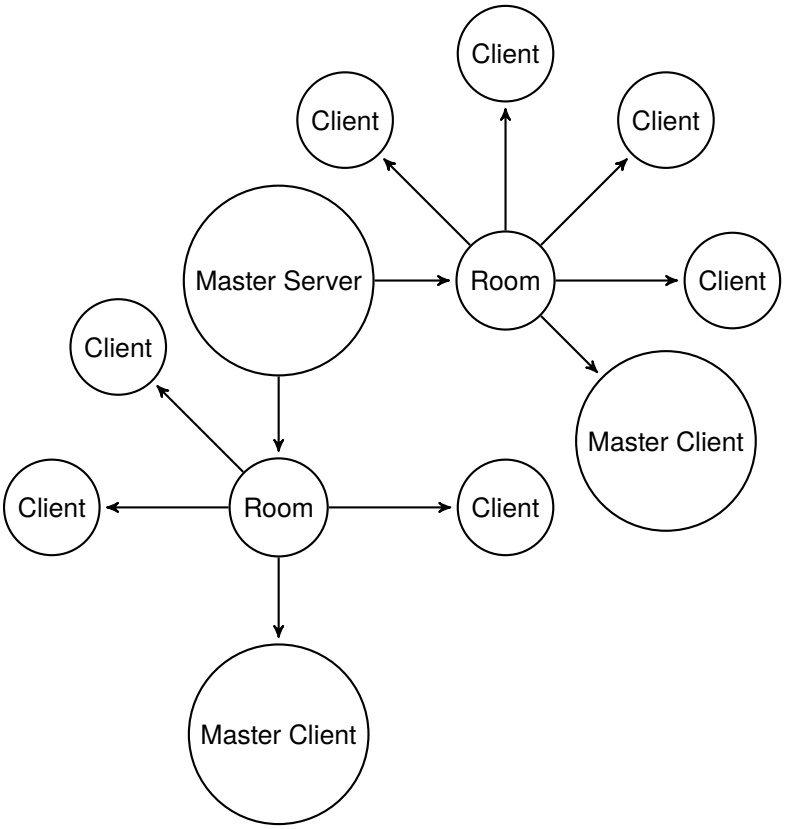
Game Server

Game server či Room (místnost) je oddělená část Master serveru. Do místnosti se hráči připojí, jakmile proběhne match making. Místnost je uzavřené, tudíž dvě různé místnosti nemůžou mezi sebou komunikovat. Hráč, který vytvoří místnost se stane Master client (definováno v sekci 3.4). Přes Game server si klienti posílají zprávy (jejich pozici, rychlost a další). Topologie serverů je zobrazena na obrázku 3.4. Vnitřní topologie místnosti je zobrazena na obrázku 3.5.



Obrázek 3.4: Diagram, který značí postup připojení klienta do místnosti. První se klient připojí na Name server od něho získá seznam Master serverů. Pak se na jeden z nich připojí a získá od něj seznam Game serverů.

[13] https://assetstore.unity.com/packages/tools/network/pun-2-free-119922
[14] Je proces vyhledání a spojení hráčů k účelů hraní online hry.



Obrázek 3.5: Ukázka vnitřní topologie Game Serveru v PUN 2. Můžeme z ní vyčíst, že se na Master serveru nacházejí dvě místnosti. V jedné z nich jsou čtyři klienti a v druhé jich je pět.

Master Client

Klient, který místnost vytvoří se automaticky stává Master klientem. Pokud se Master klient odpojí, tak nastává host migration[15]. Jelikož PUN nemá server, který by kontroloval „logiku na straně serveru“. Tuhle roli zastupuje právě Master Client. Probíhá to tak, že pokud se událost stane na Master klientovi (například kolize dvou hráčů, prohra jednoho hráče), tak on o ní informuje ostatní hráče.

Photon View

Je komponenta, která identifikuje objekt přes síť. Má unikátní číslo, které je stejné na všech klientech v místnosti. Photon view uchovává informace o tom, který klient je jejím vlastníkem a který klient ji vytvořil. Photon view synchronizuje vybrané atributy (pozici, pozici animace) přes síť k ostatním klientům.

Remote Procedure Calls

Remote procedure calls, dále RPC, je funkcionalita, kterou nabízí PUN. Dokáže zavolat metodu na vzdálených klientech. Objekt, na který je metoda volána musí mít na sobě

[15] Je proces předání kontroly nad hrou jinému klientu.

Photon view. RPC se využívá k informování o úmrtí hráče, o změně místnosti či nastavení. Lze ji zavolat na všech klientech, nebo pouze na konkrétním (pokud klient potřebuje poslat serveru informaci, že vyvolal akci).

3.5 Alternativní řešení hry pro více hráčů

Prozkoumány také byly jiné řešení multiplayeru pro Unity.

- Řešení od Unity – Dříve bylo v Unity řešení pro multiplayer, které se jmenovalo UNet (Unity Networking). Toto řešení již není oficiálně podporováno. V současné době je ve vývoji nové řešení „Netcode for GameObjects“. Z toho důvodu, že je řešení ještě ve vývoji nebyla tahle práce pomocí něho implementována.
- Mirror Networking – Mirror Networking[16]. Je open source řešení multiplayer pro Unity. Je uzpůsobeno spíše hrám typu MMORPG[17]. Mirror využívá spousta herních projektů. Také jej využívá „uMMORPG“[18], což je velmi populární MMORPG engine dostupný na Unity Asset Store.

[16] https://mirror-networking.com/
[17] Žánr počítačových her, ve kterém hráči (společně se stovkami jiných hráčů) hrají za svého hrdinu.
[18] https://assetstore.unity.com/packages/templates/systems/ummorpg-51212

Kapitola 4

Návrh

Kapitola popisuje návrh uživatelského rozhraní aplikace a také návrh mini her. Navrženo bylo vícero mini her. Tyto hry se liší v použitých mechanikách, ale sdílejí stejné principy jako je například hra pro více hráčů po internetu. Pro demonstraci vybraných mechanik je dostačující 2D reprezentace scény. Cílem hráčů je úspěšně zvládnout mini hru a získat tak bodové hodnocení. Po odehrání několika miniher se zvolí vítěz. Tím je hráč s nejvyšším počtem bodů. Tituly zmíněné v sekci 2 sloužily jako inspirace pro vybrané herní mechaniky. V sekci 4.1 se nachází popis navrženého uživatelského rozhraní. V sekcích po ní následujících je popis mechanik v navržených hrách.

4.1 Uživatelské rozhraní

Uživatelské rozhraní bylo navrženo na formát obrazu 16:9 na rozlišení FullHD. Pokud má uživatel jiné rozlišení monitoru, tak by se mu mělo uživatelské rozhraní přizpůsobit (to má na starost Canvas Scaler, který je vysvětlen s sekci 3.2). Cílem bylo navrhnout jednoduché, přehledné a minimalistické uživatelské rozhraní. Inspirací pro tvorbu GUI byla hra Boomerang Fu[1].

Úvodní obrazovka

Po spuštění hry se zobrazí úvodní obrazovka (znázorněna na obrázku 4.1). Ta obsahuje dvojici tlačítek `Play` a `Exit`. Tlačítko `Play` bude při spuštění hry deaktivované. Aktivuje se až v momentě, když se hra připojí na server. Poté s ním jde interagovat a přesunout se tak na další obrazovku.

[1] https://www.boomerangfu.com/

Obrázek 4.1: Návrh úvodní obrazovky hry.

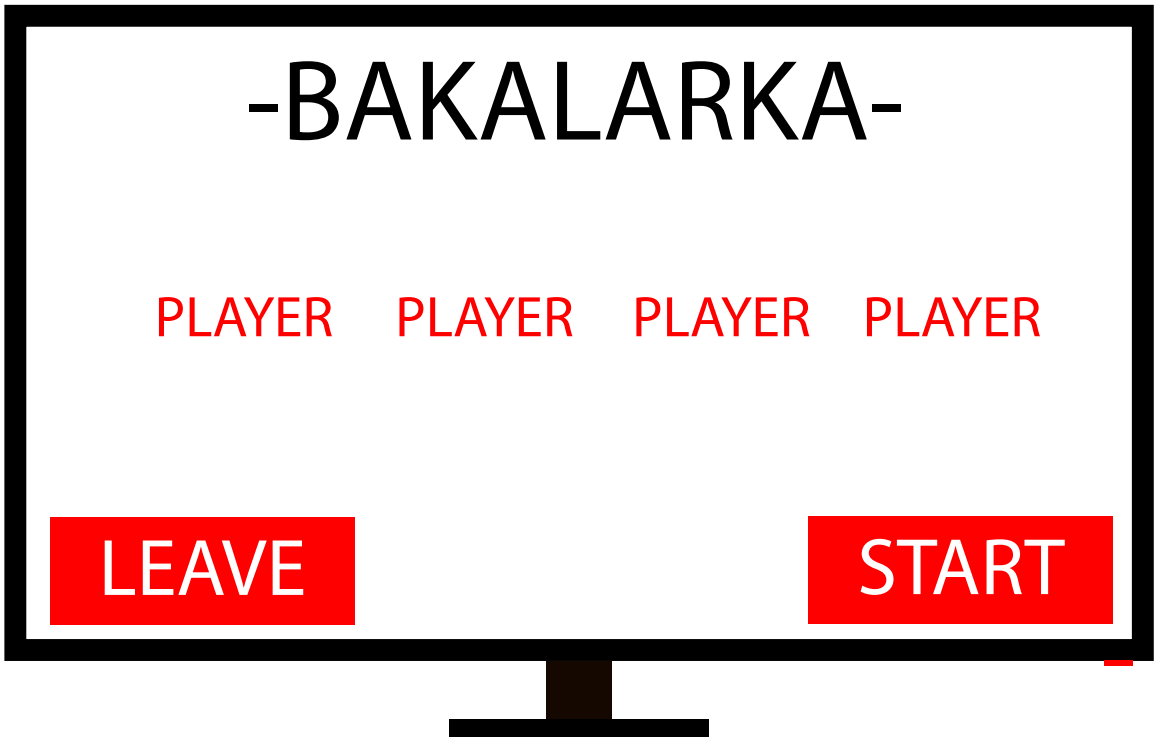
Vytváření místnosti

Obrazovka je rozdělena na dvě hlavní části – dvě tlačítka lze vidět na obrázku 4.2. Tlačítka slouží k vytváření hry anebo k připojování již do vytvořené hry. Vedle tlačítka pro připojení bude také možnost zadat kód místnosti pomocí kterého se půjde dát připojit do konkrétní místnosti. Pokud bude zadán špatný kód, nic se nestane. Pokud správný, tak se obrazovka změní na následující.

Obrázek 4.2: Návrh vytváření místnosti.

