České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra počítačů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Sedlár Ondřej

Studijní program: Otevřená informatika
Obor: Softwarové systémy

Název tématu: Počítačová hra Songs of the Wind

Pokyny pro vypracování:

Vytvořte rešerši existujících her, jejichž průběh (gameplay) je založen na využití hudby coby prostředku interakce. Sepište návrhový dokument (design document) původní počítačové hry, jejíž průběh bude též stát na hudební interakci. Významnou složkou gameplay nechť je melodie, kterou hráčova postava hraje na píšťalu dle pokynů hráče skrze uživatelské rozhraní.
Seznamte se s potřebnými technologiemi, zejména Unity3D, a tuto hru realizujte.
V rámci práce realizujte vhodný subengine pro konkatenativní syntézu PCM signálu zmíněné píšťaly. Subengine nechť dokáže syntetizovat tóny různé délky příběhem definované stupnice a propojovat tyto tóny legatem.
Celou hru podrobte testování použitelnosti dle [1], aby vyhověla kritériím uvedeným v kap. 2.2, a syntézu signálu píšťaly otestujte jednoduchým online percepčním testem.

Seznam odborné literatury:

[1] Tekinbas, Zimmerman. Rules of Play: Game Design Fundamentals. MIT Press.
[2] Nielsen: Usability Engineering. 1993. Elsevier, Academic Press.

Vedoucí: Ing. Adam Sporka, Ph.D.

Platnost zadání do konce letního semestru 2017/2018

[redacted]

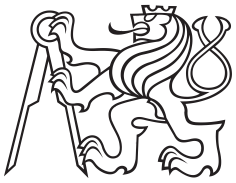
prof. Dr. Michal Pěchouček, MSc.
vedoucí katedry

[redacted]

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 20.2.2017

Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Počítačová hra Songs of the Wind

Ondřej Sedlár

Vedoucí: Ing. Adam Sporka, Ph.D.
Obor: Softwarové systémy
Studijní program: Otevřená informatika
Květen 2017

Poděkování

Chtěl bych zde poděkovat mému bratrovi Tomáši Sedlárovi za úžasnou grafiku, kterou pro tento projekt vytvořit a mému vedoucímu práce Adamu Sporkovi za pomoc a vedení při práci na tomto projektu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškeré použité zdroje a technologie.

V Praze, 20. května 2017

Abstrakt

Cílem této práce bylo vytvořit kompletní počítačovou hru pomocí herního enginu Unity. Hra patří do hudebního herního žánru, jejíž hlavní složka hratelnosti je založena na využití hudby coby prostředku interakce. V rámci přípravy na vývoj výsledné hry byl vyhotoven návrhový dokument (game design document) a rešerše mapující počítačové hry spadající do hudebního žánru.
V rámci rešerše bylo zmapováno celkem 17 hudebních her z hlediska prodejů, hodnocení i popisu hratelnosti. Ukázalo se, že mezi nalezenými hrami bylo velmi málo her, jenž hratelnost je alespoň částečně podobná hře z tohoto projektu.
Návrhový dokument již zcela popisuje všechny herní mechaniky a aspekty hry.
Vývoj výsledné hry byl úspěšný a podařilo se vytvořit ucelenou hru. Jedná se o tzv. 2D skákačku, jejíž hratelnost je založena na hře na virtuální flétnu. Ta je realizována pomocí konkatenativní syntézy PCM signálu.

Klíčová slova: hudební hra, počítačová hra, 2D, konkatenativní syntéza PCM signálu, Unity3D

Abstract

The aim of this work was to create a complete computer game using the Unity game engine. The game belongs to a musical genre whose main component of gameplay is based on the use of music as a means of interaction. As part of the preparation for the development of the final game, a game design document and a research for a computer game mapping of the musical genre were drawn up.
In the framework of the research, a total of 17 musical plays have been mapped in terms of sales, ratings and gameplay descriptions. It turned out that there were very few games among the games found, which is at least partially similar to the game from this project.
The design document already fully describes all of the game mechanics and aspects of the game.
The development of the resulting game was successful and managed to create a coherent game. This is a so-called 2D jumping game whose gameplay is based on the virtual flute play, which is realized through the concatenative synthesis of the PCM signal.

Keywords: music game, computer game, 2D, concatenative PCM signal synthesis, Unity3D

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

1.1 Cíle práce

Tato bakalářská práce má čtyři hlavní cíle. Prvním je vytvořit rešerši oblasti, které se práce týká (tj. hudební počítačové hry). V rešerši podrobně zmapovat stávající hudební hry, to znamená: mít přehled o roce, ve kterém byly vydané, najít nějaký jednotný zdroj hodnocení her, zjistit počet prodaných kopií (pokud bude možné) a zjistit jejich hlavní herní mechaniky. Takto zmapované hry následně rozdělit do kategorií v závislosti na jejich využívání hudby v rámci hratelnosti (gameplay[Cra]). V závěru také vzít v potaz jakým zařízením se hra dá ovládat.
Druhým cílem je vytvořit návrhový dokument (neboli Game Desing Document, zkráceno GDD). V něm popsat základní design hry. To zejména obecný popis a zasazení, příběh, herní mechaniky (alespoň ty nejzásadnější), navrhnout ovládání, herní prostředí / objekty a typy nepřátel.
Třetím a stěžejním bodem je vytvořit, navrhnout a naprogramovat samotnou počítačovou hru ([Kat03, Kapitola 8]). Nutnou podmínkou je, aby se dala hra označit, jako hudební. Významnou složkou hratelnosti musí být melodie, kterou hráčova postava hraje na flétnu dle pokynů hráče skrze uživatelské rozhraní. V rámci toho je třeba realizovat vhodný subengine pro konkatenativní syntézu PCM signálu (viz. [Die00]) zmíněné flétny. Subengine musí umět syntetizovat tóny různé délky příběhem definované stupnice a propojovat tyto tóny legatem.
Posledním cílem je výslednou hru podrobit testování použitelnosti dle [Jak93] a následně na jeho výsledcích provést druhou iteraci vývoje.

1.2 Technické specifikace

- **Herní engine**
  Hru dělám v enginu Unity[Unib] a to z následujících faktorů:
  - je zdarma (i pro komerční účely)
  - má velkou komunitu a dobrou podporu
  - pro malá indie studia, či jednotlivce je snazší a rychlejší se s ním naučit a vyvíjet hry (než v konkurenčních enginech: UnrealEngine[1] atd.)
- **Programovací jazyk**
  Výběr programovacího jazyka určil výběr herního enginu. Vybraný engine podporuje tyto programovací jazyky: C# a JavaScript. Z osobní preference jsem si vybral C#, protože ho umím a již jsem v něm psal. A také proto, že v něm píše většina vývojářů, kteří tento engine používají. Proto je pro C# velká komunitní pomoc.
- **Vývojové prostředí**
  Jako vývojové prostředí pro psaní v C# jsem zvolil Visual Studio 2015[2] (čili oficiální vývojové prostředí od Microsoftu[Mic]).

[1] https://www.unrealengine.com/what-is-unreal-engine-4
[2] https://www.visualstudio.com/cs/

Kapitola 2

Rešerše

2.1 Úvod

Cílem je prozkoumat trh počítačových her a zjistit, jaké druhy hudebních her (hlavně her, které využívají hudbu, jako prostředek interakce) již byly udělány. A zjistit, jakým způsobem je hudba ve hrách využívá.

U nalezených her je uvedeno jejich hodnocení na službě Steam[Stea]. Hru může ohodnotit každý, kdo ji má na službě Steam zakoupenou a to možností „Doporučuji“, nebo „Nedoporučuji“. Hodnocení se skládá z procenta, které představuje hráče, kteří hru doporučili. Dále je uveden celkový počet hodnotících uživatelů.

Bylo rozhodnuto o uvedení hodnocení pouze ze služby Steam kvůli následujícím faktorům:

1. Velká část níže uvedených her nemá recenze na velkých a ověřených webových stránkách recenzujících hry.
2. Aby byla zachovaná alespoň částečná ucelenost hodnocení, která pochází od jednoho zdroje.

Některé hry nejsou na službě Steam a proto (pokud bude možnost) bude

Steam hodnocení nahrazeno alternativním (viz. popis u konkrétních her).

Počet prodaných kopií hry, který je zde uveden, je převzat v internetové stránky SteamSpy[Steb]. Je důležité zmínit následující dodatky:

1. Tyto údaje nejsou oficiální a jedná se pouze o odhad založený na web API, které je poskytováno službou Steam.
2. Čísla prodejů jsou pouze v rámci služby Steam (nejsou zde započítány prodané fyzické kopie hry (které se neaktivují přes Steam), ani kopie prodané na jiné službě, než je Steam).
3. ‚Majitelé' je údaj, který udává počet všech aktivovaných her na službě Steam.

Veškeré obrázky her (pokud není uvedeno jinak) jsou vzaty ze služby Steam.

2.2 Kategorie

Nalezené hry byly rozděleny do následujících kategorií:

1. Hudba je výrazný prvek designu (ale ne interakce)
2. Hudba je výrazný prvek interakce
    a. Hráč reaguje na hudbu
    b. Hráč reaguje hudbou na jinou hudbu
    c. Hráč produkuje hudbu

Hry patřící do kategorie 1 v podstatě nejsou hudební hry, a proto bude uveden pouze jeden zástupce a dále se na kategorii nebude brát zřetel. Naopak kategorie, které budou hlavním cílem rešerše jsou 2a a 2c. Podkategorie 2b byla přidána, kvůli významné staré hudební hře, která ale nespadá do žádné jiné kategorie.

2.3 Nalezené hry

Zde jsou jednotlivé hry, které jsem zmapoval. U každé níže popsané hry jsou uvedeny následující informace:

- Vývojář/i
- Rok vydání
- Hodnocení (jeli dostupné)
- Celkem majitelů (jeli dostupný)
- Celkem hráčů (jeli dostupný)
- Popis hry
- Obrázek ilustrující ukázku ze hry

Hry jsou logicky rozděleny do příslušných kategorií dle kapitoly Kategorie.

2.3.1 Hudba je výrazný prvek designu

Seasons after Fall. [Swi16]

- Vývojář: Swing swing Submarine
- Rok vydání: 2016
- Hodnocení: 88% z 325
- Celkem majitelů: 28,457 ± 4,369
- Celkem hráčů: 16,864 ± 3,363 (59.26%)
- Popis: Seasons after Fall je 2D atmosférická, logická skákačka s úžasným soundtrackem a ozvučením. Ve hře hráč ovládá lišku, která dokáže měnit roční období. Změnou ročního období se částečně změní i prostředí ve hře. Na tomto principu jsou postaveny environmentální puzzly. Hra klade důraz na atmosféru, čehož dociluje originální grafickým zpracováním a hudbou.

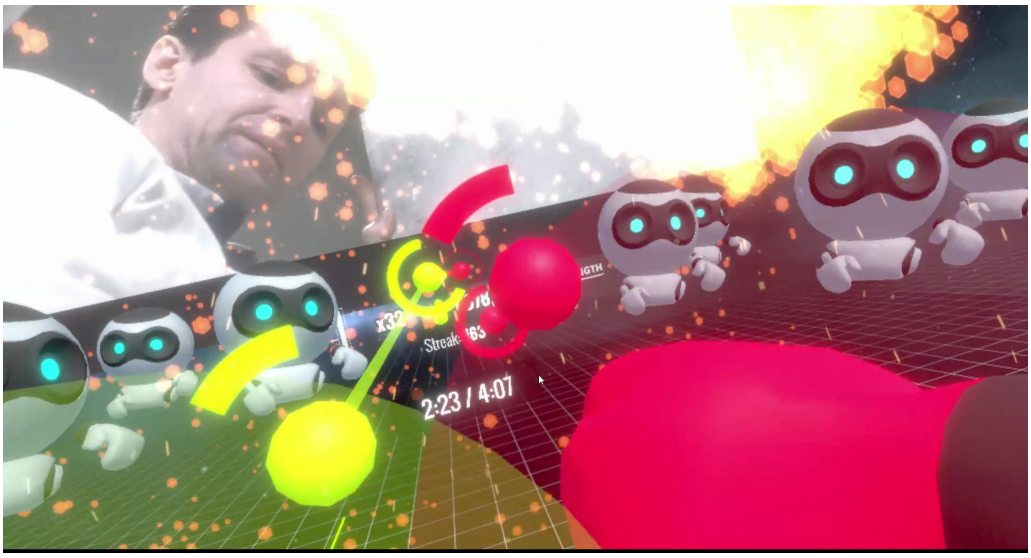
Obrázek 2.1: Ukázka ze hry Seasons after Fall

2.3.2 Hudba je výrazný prvek interakce

Hráč reaguje na hudbu

Soundboxing. [Max16]

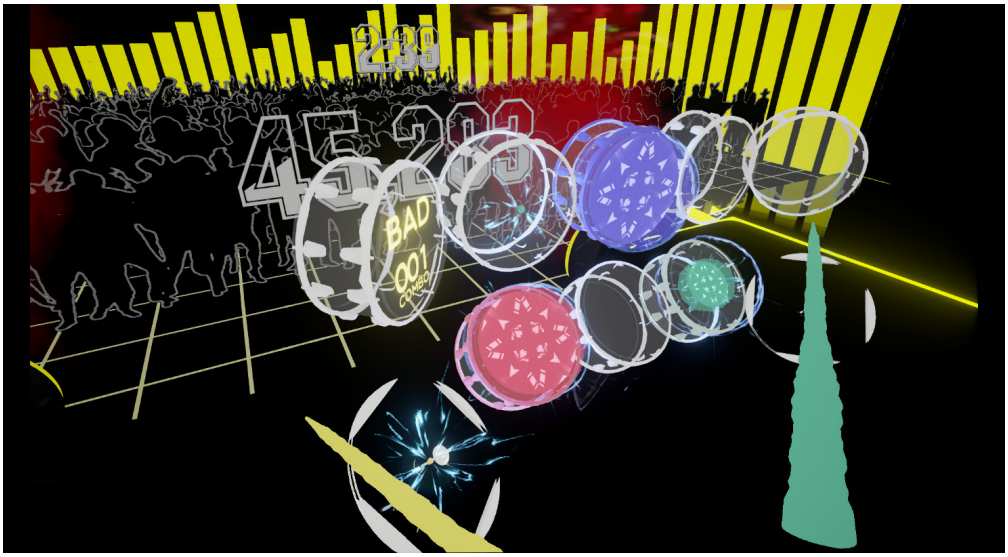
- Vývojář: Maxint LLC
- Rok vydání: 2016
- Hodnocení: 91% z 58
- Celkem majitelů: 2,301 ± 1,247
- Celkem hráčů: 2,124 ± 1,198 (92.31%)
- Popis: Soundboxing je rytmická hra pro VR. Hráč, který drží v rukách pohybové ovladače má za úkol trefovat objekty, které k němu přilétají v rytmu písničky. Hráč má dvě boxerské rukavice, které jsou barevně odlišené. Stejně tak jsou barevně odlišené i objekty, které musí hráč trefovat. Danou rukou může trefit pouze objekt stejné barvy.



