DELTA – Střední škola informatiky a ekonomie, Základní škola a Mateřská škola s.r.o.

Ke Kamenci 151, PARDUBICE

MATURITNÍ PROJEKT

2D BOJOVÁ HRA

Příjmení, jméno: Štorek Daniel
Třída: 4.A
Studijní obor: Informační technologie 18-20-M/01
Školní rok: 2021/2022

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení:	*Daniel Štorek*
Školní rok:	*2021/2022*
Třída:	*4. A*
Obor:	*Informační technologie 18-20-M/01*
Téma práce:	*2D bojová hra*
Vedoucí práce:	*Mgr. Jan Mottl*

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Cílem projektu bude vytvoření 2D hry jako aplikace pro PC. Žánrově se bude jednat o bojovou hru pro jednoho hráče s možností omezené kompetitivní hry v hráčích dvou na jednom PC. Hra bude obsahovat minimálně 2 postavy, z nichž každá bude mít vlastní animace a schopnosti, které bude moci proti soupeři využívat.

Hra bude využívat prvků achievementů, které si hráč bude moct získat a díky nim bude moct používat nové postavy.

Hra pro dva hráče bude obsahovat pouze možnost kompetitivního souboje mezi dvěma postavami.

Použitý programovací jazyk: C#

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

Září – studium programovacího jazyka a SW pro vytvoření hry, základní mechaniky hry (pohyb, útoky), první mapa, první postava se svými schopnostmi

Listopad – Hlavní menu, Multiplayer, nové postavy a mapy

Prosinec – vyladění grafické stránky hry, testování hry

Únor – možné dodělání singleplayer kampaně s bossy

Březen – Finální ladění nalezených chyb během testování; Sepisování dokumentace

Prohlášení

Prohlašuji, že jsem maturitní projekt vypracoval samostatně, výhradně s použitím uvedené literatury.

V Pardubicích dne .....

Podpis

Poděkování

Děkuji Mgr. Janu Mottlovi za studijní materiál a odborné vedení při zpracovávání maturitního projektu. Dále bych rád poděkoval rodině a přátelům za morální podporu.

Anotace

Tato dokumentace popisuje vývoj 2D bojové hry. Jsou zde popsány všechny nástroje a technologie využité k tvorbě hry. Součástí dokumentace je také popis celé hry a popis algoritmů.

Klíčová slova

C#, WPF, 2D, hra, arkáda, bojová hra, Visual Studio

Annotation

This documentation describes a process of creating a 2D fighting game. All tools and technology used to create this game are described there. A part of the documentation is also a description of the whole game and a description of algorithms.

Keywords

C#, WPF, 2D, game, arcade game, fighting game, Visual Studio

Obsah

1. Úvod

Již odmala jsem měl obrovskou zálibu ve videohrách, hrál jsem všechny žánry a nesetkal jsem se s žádným, který by mě přímo nebavil. Postupem času jsem začal přemýšlet o tom, jak vlastně hry vznikají a začal jsem snít o tom, že si jednou vytvořím svoji vlastní hru podle mých představ. Při výběru maturitního projektu jsem si řekl, že je ten správný čas zkusit si nějakou hru naprogramovat.

Cílem projektu je vytvořit rychlou 2D bojovou hru vyžadující rychlé reakce. Svým stylem se podobá hře vydanou společností Ubisoft s názvem Brawlhalla.[1]

Hra obsahuje rychlou hru pro jednoho hráče, režim věže a hru pro dva hráče. Zápasí se vždy ve stylu jeden na jednoho a každý zápas se hraje na 2 vítězná kola. Režim věže není stavěn pouze na jediný zápas, musíte se dostat přes 4 zápasy s různými soupeři až k finálnímu bossovi celé hry, který je těžší než zbytek postav díky svým schopnostem.

Za zmínku dále stojí i systém achievementů. Ze začátku začínáte pouze s jednou hratelnou postavou, další postavy si můžete odemknout sbíráním daných achievementů. Výjimka platí ve hře pro dva hráče, kde si můžete vybrat i ze zamčených postav.

