Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Puzzle platformová počítačová hra

2022

Michal Tomášek

Vedoucí práce:
Mgr. Petr Osička, Ph.D.

Studijní obor: Aplikovaná informatika,
prezenční forma

Bibliografické údaje

Autor:	Michal Tomášek
Název práce:	Puzzle platformová počítačová hra
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2022
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Petr Osička, Ph.D.
Počet stran:	26
Přílohy:	1 flash disk
Jazyk práce:	český

Bibliograhic info

Author:	Michal Tomášek
Title:	A style for thesis
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2022
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Petr Osička, Ph.D.
Page count:	26
Supplements:	1 flash disk
Thesis language:	Czech

Anotace

Tématem práce je vytvoření počítačové hry, která spojuje prvky puzzle hry s platformovým typem hry. Čtenáře seznámím s enginem Unity, který jsem k implementaci hry použil. Poté představím implementovanou hru a popíšu podstatu herních elementů

Synopsis

The theme of the work is to create a computer game that combines elements of a puzzle game with a platform type game. I will introduce the reader to the Unity engine, which I used to implement the game. Then I will introduce the implemented game and describe the essence of game elements

Klíčová slova: počítačová hra; Unity;

Keywords: computer game; Unity

Děkuji Mgr. Petru Osičkovi, Ph.D. za vedení této práce a za poskytnutí cenných rad. Děkuji také mé rodině, přátelům a přítelkyni za nepřetržitou podporu.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam zdrojových kódů

1 Úvod

Videohry jsou jeden z druhů zábavy a v dnešní době dost rozšířeným. Stejně jako se technologie neustále vylepšují, tak se i herní průmysl zlepšuje. Hraní her je dostupné pro každého, ať už úplně zdarma a nebo za ne vždy malý poplatek. Takovou hru v dnešní době může i téměř každý vytvořit. Cílem této práce bylo právě vytvoření počítačové 2D hry pro jednoho hráče se zaměřením na logický puzzle systém spojený s platformovou skákačkou. Hra byla inspirována titulem It's a wrap od Chanko studios.

Obrázek 1: Ukázka hry It's a wrap. Zdroj: chankostudios.itch.io/its-a-wrap

1.1 Puzzle platformová hra

Platformová hra, zjednodušeně jako platformovka a někdy nazývaná jump 'n' run game, je pod-žánr akčních videoher, ve kterém je hlavním cílem pohybovat s postavou hráče mezi body v prostředí. Hra má plno různých levelů s různými překážkami a rozdílnou obtížností, většinou postupně gradující.[1] K pohybu je převážně využíváno skákání a běhání do stran, ale hra může obsahovat i šplhání, zhoupnutí se na provaze a jiné možnosti.

Puzzle hry, jak už vyplývá z názvu, jsou videohry, které kladou důraz na řešení hádanek. Typy hádanek mohou otestovat dovednosti při řešení problémů, včetně logiky, rozpoznávání vzorů, řešení sekvencí, prostorového vnímání a dokončování slov.[2]

Puzzle platformová hra spojuje prvky z obou žánrů a to tak, že její výzva je primárně odvozena od hádanek, zatímco k řízení hry je používána struktura platformovky.

2 Obecný popis hry

2.1 Levely

2.1.1 Cíl hry

Cílem hry je projít všechny levely. Hráč si v menu před začátkem hry zvolí ten, který chce hrát. Na úplném začátku je odemknutý pouze první z nich a následující levely se odemykají jejich postupným procházením. Jakmile jsou jednou odemknuty, tak si je hráč může kdykoliv znovu zahrát jejich zvolením v hlavním menu. Aby hráč mohl dokončit level, tak musí splnit dva úkoly. Prvním úkolem, je napolohovat překážky tak, aby byl schopný se dostat na druhý konec a druhým úkolem už pak je jen napolohované překážky překonat a dostat se tak k cíli.

2.1.2 Fáze levelů

Základní verze hry obsahuje osm levelů. Jedná se o statické levely, jejichž velikost odpovídá velikosti oknu hry a kamera se tím pádem nepohybuje. Všechny se navzájem liší a postupně nabývají na obtížnosti, která se dělí na několik fází. První fáze zahrnuje složitější umístění překážek, kterými hráč může polohovat. Postupně se stává více esenciálním pro hráče přepínat mezi stavy a nabízí se mu překážku využít třeba jako výtah. Druhá fáze zahrnuje posouvání těles, jeho tlačením svou postavou. Dále jsou v této fázi zahrnuty ostny, které mohou hráče zabít. Poslední fáze zahrnuje animace, které jsou použity v posledních levelech a mají za úkol jejich projití značně znepříjemnit. Hráči tak přibude problém, za jaký čas může level dokončit. Po určité chvíli se totiž animace dostane do bodu, kdy hráč může zemřít a nebo už nemůže dál pokračovat a je nucen level restartovat.