Místnost

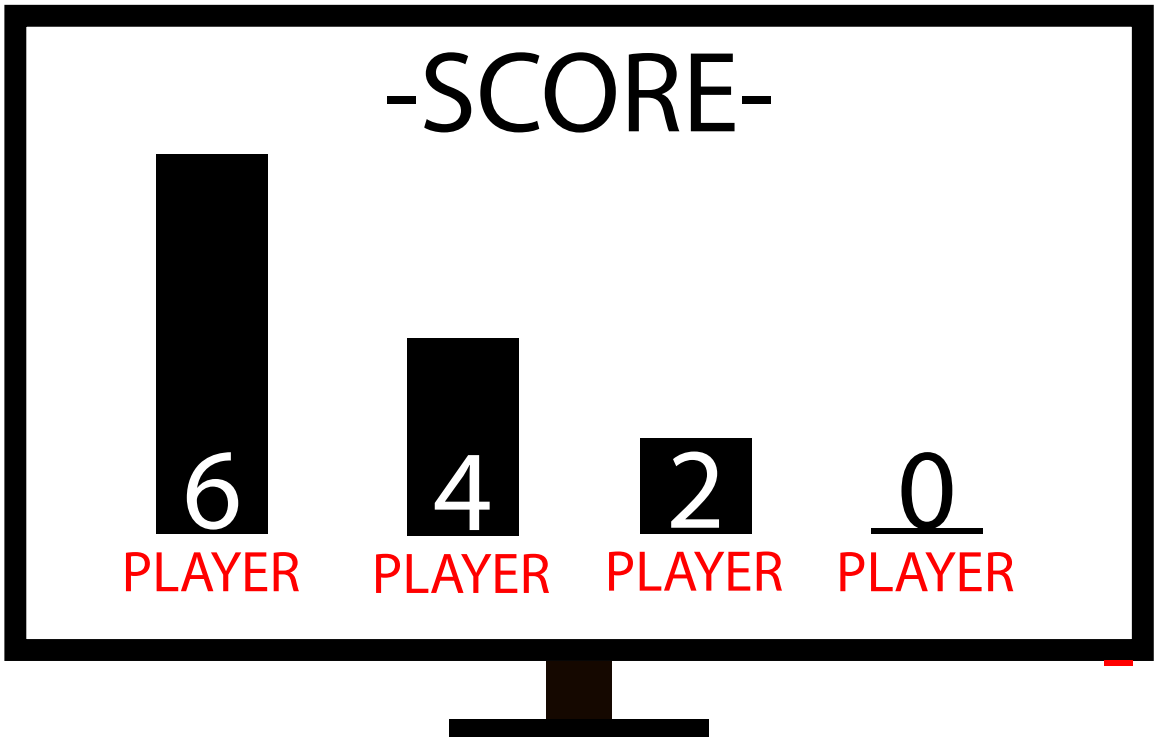
Obrázek 4.3 ukazuje rozložení místnosti, ve které hráči čekají, než začne hra. Střední část obrazovky je rozdělena na čtyři pozice pro hráče. Každá pozice má barvu, podle které se hráč identifikuje ve hře, tedy jeho postava bude mít v mini hrách příslušnou barvu. Pokud je pozice hráče prázdná má základní hodnotu. Po připojení se namísto základní hodnoty zobrazí jméno hráče. Tlačítko na start, které se nachází v pravé dolní části je dostupné pouze pro hráče, který místnost založil.



Obrázek 4.3: Návrh čekací místnosti.

Finální skóre

Obrazovka je rozdělena na čtyři části. Každá část zobrazuje hráče s hodnotou skóre, na které dosáhl. Hodnota skóre je také zobrazena graficky, a to výškou sloupce což lze vidět na obrázku 4.4.



Obrázek 4.4: Návrh místnosti zobrazující skóre.

4.2 Mini hra - Spiky Wheel

Spiky Wheel je 2D plošinová hra. Hráči v ní mají za úkol zůstat co nejdéle na živu. Mají možnost se pohybovat ze strany na stranu, skákat a také zastavit ostatní hráče. Ze stran na ně přijíždějí ostnatá kola. Ostnatých kol je více velikostí a druhů ale dělí se na dva hlavní typy.

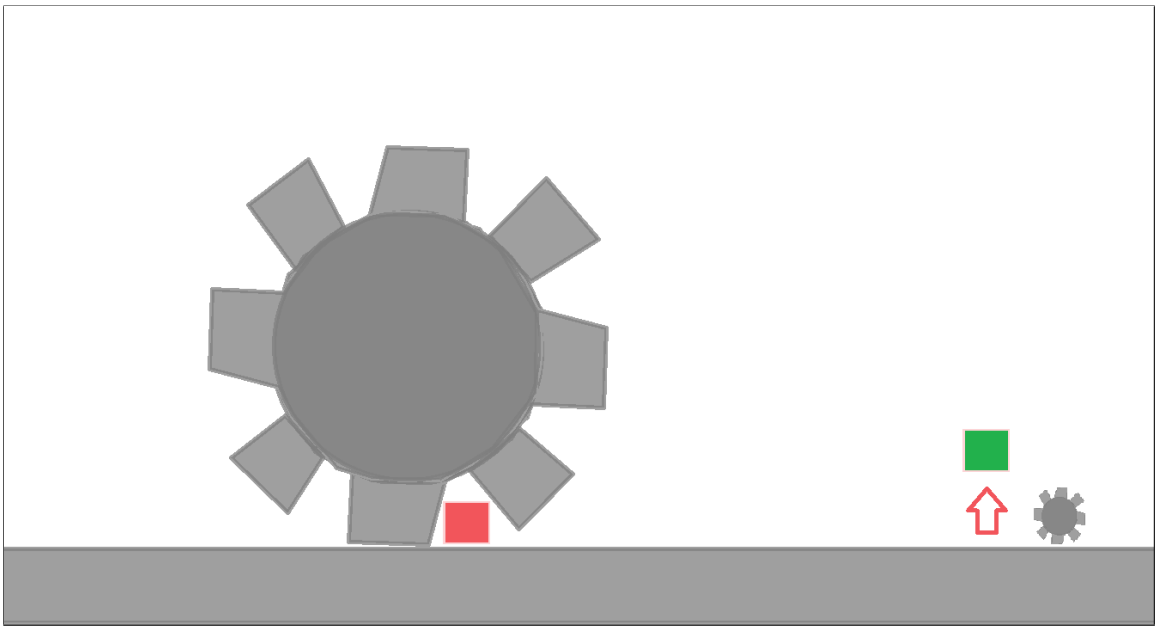
Hlavní typy ostnatých kol:

- Menší kola – jsou plná a pohybují se rychleji. Jdou přeskočit.
- Větší kola – ty mají v sobě prostor, do kterého se hráč může schovat a docílit toho, aby ho kolo nepřejelo. Kola jsou velká a z toho důvodu nejdou přeskočit, lze pozorovat na obrázku 4.5.

Pokud se hráč dotkne vystouplé části – ostnů, tak prohrál a je odstraněn ze hry. Kola přijíždí v předem připravených formacích. Náhodné generování není vhodné, protože by bylo obtížné docílit toho, aby byla hra zábavná a zároveň aby se nestalo to, že se hráči nemají, jak překážce vyhnout.

Předem připravené formace kol:

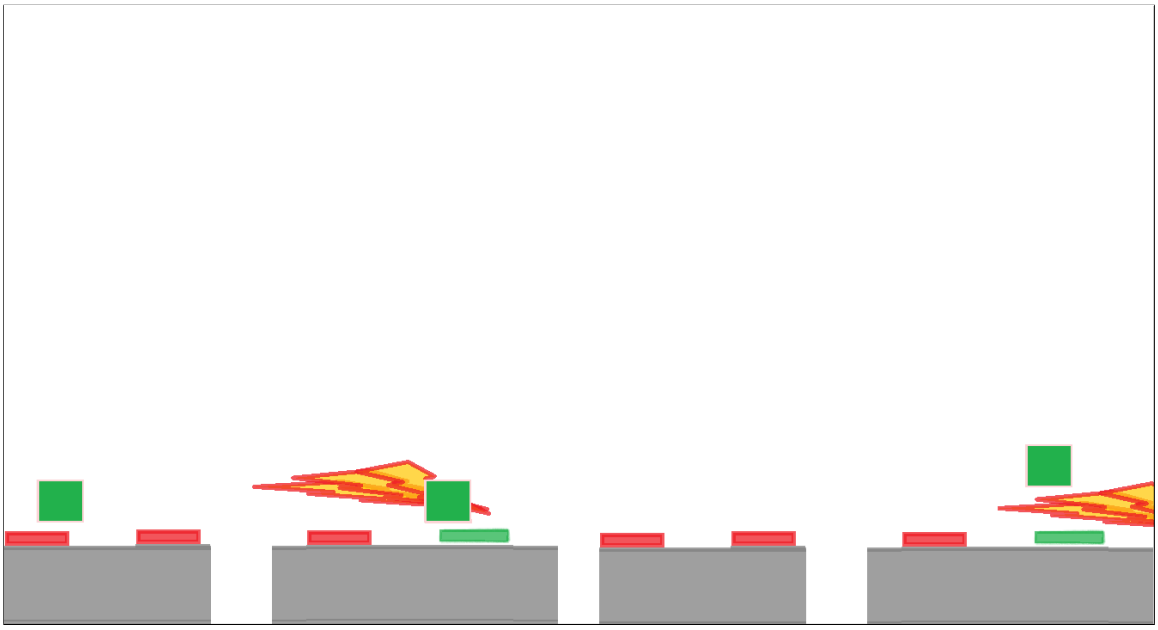
- Z každé strany přijede menší kolo.
- Z jedné strany přijede velké pomalé kolo, kterému trvá, než mapu projede. Z toho důvodu má kolo velký prostor, do kterého se lze schovat.
- Z jedné strany přijede velkou rychlostí menší kolo.
- Z jedné strany přijede velkou rychlostí větší kolo.



Obrázek 4.5: Návrh mini hry Spiky Wheel. Na obrázku můžeme pozorovat hráče červené barvy, který se schoval mezi výstupky ozubeného kola. Napravo od něj se nachází hráč zelené barvy, ten přeskakuje menší ozubené kolo.

4.3 Mini hra - Jump Rope

Každý hráč se nachází na své platformě. Platformy jsou od sebe odděleny a nelze se z jedné dostat na druhou. Na platformě jsou tlačítka – normální tlačítko a speciální. Po aktivování normálního tlačítka se objeví paprsek na platformě jiného hráče. Ten ho musí přeskočit což lze vidět na obrázku 4.6. Tlačítko se deaktivuje, dokud nebude připraveno být znovu použito. Aktivace speciálního tlačítka vytvoří paprsky na všech ostatních platformách, a to z jedné i z druhé strany. Doba načítání speciálního tlačítka je výrazně větší.