Základem hry obou režimů pro jednoho hráče je algoritmus počítače, který je naprogramován zvlášť pro každou postavu, aby se dokázal přizpůsobit jejím schopnostem a dokázal je využít.

2. Využité technologie

2.1. C#

C# je moderní, objektově orientovaný programovací jazyk vyvinutý společností Microsoft. Patří do rodiny jazyků C a mezi programovacími jazyky se mu nejvíce podobají C, C++, Java a JavaScript.[2]

Jak již bylo zmíněno výše, C# je objektově orientovaný jazyk, tudíž programátorům velice usnadní práci při vývoji robustních aplikací. Ti mohou vytvářet různé objekty, které lze následně využít ve více částích programu.[3] Objekty také umožňují snazší orientaci v kódu.

2.1.1. .NET a WPF

Základní vývojářskou platformou C# je .NET umožňující vývoj webových aplikací, mobilních aplikací, desktopových aplikací, videoher a IoT.[4] Součástí .NET je i rozhraní uživatelského prostředí WPF, pomocí kterého je vykreslována i hra popisovaná v této dokumentaci. WPF využívá jazyk XAML (Extensible Application Markup Language) a

dokáže naplno využít výhody moderního grafického hardwaru při vykreslování 2D grafiky, 3D grafiky a animací.[5]

2.2. XML

XML (eXtensible Markup Language) je velmi používaný formát pro ukládání dat, svým zápisem se podobá jazyku HTML. To zaručuje, že na rozdíl od binárního zápisu soubory XML dokáže přečíst i člověk, přesvědčit se můžete na ukázce níže.[6]

Ukázka zápisu XML souboru:

```
<clenove>
  <clen jmeno="Karel" vek="23" />
  <clen jmeno="Milan" vek="25" />
</clenove> [6]
```

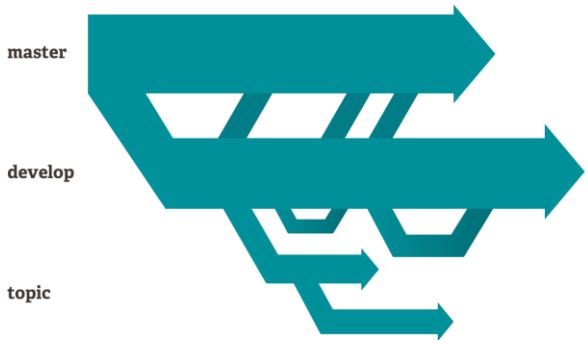
3. Využité nástroje

3.1. Visual Studio 2022

Visual Studio 2022 je integrované vývojové prostředí (IDE) od Microsoftu, to je velice důležitým prvkem při vývoji jakéhokoliv softwaru. Visual Studio 2022 podporuje mnoho programovacích jazyků a umožňuje nám vytvářet, upravovat a ladit kód. Vytváření a úpravu kódu nám velmi usnadňuje funkce IntelliSense, která zobrazuje informace o kódu přímo v editoru, dokonce za nás zvládne dokončit i část kódu. Integrované nástroje ladění nám umožní projíždět kód příkaz po příkazu, díky tomu můžeme přesně zjistit, co se v našem programu zrovna děje. Další funkce mohou být přidány pomocí rozšíření.[7]

3.2. Git

Git je open-source distribuovaný systém správy verzí.[8] Jedna z jeho hlavních předností je možnost větvení (branching), to umožňuje mít několik lokálních větví najednou, aniž by na sobě nějakým způsobem závisely, takže můžeme experimentovat s kódem beze strachu, že bychom ho porušili v jiné větvi.[9] Příklad větvení můžete vidět na obrázku 1.