2.2 Stavy

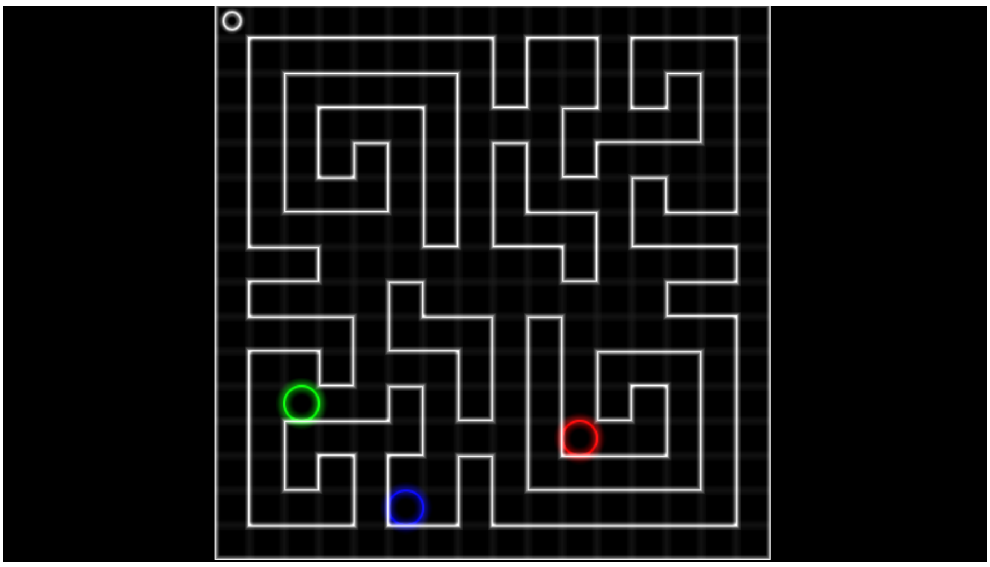
K tomu, aby hráč mohl dokončit level, tím pádem splnit potřebné dva úkoly napolohování překážek a jejích zdolání, má hráč k dispozici dva stavy, mezi kterými si může libovolně přepínat. Stav, kde je pohyb hráče zablokovaný a polohování překážek je k dispozici a stav, kde je to přesně naopak. Bez kombinace těchto dvou stavů se nedá žádný level splnit. Buď se dají využít jednoduchou kombinací napolohování a už jen následné přeskákání překážek, a nebo u pozdějších a těžších levelů se musí hráč chytře přepínat a po kousíčkách tak pomalu a postupně postupovat.

2.3 Nástrahy

Hra obsahuje dva druhy nástrah: ostny a animace. Jakmile se hráč dotkne ostnů, tak zemře a musí opakovat celý level. Instantní smrt místo určitého počtu životů a jeho ubíráním je zvolena z důvodu rovnováhy obtížnosti. Animace samotné hráče nezabijí. Ve hře jsou ale použité v kombinaci s ostny a to například v podobě bloku na němž jsou ostny umístěny, a který se posouvá, tudíž když se hráč nestihne přemístit do bezpečného místa, tak je zabit. Jako neoficiální nástraha, která hráče nezabije, ale znemožní mu dokončení levelu, pokud ho nerestartuje, se dá považovat místo, do kterého hráč zapadne a už z něho nemůže vylézt.

2.4 Grafika

Veškerá grafika hry je originální, ať už nakreslena v programu Adobe Photoshop a nebo ze základní nabídky Unity Enginu. Vycházel jsem a inspiroval jsem se grafikou hry Slidewayz the game, kterou jsem si upravil a předělal do své podoby. Grafika každého levelu je stejná, přehledná a působí minimalistickým dojmem. V prvním levelu má hráč v pozadí zobrazený návod základního ovládání hry. Překážky a lajny, díky nimž s nimi hráč polohuje, jsou navrhnuty různými barvami, aby bylo poznat jaká lajna patří k jaké překážce. Když se hráč pohybuje, zanechává za sebou trail stejného zbarvení.



Obrázek 2: Ukázka hry Slidewayz the game. Zdroj: simple-media.co.uk/games.htm

3 Použité technologie

3.1 Unity engine

3.1.1 Obecně o Unity

Unity engine byl poprvé vydán v roce 2005. Je to multiplatformní herní engine od společnosti Unity Technologies. První verze nabízela možnost vyvíjet hry pouze pro Mac OS, avšak od té doby byl rozvinut již o více než patnáct dalších platforem. Podporuje vývoje pro 2D i 3D, tvorbu skriptů v jazyce C# a grafického prostředí pro tvorbu. Možnost psaní kódu v jazyce C# společně s jednoduchostí enginu byly jedni z důvodů, proč jsem se rozhodl hru vytvářet právě v něm.

Je to velice oblíbený engine jak u začátečníků, tak i u pokročilých vývojářů. Nabízí 4 základní licence – Personal, Plus, Pro a Enterprise. Licence se od sebe drobně liší funkčností. Jediné za zmínku snad stojí *Industry specific soulution toolkits* dostupné při licensi Enterprise a nebo možnost nákupu přístupu k zdrojovým kódům unity dostupné licencí Pro a Enterprise. Je třeba říct, že důležitým aspektem při výběru licence jsou příjmy společnosti. Čím vyšší příjmy, tím dražší licenci je nezbytné používat.

3.1.2 Entity component system (ECS)

ECS je druh paradigma psaní kódu a jádrem Unity. Používá se většinou pro reprezentaci objektů herního světa při vývoji her. Programy se tvoří podle datově orientovaného vzoru. ECS zahrnuje entity složené z komponent dat se systémy, které fungují na komponentách entit.[3]. Z názvu vyplývá, že ECS má 3 základní části – Entity, komponenty a systémy. Entity představují objekty, které naplňují hru nebo program. V kontextu herního enginu je každý objekt entita. Komponenty uchovávají data spojená s entitami. Například každý objekt, který může utrpět poškození, může mít s jeho entitou spojenou komponentu zdraví.[3] Data jsou organizována podle samotných dat, nikoli podle entity. To je jeden z hlavních rozdílů mezi objektově orientovaným a datově orientovaným vzorem. Systémy jsou procesy, které působí na všechny entity s požadovanými komponenty a data komponenty transformují z jejich aktuálního stavu do dalšího stavu.

3.1.2.1 Klíčové komponenty

Vykreslovací engine zajišťuje renderování, neboli vykreslování reálně vypadajících obrazů. Používá k tomu data zpracované grafickou kartou. Tento proces Unity poskytuje buď v základním vestavěném řešení, nebo pomocí skriptů funkcionalitou *Scriptable rendering pipeline*. To umožní například vytvářet neuvěřitelně realistickou HD grafiku.

Fyzikální engine slouží k simulaci gravitace, pohybu, působení síly a dalších fyzikálních zákonů. Aby ale objekt mohl mohl být řízen fyzikálním enginem, musí k němu být připojena komponenta `Rigidbody (Rigidbody 2D)`. Mimo jiné komponenta zajišťuje i kolizi mezi objekty s kombinací s *Collidery*, které udávají hranice objektu. Zbytek je řešeno pomocí scriptů.