Obrázek 2.2: Ukázka ze hry Soundboxing

Music Inside. [Rea16]

- Vývojář: Reality Reflection
- Rok vydání: 2016
- Hodnocení: 79% z 24
- Celkem majitelů: 885 ± 773
- Celkem hráčů: 1,046 ± 1,046 (120.84%)
- Popis: Jedná se o další hru pro VR. Hráč má za úkol hrát na bicí určitou písničku. Ve hře pohybové ovladače reprezentují paličky na bicí. Jednotlivé části bicí soupravy se podle písničky barevně rozsvěcí. Na tuto část pak musíte paličkou příslušné barvy zahrát.



Obrázek 2.3: Ukázka ze hry Music Inside

Audioshield. [Dyl16]

- Vývojář: Dylan Fittere
- Rok vydání: 2016
- Hodnocení: 86% z 7,626
- Celkem majitelů: 885 ± 773
- Celkem hráčů: 66,564 ± 6,708 (77.37%)
- Popis: Hra pro VR velmi podobná již výše uvedené hře Soundboxing. Zde má hráč místo boxerských rukavic štíty, jimiž přilétající objekty odráží. Viz. Popis hry Doundboxing.

Obrázek 2.4: Ukázka ze hry Audioshield

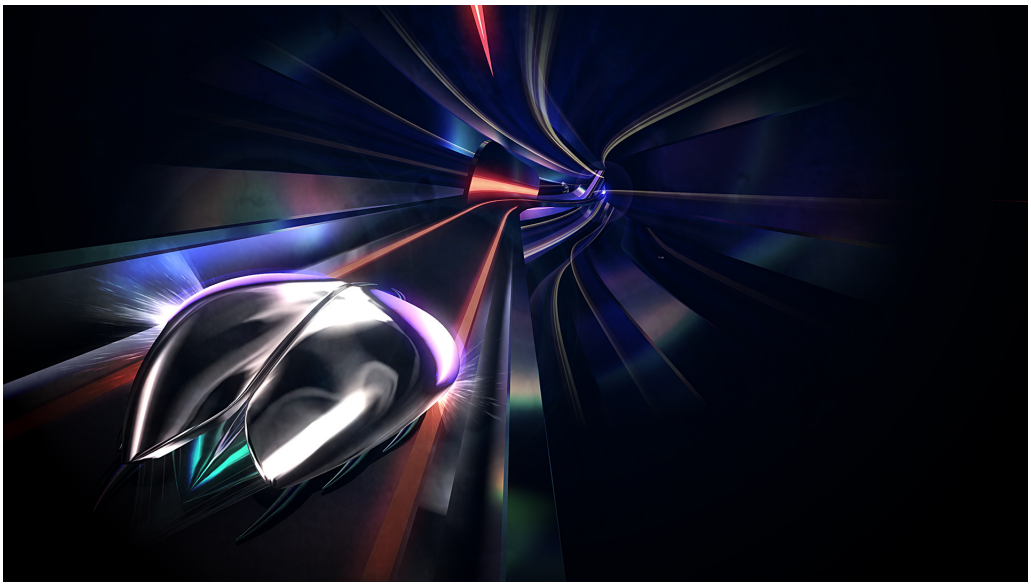
The Metronomicon. [Puu16]

- Vývojář: Puuba
- Rok vydání: 2016
- Hodnocení: 97% z 75
- Celkem majitelů: 3,718 ± 1,585
- Celkem hráčů: 2,833 ± 1,383 (76.19%)
- Popis: Jedná se o RPG hru, ve které hráč ovládá a spravuje skupinku hrdinů. Těm může postupem času dávat lepší vybavení a zlepšovat jejich schopnosti. Hlavní gameplay je souboj hráčovi skupiny proti nepřátelům. Během souboje může hráč v danou chvíli ovládat pouze jednoho z hrdinů. Aby hrdina provedl hráčem požadovanou akci musí nejprve zahrát určitou písničku. To probíhá stejným mechanismem, jako u hry Série Guitar Hero (viz. níže). Hráči se tedy objeví nad hrdinou dráha rozdělená na čtyři části. Každá část dráhy patří jedné šipce (doleva, dolu, nahoru, doprava). Po těchto částech v rytmu písničky jezdí příslušné šípky. V okamžiku, kdy šipka dojede na konec dráhy musí hráč danou klávesu se šipkou stisknou, čímž zahraje část písničky. Po úspěšném zahrání udělá hrdina hráčem požadovanou akci.

Obrázek 2.5: Ukázka ze hry The Metronomicon

Thumper. [Dro16]

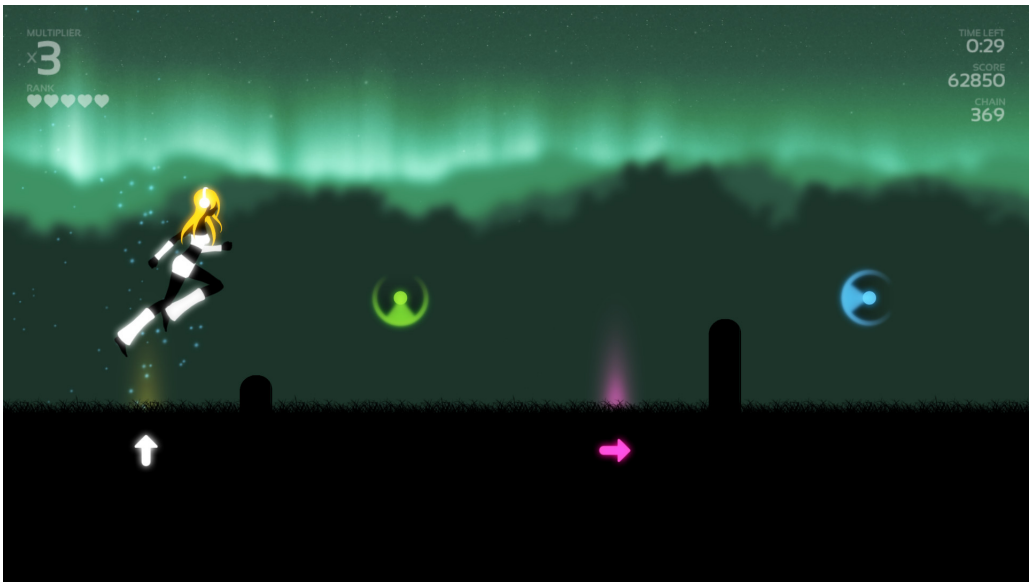
- Vývojář: Drool
- Rok vydání: 2016
- Hodnocení: 95% z 358
- Celkem majitelů: 15,225 ± 3,208
- Celkem hráčů: 14,694 ± 3,152 (96.51%)
- Popis: Jedná se o tunelovou, rytmickou hru. Hráč ovládající skaraba se pohybuje pouze po předem daném koridoru dané úrovně. Čili neovládá směr pohybu. Na dráze se v rytmu písničky objevují překážky, které musí hráč přeskakovat, odrážet se od nich atd.



Obrázek 2.6: Ukázka ze hry Thumper

Melody's Escape. [Ice16]

- Vývojář: Icetesy SPRL
- Rok vydání: 2016
- Hodnocení: 93% z 2774
- Celkem majitelů: 77,717 ± 7,248
- Celkem hráčů: 72,937 ± 7,022 (93.85%)
- Popis: Escape je rytmická 2D skákačka, ve které hráč musí proběhnout úrovní, která se generuje podle písničky, která při jejím hraní hraje. V úrovni jsou vygenerovány různé překážky, které je potřeba přeskakovat atd. Každý typ překážky je přidělen k určitému tlačítku na ovladači.



Obrázek 2.7: Ukázka ze hry Melody's Escape

Riff Racer. [FOA16]

- Vývojář: FOAM Entertainment
- Rok vydání: 2016
- Hodnocení: 88% z 596
- *U* hry není uveden počet majitelů. Tato hra není na internetové stránce SteamSpy[2] uvedena.
- Popis: Riff Racer je závodní hra, ve které se mapa generuje podle zvolené písničky. Generují se nejen překážky, do kterých nesmí hráč narazit, ale také tvar trati samotné.

Obrázek 2.8: Ukázka ze hry Riff Racer

Série Guitar Hero. *Poslední díl série:* [Fre16]

- Vývojáři:
  - Harmonix (2005–2007)
  - Neversoft (2007–2010)
  - Budcat Creations (2007–2009)
  - Vicarious Visions (2007–2010)
  - FreeStyleGames (2015–současnost)
- Hodnocení od Metacritic[Met]: 80% *(*poznámka: hodnocení je pro nejnovější díl z roku 2015)
- *U* hry není uveden počet majitelů, protože hra není dostupná na platformu PC a tím pádem není na službě Steam.
- Popis: Asi jedna z nejznámějších hudebních her. Největším rozdílem je ovladač v podobě kytary. Ta má na místě pražců tlačítka sloužící pro hraní akordů a místo, kde jsou struny a snímače má „páčku“ simulující úder do strun. Ve hře je vidět hmatník a pražce. Hmatník je rozdělen tak, aby korespondoval s tlačítky na ovladači (kytaře). Po hmatníku přijíždějí objekty, které představují stisk/podržení určitého tlačítka. To hráč stiskne/podrží ve chvíli, kdy objekt dojede k poslednímu pražci, který je zvýrazněn. Ve stejnou chvíli ale musí udeřit do strun/zmáčknout „páčku“.

Obrázek 2.9: Ukázka ze hry Guitar Hero *(*zdroj obrázku: [1])

[1] GamesCZ. `https://games.tiscali.cz/guitar-hero-live-17078/galerie` [Online]

Dance Dance Revolution (DDR). [Kon98]

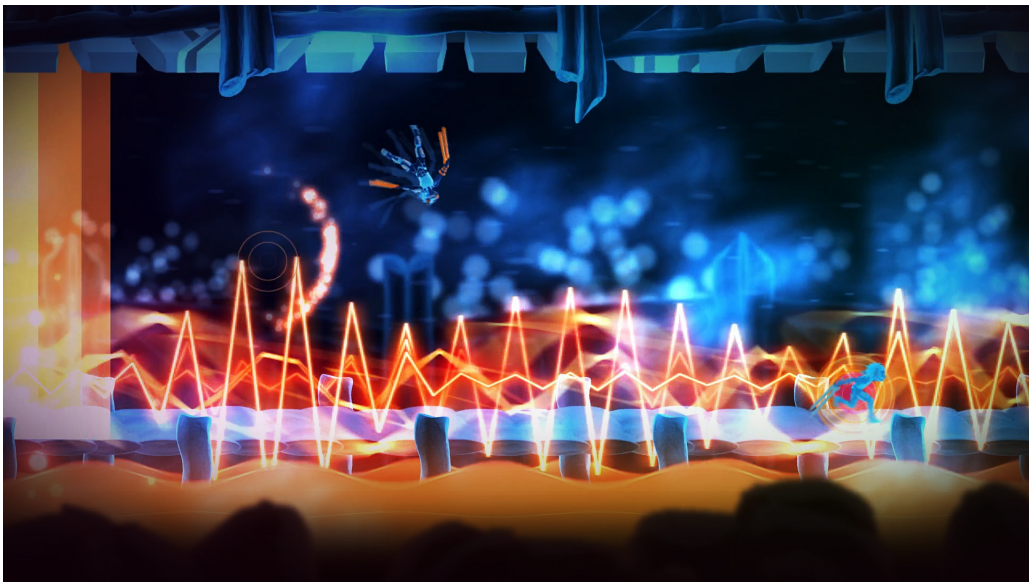
- Vývojář: Konami
- Rok vydání: 1998
- Jedná se o taneční automat. Hráč stojí na desce automatu, která je rozdělena na devět polí. Z toho jsou čtyři aktivní (směr tlačítek nahoru, dolu, doleva a doprava). Na obrazovce jsou nahoře čtyři šedé šipky. Ze spodní části obrazovky přijíždějí barevné šipky (barva šipky určuje její rytmus). Ve chvíli, kdy barevná šipka dorazí k šedé musí hráč nohou zmáčknout příslušné pole na desce. Tímto způsobem hráč tančí na písničku a podle přesnosti zmáčknutí polí se určuje jeho skóre.

Obrázek 2.10: Ukázka automatu DDR *(*zdroj obrázku[2]*)*

[2] `https://cz.pinterest.com/pin/352195633328781999/` [Online]

Klang. [Tin16]

- Vývojář: Tinimations
- Rok vydání: 2016
- Hodnocení: 95% z 24
- Celkem majitelů: 877 ± 766
- Celkem hráčů: 1,070 ± 1,046 (122.02%)
- Popis: Klang je 2D rytmická skákačka. Hra obsahuje exploraci, klasické skákací pasáže i souboj. Hráč by měl načasovat všechny akce do rytmu audio vizuálních podmětů.