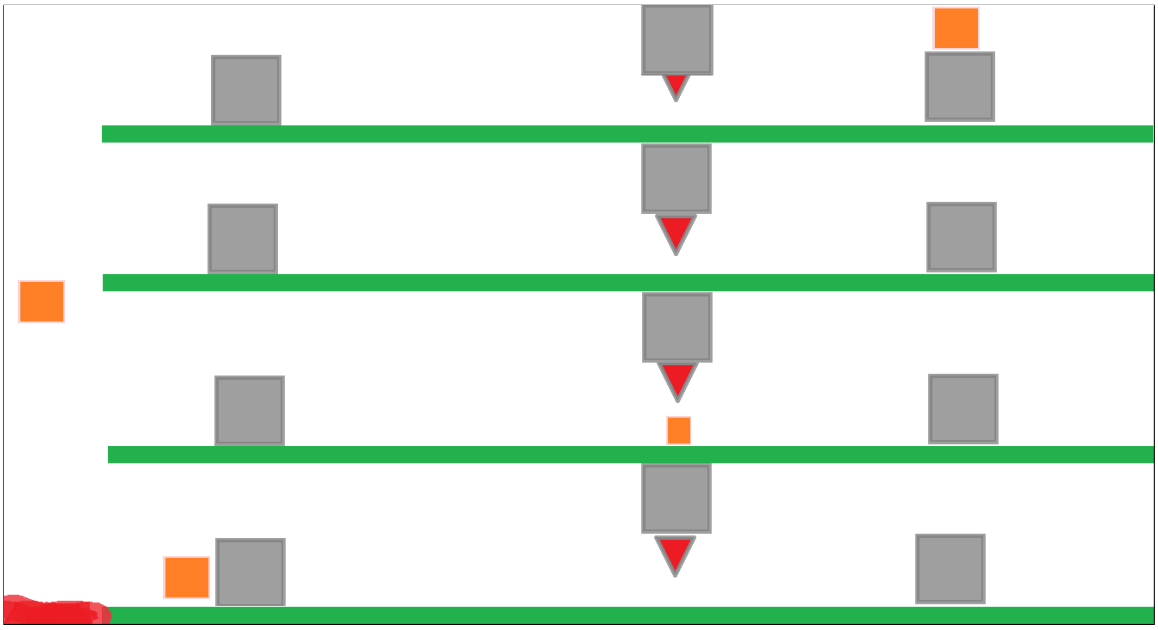
Obrázek 4.6: Návrh mini hry Jump Rope. Na obrázku lze pozorovat 4 samostatné plošiny. Na třech ze čtyř plošin se nachází hráčské postavy. Plošina v levé části, nemá aktivní žádné tlačítka (zobrazeno červenou barvou). Plošina napravo od ní zobrazuje hráče, který byl zasažen paprskem. Další plošina je prázdná, a to z toho důvodu, že hráč, který se na ní nacházel prohrál. Na poslední plošině je zobrazen hráč, který se vyhýbá paprsku pomocí skoku.

4.4 Mini hra - Death Jump

Každý hráč je na oddělené platformě. Má za úkol se vyhýbat objektům, které se přibližují z pravé části obrazovky. Hráč má k dispozici dvě akce – skok a smrštění. Vlastní vůlí se nemůže pohybovat po ose X. Objekty, co na hráče přijíždějí jsou dvojího typu:

- Box – hráč musí objekt přeskočit, pokud se mu to nepodaří, tak ho objekt posouvá po hrací ploše.
- Box s ostnem – hráč se musí smrštit, aby se vyhnul ostnu. Pokud se nevyhne, tak ho osten usmrtí.

Na levé straně hrací plochy se nachází propast s lávou (lze pozorovat na obrázku 4.7). Pokud do ní hráč spadne tak pro něho hra končí. Cílem hry je se vyhýbat překážkám a držet se co nejdál od propasti.



Obrázek 4.7: Návrh mini hry Death Jump. V levé části obrázku lze vidět prázdný prostor, kterým jeden z hráčů (hráči jsou zobrazeni oranžovou barvou) padá. Pod hráčem se nachází láva, které když se dotkne, tak prohraje. Zbytek úrovně je rozdělen na 4 stejné platformy (každá pro jednoho hráče). Na platformách lze vidět 2 typy překážek. První je zobrazena jako šedá kostka. Tu musí hráč přeskočit (lze vidět v pravé horní části obrázku). Další překážka je zobrazena jako kostka s červeným ostnem. K překonání této překážky se musí hráč zmenšit (lze vidět na druhé platformě od spodku).

4.5 Mini hra - Crazy Run

Je 2D topdown[2] hra typu Endless Runner[3]. Hráči jsou každý ve svých linkách. Pohybují se dopředu konstantní rychlostí. Rychlost pohybu či směr nejdou ovládat. Jediné, co má hráč pod kontrolou je skok a míření zbraní. Linky mají díry, do kterých lze spadnou a tím hra pro hráče skončí. Na linkách se objevují náboje do zbraně, tyhle náboje může hráč vzít. V jednu chvíli může mít pouze jeden náboj.

Použití těchto nábojů má následující efekty.

- Telebolt – hráč který je zasažen tímto nábojem si vymění pozice s hráčem, který náboj vystřelil.
- Blind Shell – hráč je na krátkou chvíli oslepen. Jeho obrazovka se na chvíli ztmaví.
- Ground Switch – pokud je použitý na prostor, kde není žádné políčko, tak políčko vytvoří. Pokud je naopak použit na prostor s políčkem, tak ho zničí.

Cílem mini hry je zbavit se ostatních hráčů. To lze docílit tak, že se hráčům nepodaří přeskočit díru v jejich linkách.

[2]Hra, ve které je pohled na hrací plochu shora dolů.

[3]Žánr hry, ve které hráč před něčím utíká a vyhýbá se přitom překážkám. Hra pokračuje až do doby, kdy se mu překážce nepodaří vyhnout.

4.6 Mini hra - Go On Red

Go On Red je 2D topdown mini hra. Hráči stojí vedle sebe na jedné straně úrovně. Na druhé straně se nachází semafor. Semafor v nepravidelných intervalech blikne červenou či zelenou barvou. Hráč má k dispozici pouze jednu akci. Může zmáčknout tlačítko. Pokud zmáčkne tlačítko, když je na semaforu červená barva, tak se posune o kousek dopředu. Pokud zmáčkne tlačítko, když je na semaforu zelená barva, tak se posune dozadu. V případě, že zmáčkne tlačítko, když je semafor „vypnutý“ (ve chvíli, když neblikne), tak se posune dozadu o větší kus.

Cílem mini hry je dostat se na druhý konec mapy jako první. Hra testuje reakce hráčů. Tím že se hráč má pohybovat, když svítí červená barva (oproti klasické zelené) se ho snaží zmást.

Kapitola 5

Implementace

Následující kapitola popisuje implementaci herního dema. Celá aplikace je implementována v jazyce C#. K vývoji byl použit herní engine Unity (verze 2020.3.28f1) a vývojové prostředí Rider[1]. V aplikace jsou použity grafické prvky od jiných autorů. Autoři jsou uvedeni v sekcích 5.6 a 5.4. V kapitole se dále popisuje struktura aplikace a adresářová struktura projektu.

5.1 Struktura scén

Herní demo se skládá z celkem čtyř scén. Scéna s názvem `Main` je vždy načtena jako první. K ní se načítají scény s jednotlivými mini hrami. V jeden moment můžou být načteny maximálně dvě scény. Z důvodů, že aplikace využívá návrhový vzor Singleton[2] a že k funkci aplikace je vyžadováno, aby se aplikace nejprve připojila k internetu, je nutné aplikaci spouštět přes scénu `Main`.

K tomuto účelu byl vytvořen skript `Bootstraper`, který má atribut `RuntimeInitialize`[3]. Tím se docílí to, že ať je aplikace v editoru spuštěna z jiné scény, než je scéna `Main`, tak se ta scéna zavře a načte se scéna `Main`. Tohle předchází chybám, které by nastaly při spuštění scény s mini hrou před tím, než jsou načteny potřebné věci.

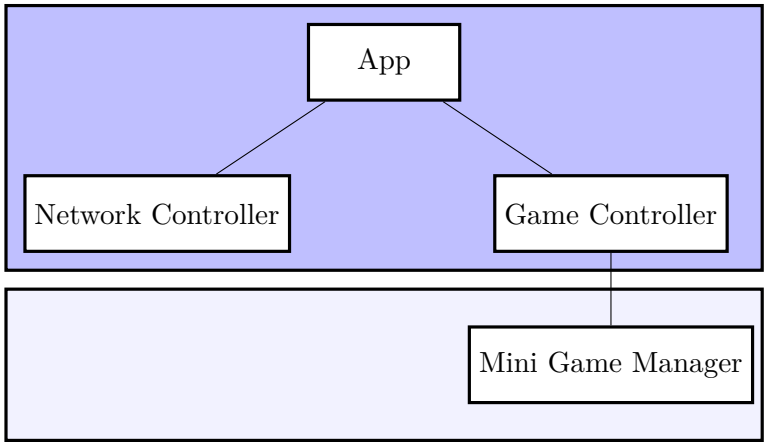
[1] https://www.jetbrains.com/rider/

[2] Návrhový vzor, který zaručuje, že třída má pouze jednu instanci [7].

[3] https://docs.unity3d.com/ScriptReference/RuntimeInitializeOnLoadMethodAttribute-ctor.html

V herním demu jsou scény dvojího typu:

- Hlavní scéna – Scéna `Main` se načítá jako první a uchovává v sobě komponenty, které jsou stěžejní pro korektní funkci aplikace. Ty jsou vidět na diagramu 5.1.
- Scéna s mini hrou – To jsou scény `Spiky Wheel`, `Jump Rope`, `Death Jump`. V každé scéně, ve které je mini hra se nachází komponenta typu `MiniGameManager`. Její přítomnost je nutná.



Obrázek 5.1: Diagram základní struktury aplikace. Komponenty v tmavě modré části se nachází ve hlavní scéně. Komponenta ve světle modré se vždy nachází ve scény s mini hrou.

Adresářová struktura

Obrázek 5.2 zobrazuje základní adresářovou strukturu Unity projektu. A objasňuje, co se v jednotlivých adresářích nachází.

`Assets`
- `3rdParty` ..... Složka obsahující balíčky třetích stran.
  - `Photon` ..... Balíček použitý k řešení multiplayeru.
  - `TextMeshPro` ..... Balíček Unity, k vytváření textových polí.
- `Core` ..... Obsahuje pomocné skripty, které používám v ostatních projektech.
- `Game` ..... Obsahuje skripty a grafické prvky z hry.
- `Scenes` ..... Obsahuje všechny scény v projektu.
- `Resources` ..... Obsahuje objekty, které se vytvářejí za běhu projektu.

Obrázek 5.2: Obrázek zobrazuje základní adresářovou strukturu projektu.