Skriptování Pomocí scriptů je řesena celá herní logika a všechny její mechaniky. Dá se pomocí nich například číst vstupy a výstupy uživatele, řídit kolize mezi objekty, uchovávat důležité hodnoty, určovat pravidla pro dané objekty atd. Unity pro psaní scriptů podporuje programovací jazyk CC#. Ve starších verzích byly podporovány i jazyky Boo a UnityScript, ale podpora těchto jazyků byla v roce 2017 ukončena.

Zvukový engine je spolu s grafickou a logickou částí nedílnou součástí projektu. Postup a základní funkcionalita je načtení zvukové stopy, její dekoprimace a následné přehrání. Pomocí pokročilejších funkcích, které Unity nabízí, lze taky například spojit více různých zdrojů zvuku do jednoho. Uživateli také nabízí kontrolu nad těmito zdroji. Mezi takové funkce patří produkce ozvěn, změna výšek tónů a podobně.

Další komponenty jsou například Multithreading a s tím související správa paměti. Síťování, animace, AI a další.

3.1.3 Skriptování

Chování herních objektů je řízeno komponenty, které jsou k nim připojeny. Ačkoliv komponenty dokáží pokrýt plno funkcí, tak většinou stejně musíte jít nad jejich rámec a implementovat tak své vlastní herní funkce. Unity umožňuje vytvořit si vlastní komponenty za pomocí skriptů, které jsou tvořeny přímo v Unity. Ten se pak může připojit k určitému hernímu objektu a všechno implementované ve skriptu se k němu vztahuje. Skripty jsou uloženy ve složce, která je aktuálně vybraná v okně s adresářem. Po dvojkliku na skript, se otevře v text editoru, základně ve Visual Studiu.

3.1.3.1 Struktura skriptu

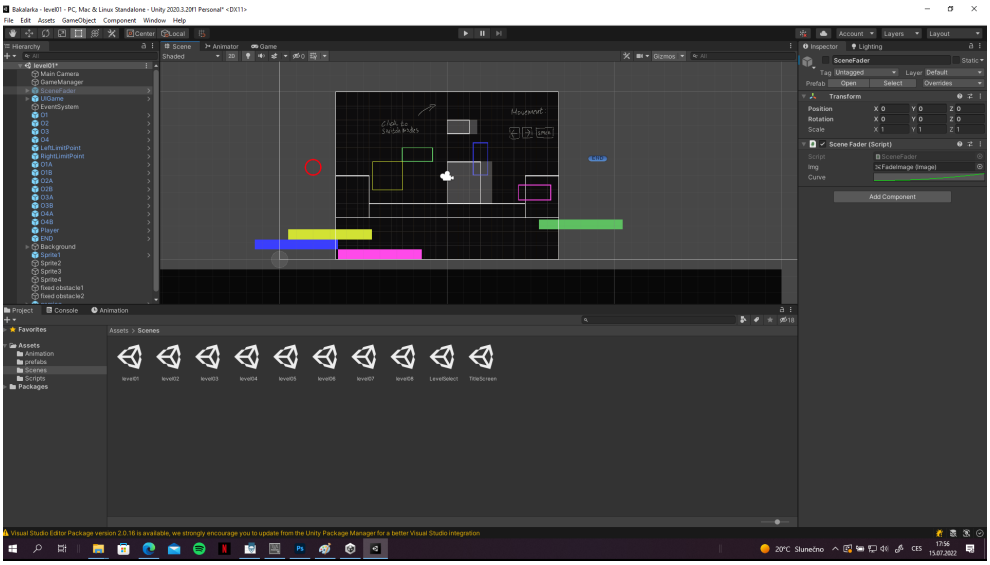
```
1  using UnityEngine;
2  using System.Collections;
3
4  public class NewBehaviourScript : MonoBehaviour {
5
6      // Use this for initialization
7      void Start () {
8
9      }
10
11     // Update is called once per frame
12     void Update () {
13
14     }
15 }
```

Zdrojový kód 1: Základní struktura skriptu

Jméno třídy se shoduje s názvem skriptu a tím, že dědí z vestavěné třídy `Monobehaviour`, tak je propojen s vnitřním fungováním Unity. Třída hned zpočátku obsahuje dvě metody `Start` a `Update`. Všechno co je implementované metodou `Start` se provede před začátkem hry, takže i před prvním zavoláním metody `Update`. Touto metodou implementujeme vše co bude zpracovávat aktualizaci snímku pro daný herní objekt. Metodu `Update` lze nahradit metodou `FixedUpdate`. Ta oproti `Update`, která se provede jednou za snímek, se může provést jednou, vůbec a nebo vícekrát za snímek v závislosti na tom, kolik fyzikálních snímků za sekundu je nastaveno v nastavení času a jak rychlá/pomalá je snímková frekvence. Z tohoto důvodu by se `FixedUpdate` měla používat při aplikaci sil, točivého momentu nebo jiných funkcí souvisejících s fyzikou, protože bude provedena přesně synchronizovaně se samotným fyzikálním enginem. Zatímco `Update` se může lišit v závislosti na fyzikálním enginu.[4]

3.1.4 Práce v Unity

Unity hned po spuštění Unity Hubu, což je správce všech Unity projektů, jenž si uživatel vytvoří, nabízí několik šablon na zvolení. Liší se od sebe zejména v použité *render pipelině* a jejím nastavení. Například pro vysokou úroveň kvality grafiky je určena šablona HDRP (High definition render pipeline), avšak pro její použití musí být již výkonné zařízení. Pokud však není třeba co nejvyšší realističnost, tak vhodnou volbou je výkonnostně optimalizovaná šablona URP (Universal render pipeline). Samotná práce je poměrně jednoduchá. Rozhrání programu tvoří pár oken, ve kterých se vše odehrává. Okno s hierarchií objektů dané scény. Okno s adresářem. Okno, kde uživatel přidává, odstraňuje a upravuje komponenty označeného objektu a samozřejmě okno s náhledem.