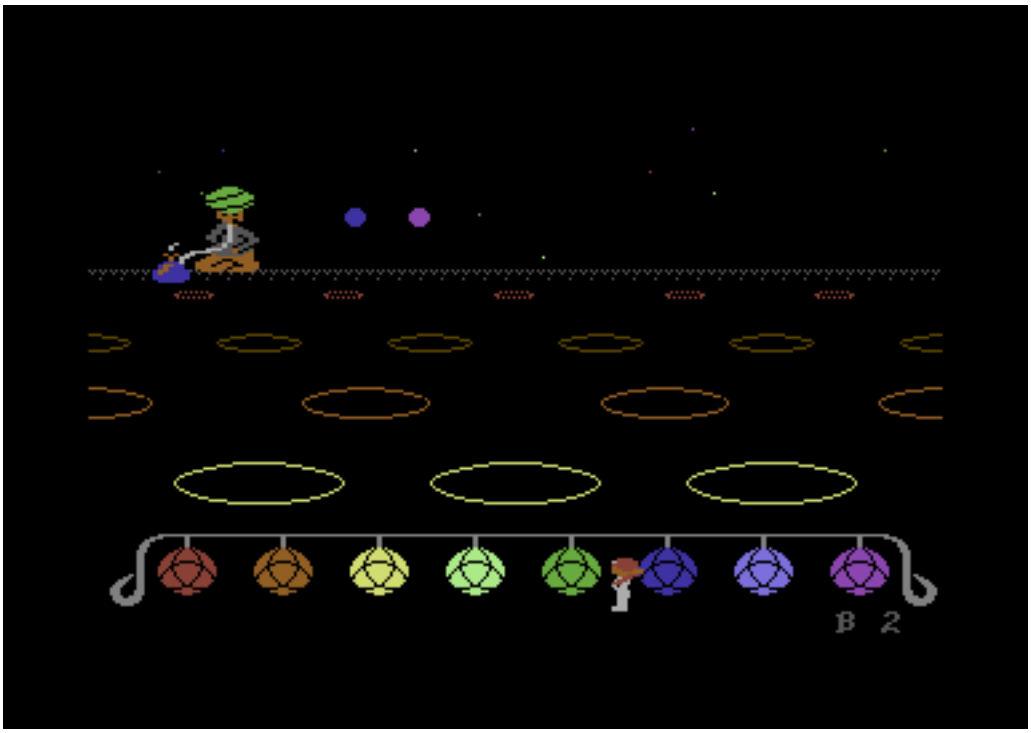


Obrázek 2.11: Ukázka ze hry Klang

Hráč reaguje hudbou na jinou hudbu

Master of the Lamps. [Act85]

- Vývojář: Activision Blizard
- Rok vydání: 1985
- U hry není uvedeno hodnocení, ani počet majitelů, protože se jedná o starou hru na Commodore 64 a proto není dostupná na službě Steam
- Popis: Hra se rozděluje na dvě fáze. V první fázi hráč musí proletět tunelem na létajícím koberci. To znamená prolétnout všemi bránami. Když se to hráči povede, tak se dostane do doupěte džina. Tím začíná druhá fáze. V ní je hráč u osmi gongů. Džin udává sekvenci tónu, které musí hráč na gongy zopakovat. Správně zahranými sekvencemi džina porazí.



Obrázek 2.12: Ukázka ze hry Master of the Lamps *(*zdroj obrázku[3]*)*

[3] `http://www.myabandonware.com/game/master-of-the-lamps-62r` [Online]

Hráč produkuje hudbu

LOOM. [Luc90]

- Vývojář: LucasArts
- Rok vydání: 1990
- Hodnocení: 89% z 228
- Celkem majitelů: 119,319 ± 8,981
- Celkem hráčů: 52,401 ± 5,952 (43.92%)
- Popis: Tato stará adventura z roku 1990 je založena na hraní melodií. Hráč muže klikat myší na jednotlivě tóny a tím hrát melodie. Princip je naučit se na co, která melodie je a pak je ve správném kontextu zahrát na interaktivní předměty ve hře. Využívá se například systém obrácené melodie (při zahrání melodie obráceně získá hráč opačný efekt).

Obrázek 2.13: Ukázka ze hry LOOM

The Legend of Zelda: Ocarina of Time. [Nin98]

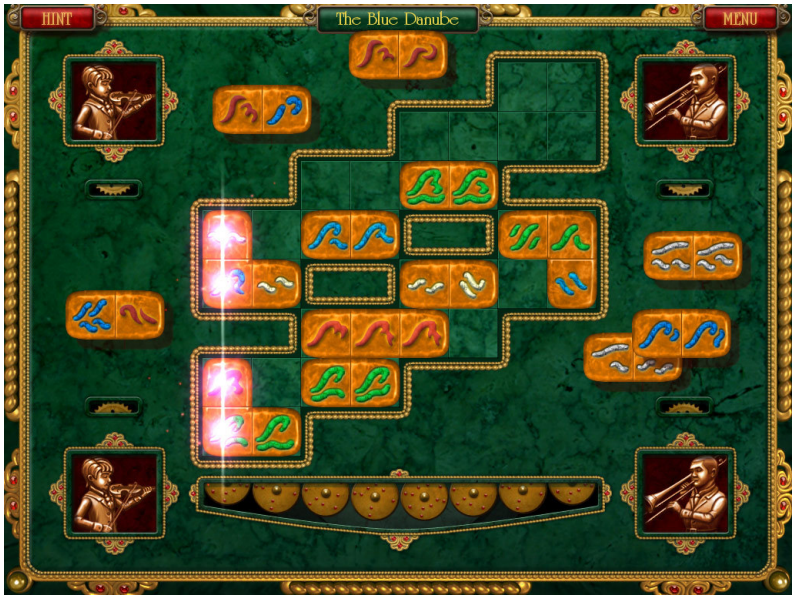
- Vývojář: Nintendo
- Rok vydání: 1998
- Hodnocení od Metacritic: 99%
- U hry není uvedeno hodnocení, ani počet majitelů, protože se jedná o hru na Nintendo 64 a není dostupná na službě Steam
- Popis: Tato hra je 3D akční adventura/skákačka, jejíž hlavní náplní jsou logické hádanky a souboje s nepřáteli pomocí standardních zbraní. Důvod, proč je hra uvedena v rešerši je její následující herní mechanika: hraní na okarínu, kterou má k dispozici v inventáři. Při jejím používání se hra přepne do módu hraní, ve kterém nejde postavička ovládat (hráč místo toho hraje na okarínu). Způsob hry na okarínu je následující: hráč musí zmáčknout tlačítka ve správné kombinaci (jiné tlačítko = jiný tón) a tím zahrát správnou písničku, která má určitý effekt.

Obrázek 2.14: Ukázka ze hry The Legend of Zelda: Ocarina of Time *(*zdroj obrázku [4])

[4]Nintendo. `http://nintendo.wikia.com/wiki/File:Dampe.jpg` [Online]

Musaic Box. [Kra08]

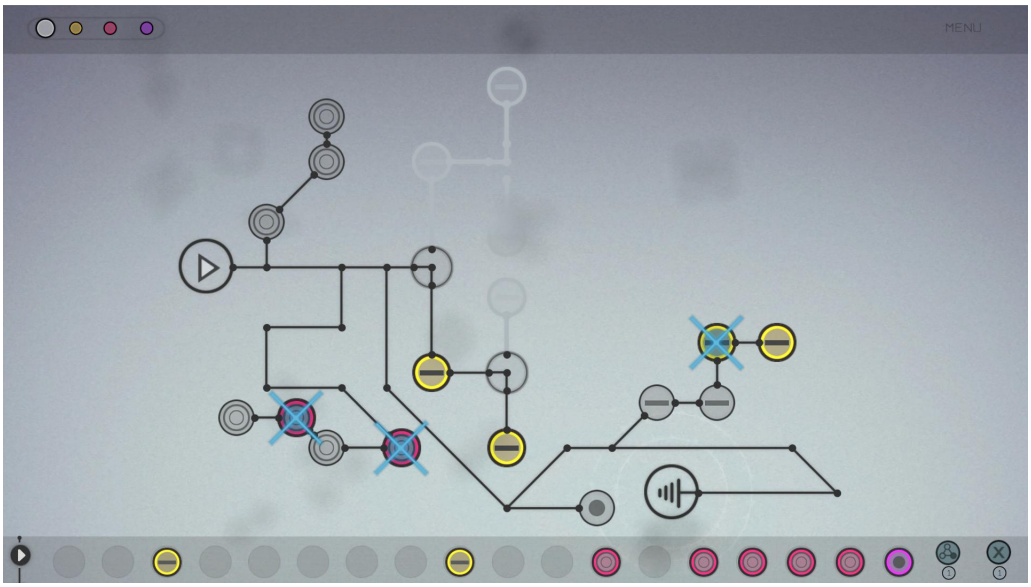
- Vývojář: KranX Productions
- Rok vydání: 2008
- Hodnocení: 91% z 130
- Celkem majitelů: 40,009 ± 5,201
- Celkem hráčů: 8,675 ± 2,421 (21.68%)
- Popis: Musaic Box hudební puzzle hra, jejíž hlavní herní mechanika je složit skládačku pomocí obrázkových dominových kamenů. Obrázky jsou vždy čtyř barev. Každá barva představuje jednoho hudebníka. Symbol na kamenu představuje, co daný hudebníky zahraje. Cílem je, aby hudebníci zahráli danou písničku, přičemž se bere pořadí zahraných částí z leva do prava po sloupcích (na obrázku zvýrazněno bílou zářící čárou).



Obrázek 2.15: Ukázka ze hry Musaic Box

Circuits. [Dig14]

- Vývojář: Digital Tentacle
- Rok vydání: 2014
- Hodnocení: 90% z 1657
- Celkem majitelů: 126,400 ± 9,243
- Celkem hráčů: 97,013 ± 8,098 (76.75%)
- Popis: Ve hře Circuits je hráčovým úkolem složit předem danou písničku. Písničku skládá stavěním „elektrického obvodu“. Kde součástky obvodu generují, nebo upravují hudební signál. Signál jde od součástky startu (zvýrazněno klasickou šipkou „start“), přes dráty postupně do všech ostatních součástek. Cílem je postavit obvod se součástkami tak, aby zahrál určenou melodii.



Obrázek 2.16: Ukázka ze hry Circuits

Flute master. [Ins16]

- Vývojář: Insignio Labs
- Rok vydání: 2016
- Hodnocení: 100% z 2
- Celkem majitelů: 1,770 ± 1,094
- Celkem hráčů: 1,066 ± 1,042 (60.22%)
- Popis: Hra Flute master se od ostatních her odlišuje tím, že se ovládá reálnou flétnou (tedy je potřeba mít připojen mikrofon). Principem hry je zabránit letícím upírům přeletět přes věž. Toho se dá docílit zapálením ohně v okně věže. Přičemž upíři létají v jednotlivých úrovních oken. Věž reprezentuje flétnu a okna otvory ve flétně. Podle toho, kolik otvorů je při zahrání ucpáno (jaký byl zahrán tón), v tolika oknech ve věži se rozhoří oheň.

Obrázek 2.17: Ukázka ze hry Flute master

2.4 Vyhodnocení

Zde je tabulka zobrazující přehled výsledků rešerše. V tabulce jsou uvedena čísla her podle kapitoly Nalezené hry. Některé hry lze ovládat více způsoby.

kategorie \způsob ovládání	Klávesnice/gamepad	VR	Speciální ovladač
2a	The Metronomicon Thumper Melody's Escape Riff Racer Klang	Soundboxing Music Inside Audioshield Thumper	Série Guitar Hero DDR
2c	LOOM Musaic Box Circuits LoZ: Ocarina of Time	——	Flute master

Tabulka 2.1: Výsledky rešerše

2.5 Závěr rešerše

Ve výsledcích rešerše se ukázalo, že naprostá většina hudebních her spadá do kategorie 2a. Přičemž největší podíl způsobu ovládání má klasická klávesnice/gamepad. Z těchto her si pro design hry nemohu vzít nic, jelikož ta bude spadat do kategorie 2c.
Co se týče kategorie 2c, tak opět převládá způsob ovládání na klávesnici/gamepadu. Je logické, že když je tato kategorie minoritou, tak zde nebude moc her na VR, jelikož je to zatím velmi nový a nerozšířený způsob ovládání.
Ukázalo se, že základní herní mechanika hry LOOM je velmi podobná té, kterou budu používat ve své hře. Je přínosné zamyslet se nad mechanikou obrácené melodie. Také hra The Legend of Zelda: Ocarina of Time má podobnou mechaniku skládání písní, hraním na hudební nástroj. Zde se ale jedná o minihru, zatímco v této práci půjde o hlavní mechaniku, která se bude používat plynule během samotného hraní. Jinak si z ostatních her této kategorie neodnesu nic, protože se jedná a velmi originální hry, které se neshodují s designem mé hry.
V kategorii 2b je pouze jedna hra: Master of the Lamps. Rád bych implementoval princip boje s džinem (opakování melodie zahrané džinem), jako finální boss souboj.

Kapitola 3

Game Design Document (GDD)

3.1 Úvod

Hra je 2d, se světě stavěném ve stylu „metroidvania“ her. Což znamená, že hráči bude svět na začátku otevřený, ale hráč pro postup do dalších oblastí a částí světa musí naučit jisté schopnosti (jsou mu kladeny přírodní bariery). Hlavní mechanika hry je simulace hry na flétnu přes klávesnici. Pomocí melodie hlavní postava ovlivňuje svět kolem sebe, bojuje s nepřáteli atd. Půjde zahrát šest tónů ve stupnici D dur.

3.2 Příběh

Zasazení. Hra se odehrává ve světě Casadh-Ceol, který je v jiné Dimenzi, než ten náš. Není nijak velký, jen jedna hora, obří les a louky… a jedna řeka. Tento svět napadnou takzvaní Jezdci. Ti se vydávají z nižších Dimenzí do světů, do nichž se snaží přivést Tmu a tím nechat svět spadnout do nižších Dimenzí, ve kterých sami žijí. To dělají pomocí nákazy. Ta má převážit nad Světlem.
Jezdcům je však na stopě Lovec, jenž je astrální bytostí tvořenou Stvořitelovým Světlem a Větrem. Ten byl vyslán Mezigalaktickou Radou, aby Jezdce zneškodnil a zabránil zničení dalších světů.
Tak se hráč v kůži Lovce pomocí Písní pokusí zachránit tento svět a rozbít

legie Jezdců.