Obrázek 1 Příklad větvění v Gitu

3.3. GIMP

GIMP je open-source nástroj pro úpravu obrázků. Podporuje Windows, Mac OS X i Linux (na většině Linuxových distribucí je dokonce nainstalován automaticky), tudíž ho můžeme využít na všech hlavních desktopových operačních systémech. Svým vysokým počtem možností, které s obrázky zvládne, se vyrovná i placeným editorům. Pro ty, kterým přeci jen v programu nějaké funkce chybí, GIMP nabízí podporu plug-inů a rozšíření, nechybí zde ani podpora skriptů, takže se nemusíme zdržovat se zdlouhavými činnostmi, které jdou vykonat spuštěním jediného skriptu.[10]

4. Popis částí hry

4.1. Hlavní menu

Hlavní menu je prvním oknem, které se otevře po spuštění hry. Je takovým rozcestníkem, ve kterém si uživatel vybere následující akci, na obrázku 2 můžete vidět všechny dostupné možnosti. Po rozkliknutí tlačítka „Hra pro jednoho hráče“ si hráč ještě může vybrat, zda chce hrát pouze jeden rychlý zápas, nebo chce hrát průchod věží. Kliknutím na ozubená kola se otevře okno s nastavením.

Obrázek 2 Ukázka hlavního menu

4.2. Nastavení

Nastavení nabízí jednoduché rozhraní pro rozvržení kláves, kterými hráč ovládá postavu. Samozřejmostí je také možnost obnovit výchozí rozvržení kláves.

4.3. Úspěchy a žebříček

V této obrazovce se hráč může podívat, které achievementy již získal a které mu stále chybí. Popis achievementu prozradí, která postava se jeho získáním odemkne. Nechybí ani žebříček, který je seřazen podle dosaženého skóre při souboji.

4.4. Výběr postav a mapy

Toto okno se zobrazí po výběru herního režimu. Hráč zde může zvolit svoji a soupeřovu postavu, v případě režimu průchodu věží si hráč volí pouze svoji postavu společně s obtížností hry.

Pokud hráč nehraje průchod věží, má na výběr také mapu, na které se souboj uskuteční.

4.5. Seznam postav

Hra obsahuje celkem 5 postav, 4 z nich jsou hratelné a poslední z nich slouží jako finální boss ve věži. Každá postava má o něco jiný styl hry, který se odvíjí od toho, zda chce hráč útočit na dálku, nebo na blízko.

4.5.1. Lukostřelec

Lukostřelec je jediná postava, kterou má hráč k dispozici hned ze začátku hry. Postava disponuje nižší rychlostí, ale zato může vystřelit na nepřítele z dálky šípy a její druhý útok je TNT, které po krátkém časovém úseku vybouchne a zraní postavu, která u TNT stojí (může zranit i lukostřelce, který TNT použil).

Obrázek 3 Lukostřelec

4.5.2. Válečník

Válečník se hráči odemkne po získání úspěchu „Začátečník“ (hráč musí odehrát jeden zápas v režimu pro jednoho hráče). Tato postava je velmi rychlá a primárně útočí na krátkou vzdálenost, k dostižení soupeře jí pomůže právě již zmíněná rychlost a může na soupeře vystřelit tornádo, které na chvíli zmrazí soupeře na místě (nemůže se hnout, ale útočit může).

Obrázek 4 Válečník

4.5.3. Kouzelník

Kouzelník je odemčen po získání úspěchu „Co je to za věž?“ (hráč si musí vyzkoušet režim věže). Kouzelník je stejně jako lukostřelec postava zaměřená na útok z vyšší vzdálenosti, jeho první útok vystřeluje magické koule a ty udělují větší poškození než lukostřelcovy šípy, ale střílí pomaleji a útok stojí více energie, takže si na ni musí hráč dávat pozor. Jeho druhý útok je nahrazen dočasným ochranným štítem, který redukuje obdržené poškození na minimum.

Obrázek 5 Kouzelník

4.5.4. Minotaur

Obrázek 6 Minotaur

Za minotaura si hráč může zahrát po získání úspěchu „Krásný pohled ze shora“ (hráč dokončí celý režim věže). Minotaur je postava útočící na blízko, jeho první útok je shodný s útokem válečníka a jeho druhý útok je rozběhnutí se na soupeře. Jeho rychlost je při tomto útoku více než 3x větší, pokud při něm navíc trefí soupeře, způsobí mu obrovské poškození.