Obrázek 3: Rozhraní herního enginu Unity

3.2 Další použité technologie

C# je vysokoúrovňový objektově orientovaný, statický programovací jazyk vyvinutý v roce 2000 firmou Microsoft. Vychází z kombinace jazyků C++ a Java a je tedy nepřímým potomkem jazyka C. Lze jej využít k tvorbě databázových programů, softwaru, webových aplikací, stránek, služeb a plno dalšího. Nejnovější verze jazyka je C# 10, který je však podporován pouze v .NET 6 a novějších verzích.

Vývojové prostředí Visual Studio. Všechny skripta celého projektu byly vytvořeny v prostředí Visual Studio od společnosti Microsoft. Visual studio nabízí tři různé verze ke stažení – Community, Proffesional a Enterprise, z toho Community je bezplatná verze. K vývoji byla použita právě tahle verze a to verze Community 2019.

Adobe Photoshop poprvé vyšlo v roce 1990 pro Mac OS. Jedná se o bitmapový grafický editor pro tvorbu a úpravy bitmapové grafiky od firmy Adobe Systems. V něm proběhl grafický vývoj celého projektu. Byla použitá verze 13.0, neboli CS6.

4 Programátorská dokumentace

Projekt se skládá z několika částí

- Vestavěné třídy v Unity
- Herní logika
- Hráč a překážky
- Animace
- Grafika
- Menu

4.1 Vestavěné třídy v Unity

V úvodu je třeba zmínit pár důležitých a v této práci užívaných vestavěných tříd.

GameObject je třída, která reprezentuje všechno co existuje v dané třídě, tj. všechny herní objekty. Poskytuje různé metody, díky kterým se dá s objektem pracovat v kódu. Například jen jeho nalezení pomocí tagu nebo jména objektu v Unity, jeho zapnutí nebo vypnutí a další.

Transform je třída, která poskytuje plno možností v rámci změny pozice, rotace a změny velikosti herního objektu.

Vectors popisují základní vlastnosti, jako například poloha postavy, rychlost pohybu a podobně. Třída `Vector2` rezprezentuje 2D vektory a body, třída `Vector3` 3D vektory a body a třída `Vektor4` čtyř dimenzionální vektory.

Time třída posyktuje práci s hodnotami souvisejícími s časem. Zejména nejdůležitější vlastností této třídy je `deltaTime`, která vrací množství času v sekundách, které uplynulo od posledního dokončeného snímku. Tato hodnota se liší v závislosti na počtu snímků za sekundu (FPS), s jakou hra nebo aplikace běží.[5]

Mathf třída poskytuje běžné matematické funkce

4.2 Herní logika

Každá třída má svůj vlastní Skript. Ten nejdůležitější, který má třídu, jež obsahuje všechny klíčové metody se jmenuje `GameManager`. Dále budu v kapitole psát o skriptech jako o třídách, protože název třídy odpovídá názvu skriptu. Třída `GameManager` se například stará o start hry, dokončení levelu, načtení dalšího levelu atd. Řeší i to nejdůležitější, a to je přepínání mezi dvěma stavy `puzzling`

a gameOn, implementované pomocí struktury enum. Stav puzzling je počáteční stav každého levelu. Přepínání mezi nimi pak jde libovolně pomocí tlačítka v horní části obrazovky, který zavolá metodu StartGame ve třídě GameManager. Ta podle toho jaký stav je právě aktivní zavolá metodu příslušnou druhému stavu na jeho zapnutí. Třída obsahuje plno atributů ve kterých jsou uloženy komponenty z Unity. Například pro načtení dalšího levelu má třída atribut nextLevel, ve kterém má uloženou scénu s dalším levelem, a zároveň atribut levelToUnlock, ve kterém má uloženo číslo následujícího levelu, aby se odemknul ve scéně pro volbu levelu. GameManager je zároveň herní objekt, který nejde vidět, ale musí být v každé scéně levelu. Na tento objekt je připojen Skript stejného názvu a tudíž je brán jako komponenta. Veškeré public atributy třídy daného skriptu jdou vidět přímo v Unity, tudíž jejich hodnota se dá upravit přímo v něm a nemusí se již zasahovat do kódu.

4.3 Hráč a překážky

Chování hráče je implementováno třídou PlayerMovement. Třída obsahuje atributy movementSpeed pro nastavení rychlosti hráče a jumpForce pro výšku skoku hráče. Zbytek fyziky je řešen pomocí komponenty v Unity enginu Rigidbody2D. Třída potom řeší jen pohyb hráče do stran a jeho výskok. To řeší jediná metoda ve třídě – FixedUpdate, která na svém začátku kontroluje, jaký stav je právě aktivní. Ve hře jsou dva druhy překážek – pohyblivé a fixní.