Průběh. Lovec se postupně dostává do centra nákazy a vzpomíná si na Písně, bez kterých by stále silnější nepřátele nedokázal porazit. Když se dostane ke zdroji nákazy (boss) a zneškodní ji, zjistí, že Jezdci jsou již dávno pryč. Zničením zdroje nákazy zachránil svět, ale další světy jsou v ohrožení... jeho práce neskončila... sále musí jezdce zničit. Proto se zbavuje svého fyzického těla, jenž mu bylo propůjčeno, aby mohl svět Casadh-Ceol zachránit a vydává se po stopě Jezdců...

3.3 Atributy hráče

1. **Životy**
   Není třeba zbytečně popisovat, snad jen upozornit na mechaniku Obnova zdraví a Umírání.

2. **Dech**
   Určuje maximální délku zahrané písně. Čím lepší dech, tím jde déle hrát. Dech ubývá i při určitých pohybech postavy.

3.4 Herní mechaniky

3.4.1 Flétna a písně

Hra na flétnu. Klávesy pro pravou ruku představují jednotlivé tóny. Mapování tónů na klávesy je zobrazeno v tabulce 3.1.

Tón	Klávesa
D	J
E	I
F	K
G	O
A	L
H	P

Tabulka 3.1: Mapování kláves na tóny

Přesnost zahrání. U písní je důležité zahrát každý tón stejně rychle (nezáleží na celkové rychlosti). Podle celkové odchylky jednotlivých tónů je určí efektivnost písní. Po překročení určité tolerance se písnička nepodaří.
U písní Vstřícný kámen a Úsměv (viz. následující odstavec) se efektivita nijak nesnižuje (registruje se jen překročení tolerance).

Písně. Písně se skládají ze dvou a více tónů. Každá píseň je na něco jiného. V tabulce 3.2 jsou všechny zatím vymyšlené druhy.

Píseň	Popis
Světelný štít	Vytvoří pruh světla, který zpomalí nepřítele.
Úder větru	Odhodí a zraní nepřítele
Ukolébavka	Dokud je píseň hrána, tak nepřátelé v okolí postupně usínají. Pokud melodie skončí, tak se v závislosti na úrovni začínají probouzet.
Neotřelý vítr	Po zahrání písně se vybrané objekty budou vznášet ve vzduchu.
Vstřícný kámen	Dokáže na určitých místech udělat průchod v kameni.
Úsměv	Dokáže po zahrání vyléčit nakažená místa v okolí.
Mrak	Vyčaruje na určitou chvíli mrak, na který může hráč vyskočit.

Tabulka 3.2: Seznam písní

Jak získat písně. Na začátku si Lovec žádné písně nepamatuje, a bude si muset vzpomenout. Proto budou po světě místa Procitnutí, kde bude daná píseň znít. Když ji pak Lovec správně zahraje, tak se ji naučí (odemkne si ji v nápovědě). Jinak písně účinkují i když je lovec nejprve neuslyší u místa Procitnutí. U některých míst bude hned vedle předmět, kterého se Píseň týká. Naopak u některých si bude muset hráč domyslet k čemu jsou (nemůže pokračovat dál, ale naučil se novou píseň a ví, že cestou narazil na překážku, kterou nemohl vyřešit).

Jak zahrát písně. Aby hra zaregistrovala, že hráč zahrál píseň, tak ji musí zahrát bez přerušení. Tj. před stiskem další klávesy pro tón nesmí minulou klávesu postit (pustí ji až po stisku klávesy nové).

Úrovně. Každý druh písně má několik úrovní. Vyšší úroveň je složitější, co se zahrání týče, ale také silnější a účinnější. Na nepřátele / předměty vyšší úrovně neplatí píseň nižší úrovně. U písně Úder větru to platí jinak. Tam nepřítel dostane zranění, ale jen velmi malé.

Konkrétní hodnoty písní.

1. **Standardní písně**
   a. Vstřícný kámen
      (i) level = 1
      tóny = D|H|G
      (ii) level = 2
      tóny = D|H|G|A|G
   b. Neotřelý vítr
      (i) level = 1
      tóny = F|G|E, efekt = 2 vteřiny
      (ii) level = 2
      tóny = F|G|E|H|G, efekt = 2 vteřiny
2. **Vytvářecí písně**
   a. Mrak
      (i) level = 1
      tóny = A|H|G, efekt = 3 vteřiny
   b. Světelný štít
      (i) level = 1
      tóny= D|G|F, efekt = 5 vteřiny
      (ii) level = 2
      tóny = D|G|F|A|G, efekt = 5 vteřiny
3. **Bojové písně**
   a. Úsměv

(i) level = 1
tóny = A|H|F, efekt = 2 vteřiny

(ii) level = 2
tóny = A|H|F|G|A, efekt = 2 vteřiny

b. Úder větru

(i) level = 1
tóny = H|A|H|D, efekt = 20 poškození

(ii) level = 2
tóny = H|A|H|D|G|H, efekt = 25 poškození

c. Ukolébavka

(i) level = 1
tóny = D|E|F|E, efekt = 2 vteřiny

(ii) level = 2
tóny = D|E|F|E|G|F, efekt = 2 vteřiny

Vylepšení. Žádné RPG systémy ve hře nejsou (vylepšování flétny, zlepšování statistik a úrovně). V původním návrhu byly, ale ukázalo se, že se do hry příliš nehodí.
Veškeré Písně také fungují od začátku, takže jediné zlepšení tedy je hráčova dovednost / schopnost hrát na flétnu. Postupně si bude osvojovat složitější písně a bude ty jednoduché hrát rychleji.

3.4.2 Ostatní

1. **Obnova zdraní**
   Zdraví se obnovuje až po smrti.

2. **Umírání**
   Při každé „smrti“ hráč přijde o fyzické tělo (jeho podstata je ale nesmrtelná, čili nemůže doopravdy umřít).
   V tomto okamžiku se celý svět restartuje.

3. **Ukládání**
   Hráčova pozice a stav světa je automaticky ukládán při aktivování záchytného bodu, nebo vstupem do nové oblasti.

Pohyb postavy

Kromě základních pohybů doleva / doprava a skok může postava provádět speciální pohyb, které ji stojí Dech (úskoky do stran).

Mapování kláves na ovládání

Akce	Klávesa
Pohyb doprava	D
Pohyb doleva	A
Skok	W
Úskok	Mezerník
Uzamčení pohybu	L Shift
Pauza	Esc

Tabulka 3.3: Mapování kláves na jednotlivé akce

Hudba

Aby se hudba v pozadí netloukla se zvukem flétny, bude hrát pouze hudba ambientní. Nejlépe přírodní motivy (les, louka, zpěv ptáků atd.).

3.5 Objekty ve hře

Interaktivní prostředí

Tyto objekty jsou součástí běžného prostředí a nejsou nijak nepřátelská (spíše naopak). Jejich pomocí (po jejich reakci na konkrétní píseň) řeší hráč environmentální překážky a dostává se dále.

Skála. Vnější část není schována v zemi a tak brání hráči pokračovat. Reakce na píseň: vnější část se odsune do země. Skládá se ze tří obrázků:

1. Vrchol: Zůstává statický a má na sobě kolizní vrstvu, která zabraňuje hráči projít skrz.
2. Vnitřek: Je také statický, ale lze skrz něj projít.
3. Vnější část: Má kolizní vrstvu a je to část, která reaguje na píseň.

Létající kámen. Reakce na píseň: kámen se vznese do vzduchu na přesně snanovené místo.

Vytvářené objekty

Jsou objekty, které se vytvoří jako reakce na píseň. Jsou vytvořeny pouze na určitou dobu, takže mají u sebe progress bar, který ukazuje dobu jejich trvání.

Mrak. Je takzvaná oneway platforma (tj. hráč jí může zespoda proskočit nahoru). Spouží jako pomoc při skocích na vysoká místa.

Štít. Není klasický štít, ale spíše bariéra přes kterou jdou nepřátelé velmi pomalu.

Nepřátelé

Smrtelná zóna. Není to opravdový nepřítel, ale spíše statická bariéra, která do jisté míry ohraničuje herní svět. Pokud do ní hráč vstoupí, tak okamžitě umírá.

Nákaza. Stojí jen na jednom místě. Vypadá jen, jako černý mrak. Útočí pouze, když do ní Lovec vstoupí.

Úlomek nákazy. Taková „černá koule“. Základní nepřítel. Taktika: primitivní -> pouze bezhlavě letí na Lovce. Stejně, jako Nákaza působí zranění pouze

při kontaktu. Pokud lovce ztratí na určitou dobu z „dohledu“, tak se zastaví a přestane ho pronásledovat.

Nositel. Také nepřítel tvořený nákazou. Taktika: spíše stojí stranou a posílá na Lovce „projektily“ Nákaza. Ty mají vlastní životy a hráč se jich musí zbavit. Žijí, ale omezený čas.

Ohnisko nákazy (boss). Na základě výsledků rešerše se jedná o „hudební“ souboj (půjde pouze o hru na flétnu: tj. hráč se nebude moct hýbat). Velitel bude hrát svou hudbu, kterou musí hráč opakovat, aby neutrpěl zranění. Pokud ji zopakuje správně, tím provede protiútok a bosse zraní. Boss bude mít tři sady písní, ze které bude na začátku jednu vybírat.

3.6 Design

Grafika. Celkový design a grafika budou laděny uměleckým stylem. Styl grafiky bude připomínat štětcem ručně kreslené obrazy a výjevy.

Uživatelské rozhraní. Výsledné uživatelské rozhraní (UI[Eve13]) by mělo být co možná nejvíce minimalistické, ale neztratit potřebnou informační hodnotu, nestavět hráči zbytečné překážky a neničit zážitek.
Hlavní herní UI budou pouze bary zachycující hodnotu atibutů a ikona zobrazující dostupnost úskoku.

Tutoriál. Hra bude hráče postupně učit jednotlivé herní mechaniky a ovládání a to od samotných základů. Forma výuky bude velmi nenásilná (žádná samostatná menu s hromadou textu), ale spíše minimalistické nápisy přímo ve hře. Ty budou hráče jemně upozorňovat na jednotlivé aspekty hry tak, aby se o nich hráč dozvěděl a sám je následně prozkoumával a objevoval.
Ve chvíli, kdy hráč poprvé zahraje píseň je na to upozorněn nějakou zprávou/upozorněním.

Kapitola 4

Návrh a realizace

V této kapitole je popsán návrh a realizace výsledné hry.

Herní objekty. V Unity je každý objekt ve hře potomkem třídy GameObject[Unia]. Každý takový objekt má *Tag* a *Layer* (neboli vrstvu). Tag slouží jako identifikátor určité skupiny objektů. Vrstvy slouží pro omezení interakce jistého objektu jen na vybrané vrstvy (viz. oficiální dokumentace Unity[Unia]).
Na takovýto objekt se následně „navěšují“ komponenty (většinou ne v kódu, ale pomocí grafického prostředí Unity viz. obrázek 4.1) Samozřejmě ale jde komponenty do objektu přidávat kódem.

Obrázek 4.1: Ukázka několika komponent (konkrétně Transform, Script a AudioSource) na herním objektu

Z toho vyplývá, že veškerý kód, který se má spustit, se musí napsat jako skript. Třída, která je považována za script musí dědit od třídy MonoBehaviour. Tímto mohou být přidány, jako komponenta do herního objektu. Na začátku je ve skriptu zavolána metoda Start, která slouží pro iniciaci. Dále má MonoBehaviour spoustu „speciálních“ metod, jako například metodu Update. Ta slouží, jako hlavní smyčka, v níž je herní logika skriptu (viz. oficiální dokumentace pro MonoBehaviour[Unia]).
Herní objekty mohou mít také mezi sebou vztah rodič/potomek (ne ve smyslu programátorském: dědičnosti, ale pouze ve smyslu logickém). To znamená, že potomek se pohybuje společně s rodičem (má souřadnice v lokálním souřadném systému svého rodiče).

Komunikace mezi skripty. Zde bych chtěl ve stručnosti popsat jak funguje komunikace mezi skripty. Jelikož skripty nemají žádný konstruktor a nejde na ně použít žádný návrhový vzor[Rob00] má Unity metody pro hledání objektů ve scéně pomocí tagů, či jména. Přes tyto metody (např. FindGameObjectsWithTag[Unia]) lze dostat instanci na konkrétní herní objekt a dále ji používat ve skriptu.

Detektory. Jedná se o techniku, pomocí které všechny objekty ve hře komunikují a interagují na sebe navzájem. Protože je princip stejný u všech objektů, tak ho zde na začátku popíši obecně.
Herní objekty, které musí nějak reagovat na prostředí/okolí objekty musí mít na sobě Collider. Neboli určitou kolizní vrstvu. Při kolizi dvou takovýchto objektů vznikne (poduk to je povolené) událost. Collider se v to chvíli stane spouštěčem (Trigger), nikoliv kolizním obalem. Jsou tři tipy takovýchto událostí. Událost vyvolaná, když dvě komponenty typu Collider[Unia]:

1. **OnTriggerEnter2D**
   začnou kolidovat
2. **OnTriggerStay2D**
   stále kolidují
3. **OnTriggerExit2D**
   ukončily kolizi

Tyto události se dále zpracovávají ve skriptu, kde je příslušná logika, jak na události reagovat. Pokud dojde v určité události, tak je ve skriptu zavolána speciální metoda (jmenuje se stejně, jako událost na kterou reaguje).
Pro jednoznačnou identifikaci s čím daný Collider kolidoval používám jíž zmíněné tagy. Tímto způsobem např. nepřítel detekuje hráče a začne s útokem.

Rozdělení světa. Herní svět je rozdělen na oblasti, přičemž každá oblast představuje jednu scénu[Unia]. Každá scéna je naprosto oddělaná od ostatních a musí se vždy načíst. Navíc má unikátní index, pomocí kterého se určí, že se má daná scéna načíst. Ve hře se načítání zobrazuje pomocí načítací obrazovky.

Stavba jednotlivých úrovní. Při stavně se využívají takzvané prefaby[Unia] (prefabs). Jedná se o uložený již existující herní objekt, který lze takto jednoduše „kopírovat“ a používat ho na více místech. Navíc mají všechny kopie stejné hodnoty atributů (tj. stačí změnit hodnoty pouze o jednoho a ostatním se automaticky aktualizují).