5.2 Síťové řešení

Při spuštěni aplikace se ve skriptu `Network Controller` zavolá metoda `Connect`. V této metodě se nastaví aktuální verze hry a také jméno hráče. Poté se pokouší připojit na server. Pokud je připojení úspěšné, tak skript obdrží od Photon serveru callback (zpětné volání). Uživatel je o úspěšném připojení informován pomoci pop up zprávy (vysvětleno v sekci 5.3), a pak připojen do Lobby. Dále má skript na starost vytváření místnosti. Místnosti je přiřazen

náhodný kód. Ten je vygenerován pomocí pomocné třídy `Helpers`. Délka kódu je vždy čtyři znaky a skládá se ze znaků anglické abecedy. Skript také připojuje hráče do místnosti. Pokud je zadán kód místnosti, tak ho použije. Pokud žádný zadán není, tak se připojí do náhodné místnosti. `Network Controller` dědí od třídy `MonoBehaviourPunCallbacks`. Díky tomu obdrží callback, pokud nastane nějaká z následujících situací.

Zpětná volání, které může `Network Controller` obdržet:

- OnConnectedToMaster – vyvoláno, když se uživatel připojí k serveru.
- OnJoinedLobby – vyvoláno, když se uživatel připojí do lobby.
- OnDisconnected – vyvoláno, když se uživatel odpojí od serveru.
- OnMasterClientSwitched – vyvoláno, když je uživatel ve hře a odpojí se `Master Client`.
- OnCreatedRoom – vyvoláno, když se úspěšně vytvoří místnost.
- OnCreateRoomFailed – vyvoláno, když se místnost vytvořit nepodaří. Může nastat například v případě, že je již vyčerpána kapacita místností.
- OnJoinedRoom – vyvoláno, když se uživatel připojí do místnosti.
- OnJoinRandomFailed – vyvoláno, když se nepodaří připojit k náhodné místnosti.
- OnLeftRoom – vyvoláno, když se nepodaří připojit ke konkrétní místnosti.
- OnPlayerEnteredRoom – vyvoláno, když se jiný uživatel připojí do místnosti, ve které se nacházíme.
- OnPlayerLeftRoom – vyvoláno, když se jiný uživatel odpojí z místnosti, ve které se nacházíme.

5.3 GUI

V herním demu se nachází celkem devět různých grafických rozhraní. Tyhle rozhraní jsou ve scéně `Main` jako potomci objektu `Menu`. Finální implementace rozhraní se částečně odlišuje od jeho prvotního návrhu zobrazeného v kapitole 4.1. Hlavním rozdílem je vzhled tlačítek, ty mají v původním návrhu výrazné červené pozadí a bílý text. Aktuální implementace tlačítek ale nemá žádné pozadí a má červený text. V rámci průběžného testování byl vzhled změněn na základě zpětné vazby jednoho z uživatelů.

Objekt `Menu` má na sobě následující Unity komponenty:

- Rect Transform
- Canvas
- Canvas Scaler

Poté je na objektu komponenta `Canvas Manager`, která se stará o přepínaní mezi jednotlivými obrazovkami. Aby bylo možné obrazovku přepnout, tak na sobě musí mít komponentu `Canvas Controller`. Ta určuje typ obrazovky.

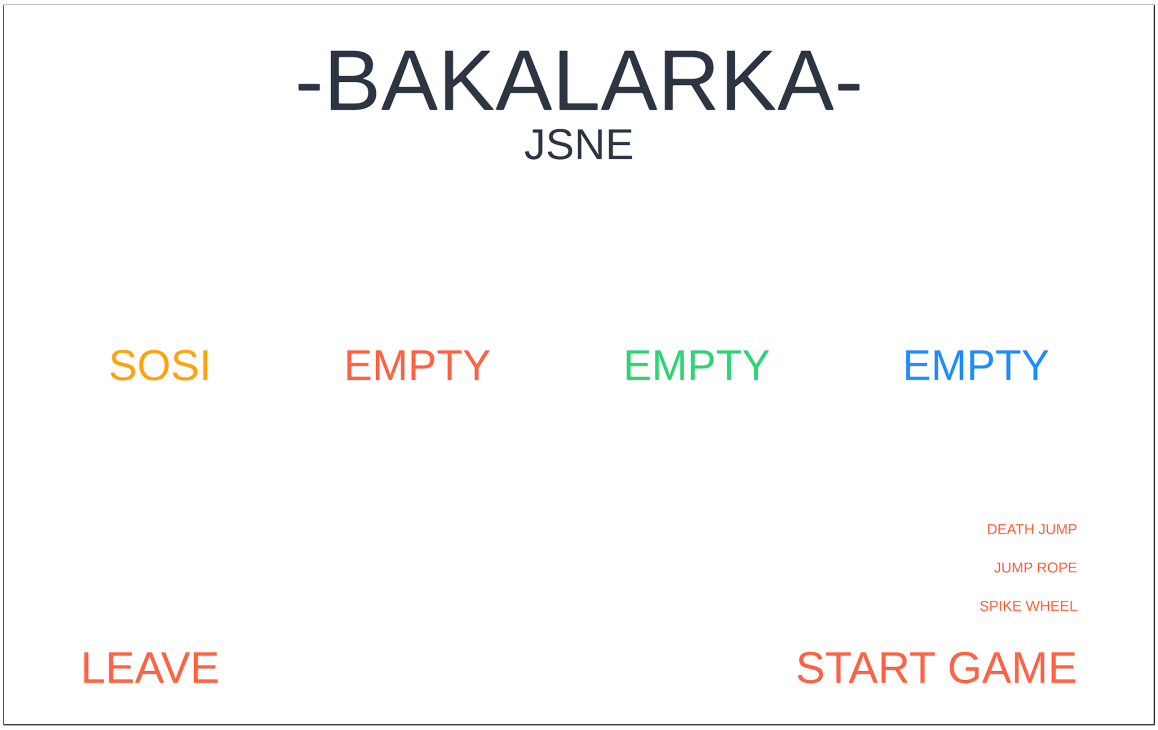
Veškeré typy obrazovek v herním demu:

- Splash Screen – Obrazovka, která je zobrazena před tím, než se aplikace připojí k Photon serveru.
- Menu – Obrazovka, která je zobrazena, jakmile se aplikace připojí k Photon serveru. Z obrazovky je možné přejít na obrazovku další po stisknutí jakékoliv klávesy na klávesnici (kromě klávesy Esc), k tomuto kroku je uživatel vyzván textem na obrazovce.
- Join Menu – Obrazovka, ze které může uživatel založit hru, či se připojit do již založené. Zde si uživatel taktéž může změnit jméno. Při prvním spuštění hry se uživateli přidělí náhodné jméno, a to ve tvaru `Player_XY`, kde „XY“ je číslo od 0 od 99. Jméno je uloženo na lokálním zařízení pomocí třídy `Player Prefs`[4]. Při zadání jména, které není validní nebo při nezadání žádného jména se použije jméno, které bylo zadáno jako poslední. Dále je zde pole, do kterého lze zadat kód místnosti a připojit se tak do konkrétní místnosti. Obrazovka je zobrazena na obrázku 5.3.
- Room – Obrazovka, která se uživateli zobrazí při založení hry nebo po připojení se do hry. Obrazovka je popsána na obrázku 5.4.
- Loading – Obrazovka, která se ukáže po odstartování hry.
- Game Info – Obrazovka, která se ukáže před startem každé mini hry. Informuje hráče o tom, jaká mini hra následuje, jaký je její princip a jak se ovládá.
- Mini Game – Obrazovka, která je zobrazena při každé mini hře. Obsahuje časomíru.
- Score – Obrazovka, která se ukáže po dohrání poslední mini hry.

Komponenta Canvas Manager přepíná obrazovky, když je vyvolána akce, která vyžaduje zobrazení konkrétní obrazovky. V jeden moment může být aktivní pouze jedna obrazovka, kromě obrazovky `Exit Menu`, která vždy aktuální obrazovku překryje. Obrazovka `Exit Menu`, se zobrazí při stisknutí klávesi `Esc` a při jejím opakovaném stisknutí se zavře.

[4] https://docs.unity3d.com/ScriptReference/PlayerPrefs.html

Obrázek 5.3: Vzhled obrazovky `Join Menu`. Ve vrchní části můžeme vidět textové okno, kde lze změnit jméno uživatele. Pod ním se nachází tlačítko, které slouží k připojení do místnosti. Pokud je zadán kód místnosti, tak se pokusí připojit do místnosti s tímto kódem. Pokud je okno pro kód prázdné, tak se připojí k náhodné místnosti, která je dostupná.



Obrázek 5.4: Vzhled obrazovky `Room`. Ve vrchní části můžeme vidět název projektu a pod ním kód místnosti, pomocí kterého se lze do místnosti napojit. Ve střední části jsou místa pro hráče, kde je jedno místo zaplněno. Pod nimi pak tlačítka na opuštění místnosti a start hry.

Pop up message

`PopUpMessageController` je třída, která na obrazovce vytváří zprávy, které po určitém časovém úseku zmizí. V návrhu nebylo s nějakou podobnou funkcionalitou počítáno. Nápad ji vytvořit přišel až z nutnosti informovat uživatele například o tom, že zadal jméno, které není validní nebo že se snaží připojit do místnosti, i když žádná místnost není vytvořena. Vytvoření zprávy je umožněno přes metodu `InfoPopUp` a `GameInfoPopUp`. První se používá k informováni uživatele v menu a druhá ve hře. Při zavolání jedné z metod se zpráva přidá do seznamu. Zprávy ze seznamu se uživateli postupně zobrazují. Zpráva, která je uživateli zobrazena, se po chvíli začne zmenšovat a poté zanikne. Zmenšování zprávy je implementováno pomocí `Animation Controller`.

5.4 Herní postava

Následující kapitola popisuje herní postavy hráče, které ovládá v mini hrách. Jsou celkem 3 druhy postav a to `SpikeWheelPlayer`, `JumpRopePlayer` a `DeathJumpPlayer`.

Všechny postavy mají na sobě následující Photon komponenty:

- Photon View – identifikuje objekt hráče na síti.
- Photon Transform View – synchronizuje pozici hráče přes síť.
- Photon Animator View – synchronizuje animace hráče přes síť.

A také Unity komponenty:

- Rigidbody 2D – dělá z hráče objekt, ovlivnitelný fyzikou.
- Box Collider – vytváří kolizní box hráče.
- Sprite Renderer – renderuje vzhled hráče do scény.
- Animator – animuje postavu hráče.
- TextMeshPro – slouží k zobrazení jména hráče.

Interakce hráče s prostředím mini her je umožněna pomocí jeho postavy. Ne ve všech mini hrách má hráč k dispozici všechny akce. Záleží, jestli má na sobě postava komponentu, která konkrétní akci povoluje.