4.3.1 Pohyblivé překážky

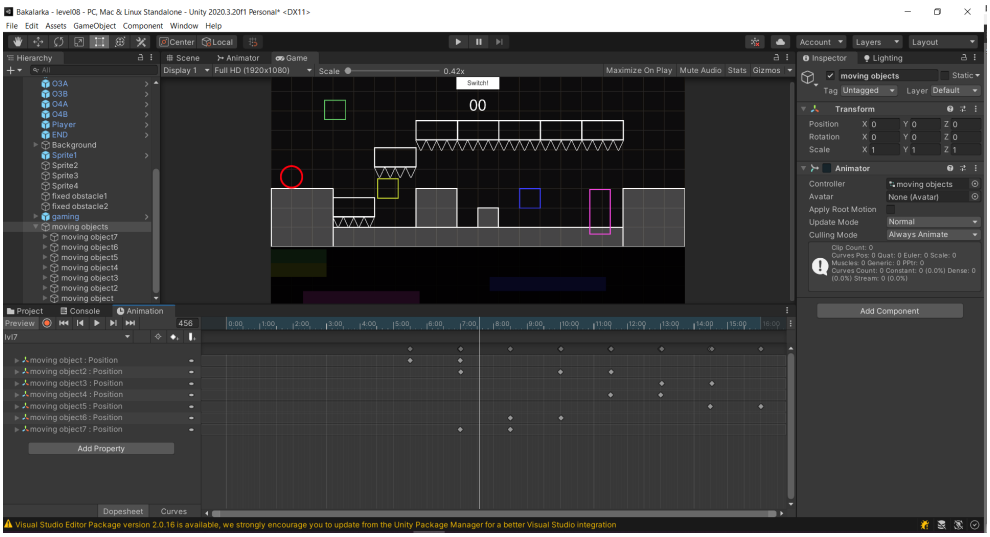
Jsou implementované třídou ObstacleMovement. Lišty, kterými posouváme, abychom posunuli překážkou, jsou implementované třídou LinerMovement. Každá překážka má své 2 různé body, mezi kterými se může posouvat. Ty jsou implementovány jako objekty v Unity a jejich připojení do atributů třídy ObstacleMovement. Pro všechny lišty také existují 2 takové body, ale jsou pro všechny lišty stejné. Ty jsou připojeny do atributů třídy LinerMovement. Pohyb je implementován procentuálně. 100 % dráhy lišty se rovná 100 % dráhy překážky. Na začátku hry se spočítá vzdálenost bodů, poté se spočítá procentuální pozice lišty trojčlenkou v rámci metody GetPercentPosition ve třídě LinerMovement a nastaví se stejná pozice překážky vůči její dráze. To je implementované metodou Positioning ve třídě ObstacleMovement. Ta se potom volá při každém pohybu.

4.3.2 Fixní překážky

Jedná se o jednu z nástrah. Jsou implementované třídou Spikes, která má pouze metodu řešící co se stane při střetu hráče s nástrahou. Zavolá se metoda Toggle, ze třídy PauseMenu, s parametrem true. Tato metoda podle parametru, který jí byl předán, nastaví obrazovku pro pause menu nebo smrt aktivní.

4.4 Animace

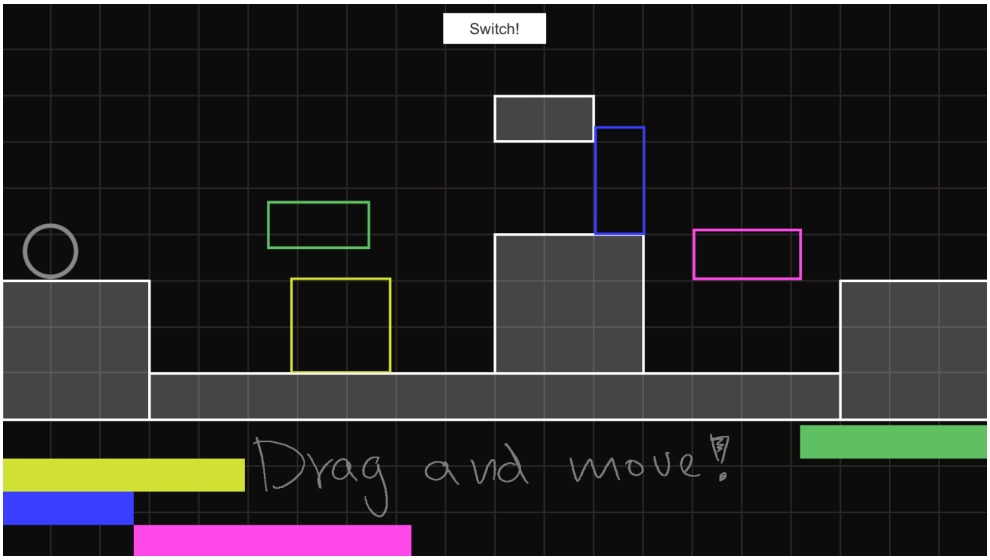
Animace jsou řešeny čistě prostřednictvím Unity enginu a jeho rozhraní pro tvorbu animací. Na časové ose si navolím co potřebuji, aby se v daný moment stalo, ať už posunutí předmětu, jeho zmizení a plno dalšího. Nástraha, která má animaci má na sobě buďto fixní překážku, nebo je přímo implementovaná stejnou třídou, a to třídou `Spikes`.



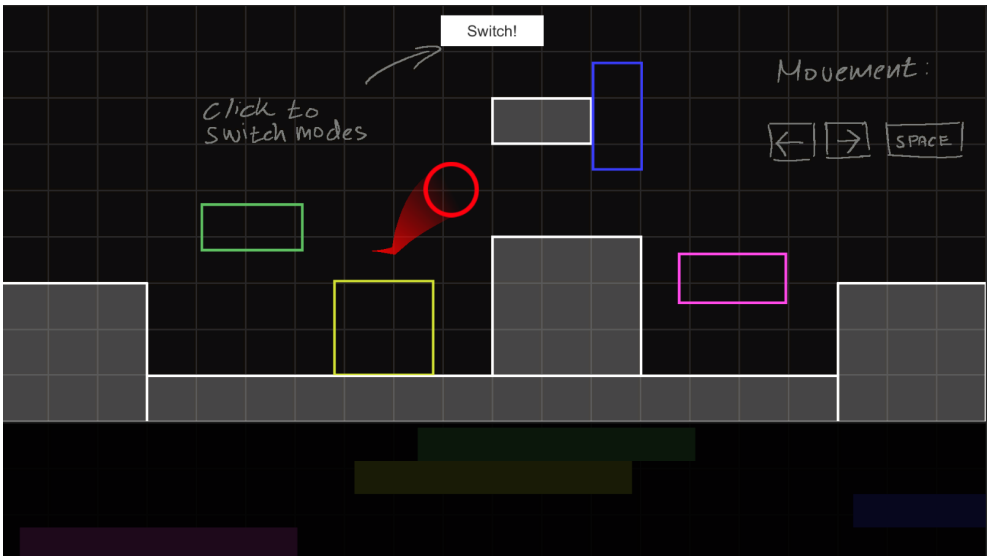
Obrázek 4: Ukázka tvoření animací v Unity