4.5.5. Ork

Obrázek 7 Ork

Ork je jedinou nehratelnou postavou ve hře a slouží jako finální soupeř ve věži, také má jako jediná postava více než 2 útoky. Jeho primární útok je stejný jako útok minotaura a válečníka, druhým vystřelí hák, kterým hráče přitáhne k sobě. Při celém souboji na hráče hází TNT a střílí z kraje pole magické koule (stejně jako primární útok kouzelníka).

4.6. Zápas

Na obrázku 8 je vyobrazeno nejdůležitější okno, zde již probíhá samotný souboj.

Obrázek 8 Ukázka zápasu

Každá mapa má různě navržené platformy s jiným pozadím, mapa vyobrazená na obrázku 8 je unikátní tím, že se z ní dá vypadnout. Na platformách se náhodně objevují bonusy, které postavám po sebrání poskytují určitou výhodu. Jsou celkem tři typy bonusů: doplnění životů, dočasné zvýšení poškození a dočasné zvýšení rychlosti.

Hráči začínají kolo na nejnižší platformě v rohu. Pro výhru v celém zápase musejí soupeře porazit dvakrát, u bosse ve věži je to jiné, tam se hraje pouze na jedno kolo.

V horní části obrazovky se nachází UI, které obsahuje název hráče, životy, energii, cooldowny (doba, po kterou se schopnost nesmí opakovaně použít) a aktuální počet vyhraných kol.

4.7. Bot

Bot má za úkol se ve hře co nejvíce podobat chování normálního lidského hráče, aby si hráč mohl užít hru, aniž by musel hrát s nějakým kamarádem.

4.8. Průchod věží

Průchod věží je herní mód pro jednoho hráče, ve kterém si zvolí postavu, se kterou bude muset hrát při celém průchodu.

Ve věži na hráče čeká celkem 5 kol, ve kterých musí zvítězit, aby zdolal tento mód a získal úspěch „Krásný pohled ze shora“. Jakmile prohraje jediný zápas, hra pro něj končí a musí hrát od začátku.

V posledním zápase na hráče čeká boss a zápas se hraje pouze na jedno kolo. Tento zápas je také těžší v tom, že boss má k dispozici více schopností, než mají ostatní postavy.

4.9. Cheaty

Do hry jsou zakomponovány také 2 cheaty původně určené pro testování hry, ale zůstaly zachovány pro případ, že by hráč chtěl například dělat různé pokusy ve věži atd. Slouží i jako taková vzpomínka na starší hry, kde byly cheaty samozřejmostí.

Dostupné cheaty jsou:

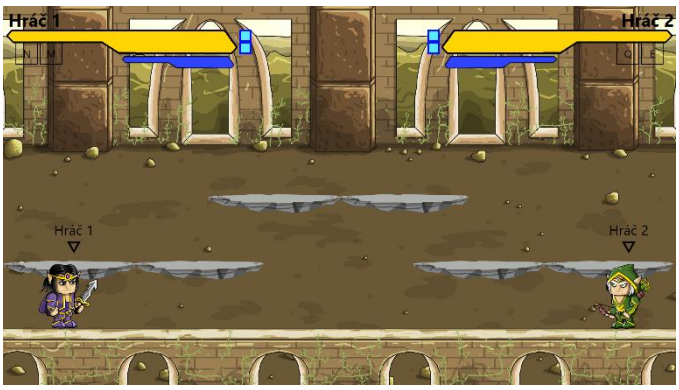
- SVALY – první hráč (začínající na levé straně obrazovky) dostane nesmrtelnost, má zvýšenou rychlost a soupeře porazí na jeden zásah
- NAHIT – druhému hráči se sníží počet životů na minimum, aby na jeho poražení stačil pouhý zásah

5. Popis řešení

5.1. Responzivita hry

Celá hra je navržena pro rozlišení 1920x1080 a pozice různých elementů v UI, dokonce i pozice objektů v samotné hře jsou na to přizpůsobeny. Mohlo by se zdát, že hra bude nehratelná na jiných rozlišeních, ale WPF mi v tomto ohledu velmi pomohlo objektem *ScaleTransform*, pomocí kterého lze škálovat velikost plochy, aniž by to porušilo např. původní vzdálenosti od okraje atd. Na obrázku 9 níže můžete vidět, že hru lze hrát i na jiných poměrech obrazovky než 16:9, který je pro porovnání na obrázku 10. Hra sice nebude vypadat tak přirozeně, ale je hratelná.