Komponenta Input

Komponenta `Input` se stará o přihlašování a odhlašování akcí z Unity Input Systému[5]. V momentu, kdy `MiniGameManager` vyvolá akci `OnMiniGameStartAction`, se všechny vstupy povolí, a hráč se může začít dělat příslušné akce (pohyb, skok a další).

Komponenta Ground

Komponenta `Ground`, kontroluje, zda je postava hráče na zemi anebo je ve vzduchu. Kontrola proběhne, pokud nastane kolize. Průběh kontroly je popsán následujícím pseudokódem 1.

Algorithm 1 Pseudokód pro zjištění kolize se zemí. Pokud nastane kolize, tak `Rigidbody2D` zjistí všechny body, kde se dva objekty dotýkají. Souřadnice Y normály z bodů jsou pak porovnány s „magickou konstantou“ `0.99`. Pokud je souřadnice Y větší nebo než magická konstanta, tak nastala kolize se zemí. Magická konstanta má hodnotu `0.99`, aby se jako země detekovala i ta plocha, které není zcela horizontální.

```
1: if nastal kolize then
2:     for bod v kolizi do
3:         if Y souřadnice normály bodu >= 0.99 then return true
4:         end if
5:     end for
6: end if
```

[5] https://docs.unity3d.com/Packages/com.unity.inputsystem@1.0/manual/

Komponenta Sprite

Komponenta `Sprite` zařizuje to, aby měly postavy barvu a jméno hráče ke kterému patří. To zajistí tak, že při startu mini hry zjistí pořadí hráče v místnosti a podle něj mu nastaví barvu, kterou získá z nastavení hry.

Komponenta Move

Je komponenta, ve které se implementuje pohyb hráče. V `Update` metodě se nastavuje proměna `desiredVelocity`. Ta uvádí, jakým směrem a jak rychle se má postava pohnout. Pokud není aplikován žádný vstup na pohyb hráče, tak je tahle proměnná 0. V metodě `FixedUpdate` se pak tahle rychlost aplikuje na `Rigidbody2D` a to pouze na ose X. V případe, že je `desiredVelocity` nula se `Rigidbody2D` přestane pohybovat.

Komponenta Jump

Komponenta, která má na starost kontrolu skoku a pádu. Využívá při tom několik technik, které zlepšují požitek ze skoku. Jednou z nich je takzvaný `Jump Buffering`. Pokud je vyvolána akce skoku. Tak se nastaví float hodnota `lastJumpPressedTime` na aktuální čas. K ní se přičte konstanta `jumpBufferTime`. Pokud je součet těchto proměnných menší nebo roven aktuálnímu času, tak je povolen skok a v momentu, kdy se hráčova postava dotkne země se skok vykoná. Tímhle docílíme toho, že když uživatel zmáčkne tlačítko pro skok těsně před tím, než dopadne na zem, tak se skok vykoná. Bez téhle techniky by postava působila neresponsivně . Další technikou je změna `Gravity Scale`[6]. Při skoku postavy se hodnota `Gravity Scale` nastaví na menší než, je při pádu postavy. Tím se docílí toho, že postava „nepoletuje ve vzduchu“ a působí dojmem, že má nějakou váhu. Zároveň, pokud je při skoku drženo tlačítko pro skok, je hodnota `Gravity Scale` nastavena na hodnoto při pádu. Tímhle hráč dokáže ovlivnit výšku skoku.

Komponenta Duck

Komponenta `Duck` zařizuje to, aby se postava hráč zmenšila, pokud je levé tlačítko myši drženo, a zvětšila, jakmile bude opět puštěno. Tahle mechanika je implementována pomocí zmenšení měřítka game objektu hráče.

Komponenta Death

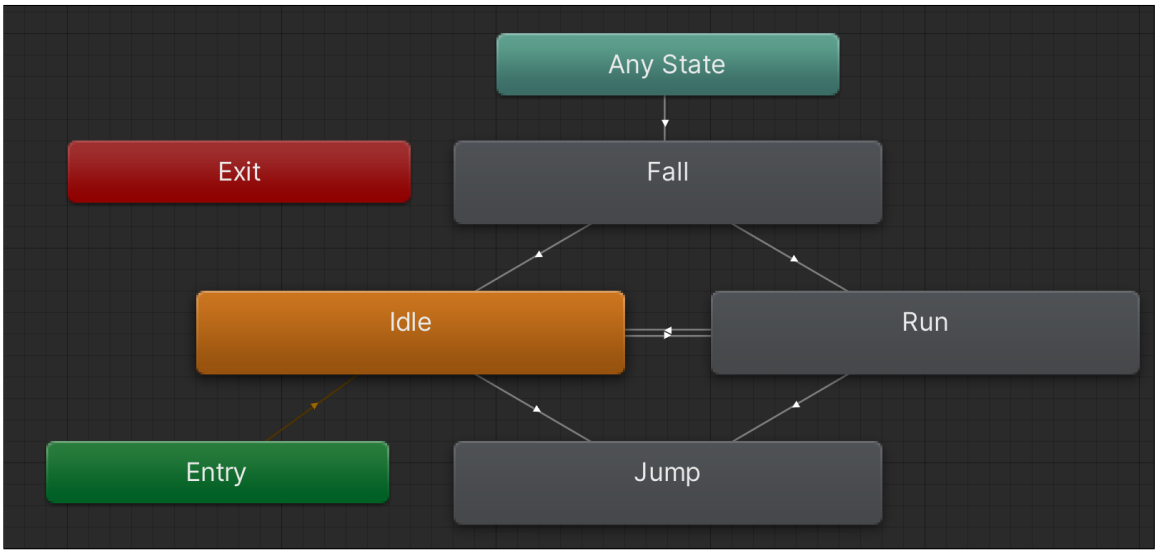
Komponenta `Death`, kontroluje, zda má hráčova postava kolizi s objektem, který ji může zabít. Tahle kontrola probíhá pouze u objektů u Master klienta. Pokud taková kolize nastane, je pomocí RPC zavolání funkce `RPC_PlayerDead`, která vypne kolizní box hráče. Vypnutí kolizního boxu způsobí že hráč propadne hrací plochou, čímž se znázorňuje že hráč prohrál. Dále je hráči zamezeno v ovládání postavy. Poté je o smrti hráče informován `MiniGameManager`.

Animace

Animační klipy hráče jsou převzaty z volně dostupného assetu [1]. Animace hráče má celkem čtyři stavy (viditelné na obrázku 5.5), ve kterých se může nacházet. Přechody mezi těmito

[6] Jedna z atributy Rigidbody2D, která určuje, jak moc je Rigidbody2D ovlivňováno gravitací. Pokud je hodnota 1 gravitace je stejná. Pokud je hodnota 2, tak na objekt působí 2x větší gravitace.

stavy, se uskuteční, pokud je splněna příslušná podmínka. Vstupním stavem animátoru je stav `Idle`. V tomhle stavu hráč setrvává, pokud se nepohybuje v ose X ani v ose Y. Pokud bude hráč padat (hodnota rychlosti na ose Y je menší než 0), tak se z jakéhokoliv stavu dostane do stavu `Fall`. Z tohoto stavu se dostane až po dopadu na zem. Poté přejde do stavu `Idle` či `Run`, podle toho, jestli drží tlačítko pro pohyb. Posledním stavem je stav `Jump`, do tohoto stavu se dostane, pokud je jeho rychlost na ose Y větší než 0.



Obrázek 5.5: Pohled na `Animation Controller`. Zobrazuje všechny stavy, ve kterých se může animace nacházet a také přechody mezi nimi.

5.5 Game Manager

Úkolem třídy `Game Manager`, je kontrola aplikace poté, co se uživatel připojí do místnosti. Skript startuje sérii mini her, a to buď přes metodu`StartGame` nebo `StartGameForced`. První z nich spustí tři kola náhodně vybraných mini her. Druhá spustí tři kola jedné a té samé mini hry. Mini hru může spustit pouze uživatel, který hru založil a hru lze spustit jen pokud je počet hráčů v místnosti větší nebo roven dvěma. Pokud je tahle podmínka splněna, tak se pošle všem hráčům `RPC` zpráva `RPC_StartMiniGamesRounds`. Ta u každého uživatele přepne obrazovku na obrazovku načítaní. Poté `MasterClient` všem pošle další RPC zprávu a to `RPC_LoadMiniGame` s informací o tom, kterou mini hru mají načíst. To má za následek to, že se u každého uživatele spustí Coroutine `LoadMiniGame`. Ta má na starost odstranit současnou scénu s mini hrou (pokud nějaká je) a načíst scénu novou. K načítaní scén u všech hráčů existuje ve Photon metoda. V tomhle projektu však nemohla být použita. Jelikož metoda `PhotonNetwork.LoadLevel` neumožňuje načtenou scénu přidat k aktuálně načteným scénám. A jelikož se v projektu využívá současné načtení vícero scén, tak tahle metoda nemohla být použita a byla použita vlastní implementace.

Pokud se v průběhu mini hry některý z hráčů odpojí, tak třída automaticky hru ukončí a resetuje vnitřní stav aplikace na stav před zapnutím mini hry. Jestliže hráči sérii mini her dohrají, tak skript zobrazí hráčům jejich skóre a chvíli poté je odpojí ze hry.

5.6 Mini hry

Při načtení scény s mini hrou, se uživateli zobrazí obrazovka, která ho informuje o tom, jak se mini hra ovládá. Zároveň je hráč vyzván, aby stiskl klávesu `mezerník`, pokud je připraven. Jakmile jsou všichni hráči připraveni, tak se zobrazí scéna s mini hrou a spustí se odpočet do startu hry. Po skončení odpočtu hra začne a uživateli je zpřístupněno ovládání. Každá mini hra končí buď tím, že vyprší čas anebo v okamžiku co je ve hře jeden hráč. Za výhru v mini hře (výhrou se myslí to, že hráč zůstane nejdéle na živu) dostane hráč bod. Po konci mini her se hráčům, zobrazí graf s počty získaných bodů. Grafické prvky, které byly ve hrách použity jsou z volně dostupného balíčku[7] vydaného pod MIT licencí (licence je obsažena v projektu).

V každé mini hře je komponenta, která dědí od abstraktní třídy `MiniGameManager`. V téhle třídě je implementována funkcionalita, která je společná pro všechny mini hry. Při načtení scény s mini hrou vyvolá `GameController` akci `OnMiniGameSceneLoadedAction`. K téhle akci je přihlášena funkce `SetUpGame`. Ta vytvoří instance hrací postavy ve scéně. Jakmile je vyvolána akce `OnAllPlayersReadyAction`, tak se spustí Coroutine s názvem `CountDown`. Ta spustí odpočet, který trvá 3 sekundy a poté vyvolá akci `OnMiniGameStartAction`, která informuje o startu hry a zavolá metodu `GameLoop`. Tahle metoda je v každé třídě, která dědí od třídy `MiniGameManager` jinak implementována a stará se o funkcionalitu mini hry.