4.5 Grafika

Grafika celé hry má za cíl působit minimalistickým dojmem. Návod pro ovládání hry, jsem nenápadně vložil do pozadí prvního levelu. Když se hráč nachází ve stavu `puzzling`, tudíž se nemůže pohybovat se svou postavou, tak barva postavy hráče je zašedlá. V momentě, když hráč přepne do stavu `gameOn`, tak se postavě hráči změní barva na červenou a ve stejné barvě má za sebou při pohybu trail, pro dodání živosti hry. Zároveň prostor s lištami pro polohování překážek se zatemní, aby bylo lépe poznat, že již nadále hráč nemůže překážky polohovat, pokud se nepřepne do druhého stavu. Mezi jednotlivými levely, spuštění levelu z menu, vrácení se do menu a překlikávání obrazovek v menu jsou implementovány přechody třídou `SceneFader`, pro jemnost přechodu mezi scénami. Třída obsahuje 2 metody `FadeIn` a `FadeOut`. První metoda se stará o přechod při přepnutí na novou scénu a to tak, že alfa kanál čistě tmavého obrázku se postupně zmenšuje a obrázek tím mizí. Druhá metoda naopak alfa kanál toho stejného obrázku zvyšuje a scéna postupně tmavne a mizí.



Obrázek 5: Vzhled prvního levelu v puzzling stavu

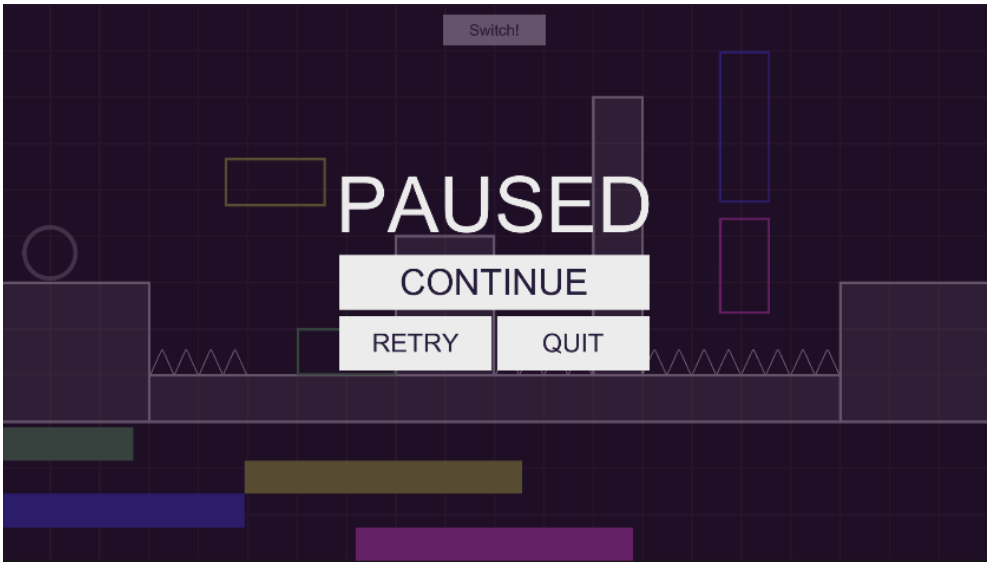


Obrázek 6: Vzhled prvního levelu v gameOn stavu

4.6 Menu

Po zapnutí hry se objeví titulní obrazovka s názvem hry a možností hrát nebo hru vypnout. Obě tlačítka jsou implementované ve třídě `MainManu` metodami `Play` a `Quit`. Jakmile hráč klikne na play, přesune se do obrazovky s výběrem levelu. Výběr je implementován třídou `LevelSelector`, kde v metodě `Start` si v proměnné `levelReached` hra pamatuje jaké všechny levely už hráč odehrál. Zbytek dlaždic levelů jsou zašedlé a neaktivní. Po zvolení levelu je metodou `Select` hráč přesměrován na scénu daného levelu. Ve hře je implementováno i pause menu třídou `PauseMenu`. Dostat se do něj může hráč pomocí dvou kláves

„Esc“ nebo „P“. Zavolá se metoda `Toggle`, tím se aktivuje scéna pro pause menu, čas hry se stopne a s tím i všechno ostatní. Menu obsahuje 3 tlačítka continue, retry a quit. Když hráč klikne na continue, znovu se zavolá metoda `Toggle`, scéna se deaktivuje a čas pokračuje. Při kliknutí na retry se zavolá metoda `Retry`, která načte aktivní scénu a level tím restartuje. Při kliknutí na quit se zavolá metoda `Menu`, která načte scénu s titulní stranou. Obě metody zároveň volají metodu `Toggle` pro deaktivování pause menu a spuštění času hry. Třída se zároveň stará o to, když hráč zemře a to tak, že obsahuje dvě proměnné `uiPause` a `uiDeath`, ve kterých jsou uložené scény s pause menu a se smrtí. Podle toho, jestli je předán parametr `true` nebo `false` metodě `Toggle`, tak se aktivuje scéna pro pause menu, nebo smrt. Na scéně pro smrt, jsou dvě tlačítka retry a quit, které využívají stejné dvě metody jako tlačítka v pause menu.