Obrázek 10 Ukázka 4:3 rozlišení



Obrázek 9 Ukázka 16:9 rozlišení

Jediný problém, na který jsem narazil u *ScaleTransform*, nastane při získávání souřadnic kurzoru myši. Pokud počítáte s určitým rozlišením, ale uživatel používá jiné, tak nepomůže ani *ScaleTransform*.

5.2. Aktualizace plochy hry

Již v začátcích vývoje jsem se musel rozhodnout, jakým způsobem bude nejlepší aktualizovat plochu, aby hra působila plynule. Na výběr jsem měl mezi několika Timery, ale nakonec jsem se rozhodl pro *DispatcherTimer* kvůli lepší integraci s UI prvky. *DispatcherTimer* se totiž dá spustit na vlákně, na kterém běží i UI, díky tomu je přístup k prvkům jednoduchý. Klasický *Timer* má s přístupem k UI prvkům problémy, protože běží na jiném vlákně.

Jedinou nevýhodou *DispatcherTimeru* je, že nezaručuje úplně přesný interval opakování, ale pro hru je tato nepřesnost zanedbatelná, protože i tak zvládne WPF pomocí *DispatcherTimeru* vykreslovat 60 snímků za sekundu. [11]

5.3. Pozice a pohyb objektů

Pozice všech objektů (platformy, postavy, bonusy, útoky postav) je závislá na jejich vzdálenosti od levého kraje obrazovky (osa X) a od spodního kraje obrazovky (osa Y), tudíž z vlastnosti *margin* mohu zjistit přesné souřadnice daného objektu.

Pro plynulost pohybu postav do stran jsem založil číselnou proměnnou, která udává o kolik se hráč posune na ose X. Po stisku klávesy chození se začne hodnota proměnné navyšovat (pokud hráč jde doleva, tak snižovat), dokud nedosáhne maximální rychlosti postavy. Díky postupnému navyšování aktuální rychlosti pohyb působí přirozeně a postava hned při zahájení chůze nemá plnou rychlost.

Skok funguje na bázi ticků (1/60 sekundy v případě hry) poskytnutých *DispatcherTimerem*. Po zahájení výskoku se dalších 12 ticků postava posune o 20 pixelů nahoru, takže po dokončení skoku bude hráč o 240 pixelů výš oproti jeho původní pozici.

Seskok postavu posune o 20 pixelů níže a o zbytek se postará algoritmus gravitace, díky kterému postava spadne, jelikož už nebude na platformě, ze které seskočila.

5.4. Platformy

Důležitou součástí hry jsou platformy, které výrazně pomáhají v souboji se soupeřem, protože proskakováním mezi nimi se hráč může efektivně vyhýbat soupeřovým útokům. Každá mapa má různě navržené platformy, které jsou uloženy v *Listu* objektů. Jedná se o objekty typu Image, aby na ně šel jednoduše použít obrázek a jejich *Margin* udává pozici na mapě.

Jelikož algoritmus gravitace počítá s tím, že pod platformu položenou nejníže se nedá dostat, tak pod tuto úroveň nefunguje. Ve většině případů by to nebyl takový problém, ale v případě druhé mapy, ze které se musí dát vypadnout, už by to problém byl. Ten jsem vyřešil tak, že jsem pod mapu položil neviditelnou platformu, díky které gravitace mohla normálně fungovat a zároveň se dalo vypadnout z mapy.