Třída dále kontroluje, kolik hráčů je aktuálně naživu a v případě že již zbývá jen jeden hráč, tak zobrazí jeho jméno a že se stal vítězem. Poté vyvolá akci `OnMiniGameEndAction`.

Spiky Wheel

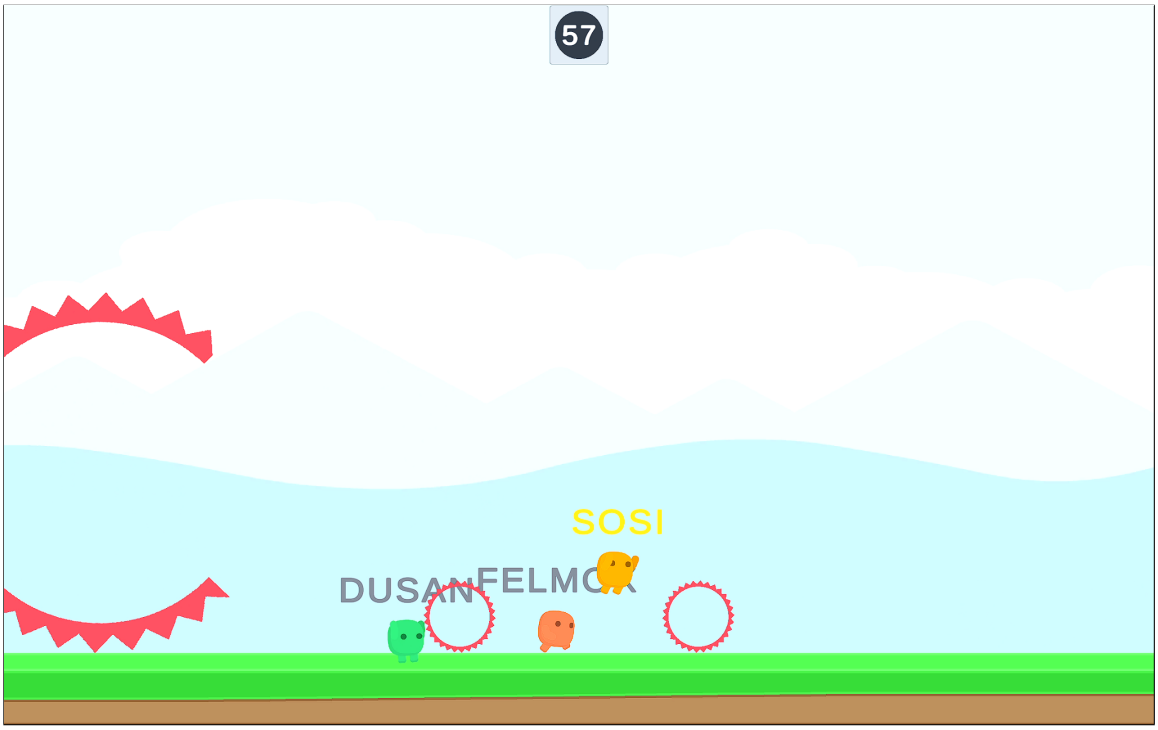
V mini hře `Spiky Wheel`, je hráč při vyvolání akce `OnMiniGameSceneLoadedAction` instanciován na spawn point[8]. Po vyvolání akce `OnMiniGameStartAction`, je spuštěna herní smyčka mini hry. Ta skončí, pokud vyprší čas nebo ve hře zůstane jeden hráč. Smyčka je implementovaná pomocí `Coroutine` (pojem Coroutine je vysvětlen v sekci 3.2). Její vnitřní implementaci popisuje algoritmus 2. Ten na hrací plochu přidává „ozubená kola“. Díky tomu, že je herní smyčka implementována pomocí `Coroutine`. Na obrázku 5.6 lze vidět implementovanou mini hru Spiky Wheel.

Algorithm 2 Herní smyčka mini hry Spiky Wheel

```
1: if jsem master then
2:     while zbývající herní čas je >= 0 do
           vyber náhodný set kol
           přidej ho na herní plochu
           počkej určitý čas
3:     end while
4: end if
```

[7] https://bayat.itch.io/platform-game-assets/

[8] Anglický výraz pro místo, na kterém se hráč objeví při zrození.

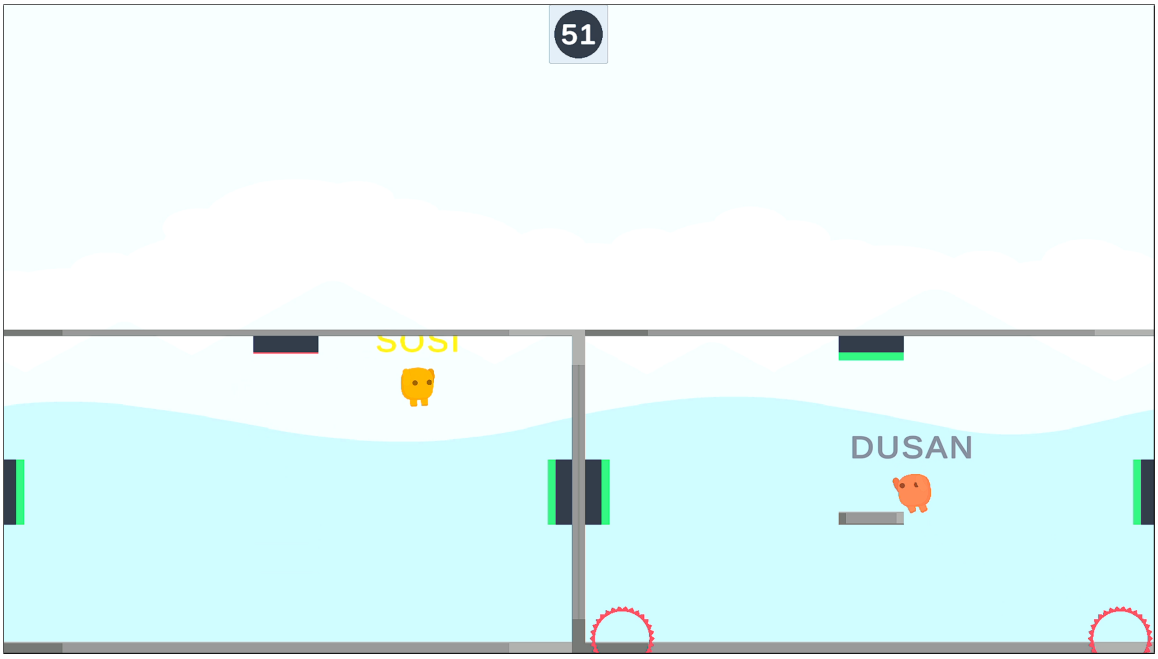


Obrázek 5.6: Pohled na mini hru `Spiky Wheel`. V levé části obrázku lze vidět ozubené kolo, které se přibližuje k hráčům ve středu obrázku. Zelený hráč byl zasažen menším ozubeným kolem, a tak padá ven z mapy. Oranžový hráč se snaží uniknout kolu po jeho levé straně. Žlutý hráč právě přeskočil kolo, které se nachází pod ním. Ve vrchní části obrázku je časomíra, která zobrazuje zbývající čas.

Jump Rope

V mini hře `Jump Rope` je každý hráč umístěn na svou uzavřenou platformu. Počet platforem se odvíjí od počtu hráčů. Tlačítka, která jsou na platformě, jsou ze začátku neaktivní. Chování tlačítka má na starost skript `Pressure Plate`. Tlačítka se aktivují po uběhnutí „doby nabíjení“. To je znázorněno změnou barvy komponenty SpriteRenderer a zasunutím tlačítka. Pokud hráč vstoupí na tlačítko, tak se tlačítko opět deaktivuje. Vstup hráče na tlačítko je implementován v metodě `OnTriggerEnter2D`[9]. Aktivací tlačítka je vyvolána akce `OnActivatedAction`. K akci je přihlášen skript `Platform Controller`, který se nachází na každé plaformě. Jakmile je akce vyvolána, tak skript zjistí aktuálně dostupná místa (místa na platformách na kterých ještě je hráč). A poté na tomhle místě vytvoří malé ozubené kolo. Tohle kolo se po krátké chvíli rozjede. Kolo se zničí, jakmile dorazí na druhý konec platformy (na každém konci platformy je umístěná oblast s Triggerem, která tyto kola zničí). V prostředí části platformy se nachází blok, který je potřebné k tomu, aby se hráč mohl dostat ke speciálnímu tlačítku. Implementace chování bloku je ve skriptu `Help Block`. Tohle políčko je „nestabilní“ a poté co se ho hráč dotkne spadne (implementováno tím, že se stane ovlivnitelné gravitací) a také mu je vypnut Collider 3.2 (aby mohlo propadnout a nemít kolizi s hráčem či platformou). Speciální tlačítko má „doby nabíjení“ delší než tlačítka normální. A při jeho aktivaci se zjistí všechny ostatní dostupné místa pro vytvoření kola

a na všech místech se kola vytvoří. Obrázek 5.7 ukazuje implementovanou mini hru Jump Rope.



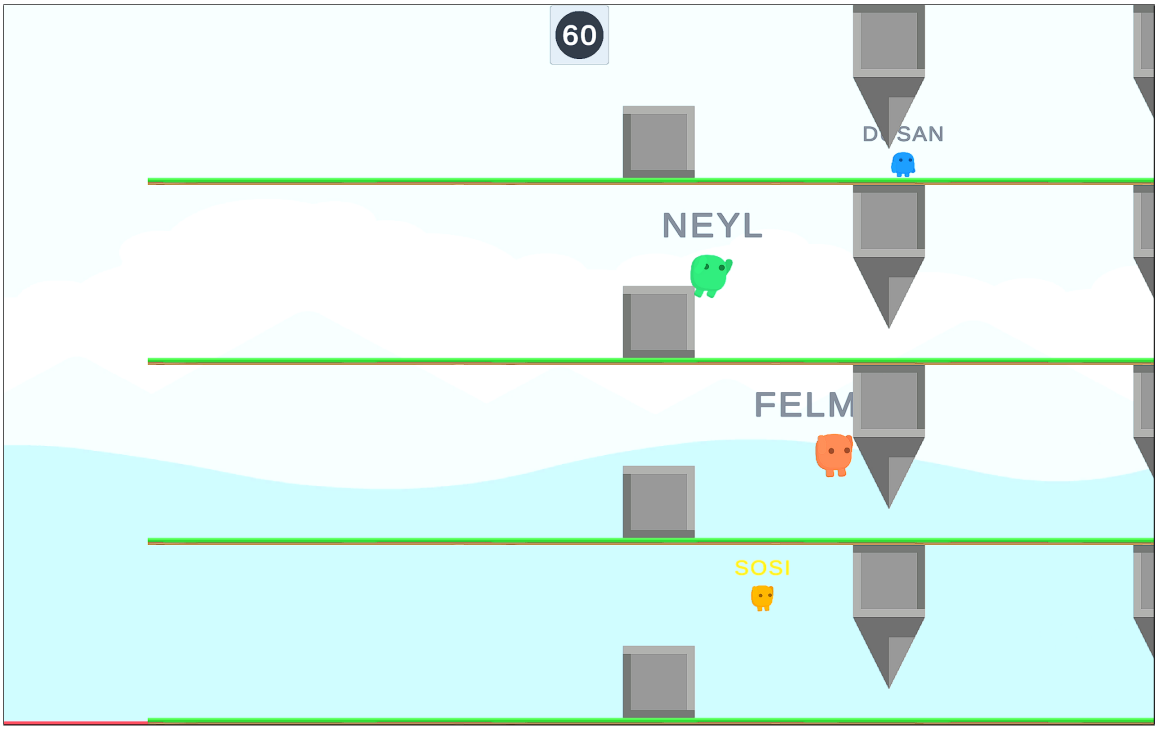
Obrázek 5.7: Pohled na mini hru `Jump Rope`. Hra probíhá pouze ve dvou hráčích (lze poznat podle absence vrchní části mapy). Na obrázku lze vidět žlutého hráče, který právě aktivoval prostřední speciální tlačítko. Dále lze vidět červeného hráče při skoku na plošinu. Pod ním se nachází ozubená kola, která se objevila poté, co žlutý hráč aktivoval speciální tlačítko.