Obrázek 7: Pause menu

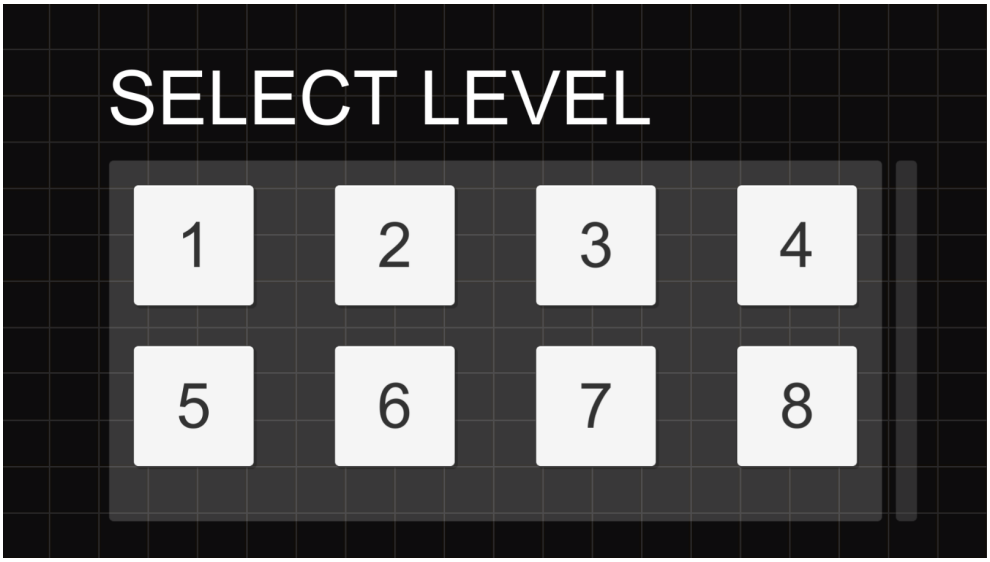
5 Uživatelská příručka

V této kapitole uživatele seznámím s hlavním menu, ovládáním hry a stručně popíšu základ hry.

5.1 Hlavní menu

Po spuštění hry se hráč objeví v titulní obrazovce, která slouží zároveň jako hlavní menu. Hráč má 2 možnosti, a to buď kliknout na tlačítko play, aby mohl hrát a nebo na tlačítko quit, pro vypnutí hry. Pro navigaci v menu lze použít pouze myš. Kliknutí na tlačítko play hráče přesune k nabídce levelů. Hráč má možnost si vybrat jeden z odemknutých levelů, který chce hrát. Pro návrat na titulní obrazovku slouží tlačítko back. Grafika menu a celé hry byla zvolena s cílem udělat hru přehlednou a uživatelsky atraktivní.

Obrázek 8: Hlavní menu



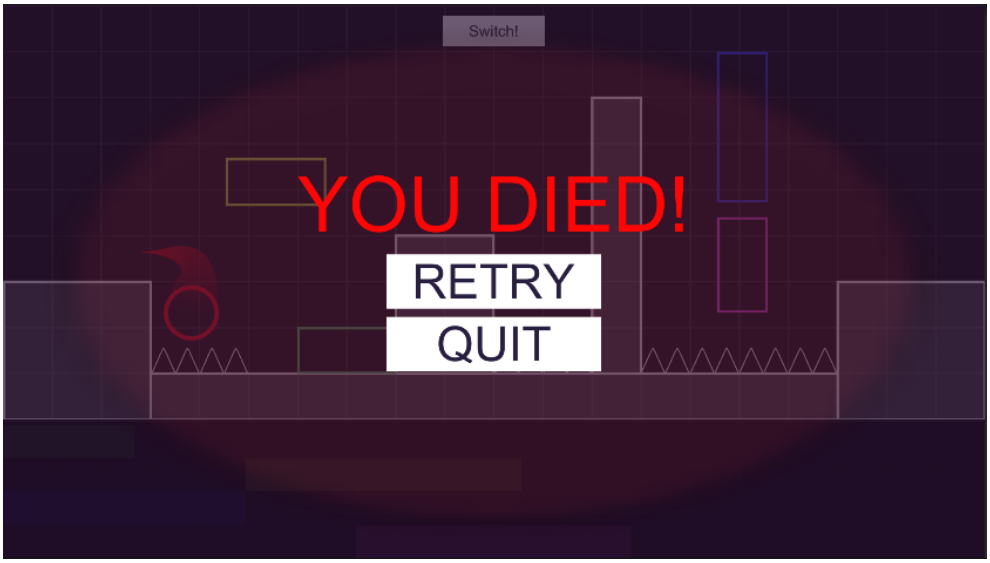
Obrázek 9: Obrazovka s výběrem levelů

5.2 Ovládání

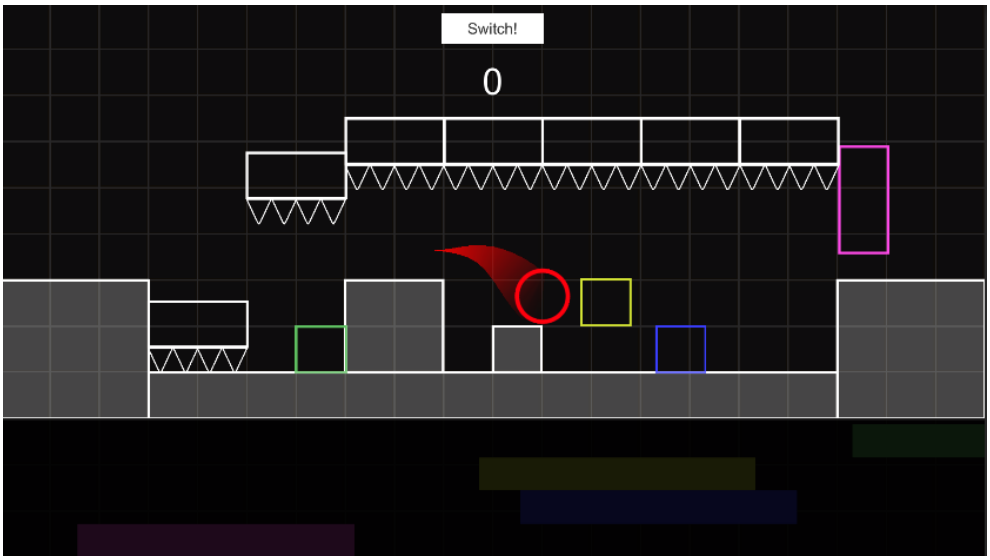
Hra se ovládá pomocí klávesnice a myši. Myš slouží pro navigaci v menu a ovládání v režimu `puzzling`, kdy držením levého tlačítka a táhnutí myši do strany hráč polohuje překážky. Poté v režimu `gameOn` se hra ovládá pouze pomocí klávesnice. Postava hráče se ovládá šipkami, nebo klávesou „A“ a „D“ pro pohyb a klávesou „Space“ pro výskok. Klávesou „Esc“ nebo „P“ hráč zastaví hru a objeví se pause menu. Návod k ovládání hry je součásti pozadí prvního levelu.