5.5. Bonusy

Bonusy hráči po sebrání poskytnou dočasnou výhodu. Po zahájení kola se vygeneruje počet milisekund v rozmezí 7500 až 15000 sloužící k tomu, aby se bonusy neobjevovaly vždy po stejné době, ale pokaždé jindy. O jejich spawn (objevení na mapě) se stará časovač poskytnutý třídou *System.Diagnostics.Stopwatch*, ten je spuštěn hned po začátku kola. Pokud je jeho vlastnost ElapsedMilliseconds (udává počet milisekund, který uběhl od spuštění) vyšší než vygenerovaný čas potřebný ke spawnu bonusu, vylosuje se číslo od 1 do 3, pomocí kterého se určí, který bonus se na mapě objeví. Dále se náhodně vybere číslo platformy a vzdálenost od počátku platformy určující jeho přesnou polohu. Když už mám

všechny potřebné údaje, stačí mi jen založit objekt s daným bonusem, přidat ho do *Listu* aktivních bonusů a vykreslit ho na obrazovce. Díky tomu, že mám platformy uložené v Listu, se mi proces hodně zjednodušil, jelikož mi při výběru platformy stačí generovat číslo od 0 do počtu platforem – 2 (poslední platforma v listu je neviditelná pod mapou, vysvětlena je výše v kapitole 5.4.).

Po tomto celém procesu se časovač vyresetuje, vygeneruje se čas spawnu dalšího bonusu a takto to pokračuje dál.

Během hry se každým *tickem* prochází pole aktivních bonusů. Když algoritmus pozná, že hráč stojí na bonusu, vymaže se bonus z plochy hry a spustí se u hráče časovač udávající, jakou dobu bude bonus působit. Pokud se jedná o bonus na doplnění životů, nemusí se spouštět žádný časovač, protože se životy hráči přičtou hned.

5.6. Útoky postav

Již od začátku vytváření útoků jsem si založil abstraktní třídu *Aktualizovatelne*, ze které budou dědit všechny útoky vyžadující pravidelnou aktualizaci (pohybující se výstřely, vybuchující TNT, atd…). Třída v sobě má předem vytvořenou metodu, která útok nastaví jako neaktivní (když se výstřel dostane z mapy, když TNT vybuchne). Dále je zde abstraktní metoda, která se bude spouštět každý *tick* a má za úkol obnovovat vlastnosti daného útoku (např. měnit pozici, zjišťovat kolizi se soupeřem), ta je abstraktní, protože u každého útoku se musí aktualizovat jiné vlastnosti.

Ukázka třídy *Aktualizovatelne*:

```
abstract class Aktualizovatelne
    {
        bool aktivni = true;
        public abstract void Tick();
        public abstract Image ReturnImage();
        public void Neaktivni()
        {
            aktivni = false;
        }
        public bool getAktivni()
        {
            return aktivni;
        }
    }
```

Každá postava má různé cooldowny (doba, kterou hráč musí čekat před vysláním dalšího útoku), ty jsou uloženy jako vlastnosti v třídě postavy. Do proměnné typu *DateTime* ukládám čas vyslání útoku a pokud od tohoto času neuběhne požadovaný počet milisekund udávaný cooldownem útoku, hráč nemůže daný útok využít.

5.7. Animace postav

Práce na animování postav mi zabrala spoustu práce, ze začátku jsem ve hře měl pouze pohybující se postavy bez jakýchkoliv animací a nijak jsem je moc neřešil. Když jsem se ale poté hru snažil graficky vylepšit, dal jsem si za cíl udělat i pořádné animace, bez kterých by to prostě nebylo ono. Již v tuto dobu jsem věděl, že to nebude nic lehkého.

Začal jsem tím, že jsem si připravil několik obrázků ke každé činnosti dané postavy, ze které potom vznikly animace. Ke každé činnosti jsem měl k dispozici většinou kolem 10 obrázků, s takovým počtem už se animace zdála být plynulá. Obrázky jsem naimportoval do projektu a vložil do *Listů* s typem *BitmapImage,* každá činnost má vlastní *List* s obrázky.