Ukládání. Hra se ukládá vždy při přechodu mezi jednotlivými oblastmi a při vstupu k záchytnému bodu. Obě události jsou realizovány pomocí detektorů. Všechna místa pro ukládání mají v rámci jedné scény unikátní tag. Ukládání má tři fáze: uložení hodnoty aktuálního hráčova zdraví, jeho pozici a stav

světa.
Hráčova pozice se skládá ze dvou hodnot. První je index scény ve které se hráč nachází. Druhý představuje již zmíněný unikátní tag místa, kde byla hra uložena. Z těchto dvou údajů lze zpětně načíst přesné místo, kde si hráč hru uložil.
U interaktivních objektů se ukládá příznak, zda na ně byla použita píseň (neboli jejich stav). Každá scéna si interaktivní objekty ukládá do vlastního souboru, jehož jméno odpovídá indexu dané scény. Uložený objekt představuje jednu řádku v soboru, která má následující tvar: *unikátní_jméno_objektu;stav*, neboli se jedná o CSV soubor[Y. 05].

UI. Celé UI[Eve13] ve hře je logicky rozděleno do samostatných panelů:

- Hra
- Pauza
- Nastavení
- Načítání

Panely odsahují standardní Unity UI komponenty[Unia]. Skriptem InGame je zařízeno, že je viditelný pouze jeden panel, neboli slouží jako přepínač.

Vizuální efekty a shadery.

- efekt parallaxu v pozadí jsem stáhl od [joe] (open source)
- shader vytvářející tzv. glow efekt jsem stáhl od [Kaa]
- shader vytvářející tzv. bloom efekt jsem stáhl od [aub]

Font. Ve hře byl použit font Jim Nightshade[Bri11].

Grafika. Autorem grafiky je můj bratr Tomáš Sedlár.

4.1 Virtuální flétna

Virtuální flétna je systém, který z obdržených dat generuje zvuk příčné flétny. Tento systém bude ovládat hráč (viz. design document). Jde o vlastní model PCM signálu flétny, umožňující plynulé přehrávání tónů a jejich legato přechody. Je zde využita konkatenativní syntéza PCM signálu, neboli výsledný zvukový signál je skládán z předpřipravených dílčích částí (samplů).

4.1.1 Data

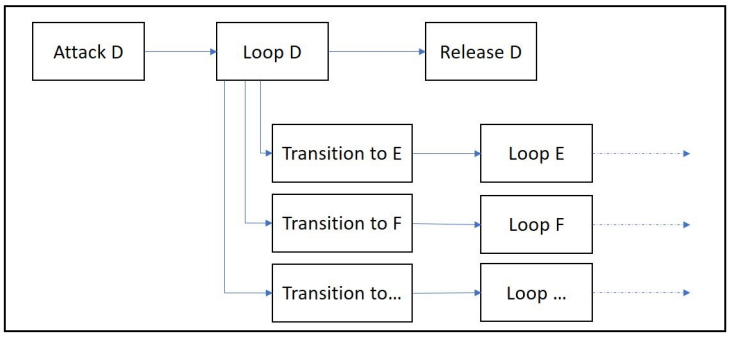
Každý z tónů je realizován třemi částmi (samply):

1. **Attack**
   Neboli začátek tónu.

2. **Loop**
   Samotný průběh tónu, který je zacyklený.

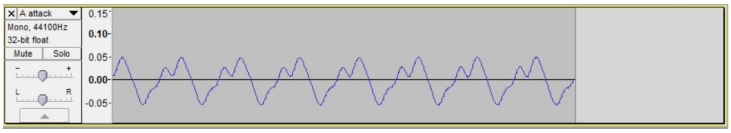
3. **Release**
   Ukončení tónu.

Pro hladké a plynulé přechody mezi jednotlivými tóny jsou navíc přidány části **Transition**, které představují přechod určitého tónu do ostatních tónů. Neboli pokud při hraní smyčky daného tónu (např. *Loop D*) se objeví požadavek pro zahrání tónu jiného (např. *Loop E*), tak se melodie nepřeruší, ale plynule se zahraje přechodová část mezi tóny (např. *Transition D-E*). Jejím přehráním se přejde z původního tónu do tónu nového. Ve chvíli, kdy přechodový sample končí se začne přehrávat smyčka požadovaného tónu (např. *Loop E*). Tímto způsobem je možno (aniž by hráč zaregistroval rušivý signál) přechodu mezi všemy tóny (viz. ukázka na obrázku 4.2).
Celkem tedy je 6*Attack + 6*Loop + 6*Release + 6*(5*Transition) = 48 samplů.



Obrázek 4.2: Ukázka matice přechodů

Originální zvukové nahrávky (formát wav), ze kterých byly samply udělány, byly zpracovány v programu REAPER[1] a nahrány pomocí programu KONTAKT player[2]. Tento program je určen pro přehrávání virtuálních hudebních nástrojů. Pro účely projektu jsem použil hudební nástroj Simple flute[3], který je zdarma (místo něj by mohla být použita klasická flétna, jelikož se používá pouze ke nahrání zvukových nahrávek).
Pro každý sample platí, že musí začínat a končit v nule (viz. obrázek 4.3), aby při navazování jednotlivých samplů nedocházelo k rušivým signálům. Proto byla finální úprava wav souborů provedena v programu Audacity[4]. Finální soubory byly nakonec převedeny (vlastním kódem) do raw formátu (formát, který obsahuje pouze čistá data bez hlaviček).

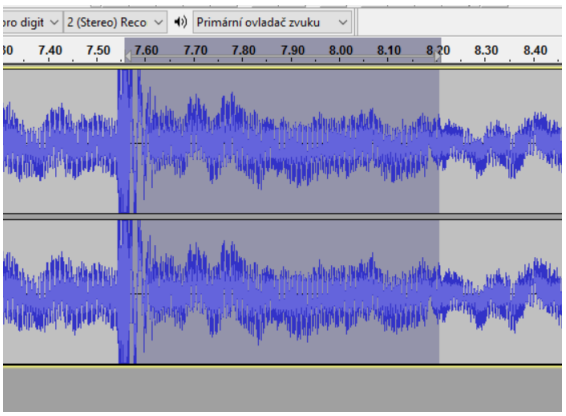


Obrázek 4.3: Ukázka ukončení samplu pro tón A (část Attack)

Pro přechody mezi samply je potřeba mít k dispozici vytvořené synchronizační body (syncpoints). V těchto bodech je zaručený bezpečný přechod mezi dvěma samply *(tyto body byly vytvořeny vlastním kódem)*. Bezpečný přechod znamená, že bude přechod plynulý a posluchač nezaznamená žádný rušivý signál (např. prasknutý).
Správně by synchronizační body měli být v 0 (tak jako v 0 začínají a končí samply), ale to způsobovalo jejich nedostatek a byla tak příliš velká odezva (cca 700 ms viz. obrázek 4.4), mezi stiskem klávesy a zahráním tónu, na který se přechází. Proto jsem do vyhledávání přidal toleranci ±0,01. Tato tolerance

[1] http://www.reaper.fm/
[2] https://www.native-instruments.com/en/products/komplete/samplers/kontakt-5/
[3] http://fluffyaudio.com/shop/simple-flute/
[4] http://www.audacityteam.org/

výrazně navýší počet bodů a zároveň pro lidský sluch není znatelná chyba při přechodu.



Obrázek 4.4: Ukázka zpoždění přechodu mezi dvěma samply

4.1.2 Kód

Jádro flétny se skládá z následujících tříd: Sample, SampleInstance, PlayList a Synthesizer.

- **Sample**
  Třída reprezentuje jeden obecný sample (jeden ze 48). Proto jsou následující atributy zcela zřejmé.
  - pole dat daného samplu, která se mají přehrávat
  - MIDI # daného tónu
  - typ samplu
  - seznam bodů přechodu (sincpoints)
- **SampleInstance**
  Třída představuje konkrétní sample, který se právě přehrává. Jeho nejdůležitější atributy jsou:
  - obecný sample, který se má přehrát
  - pozice přehrávací hlavičky (která data se mají následně přehrát)
- **Playlist**
  Playlist dostává požadavky na to co má přehrát. Drží si seznam právě přehrávaných SampleInstance, který aktualizuje a podle požadavků upravuje. Také si při vytvoření udělá seznam všech samplů, ze kterých vytváří konkrétní přehrávané instance.

- **Synthesizer**
  Je potomkem třídy Playlist, kterou rozšiřuje o metodu pro hromadné načtení dat.

- **Flute**
  Třída tvoří celkovou „flétnu“ (tohle už je skript přidaný na herní objekt Flute). Je zde ovládání a Synthesizer, který produkuje data pro přehrávání. Aby mohla být generovaná data slyšet musí být předána do komponenty AudioSource (neboli komponenta pro vytváření zvukového signálu). K tomu, aby se AudioSource vždy po přehrání všech dat v něm uložená, naplnil daty novými, používám speciální metodu OnAudioFilterRead. Tato metoda se vždy po vyčerpání dat zavolá s parametrem pole typu float, do kterého se mají nová data nahrát.

4.1.3 Flétna jako herní objekt

Herní objekt představující flétnu má na sobě výše zmíněný skript Flute a AudioSource. Tím je zajištěno, aby flétna hrála nějaké tóny. Dále je za potřebí, aby flétna ovlivňovala pouze objekty ve svém okolí. K tomu využívám techniky detektorů. Neboli herní objekt má Collider, který slouží pro vyvolávání událostí OnTriggerEnter2D a OnTriggerExit2D. Dále má samotný skript, kde se tyto události zachytávají. Skript má vnitřní seznam, na který přidává/odebírá objekty se kterými začal/přestal kolidovat. Reaguje pouze na speciální objekty (tzv. interaktivní objekty popsané níže).
Toto je vše, co herní objekt flétny potřebuje, aby mohl ve hře fungovat. Aby se flétna ve světe správně pohybovala, tak ji nastavíme, jako potomka hernímu objektu Hráč.

4.2 Obecné aplikování fyziky

Obecně fyziku (jak pro Hráče, tak pro nepřátele) zajišťuje skript CharacterPhysics. Ten využívá skript 2DCharacterController[pri] který je open source. Používám ho hlavně díky jeho podpoře pohybu po nerovných/zakřivených terénech. Skript CharacterPhysics implementuje základní rychlosti pro pohyb dopředu a dozadu, orientaci otočení (doleva/doprava), jednoduchý posun do pravého/levého směru (je rozlišeno, zda je to pohyb dopředu, nebo dozadu), skok a úskok (dash). Dále také implementuje metodu pro výpočet celkové rychlosti (po aplikování všech požadovaných pohybů).
Veškeré ovládání a případné specifikace jsou již pro konkrétní herní objekty

specifikovány v samostatných skriptech, které od obecného skriptu CharacterPhysics dědí.

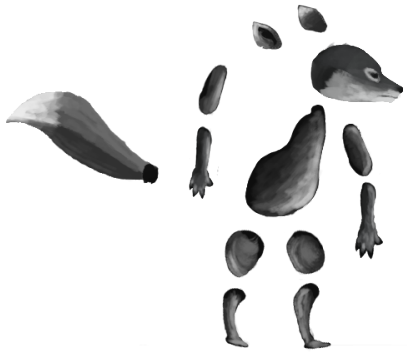
4.3 Hráč

Atributy. Veškeré atributy a podmínky s nimi spjaté jsou definovány v jediném skriptu (Player). Dle GDD má hráč pouze dva atributy a to životy a dech. Ve skriptu jsou definovány mezní hranice těchto dvou atributů. Dále síla „výdechu“ a „nádechu“, neboli hodnoty o které se hráči snižuje dech při hraní na flétnu a naopak o kolik se hráči dech nabíjí, pokud nehraje. K tomu aby věděl, jestli hraje si musí pochopitelně vzít instanci skriptu Flute.
Zásadní již zmíněné podmínky jsou:

- hráč nesmí hrát, pokud nemá již žádný dech (neboli atribut dech je roven minimální povolené hodnotě: 0)
- pokud hráč již nemá životy (neboli atribut životy jsou rovny minimální povolené hodnotě: 0) je mrtev a hra na to patřičně zareaguje.

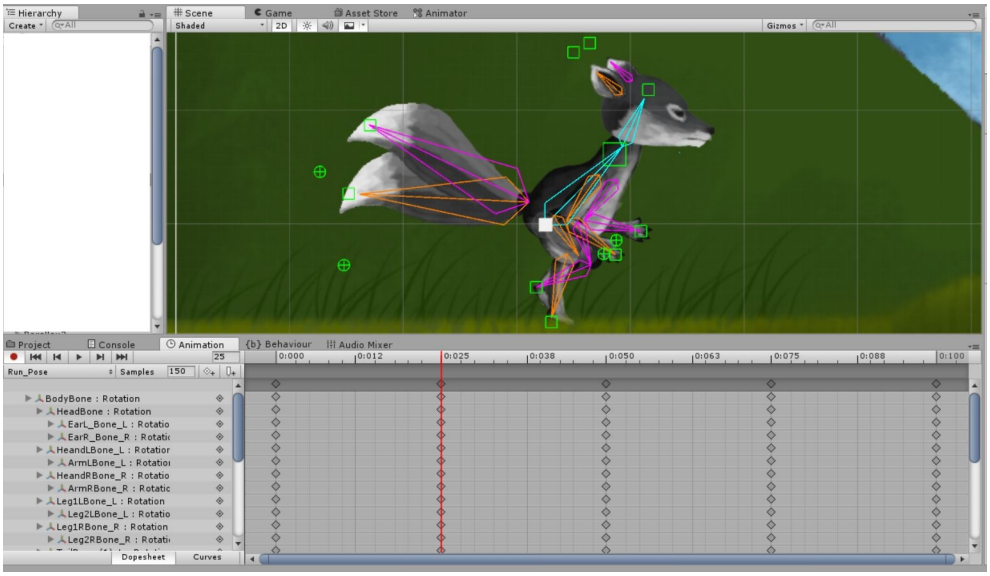
Pohyb a fyzika. Ovládání a fyzika hráče je implementována ve skriptu PlayerPhysics, která (jak je uvedeno v kapitole 4.2 dědí od CharacterPhysics). Je zde již konkrétní namapování kláves na jednotlivé pohyby a vylepšená detekce kolize se zemí (dovoluje jistou časovou toleranci, po jejíž dobu se počítá, jakoby byl hráč stále na zemi). Tuto toleranci je třeba přidat, protože terén je nerovný a občas na něm hráč lehce „nadskočí“ (ztratí kontakt s terénem). To je problém protože, jedno kritérium, aby mohl hráč vyskočit je právě kontakt s terénem.