5.3 Hra

Cílem hry je projít všech 8 levelů. Hráč má pouze jeden život, pokud zemře, tak má možnost level restartovat. Ke splnění levelu musí hráč napolohovat překážky a ty následně překonat pomocí výskoku. V pozdějších levelech se přidávají i animace, které přidávají na obtížnosti hry. Level s animací má na horní části obrazovky pod tlačítkem na přepínaní stavů časovač. Jakmile čas vyprší, animace se spustí. Hráč má tedy omezený čas na daný level a musí se dostat do určitého bodu, než ho animace odřízne. Odejít z levelu nebo vypnout hru může hráč kdykoliv pomocí pause menu a tlačítka quit. Levely se postupně odemykají jeho projítím a odemknuté levely si hráč může zahrát znovu.



Obrázek 10: Obrazovka, když hráč zemře



Obrázek 11: Ukázka posledního levelu

```
1  void Positioning(float percent)
2     {
3        float offset;
4        if (distanceX == 0)
5        {
6           offset = Mathf.Min(pointA.position.y, pointB.position.y);
7           transform.position = new Vector2(pointA.position.x, (
               distanceY * percent / 100) + offset);
8        }
9        else if (distanceY == 0)
10       {
11          offset = Mathf.Min(pointA.position.x, pointB.position.x);
12          transform.position = new Vector2((distanceX * percent /
               100) + offset, pointA.position.y);
13       }
14       else
15       {
16          float offsetX, offsetY;
17          offsetY = Mathf.Min(pointA.position.y, pointB.position.y);
18
19          if (pointA.position.x < pointB.position.x && pointA.
               position.y > pointB.position.y)
20          {
21             offsetX = Mathf.Max(pointA.position.x, pointB.position.x
                  );
22             transform.position = new Vector2(((-distanceX) * percent
                   / 100) + offsetX, (distanceY * percent / 100) +
                  offsetY);
23          }
24          else
25          {
26             offsetX = Mathf.Min(pointA.position.x, pointB.position.x
                  );
27             transform.position = new Vector2((distanceX * percent /
                  100) + offsetX, (distanceY * percent / 100) +
                  offsetY);
28          }
29       }
30    }
```

Zdrojový kód 2: Implementace pohybu překážky

Závěr

Výsledkem bakalářské práce je 2D puzzle-platformová počítačová hra pro jednoho hráče obsahující 8 levelů. Překážky každým levelem představují větší výzvu a hra tím plynule nabývá na obtížnosti. Ke hře se dá kdykoliv vrátit a pokračovat levelem, kterým hráč naposledy skončil. Hra má určitě potenciál pro vylepšení, ale s finálním výsledkem jsem velmi spokojený. Je například možné přidat více levelů, vícero druhů překážek a to nejen pasivních, ale i aktivních ve formě střelby z děla a podobně. Hra je naprogramovaná v jazyce C#. Co se týče enginu Unity, tak navzdory tomu, že jsem s ním už měl pár zkušeností i před tím, tak jsem se ledacos naučil při této práci. Pracovalo se mi s ním pohodlně a efektivně. Na internetu se k němu nachází plno návodů a videí, který dokážou naučit základní funkcionalitu za velmi krátký čas. Celkově je Unity engine velmi efektivní a široce dostupný pro vývoj her.

Conclusions

The result of my bachelor's thesis is a 2D puzzle-platform computer game for one player containing 8 levels. The Obstacles represents greater challenge with each level, allowing the game to steadily increase in difficulty. The player can return to the game at any time and continue with the level where he last finished. The game definitely has potential for improvement, but I'm very satisfied with the final result. For example, it is possible to add more levels, more types of obstacles, not only passive, but also active in the form of cannon fire etc. The game is programmed using C# programming language. As for the Unity engine, despite the fact that I already had some experience using it, I learned a lot from this thesis. I found it comfortable and efficient to work with it. There are plenty of instructions and videos on the internet, which can teach the basic functionality of it in a very short time. Overall, the Unity engine is very efficient and widely available for game development.

A Obsah přiloženého datového média

`bin/`
: Všechny potřebné soubory pro spuštění [**a**] běh hry. Hra lze spustit otevřením aplikace *MakeItJump.exe*.

`doc/`
: Text práce ve formátu PDF, včetně všech příloh nezbytné pro vygenerování dokumentu.

`src/`
: Kompletní zdrojové texty programu

Citace

[1] WIKIPEDIA (ed.). *Platform game* [online]. [cit. 2022-6-17]. Dostupný z: ⟨https://en.wikipedia.org/wiki/Platform_game⟩.

[2] WIKIPEDIA (ed.). *Platform game* [online]. [cit. 2022-6-18]. Dostupný z: ⟨https://en.wikipedia.org/wiki/Puzzle_video_game⟩.

[3] WIKIPEDIA (ed.). *Entity component system* [online]. [cit. 2022-6-10]. Dostupný z: ⟨https://en.wikipedia.org/wiki/Entity_component_system⟩.

[4] ANSWERS, Unity (ed.). *What's the difference between Update and FixedUpdate? When are they called?* [online]. [cit. 2010-2-2]. Dostupný z: ⟨https://answers.unity.com/questions/10993/whats-the-difference-between-update-and-fixedupdat.html⟩.

[5] DOCUMENTATION, Unity (ed.). *Important Classes - Time* [online]. [cit. 2022-7-17]. Dostupný z: ⟨https://docs.unity3d.com/Manual/TimeFrameManagement.html⟩.