Death Jump

V mini hře Death Jump mají hráči k dispozici dvě akce. Akci skoku a akci zmenšení. Hráčům je zamezen pohyb na ose X (nemají na sobě komponentu Move). Při startu hry jsou hráči umístěny na pravou část herní plochy. Po startu se taktéž začnou generovat překážky. Překážky se generují u každého hráče stejně (aby nebyl některý z hráčů v nevýhodě). Překážky jsou implementovány ve skriptu `Obstacle`. Pohybují se směrem k hráčům, a to konstantní rychlostí, jejich `Rigidbody2D` je nastaveno na kinematický mód. Jednou z překážek je blok. Jeho umístění je vždy na dolní části platformy. Díky tomu že je `Rigidbody2D` překážky v kinematickém mód posunuje překážka hráče. A hráč překážku nijak neovlivňuje. Další překážka je vždy umístěna na vrchní části platformy. Její velikost je nastavena tak, že hráč musí zmenšit svou velikost, aby se jí vyhnul. Pokud se hráči nepodaří překážce vyhnout. Nastane kolize a hráčova postava je zničena. Zmenšení hráče je implementováno pomocí změny měřítka objektu konstantou `duckScaleFactor`. V zadní části mapy je objekt, který má na sobě komponentu `BoxCollider2D` s atributem `IsTrigger` nastaveným na `True`. V momentu, kdy překážka vstoupí do této zóny, je mód její `Rigidbody2D` nastaven na `Dynamic`, což způsobí to, že překážka začne padat. Hráč není touhle zónou nijak ovlivněn. Pokud je hráč kostkou posunut do zadní části úrovně, kde již není nic na čem by mohl stát, tak začne padat do „lávy“. Ta má na sobě taky komponentu `BoxCollider2D` s atributem

[9] https://docs.unity3d.com/ScriptReference/MonoBehaviour.OnTriggerEnter2D.html

`IsTrigger`. Po pádu do lávy je hráč usmrcen. Obrázek 5.8 ukazuje implementovanou mini hru Death Jump.



Obrázek 5.8: Pohled na mini hru `Death Jump`. Z obrázku můžeme pozorovat, že modrý hráč je aktuálně ve vedení (nachází se nejdál od lávy). Dále můžeme vidět, že červený hráč narazil do ostnu a je tak ze hry vyřazen. V dolní části obrázku lze vidět žlutého hráče, který využil zmenšení, aby se pokusil vyhnout přibližující se překážce.

Kapitola 6

Testování

V následující kapitole je popsáno, jak probíhalo testování aplikace. A to uživatelské testování a klasické testování funkčnosti aplikace – debugging.

Testování aplikace probíhalo při celém procesu vývoje aplikace. Aplikace se během svého vývoje dost změnila. Velkou změnou prošla její grafická stránka ale i její vnitřní fungování. Když přišla řada na implementaci druhé mini hry, tak vyplynulo na povrch, že se musí předělat celé načítaní mini her, protože nebylo uzpůsobeno pro načtení mini her, které mají jinou vnitřní logiku. Z toho důvodu byla logika všech mini her sloučena do jednoho skriptu a to `MiniGameManager`. Změnou si prošel také skript na ovládání hráče. Původně bylo v plánu vytvořit jeden velký skript, který by se o všechno staral a šel by použít i ve 3D prostředí. V průběhu jeho tvorby jsem ale zjistil, že by to byl „kanón na vrabce“. Z toho důvodu je chování hráče implementováno pomocí více skriptů, které se starají o jednu konkrétní věc.

6.1 Uživatelské testování

Účastnici uživatelského testování byli před samotným začátek testováním požádání o vyplnění obecného dotazníku, který zkoumal jejich zkušenosti s hraním počítačových her. Výsledky lze vyčíst z grafů 6.1. Testování se zúčastnili celkem 4 lidé. Od každé mini hry bylo odehráno devět kol. Po každé odehrané mini hře, byli respondenti vyzváni k vyplnění částí dotazníku zabývající mini hrou, kterou právě hráli. Jako první byla otestována mini hra `Spike Wheel`, poté mini hra `Jump Rope` a nakonec mini hra `Death Jump`. Kompletní výsledky testování (13 stran) jsou přiloženy na paměťovém médiu.

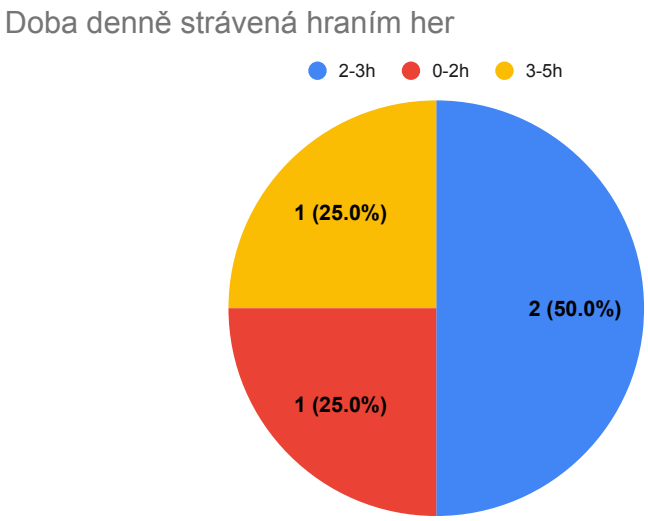
Kromě numerického hodnocení mini her, hodnotili respondenti i slovně. A to na následující obecné dotazy:

- Co by se dalo v aplikaci vylepšit?
- Čím na Vás aplikace nejvíce zapůsobila?
- Objevili se při testování aplikace nějaké problémy?
- Chcete něco dodat? (finální otázka na konci formuláře)

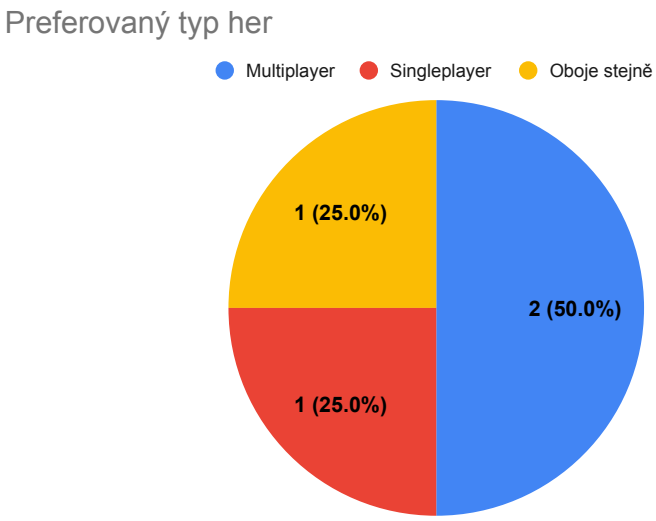
A také na dotazy k jednotlivým mini hrám:

- Co byste na mini hře vyzdvihli?
- Co byste mini hře vytkli?

Po otestovaní všech mini her je respondenti seřadili od nejlepší po nejhorší.



(a) Graf zobrazující dobu denně strávenou hraním her

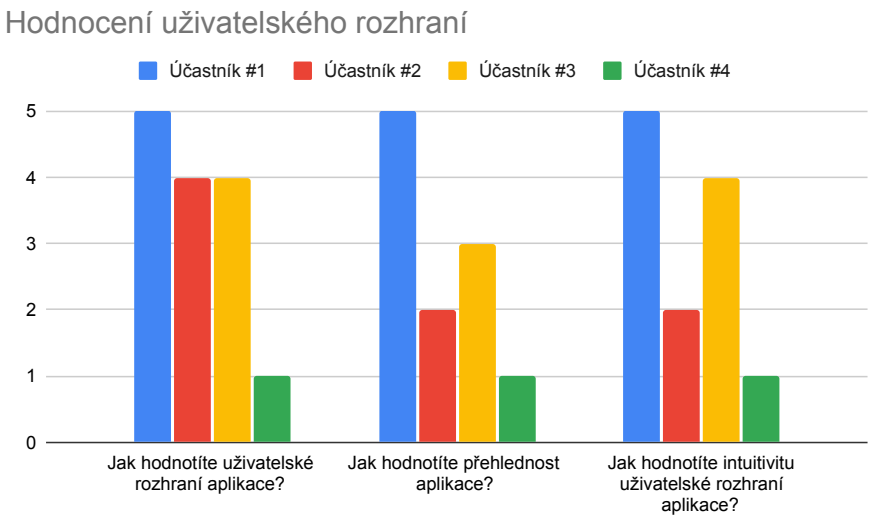


(b) Graf zobrazující preferovaný typ her

Obrázek 6.1: Grafy zobrazující odpovědi z obecného dotazníku

Grafický rozhraní

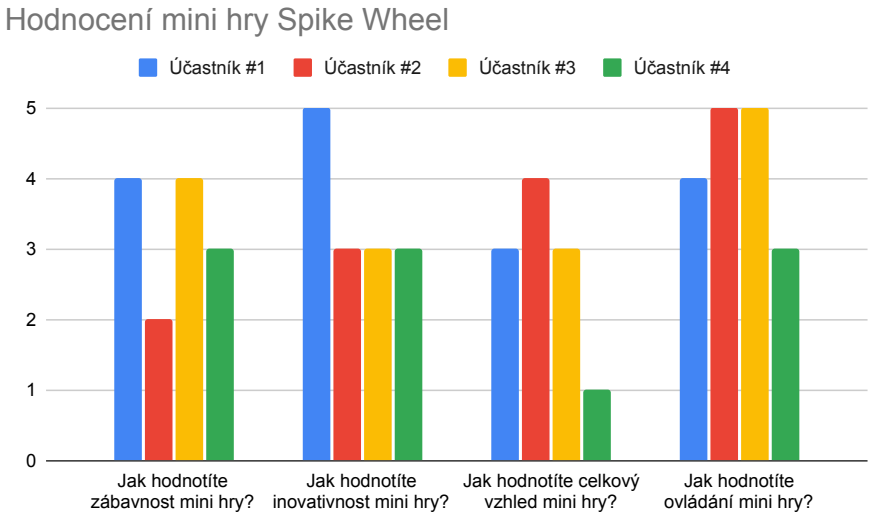
Respondenti, jak lze vyčíst z grafu 6.2, hodnotili grafické rozhraní převážně kladně. Jeden z respondentů, účastník číslo čtyři, hodnotil rozhraní převážně záporně. U slovního hodnocení uvedl: „Grafické rozhraní v menu je velice složité číst“. Účastník číslo tři, ale zase tvrdil že: „Celkový grafický design působil velice čistě“. S tímhle názorem souzněli i zbylí respondenti.