V případě animace pohybu jsem u třídy, ze které všechny postavy dědí, založil vlastnost, která v sobě drží index animace, pomocí kterého se z *Listu* vybere správný obrázek. U postavy se musí při každém *ticku* kontrolovat, zda je v pohybu. Pokud ano, zvýší se již zmíněný index animace. Při příštím průchodu *ticku* se obraz postavy nahradí dalším v pořadí (v závislosti na indexu) a tím u postavy vzniká animace. Pokud se postava zastaví a stojí na místě, index se změní na 0 a to znamená, že se postavě vrátí původní obrázek, se kterým začínala. Na obrázku 11 je pro ukázku vyobrazená animace pohybu lukostřelce.

Obrázek 11 Animace pohybu lukostřelce

U animací útoků jsem na to šel podobným způsobem, zde jsem musel navíc vyřešit to, že se animace nesmí dát zrušit a také to, že se objekt využívaný útokem (např. šíp) vyvolá, až když index animace dosáhne určité hodnoty. Za zmínku stojí to, že animace útoku mají přednost před animací pohybu, postava by jinak vypadala divně, kdyby např. střílela šípy bez natahování luku.

5.8. UI

UI je důležitým prvkem hry, pomocí kterého zjistíte aktuální stav obou postav. Základem ukazatele je tzv. healthbar, který graficky vyobrazuje počet životů, skládá se ze dvou překrývajících se objektů. Spodní objekt složí pouze jako pozadí a objekt, který je v popředí, slouží ke znázornění životů, ten se aktualizuje změnou vlastnosti *Width*, do které vložíme součin aktuálního stavu životů (v procentech) a původní šířky. Na obrázku

12 je ukázka healthbaru, který znázorňuje plné životy, na obrázku 13 znázorňuje část životů, která hráči zbývá.

Obrázek 12 Healthbar znázorňující plné HP

Obrázek 13 Healthbar znázorňující ubrané HP

Podobně jako healthbar funguje i bar zobrazující energii a ukazovač cooldownu schopností.

5.9. Algoritmus bota

Nejtěžší algoritmus v celé hře je bezpochyby chování bota. Ten byl potřeba udělat tak, aby se bot choval co nejlidštěji, jak je to jen možné. To, jaký bude další krok bota, ovlivňuje mnoho faktorů, např. zda má bot málo životů, zda vidí soupeře atd.

Algoritmus je potřeba opakovat každým *tickem* a vždy si na jeho začátku zjišťuje polohu bota, polohu soupeře, zda se nad botem nachází platforma ke skoku, zda se pod ním nachází platforma k seskoku, na jaké platformě se právě nachází, jaké jsou na mapě bonusy a aktuální počet životů bota.

Aby se snáze dalo porozumět tomu, co vše algoritmus hlídá a vykonává, vytvořil jsem vývojový diagram, který je na obrázku 14. Na něm jsou ukázány nejdůležitější činnosti bota. Jediná podrobně nepopsaná část je útok na soupeře, protože pro každou postavu je algoritmus trošku jiný, aby se mohl přizpůsobit schopnostem dané postavy.

Obrázek 14 Vývojový diagram algoritmu bota

5.10. Využitá grafika

V rané fázi vývoje jsem hru zkoušel tvořit s vlastními obrázky, ale jelikož grafika není moje silná stránka, nevypadala hra moc oslnivě, sáhl jsem tedy po volně dostupných obrázkách na internetu (všechny odkazy jsou samozřejmě součástí dokumentace). Výhodou těchto tzv. assetů je to, že byly poskytnuty i s animacemi ve formátu PNG.

Závěr

Podařilo se mi vytvořit hru, která jistě potěší všechny, kteří si chtějí zavzpomínat na staré dobré 2D hry. Podařilo se mi do hry zahrnout vše, co jsem si slíbil v zadaní, dokonce se mi navíc podařilo přidat i herní mód, který původně v plánu nebyl (režim průchodu věží). Hra tedy aktuálně nabízí funkční systém achievementů, 5 postav (4 z nich jsou hratelné), 5 map a 3 herní módy.