Animace. Pro vytvoření animací hráče byl využit doplněk do Unity Sprites And Bones[Ban], který je open source. Jedná se tzv. skeletální animaci[Nad88] (ve 2D), při které pohyb grafiky určuje pozice a rotace kostí (neboli Bones). K tomu je potřeba, aby byla grafika rozdělena na jednotlivé části, které budou zmíněnými kostmi ovládány (viz. obrázek 4.5).



Obrázek 4.5: Ukázka „rozřezání“ grafiky hráče pro skeletální animaci

Grafika byla následně vložena do scény a složena tak, aby tvořila postavu. Pro každou část těla byla následně potřeba vytvořit kost, které ji bude při animaci ovládat (při ovládání kostí působí inverzní kinematika[Luk]). Samotná animace je vytvářena pomocí tzv. klipů (jeden klip = jedna animace), ty jsou pak přehrávány v Animator[Unia] (jedna z Unity komponent). Následně je pomocí klíčových snímků (key frames) vytvořeno několik pozic, mezi kterými je následně interpolováno (viz. obrázek 4.6)

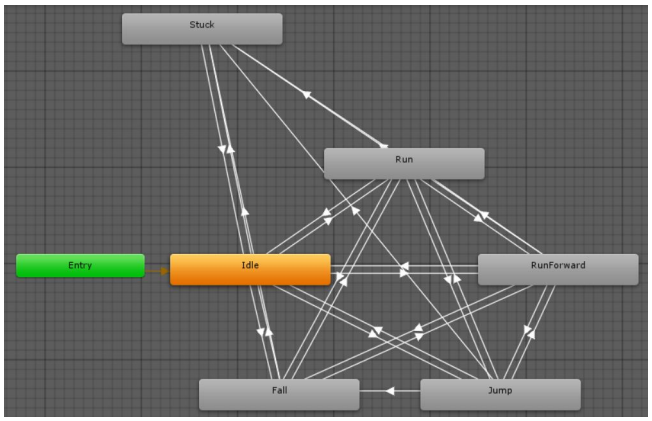


Obrázek 4.6: Ukázka jednoho klipu a jeho klíčových snímků

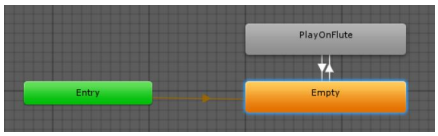
Zde je výčet všech hráčových animací:

- Idle
- Run
- Run forward
- Jump
- Fall
- Play on flute
- Stuck (jedná se o animaci, kdy se hráč nemůže bě hem finálního boss souboje hýbat (viz. GDD: popis bosse))

Výsledné animace (klipy) byly následně v Animátoru propojeny přechodovými podmínkami, které jsou následně řízeny ve skriptu. V Animatoru je možnost animace kombinovat a mixovat pomocí tzv. **Animation Layers**[Unia]. Hlavní vrstva (viz. obrázek 4.7) je určena animacím pohybu a druhá vrstva (viz. obrázek 4.8) je určena animaci hraní na flétnu (a ovlivňuje pouze ruce a hlavu) a pokud je přehrávána, tak částečně překrývá animaci ve hlavní vrstvě. Tím se docílí toho, že hráč může na flétnu hrát za běhu, při skákání, nebo když v klidu stojí na místě.



Obrázek 4.7: Ukázka animační vrstvy pro pohyb



Obrázek 4.8: Ukázka animační vrstvy pro hru na flétnu

Animace jsou nyní rozpohybovány pomocí inverzní kinematiky, čímž vzniká problém při mixování animací pohybu a hry na flétnu. Proto je potřeba animace předělat tak, aby nebyly inverzní kinematikou ovlivněny. Toho se

docílí pomocí tzv. póz. Jedná se o schopnost výše zmiňovanému doplňku Sprites And Bones, která dokáže uložit pozici a rotaci všech kostí (tím vytvoří zmiňovanou pózu). Proto z jednotlivých pozic všech animací, které jsme vytvořili pomocí inverzní kinematiky, uděláme pózy, ze kterých následně vytvoříme nové animace (klipy). Tyto animace vypadají stejně, jako ty původní ale nemusí u nich být zapnutá inverzní kinematika (nejsou jí ovládány) a mixování jednotlivých **animačních vrstev** funguje správně.

4.4 Prostředí

Místo procitnutí. Skládá se z několika obrázků a částicových systémů. Používá detektor pro zjištění přítomnosti hráče. Pro přehrávání používá skript pro automatickou flétnu a skript pro simulaci poklesu hlasitosti v závislosti na vzdálenosti od zdroje hudby (ta se počítá procentuálně pomocí trojčlenky, při znalosti maximální povolené hlasitosti, maximální možné vzdálenosti- rádius slyšitelnosti a hodnotě hráčovy pozice- přesněji na její souřadnici X).

Automatická flétna. Pracuje na stejném principu jako původní Virtuální flétna. S tím rozdílem, že nepřímá vstup od hráče, ale má pevně danou písničku, kterou neustále opakuje. Má určený čas hraní jednoho tónu a pak čas na odpočinek (prodleva mezi koncem písničky a opakovaným začátkem).

Smrtelná zóna. Jednoduchý herní objekt (prefab), který je tvořen detektorem. Při kolizi s hráčem vezme objekt hráči všechny životy.

Terén. Základní herní objekt (prefab), který obsahuje obrázek terénu a PolygonCollider (aby bylo možné po nerovném/zakřiveném povrchu chodit). U terénů, které končí okraji je navíc Collider pro detekci případného pádu nepřátel.

4.5 Interaktivní objekty

Pojmem interaktivní objekt je myšlen herní objekt, který může nějak reagovat na hráčem zahrané písně. Tyto objekty stejně jako písně mají určitou úroveň.

Ta určuje zda může být píseň na objekt uplatněna (respektive, zda má objekt na píseň zareagovat).

Tipy. Pro interaktivní objekty jsem vytvořil tři interfacy, které se dají kombinovat a určují povahu daného objektu.

1. **IInteractive**
   Tento interface je povinný, pokud má objekt být interatrivní. Předepisuje getter Level (pro získání hodnoty úrovně daného objektu) a metodu SongReaction, která se zavolá, pokud je na něj aplikovaná nějaká píseň (viz. kapitola 4.6).

2. **ICharacter**
   Zajišťuje, že objekt (stejně jako hráč) má daný počet životů a může umřít (metoda Death). Použito převážně u nepřátel.

3. **IResetable**
   Objekty s tímto interfacem mají nějaký stav (objekt zůstane rozbitý, nepřítel zůstane mrtvý). A v případě hráčovi smrti může být objekt resetován do původního stavu. Stav objektu se ukládá do souboru s názvem scény. Ukládá se jako dvojce: (jméno):(je možné objekt resetovat).

4.5.1 Interaktivní prostředí

Skála. Implementuje: IInteractiveObject, IResetable.
Stavy:

1. Základní: Vnější část není schována v zemi a tak brání hráči pokračovat.

2. Po reakci na píseň: Vnější část je pod zemí a hráč může projít. Toho se docílí přehráním animace vnější části (otřes), následně se vnější část začne posouvat směrem dolu a zapne se otřes kamery (ten je implementován v samostatném skriptu).

Dokud nedojde k restartu světa (hráč neumře), tak zůstává skála ve stavu Po reakci 2. Při restartu se ale dostane zpět do základního stavu Základní 1.

Létající kámen. Implementuje: IInteractiveObject.
Stavy:

1. Základní: Herní objekt je umístěn ve vzduchu a přehrává ve smyčce animaci pro vznášení.
2. Po reakci na píseň: Objekt se začne posouvat směrem nahoru, dokud neurazí požadovanou vzdálenost, jenž mu je na začátku zadána. Po zastavení setrvává daný čas nahoře a následně padá dolů. To je implementováno stejným způsobem, jako cesta nahoru, pouze s vyšší rychlostí (tj. na objekt neplatí gravitace).

4.5.2 Nepřátelé

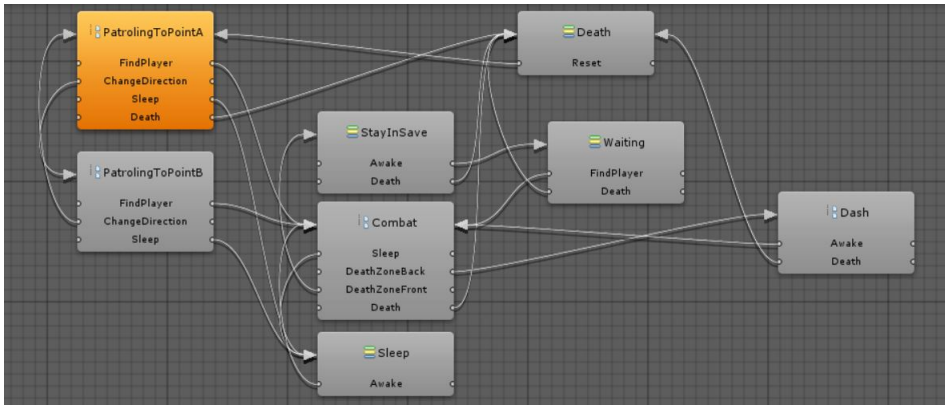
Všichni nepřátele implementují stejné interfacy. Proto jsou uvedeny zde na začátku a u jednotlivých tipů nepřátel již nebudou uváděny. Implementují: IInteractiveObject, ICharacter, IResetable

AI a rozhodovací stromy. Pro jednoduché nepřátele jako jsou Nákaza a Fragment nákazy je použita pouze jednoduchá „else-if“ metoda. Pro Nositel nákazy a Boss (zdroj nákazy) byl použit systém grafů a rozhodovacích stromů {b} Behaviour Machine[And], který je zdarma ke stažení.
Rozhodovací stromy/grafy jsou založeny na jednotlivých vrcholech/uzlech (každý vrchol/uzel je určité chování, nebo stav) a na hranách (ty představují přechodové podmínky mezi stavy). Do jednotlivých stavů se napojují skripty, které nedědí od MonoBehaviour, ale od ActionNode (který v sobě má instanci na GameObject, takže se neztrácí základní prvky použitelnosti, které má klasický skript). ActionNode není standardní součástí Unity, ale právě {b} Behaviour Machine.

Nákaza. Jednoduchý nepřítel založen na detektorech. Když přijde do kolize se hráčem (OnTriggerEnter2D) tak ho zpomalí (nastaví mu menší rychlost). Dokud hráč zůstává v kolizi s nákazou (OnTriggerStay2D), tak hráči nákaza postupně snižuje životy (v daném intervalu). Když kolize skončí (OnTriggerExit2D), tak nákaza hráči nastaví původní rychlosti.
Reakce na píseň: nákaza je zničena

Fragment nákazy. Jeho logika je jako u nákazy založena na detektorech. Má hlavní detektor, který je o poloměru cca 3x větší, než je jeho velikost. Pokud začne kolize s hráčem, tak si nastaví příznak, že má jít po hráči. Podle své a hráčovy pozice pozná, jestli je hráč napravo, nebo nalevo a pak se za ním vydá, protože útočí stejně, jako nákaza. V případě, že hráč chce fragment přeskočit, tak se pokouší po hráči skočit. Pokud ale nastane OnTriggerExit2D, tak začne odbíhat odpočet, po jehož vypršení fragment hráče „ztratí“ a přestane po něm jít.
Reakce na píseň: dostává zranění způsobené písní, ale pokud nejsou jeho životy 0, není zničen.

Nositel nákazy. Zde již je implementován stavový graf (viz obrázek 4.9).



Obrázek 4.9: Graf pro nepřítele Nositel nákazy

Objekt má tři detektory:

1. Hlavní:
   Hlídá, zda nenarazil na hráče. Pokud ano, tak volá událost pro přesun do boje.

2. Zadní:
   Hlídá, aby při couvání nenarazil na okraj, ze kterého by mohl případně spadnout.

3. Přední:
   Hlídá, aby při pronásledování hráče nenarazil na okraj, ze kterého by mohl případně spadnout.

Stavy:

- Krok CheckDeath: Není stav, ale krok, který je v každém stavu, takže je popsán zde na začátku.
  Pouze kontroluje zda nemá objekt již 0 životů. Pokud ano, tak vyvolá událost „Death“ pro přechod do stavu Death.

- PatrolingToPointA/B
  Je to startovní/výchozí stav.
  Jedná se o vnořený strom, který se skládá ze sekvence tří kroků. První je skript pro přesun ze současného místa na cílovou pozici. Ve chvíli dosažení cílové pozice začne čekací fáze na pozici. Do uplynutí dané doby se zavolá událost pro přechod do stavu PatrolingToPointB/A.

- Combat
  Jedná se o hlavní stav, který nastane pokud objekt zachytí hráče (pomocí hlavního detektu).
  Je to také, jako u stavu PatrolingToPointA/B vnořený strom, který se skládá ze dvou paralelních činností:

  - HoldDistance
    Udržuje si od hráče dostatečný odstup (v jistém rozmezí). Přitom ale zachovává orientaci pohledu vždy na hráči.
  - Shoot
    Tento skript zajišťuje vystřelování projektilů po hráči. Je implementována kadence střel (tj. je možno vystřelit pouze jednou za x vteřin, kde x je celé číslo větší, než 0). Navíc se nemusí střela povést. To z důvodu, že je zde ještě procentuální šance, která snižuje počet střel, ale navyšuje nepředvídatelnost střel. Tato omezení se zavedla, protože samotná střela je nepřítel a je obtížné bojovat proti většímu počtu střel najednou (viz. Střela).