Obrázek 6.2: Graf zobrazující odpovědi jednotlivých účastníků ohledně uživatelského rozhraní aplikace. Vyšší hodnota na ose Y znamená kladnější hodnocení.

Spike Wheel

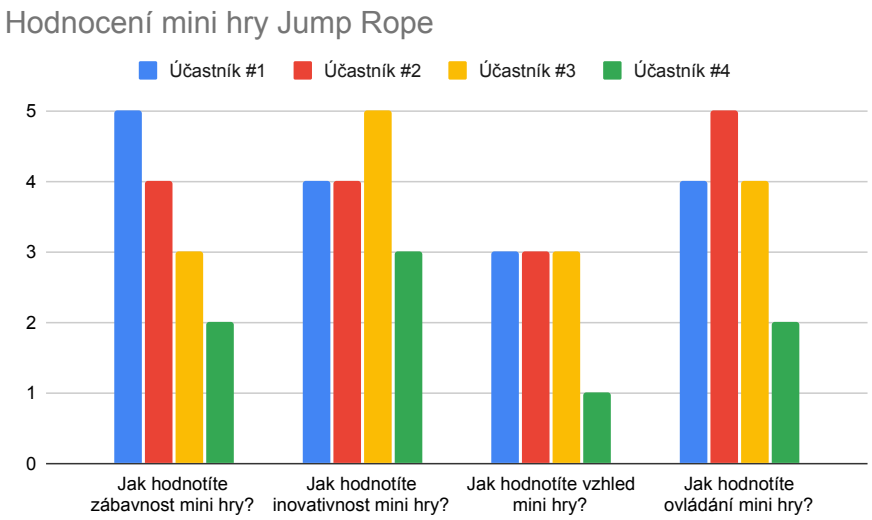
Hodnocení mini hry `Spike Wheel` byla převážně kladná. V textové části formuláře si polovina účastníků stěžovala na „nespravedlivé hit box“ a také na velkou rychlost kol. Graf 6.3, říká že se respondentům nejvíce líbilo ovládání mini hry.



Obrázek 6.3: Graf zobrazující odpovědi jednotlivých účastníků ohledně mini hry Spike Wheel. Vyšší hodnota na ose Y znamená kladnější hodnocení.

Jump Rope

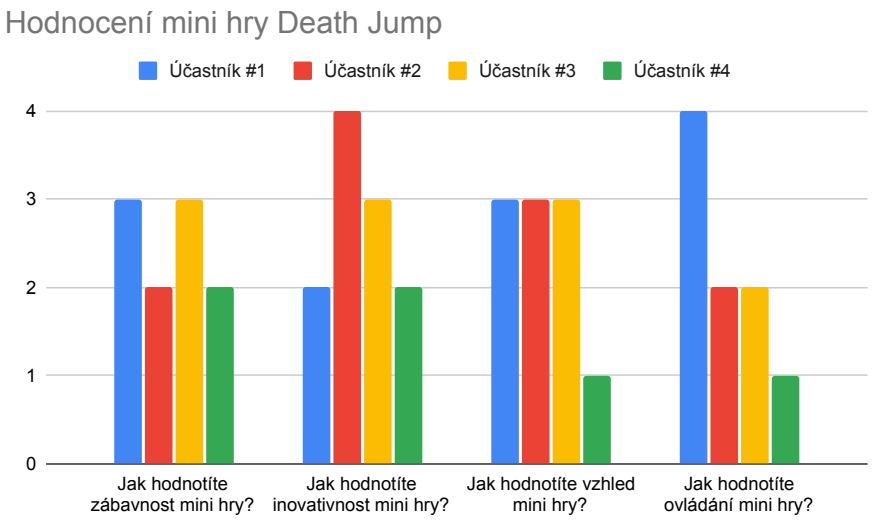
Mini hra `Jump Rope`, byla ohodnocena jako nejvíce inovativní (podle grafu 6.4) a také jako nejvíce zábavná. Naopak její vzhled byl špatně ohodnocen. Při slovním hodnocením většina respondentů hře vytkla: „Nebylo jasné kde se kolečka po stisku tlačítka objeví“.



Obrázek 6.4: Graf zobrazující odpovědi jednotlivých účastníků ohledně mini hry Jump Rope. Vyšší hodnota na ose Y znamená kladnější hodnocení.

Death Jump

Tato mini hra byla ohodnocena převážně záporně. Respondenti si nejvíce stěžovali na špatné ovládaní (což lze vidět v grafu 6.5). Negativně také hodnotili zábavnost mini hry. Podle jednoho z respondentů by hře pomohlo, kdyby se rychlost a četnost překážek postupně stupňovala. Jeho doslovné hodnocení zní: „Mini hra je příliš rychlá, někdy nestihnu zareagovat, rychlost překážek by se měla zrychlovat postupně“.



Obrázek 6.5: Graf zobrazující odpovědi jednotlivých účastníků ohledně mini hry Death Jump. Vyšší hodnota na ose Y znamená kladnější hodnocení.

Celkové hodnocení

Nejlepší mini hrou byla podle respondentů hra `Jump Rope`. Na první příčku ji umístili tři ze čtyř respondentů. Všichni respondenti se shodli na tom, že mini hra `Death Jump` byla nejhorší. Respondentům ve hře chyběl hudební doprovod a zvuky, část z nich by také ocenila možnost úpravy počtu herních kol. Jeden z nich by ve hře ocenil možnost psaní zpráv.

Kapitola 7

Závěr

Cílem bakalářské práce bylo navrhnout a implementovat herní demo, které umožňuje hrát ve vícero hráčům přes internet. A navrhnout mini hry, které jsou vhodné pro hru ve více hráčích.

K tomu bylo nutné se seznámit se herním enginem Unity a nastudovat, jak se s ním pracuje, prozkoumat dostupná řešení multiplayeru pro hry a vhodné zvolit. Tím se stalo řešení od `Photon Engine`.

Celkem bylo navrženo pět mini her. Z navržených mini her byly implementovány tři. Důvod je ten, že implementace byla náročnější, než bylo předpokládáno.

V průběhu celého vývoje bylo prováděno uživatelské testování. Na základně podmětů od uživatelů byla aplikace upravovaná až do její finální verze. Jakmile byla aplikace ve finální verzi, tak bylo provedeno uživatelské testování, ve kterém korespondenti vyplnili formulář. Výsledky testování jsou popsány v kapitole 6.1. Hodnocení referentů bylo spíše pozitivní. Nejvíce je zaujala mini hra `Jump Rope` a také chválili vzhled grafického uživatelského rozhraní. Grafickou stránku mini her hodnotili naopak jako průměrnou.

Práce na projektu mi dodala zkušenosti s vývojem her pro více hráčů. Bylo to poprvé co jsem programoval multiplayerovou hru. Ze zkušeností, které jsem při vývoji získal bych teď udělal pár věcí jinak. V první řadě bych hru zaměřil spíše na kooperativní mechaniky než na kompetitivní. S tím také souvisí další změna a to, omezení celkového počtu hráčů ze čtyř na dva. Co se týče dodatečných vylepšení aplikace, tak se jich nabízí celá řada. Hře by rozhodně neuškodilo navýšit počet mini her a také přidat různé typy mini her (jako má například hra v sekci 2.4). Přidat do hry zvuky a hudební doprovod. Grafická stránka hry je, co se uživatelského rozhraní týče, podle mě i podle uživatelů dostačující. Druhou stránkou je vzhled jednotlivých úrovní a postavy hráče. Ten by byl zapotřebí udělat vlastní či jeho tvorbu přenechat někomu zkušenějšímu. Další způsob, jak aplikaci vylepšit je předělání její síťové části buď do samotného C# nebo pomocí knihovny, která umožňuje kontrolu nad nižší vrstvou sítě.

Literatura

[1] *Free platform game assets + Gui by bayat games*. Dostupné z: https://bayat.itch.io/platform-game-assets.

[2] Engine, P. *Photon 2 dokumentace* [online]. 2022. Dostupné z: https://doc.photonengine.com/en-us/pun/current/getting-started/pun-intro.

[3] Gregory, J. 1.3 What Is a Game Engine? In: *Game Engine Architecture*. CRC Press, 2018, s. 11–11.

[4] Gregory, J. *Game Engine Architecture*. 1. vyd. CRC Press, 2018. ISBN 9781315267845. Dostupné z: https://www.taylorfrancis.com/books/mono/10.1201/9781315267845/game-engine-architecture-third-edition-jason-gregory.

[5] Griffiths, M. *IREP*. Schools Health Education Unit, Jan 1970. Dostupné z: https://irep.ntu.ac.uk/id/eprint/15272/.

[6] Kostet, R. *Theory of fun for Game Design 2ed*. 1. vyd. O'reilly, 2013. ISBN 1449363210.

[7] Nystrom, R. *Game Programming Patterns*. Genever Benning, 2014.

[8] Technologies, U. *Coroutines*. Dostupné z: https://docs.unity3d.com/Manual/Coroutines.html.

[9] Technologies, U. *Unity user manual 2020.3 (LTS)*. 2022. Dostupné z: https://docs.unity3d.com/Manual/index.html.

Příloha A

Obsah přiloženého paměťového média

program/ – spustitelné soubory aplikace

zdrojove-soubory/ – zdrojové soubory aplikace

zprava/ – zdrojové soubory zprávy

dotaznik.pdf – kompletní výsledky dotazníku

licence.txt – licence pro použitou grafiku

reademe.txt – informace o projektu

video-prezentace.mp4 – demonstrační video aplikace

xwoska00.pdf – tato zpráva