Do budoucna bych do hry chtěl nějakým způsobem implementovat jednoduchý LAN multiplayer, přidat více postav a další herní mód navíc.

Zdroje

[1] About – Brawlhalla. *Brawlhalla* [online]. [cit. 2022-03-14]. Dostupné z: https://www.brawlhalla.com/about/

[2] Prohlídka jazyka C#. *Microsoft Docs* [online]. 2022 [cit. 2022-03-13]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/tour-of-csharp/

[3] Lekce 1 - Úvod do objektově orientovaného programování v C#. *Itnetwork.cz* [online]. [cit. 2022-03-13]. Dostupné z: https://www.itnetwork.cz/csharp/oop/c-sharp-tutorial-uvod-do-objektove-orientovaneho-programovani

[4] What is .NET?. *Microsoft* [online]. [cit. 2022-03-13]. Dostupné z: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

[5] Desktop guide (WPF .NET). *Microsoft Docs* [online]. 2022 [cit. 2022-03-13]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/desktop/wpf/overview/?view=netdesktop-6.0

[6] Introduction to XML with C#. *csharp.net-tutorials.com* [online]. [cit. 2022-03-13]. Dostupné z: https://csharp.net-tutorials.com/xml/introduction/

[7] Welcome to the Visual Studio IDE. *Microsoft Docs* [online]. 2022 [cit. 2022-03-14]. Dostupné z: https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022

[8] Git. *Git* [online]. [cit. 2022-03-15]. Dostupné z: https://git-scm.com/

[9] About – Git. Git [online]. [cit. 2022-03-15]. Dostupné z: https://git-scm.com/about

[10] Chapter 1 Introduction. docs.gimp.org [online]. [cit. 2022-03-16]. Dostupné z: https://docs.gimp.org/2.10/en/introduction.html

[11] DispatcherTimer Class. *Microsoft Docs* [online]. [cit. 2022-03-19]. Dostupné z: https://docs.microsoft.com/en-us/dotnet/api/system.windows.threading.dispatchertimer?view=windowsdesktop-6.0

Odkazy na obrázky ve hře

https://pixabay.com/cs/vectors/stopky-%c4%8dasova%c4%8d-hodinky-sekundy-34107/

https://pixabay.com/cs/vectors/torn%c3%a1do-bou%c5%99ka-v%c3%adtr-ot%c3%a1%c4%8den%c3%ad-46793/

https://pixabay.com/illustrations/competition-fight-versus-art-6612629/

https://pixabay.com/cs/illustrations/bombardovat-%c4%8dasova%c4%8d-asi-vyhodit-3910553/

https://pixabay.com/cs/illustrations/bombardovat-%c4%8dasova%c4%8d-asi-vyhodit-3910551/

https://pixabay.com/cs/illustrations/bombardovat-%c4%8dasova%c4%8d-asi-vyhodit-3910550/

https://pixabay.com/cs/vectors/myslel-bal%c3%b3n-mysl%c3%adc%c3%ad-myslet-si-2655094/

https://pixabay.com/cs/vectors/koruna-kr%c3%a1l-kr%c3%a1lovna-korunky-2130770/

https://pixabay.com/vectors/chess-chess-icons-from-persian-shah-335030/

https://pixabay.com/vectors/tick-mark-checked-check-okay-yes-296754/

https://pixabay.com/cs/vectors/ozuben%c3%a9-kolo-obr%c3%a1b%c4%9bn%c3%ad-ozuben%c3%bdch-kol-145804/

https://craftpix.net/freebies/free-elven-land-game-battle-backgrounds/

https://craftpix.net/freebies/free-jump-game-items/

https://craftpix.net/freebies/2d-fantasy-elf-free-sprite-sheets/

https://craftpix.net/freebies/free-minotaur-tiny-style-2d-sprites/

https://craftpix.net/freebies/free-golem-tiny-style-2d-character-sprites/

Seznam obrázků