- StayInSave
  Aby objekt nespadl při pronásledování hráče využívá přední detektor. Ve chvíli detekce pádu začne objekt couvat do bezpečné vzdálenosti a následně volá událost pro přechod do stavu Waiting.

- Waiting
  Do tohoto stavu se dá dostat pouze prostřednictvím stavu StayInSave. Objekt v něm čeká, dokud se hráč nevrátí na plochu, po které se může pohybovat a střílí po něm.

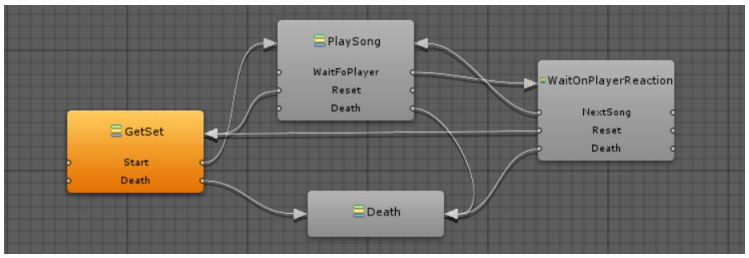
- Dash
  Aby objekt nespadl při couvání od hráče využívá přední detektor. Jedná se o útok z poslední nouze. Stejně, jako nákaza totiž zraňuje hráče při kontaktu. Proto se provede ve skriptu úskok (viz. Obecné aplikování fyziky). Tím se má hráč stáhnout. Po tomto útoku se zavolá událost pro boj.

- Sleep
  Do stavu se dostane, jako reakce na píseň (viz. kapitola 4.6). Pouze objekt na určitý čas uspí (neboli nic nedělá).

- Death
  Stav, kdy je objekt mrtví. Nic nedělá.

Střela. Je to samostatný nepřítel, na kterého platí všechny písně, které platí na ostatní nepřátele.
Střela má jen omezený dolet (po určitém čase je zničena), ale vždy letí na pozici hráče.

Boss (zdroj nákazy). Chování je implementováno pomocí stavového grafu (viz obrázek 4.10).



Obrázek 4.10: Graf pro bosse

Jednotlivé stavy na sebe navazují, takže chování popíšu uceleně po logických krocích, než popisováním stavů.

1. vybrání jedné ze tří sad písní
2. přehrání písně
3. hraje hráč nějakou píseň?
   - NE: ubíhá cooldown, hráč ztrácí životy
   - ANO: cooldown stojí, hráč neztrácí životy
4. pokud hráč zahrál písničku, před vypršením cooldownu
   - ŠPATNĚ: hráč přijde jednorázově o dané množství životů (je doprovázeno otřesem kamery)
   - SPRÁVNĚ: objekt ztratí jeden život, zvyšuje se náročnost písně, kterou objekt zahraje
5. pokud přišel objekt o poslední život, umírá
6. GOTO 2

4.6 Systém písní

Písně. Každá píseň, která byla navržena v GDD je reprezenntována jedním objektem Song (některé písně stejného jména jsou reprezentovány vícekrát, protože píseň může mít více úrovní). U každé písničky si uchovávám informaci a jméně, úrovni, seznamu not, ze kterých je složena a příznaku, zda již hráč

píseň zahrál (pokud ano, tak ji tím pádem má v menu v seznamu písní zobrazenou). Dále je to informace na jaký typ interatnivního objektu daná píseň působí.

Databáze písní. Jedná se o skript, jenž na začátku načte všechny písně a udržuje si jejich stav po celou dobu běhu programu.

Zachytávání písní. Skript „poslouchá“ flétnu (neboli sbírá informace o hráčem stisknutých klávesách / zahraných notách ze skriptu pro flétnu) a ve chvíli, kdy hráč přestane (dle kapitoly Jak zahrát písně v GDD) hrát, si skript zahrané tóny složí do písně a pokusí se jí najít v databázi. Pokud píseň v databázi najde, tak ji pošle dále k vykonání reakce na ni.
Pokud byla píseň zahraná poprvé, tak se zobrazí zpráva (UI komponenta text), která hráče upozorní, že se naučil novou píseň.

Reakce na písně. Zde se pouze podle konkrétní písně najdou příslušné objekty, které písnička ovlivňuje a podle pravidel dané písničky se na jednom (nebo více těchto objektech) zavolá metoda SongReaction, ve které objekt/y korektně zareaguje/jí. V případě vytvářecích písní se žádné objekty nehledají, ale vytvoří se příslušný objekt.

4.7 Hudba

Pro přehrávání a plynulé přechody mezi jednotlivými skladbami jsem napsal skript MusicPlayback. Plynulý přechod mezi skladbami je zajištěn pomocí fade out a fade in dvou sousedících skladeb (a to v dostatečném předstihu, než dohrávací skladba, na kterou je uplatněn fade out, skončí). Aby se při každé nové scéně nemusely skladby opakovaně načítat, tak je celý herní objekt, ve kterém je tento skript, takzvaný nezničitelný objekt[Unia], který je mezi scénami zachováván.

Seznam hudebních skladeb:

- Forest[Mor15]
- Ambient Birds[aye16]
- Grass Field[Luf08]

Seznam zvukových efektů:

- Key Chimes[Para]
- Wind[Parb]

Kapitola 5

Testování

5.1 Hra

5.1.1 Příprava testování

Cíle testování. Předmětem tohoto testu je hra Songs of the Wind (Písně Větru), která je vyvíjena v rámci této práce. Cílem je odhalit skryté nedostatky a bugy, které se během vývoje nepodařilo odchytit. Dále pak zjistit, jak si hra vede z hlediska uživatelské přívětivosti a v neposlední řadě jak je na tom samotná zábava a hratelnost.
Zvláštní důraz bude kladen na živatelskou přívětivost, neboť se hra pokouší o minimalistický design (hlavně co se týče uživatelského rozhraní, neboli UI[Eve13]).

Cílová skupina. Při testování jsem se zaměřil na osoby, které se věnují počítačovým hrám. Hra Songs of the Wind je hlavně 2D skákačka, proto jsem hledal osoby mající v oblibě tento herní žánr. Hra je ale také hrou hudební, takže jsme se snažil nalést osoby, které mají zkušenosti se současnými hudebnímy hrami (viz. rešerše). Důvodem jsou návyky z předchozích her a zejména fakt, že se od lidí s tímto profilem dá očekávat zájem o výslednou hru. Na druhou stranu je dobré hru otestovat i osobami, které vyloženě nehrají herní žánry do kterých hra spadá, nebo osobami, které hrají pouze příležitostně. Omezení/kritérium na věk účastníků není žádné.

Vybraní participanti. Testu se zúčastnilo 5 participantů, kteří byly vybrání na základě výše uvedených požadavků.

- tři účastnící byly studenti z vysoké školy (ve věku 21-23 let)
- jeden účastník byl studentem střední školy (ve věku 18 let)
- jeden účastník byl již pracující (ve věku 27 let)

Způsob testování. Testování probíhalo v neformálním prostředí za přítomnosti moderátora a pouze jednoho účastníka, který testoval hru. Moderátor měl za úkol provést parcitipanta testem, v případě nesnází mu pomoct a dělat si poznámky.

Použitý software. Hra Songs of the Wind – build aktuální k datu 15. 5. 2017.

Použitý hardware. Notebook Lenovo IdeaPad 700-15ISK[1].

Use cases. Testování mělo jeden use case: **Hraj první dvě oblasti ve hře a pochop/vysvětly její principy.** U něj byly detailněji pozorovány následující kroky:

1. zahaj novou hru
2. pochop základní ovládání hry
3. nauč se první píseň
4. zdolej první překážku
5. úspěšně se dostaň k prvnímu záchytnému bodu
6. následně projdi první oblast
7. ve druhé oblasti pochop tutoriál k soubojovému systému a poraz prvního nepřítele
8. dokonči druhou oblast

[1] http://www3.lenovo.com/us/en/laptops/ideapad/ideapad-700-series/IdeaPad-700-15ISK-/p/88IP7000671

Výchozí stav. Participant dostane hru zapnutou v hlavním menu.

Popis typů problémů.

- Kritický problém – chyba znemožňuje uživateli dále pokračovat v úkolu
- Vážný problém – chyba velmi znesnadňuje používání
- Méně závažný problém – spíše upozornění

5.1.2 Souhrn nálezů

1. **Nepochopení úrovně písní**
   - Závažnost: Kritický problém
   - Výskyt: 5/5
   - Popis: Účastníkům nebylo jasné, co je to úroveň písně, kde se větší úrověň naučí (kde ji získá). Také nebylo zřejmé, jaký interatnivní prvek (viz. GDD) je pro danou úroveň písně přijatelný (jaký objekt má již větší úroveň, než zahraná písnička)
2. **Nepochopení ovládání**
   - Závažnost: Vážný problém
   - Výskyt: 3/5
   - Popis: Účastníkům přišel základní tutoriál (popisky herních mechanik) příliš stručný a nejednoznačný.
3. **Zapamatování písniček**
   - Závažnost: Vážný problém
   - Výskyt: 5/5
   - Popis: Účastníci měli problém si po určité době hraní vzpomenout na písničku, kterou delší dobu nepoužívali.
4. **Vizualizace zraňování hráče**
   - Závažnost: Vážný problém
   - Výskyt: 3/5
   - Popis: Upozornění, že je hráč zraňován bylo znázorněno pouze health barem. Účastníci měli problém s tím, že si health baru nevšímali a následně umřeli, aniž by věděli proč.

5. **Upozornění při naučení nové písně**
    - Závažnost: Méně závažný problém
    - Výskyt: 2/5
    - Popis: Účastníci měli problém s barvou text upozornění prí naučení nové písně. Červená barva jim připadala pro upozornění nevýrazná. Také se jim nelíbilo, že nevědí nějaké informace o právě naučené písni.
6. **Délka písní**
    - Závažnost: Méně závažný problém
    - Výskyt: 2/5
    - Popis: Účastníci očekávali, že písně stejné úrovně budou stejně dlouhé (mít stejný počet not). Také jim přišli některé písně příliš dlouhé.

5.1.3 Závěr

Hra byla testována pěti účastníky testu. Mezi účastníky, byli hráči pokročilí i občasní. Nicméně nikdo z nich dříve testovanou hru nehrál. Hlavním cílem bylo otestování, zda hráči pochopí ovládání a principy hry a zda se jim bude dobře hrát. To bylo provedeno a z testu vyplynulo několik nedostatků, kterým byl přidělen typ závažnosti.

5.2 Syntéza signálu flétny

5.2.1 Příprava testování

Cíl testování. Předmětem tohoto testu je virtuální flétna (viz. 4.1), která byla v rámci projektu vytvořena pomocí konkatenativní syntézy PCM signálu. Cílem testování je zjistit jaký výsledný zvukový signál zmíněné flétny se uživatelům nejlépe poslouchá (je pro ně na poslech nejpříjemnější).

Cílová skupina. Výběr účastníků není pro tento test nijak omezen. Celkově bylo vybráno 5 účastníků.

Způsob testování. Testování probíhalo v neformálním prostředí za přítomnosti moderátora a pouze jednoho účastníka, který poslouchal zvukový výstup virtuální flétny. Účastníkovi byly postupně puštěny 4 hudební signály, které generovala virtuální flétna. Signály se měnily pomocí proměnné *pitch* v komponentě AudioSource (viz. 4). Proměnná *pitch* představuje stejnojmennou vlastnost zvuku, neboli výšku tónu. Vnímáním výšky člověk rozeznává „vyšší“ a „hlubší“ tóny (viz. [Ans06]). Proměnná byla testovaná s následujícími hodnotami:

- 1
- 0,9
- 0,8
- 0,7

U každé varianty účastník odpověděl na otázku: **Melodie se mi dobře poslouchala** (viz. následující odstavec).

Odpovědi účastníků. Účastníci museli na každou otázku odpovědět jednou z následujících možností. Ty vyjadřují míru jejich souhlasu s tvrzením otázky, na kterou odpovídají. Jedná se o tzv. Linkerovu škálu [Ren32].

- Rozhodně souhlasím
- Spíše souhlasím
- Nevím
- Spíše nesouhlasím
- Rozhodně nesouhlasím

Ohodnocení odpovědí. Každé odpovědi z Linkerovy škály byla přiřazena jiná hodnota (viz. tabulka 5.1).

Odpověď	Ohodnocení
Rozhodně souhlasím	4
Spíše souhlasím	3
Nevím	2
Spíše nesouhlasím	1
Rozhodně nesouhlasím	0

Tabulka 5.1: Ohodnocení odpovědí Linkerovy škály

Určení celkového výsledku. Výsledek se určí pomocí součtu všech odpovědí (respektive jejich ohodnocení) pro každou testovanou variantu. Čím větší součet tím je varianta oblíbenější.

5.2.2 Výsledky testování

V tabulce 5.2 jsou zobrazeny výsledky odpovědí jednotlivých účastníků. Poslední řádek reprezentuje součet hodnot odpovědí všech účastníků pro danou hodnotu proměnné *pitch*.

Účastník	pitch = 1	pitch = 0,9	pitch = 0,8	pitch = 0,7
P1	4	2	2	1
P2	2	1	1	4
P3	1	2	0	4
P4	3	3	0	3
P5	4	3	2	3
Součet	14	11	5	15

Tabulka 5.2: Výsledky testování syntéza signálu flétny

5.2.3 Závěr

Zvukový výstup flétny byl testován pěti účastníky testu. Dle výsledků je patrné, že účastníkům nejvíce vyhovovala varianta, kdy se proměnná *pitch* rovnala hodnotě 0,7.

Kapitola 6

Druhá iterace vývoje

V této kapitole jsou popsaná řešení na objevené nálezy v průběhu testování samotné hry.

1. **Nález: Nepochopení úrovně písní** Všechno interaktivní prostředí (viz. GDD) má nyní takzvaný LowLeveEfekt. Je to upozornění, že hráč zahrál správnou písničku, ale nižší úrovně, než jakou má objekt. Jedná je o UI (obrázek s číslem úrovně objektu). To je skryto (je neaktivní) a pokud nastene popisovaná situace, začně se přehrávat animace (na začátku se UI zviditelní, následně probíhá animace, která zvětšuje a zmenšuje UI a na konci se UI opět skryje).
   Také byla rozšířena nápověda popisující úrovně písní a interaktivních objektů.

2. **Nález: Nepochopení ovládání** Tutoriál byl rozšířen o doplňující informace, aby hráči lépe pochopily základy hry.

3. **Nález: Zapamatování písniček** Do menu byla přidána položka Písně, kde je kompletní seznam všech písní a jejich úrovní. Pokud hráč danou úroveň u písničky již umí, tak je u ní napsáno, jak ji zahrát (ne tóny ale klávesami, které jsou na jednotliv tóny namapované (viz. GDD)). V opačném případě tam je napsáno, že hráč písničku jěště neumí.

4. **Nález: Vizualizace zraňování hráče** Na herní objekt hráče jsem přidal skript, který kontroluje zda není hráč zraňován a pokud ano, tak mu nastaví shader tak, aby byla postava výrazně bíla a za daný časový interval dá shader zas pryč. Takto pokračuje po celou dobu, co je hráč zraňován a vytváří tím efekt, že hráč bíle poblikává.

5. **Nález: Upozornění při naučení nové písně** Dle nálezu byla barva textu změněna na žllutou, která je více výraznější, než červená. Pro více informací o nové písni, byl přidán popis jejího jména a číslo úrovně.

6. **Nález: Délka písní** Délka některých klíčových písní byla zkrácena. Písně na první úrovni byly upraveny tak, aby měli stejný počet not.

Kapitola 7

Závěr

Cíle této práce byly následující: vypracovat rešerši hudebních her, vytvořit návrhový dokument výsledné hry, naprogramovat a navrhnout samotnou hru a tu následně podrobit testování.
Rešerše mapuje jedny z nejzásadnějších hudebních her historie. Je v ní zahrnuta většina současných moderních hudebních her, které převážně vyšli na platformě Steam (ovšem některé hry mají spousty „kopií“ co se týče herní náplně a designu a proto by bylo redundantní je v rešerši uvádět). Samozřejmě, že není možné pokrýt 100% hudebních her, protože existuje velká spousta alternativ ke službě Steam a spousty indie her vůbec na těchto službách nejsou. Přesto si myslím, že rešerše dobře odráží stav herního průmyslu pro hudební hry. Dle výsledků rešerše se ukázalo, že v kategorii, do které výsledná hra spadá, je velmi málo jiných her. Pouze jedna z nalezených a zmapovaných her (The Legend of Zelda: Ocarina of Time) má lehce podobný princip hry na hudební nástroj, ale jeho zasazení do hratelnosti je odlišné. Pro design souboje s bossem, který celou hru ukončuje, byl nakonec opravdu použit styl, který se podobá herní náplni hry Master of the Lamps.
Návrhový dokument již zcela popisuje výslednou hru. A to v rámci hlavní herní mechaniky (hry na flétnu), příběhu, samotného designu (světa, nepřátel, hlavního hrdiny i písní) i konkrétních statistik jednotlivých písní.
Co se týče hry samotné, tak v rámci této práce se jedná o celistvou hru. Hra je ucelená (má jasný příběhový začátek i samotný konec zakončený soubojem s bossem). Má také drobné, ale ve hrách běžné a důležité náležitosti, jako hlavní menu, pauzu, nastavení hlasitosti tří zvukových zdrojů (hudba v pozadí, zvuk flétny a zvukové efekty) a automatické ukládání v rámci záchytných bodů (tzv. checkpointů). Subengine pro konkatenativní syntézu PCM signálu funguje (při syntéze nevznikají žádné šumy a výsledný signál je čistý bez rušivých praskání) a zvládá i rychlou změnu jednotlivých tónů. Samotná hra na flétnu byla do hratelnosti hry úspěšně začleněna. Hra se skládá ze 4

oblastí plus závěrečná oblast s bossem. Přitom je svět zcela otevřen a hráči v postupu brání jen přírodní bariéry a pokud je již zkušený (dokáže dobře hrát na flétnu) a zná příslušné písně, tak může vcelku přímou cestou jít až k bossovi a jednu oblast úplně přeskočit. Hra navíc velmi přirozeně a nenuceně učí hráče jednotlivé herní mechaniky od úplných základů.
Během testování výsledné hry bylo objeveno několik nedostatků spojených převážně s nepochopením některých herních mechanik kvůli málo podrobnému tutoriálu. Případně špatnou komunikací s hráčem, který v některých případech nebyl hrou dostatečně informován. Tyto problémy byly v druhé iteraci vývoje opraveny.
Při testování výsledného zvuku virtuální flétny se ukázalo, že optimální hodnota proměnné *pitch* je pro většinu účastníků rovna 0,7.

Plány do budoucna a zlepšení. Co se týče budoucích plánů, tak bych rád hru oficiálně vydal ale před tím je potřeba udělat následující:

- přidat anglickou lokalizaci (hra je pouze v češtině)
- přidat krátké příběhové filmečky
- zvětšit počet oblastí
- případně rozšířit seznam písní

Příloha A

Obsah přiloženého CD

- **text.zip:** obsahuje zdrojové tex soubory
- **source_code.zip:** obsahuje zdrojový kód a soubory, které byly použiity v projektu
- **final_game.zip:** obsahuje výslednou hru
- **sedlaon3_Songs_of_the_Wind.pdf:** je elektronická verze zprávy k bakalářské práci

Příloha B

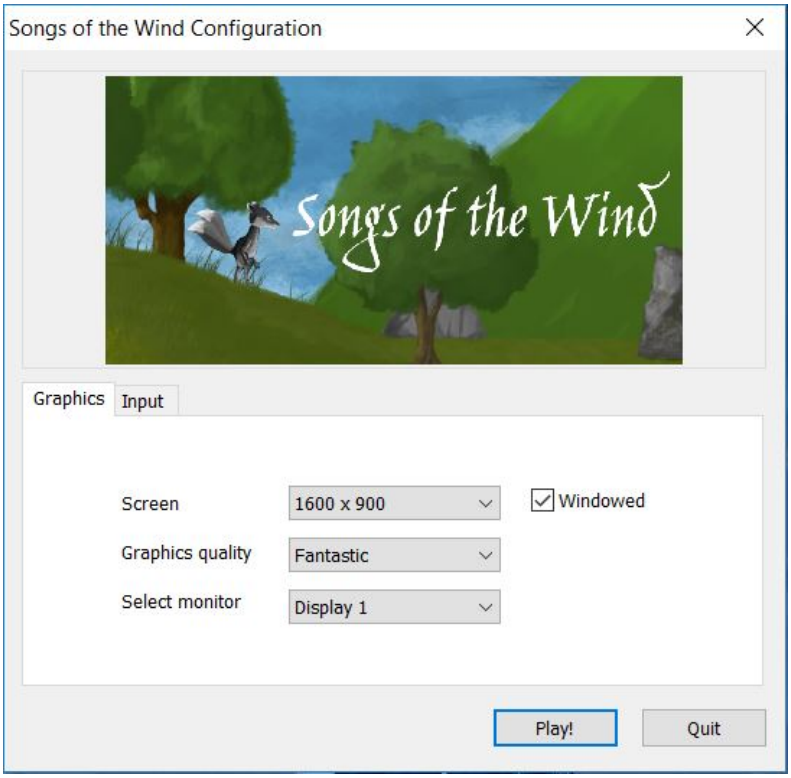
Uživatelská příručka

Požadavky pro spuštění hry:

- Hra musí být spuštěna na operačnám systému Windows
- Složky Assets a SongsOfTheWind_Data musí být ve stejné složce, ve které je soubor pro spuštění výsledné hry SongsOfTheWind.exe.

Zde je uveden jednoduchý návod k použití.

1. Začněte spuštěním souboru SongsOfTheWind.exe.
2. Zobrazí se Vám dialog s nastavením (viz. obrázek B.1)
3. Zde můžete v záložce Graphics nastavit rozlišení, kvalitu grafiky, cílový monitor a možnost, zda bude hra na celou obrazovku (*poznámka: záložku Input prosím ignorujte, na hru nemá žádný efekt*).
4. Následně klikněte na tlačítko Play!, čímž hru spustíte a dostanete se do hlavního menu.



Obrázek B.1: Ukázka úvodního dialogu hry

Příloha C

Literatura

[Act85] Activision Blizzard, *Master of the Lamps. Video game*, 1985.

[And] Anderson Cardoso, *{b} Behaviour Machine*, Available at: `http://www.behaviourmachine.com/`.

[Ans06] Anssi Klapuri, *Introduction to Music Transcription*, 2006.

[aub] aubergine, *Bloom efekt*, `https://forum.unity3d.com/threads/simple-bloom-post-process-shader.103377/`.

[aye16] ayerimv, *Ambient Birds. Song*, Available at: `https://www.freesound.org/people/ayerimv/sounds/346627/`, 2016.

[Ban] Banbury, *Sprites And Bones*, Available at: `https://github.com/Banbury/UnitySpritesAndBones`.

[Bri11] Brian J. Bonislawsky, *Jim Nightshade. Font*, Available at: `https://fonts.google.com/specimen/Jim+Nightshade?selection.family=Jim+Nightshade`, 2011.

[Cra] Craig A. Lindley and Charlotte C. Sennersten, *A cognitive framework for the analysis of game play.*

[Die00] Diemo Schwarz, *A system for data-driven concatenative sound synthesis*, 2000.

[Dig14] Digital Tentacle, *Circuits. Video game*, Available at: `http://store.steampowered.com/app/282760/Circuits/`, 2014.

[Dro16] Drool, *Thumper. Video game*, Available at: `http://store.steampowered.com/app/356400/Thumper/`, 2016.

[Dyl16] Dylan Fittere, *Audioshield. Video game*, Available at: https://www.soundboxing.co/, 2016.

[Eve13] Everett N McKay, *UI is Communication: How to Design Intuitive, User Centered Interfaces by Focusing on Effective Communication*, Morgan Kaufmann, 2013.

[FOA16] FOAM Entertainment, *Riff Racer. Video game*, Available at: http://store.steampowered.com/app/351990/Riff_Racer__Race_Your_Music/, 2016.

[Fre16] FreeStyleGames, *Guitar Hero Live. Video game*, Available at: https://www.guitarhero.com/buy, 2016.

[Ice16] Icetesy SPRL, *Melody's Escape. Video game*, Available at: http://store.steampowered.com/app/270210/Melodys_Escape/, 2016.

[Ins16] Insignio Labs, *Flute master. Video game*, Available at: http://store.steampowered.com/app/380100/Flute_Master/, 2016.

[Jak93] Jakob Nielsen, *Usability Engineering*, Elsevier, Academic Press, 1993.

[joe] joedanhol, *Parallax efekt*, Available at: https://github.com/joedanhol/Parallax2D.git.

[Kaa] Kaan-Yy, *Glow efekt*, http://answers.unity3d.com/questions/932093/make-2d-sprites-glow.html.

[Kat03] Katie Tekinbas and Eric Zimmerman, *Rules of Play: Game Design Fundamentals*, MIT Press, 2003.

[Kon98] Konami, *Dance Dance Revolution. Dance machine*, 1998.

[Kra08] KranX Productions, *Musaic Box. Video game*, Available at: http://store.steampowered.com/app/29130/Musaic_Box/, 2008.

[Luc90] LucasArts, *LOOM. Video game*, Available at: http://store.steampowered.com/app/32340/LOOM/, 1990.

[Luf08] Luftrum, *Grass Field. Song*, Available at: https://www.freesound.org/people/Luftrum/sounds/47989/, 2008.

[Luk] Lukáš Bařinka and Ing. Roman Berka, *Inverse Kinematics - Basic Method*, CTU Prague.

[Max16] Maxint LLC, *Soundboxing. Video game*, Available at: https://www.soundboxing.co/, 2016.

[Met] Metacritic, *Oficiální stránky službu Metacritic*, http://www.metacritic.com/.

[Mic] Microsoft, *Oficiální stránky společnosti microsoft*, https://www.microsoft.com/cs-cz/.

[Mor15] MorneDelport, *Forest. Song*, Available at: https://www.freesound.org/people/MorneDelport/sounds/326397/, 2015.

[Nad88] Nadia Magnenat-Thalmann and Richard Laperrière and Daniel Thalmann, *Joint-Dependent Local Deformations for Hand Animation and Object Grasping*, 1988.

[Nin98] Nintendo, *The Legend of Zelda: Ocarina of Time. Video game*, 1998.

[Para] Partners In Rhyme, *Key Chimes. Song efekt*, Available at: https://www.freesoundeffects.com/free-track/keychimes-428697/.

[Parb] _____, *Wind. Song efekt*, Available at: https://www.freesoundeffects.com/free-track/wind-428701/.

[pri] prime31, *Charactercontroller2d*, https://github.com/prime31/CharacterController2D.

[Puu16] Puuba, *The Metronomicon. Video game*, Available at: http://store.steampowered.com/app/412740/Audioshield/, 2016.

[Rea16] Reality Reflection, *Music Inside. Video game*, Available at: http://store.steampowered.com/app/520470/Music_Inside_A_VR_Rhythm_Game/, 2016.

[Ren32] Rensis Likert, *A Technique for the Measurement of Attitudes*, 1932.

[Rob00] Robert C. Martin, *Design Principles and Design Patterns*, 2000.

[Stea] Steam, *Oficiální stránky služby Steam*, http://store.steampowered.com/.

[Steb] SteamSpy, *Oficiální stránky služby SteamSpy*, https://steamspy.com/.

[Swi16] Swing Swing Submarine, *Seasons after Fall. Video game*, Available at: http://store.steampowered.com/app/366320/Seasons_after_Fall/, 2016.

[Tin16] Tinimations, *Klang. Video game*, Available at: http://store.steampowered.com/app/412660/Klang/, 2016.

[Unia] Unity3D, *Oficiální dokumentace Unity3D*, https://docs.unity3d.com/Manual/index.html.

[Unib] _____, *Unity- game engine*, https://unity3d.com/.

[Y. 05] Y. Shafranovich, *Common Format and MIME Type for Comma-Separated Values (CSV) Files*, Available at: http://www.ietf.org/rfc/rfc4180.txt#page-1, 2005, Online.

Všechny odkazy byly dostupné k 5. květnu 2017