PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

ROČNÍKOVÝ PROJEKT

Softwarová implementace deskové hry Bunny War

Květen 2016

David Jaroš
Aplikovaná informatika II. ročník

Abstrakt

Následující text obsahuje popis mého řešení zadání projektu, kterým bylo implementovat softwarovou aplikaci deskové hry Bunny War.

Obsah

Seznam obrázků

1 Popis a analýza

Ve 2. ročníku studia oboru Aplikovaná informatika v akademickém roce 2015/16 v rámci předmětů Projektový seminář 1 a 2 bylo zadáno implementovat softwarovou aplikaci deskové hry Bunny War (dále v textu jako „projekt“). Tento text popisuje mé řešení projektu.

1.1 Požadavky na projekt vznesené v zadání

- korektní implementace pravidel hry (nemožnost provést tah odporující pravidlům, správné ukončení hry a podobně)
- algoritmy pro herní strategii, nastavitelná obtížnost hry v adekvátním rozsahu (zvlášť pro každého počítačového hráče)
- možnost hry dvou lidí, člověka proti „počítači“, a „počítače“ proti „počítači“
- možnost nastavit a kdykoliv změnit obtížnost i v průběhu hry
- možnost kdykoliv zaměnit počítačového a lidského hráče (bez ohledu na to, je-li hráčem člověk nebo počítač, změna i v průběhu hry)
- nápověda „nejlepšího tahu“
- ukládání a načítání (ukončených i rozehraných) partií
- undo/redo tahů do libovolné úrovně
- prohlížení historie tahů (přehledné zobrazení provedených tahů)
- robustnost (program musí reagovat správně na nesprávné uživatelské vstupy, zejména ovládání, vadný formát nebo obsah souboru apod., aplikace nesmí havarovat)
- vestavěná nápověda
- grafické uživatelské rozhraní (GUI) zpracované podle standardů
- program ve spustitelné formě, je-li to pro zprovoznění aplikace nutné pak také instalátor (v odůvodněných případech je přípustná spustitelnost z vývojového prostředí)
- kompletní zdrojové kódy programu včetně dalších částí nutných pro sestavení aplikace
- programátorská dokumentace k projektu vytvořená podle doporučeného stylu (PDF). Musí obsahovat zejména popis struktury kódu, algoritmy hry včetně herní strategie, postup pro sestavení aplikace. Dokumentace nemusí obsahovat uživatelskou příručku

1.2 Zadání hry

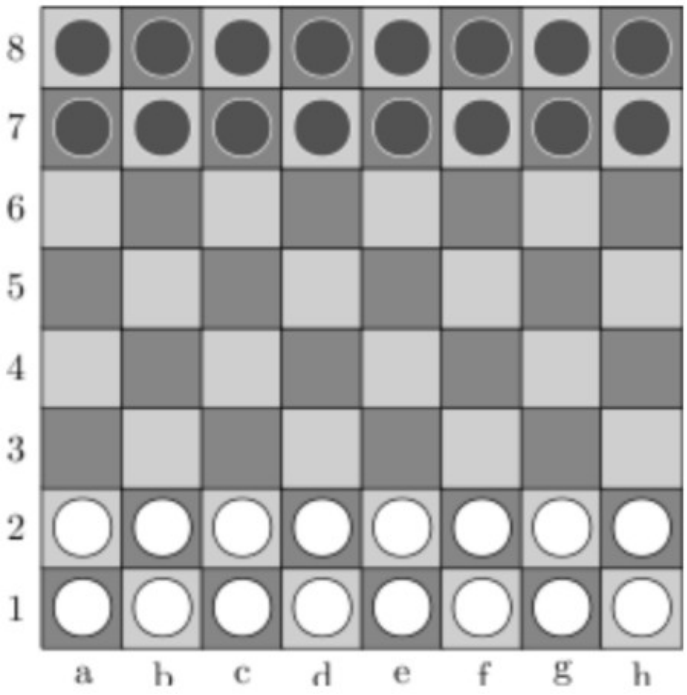
Desková hra Bunny War vznikla v roce 2001 a jejím autorem je L. Lynn Smith, zdroj informací o této hře [1].

Cíl hry

- Zajmout alespoň 15 protivníkových kamenů.

Výchozí situace

- Kameny stojí proti sobě ve dvou krajních řadách hrací desky o rozměru 8x8 (viz. Obrázek 1)



Obrázek 1 - Výchozí situace

Hra

- Kámen smí být posunut o libovolný počet volných polí v libovolném diagonálním nebo ortogonálním směru.
- Zajímat soupeřovy kameny přeskokem lze tehdy, leží-li na sousedním poli protivníkův kámen a další pole ve směru skoku je volné. Skákající kámen je umístěn na toto volné pole a přeskočený kámen je odstraněn z desky.
- Skákání není povinné, ale když hráč začne skákat, musí možnosti skákání využít v plném rozsahu, tj. hráč nesmí ukončit tah, pokud může pokračovat ve vícenásobném skoku. Po každém dopadu lze měnit směr skoku.

Konec hry

- Pokud některému z hráčů zbývá pouze jeden kámen, prohrává.
- Pokud některý z hráčů nemůže provést žádný tah, prohrává.
- Pokud po dobu 30 tahů nebyl přeskočen žádný kámen, končí hra remízou.

1.3 Podklady pro vypracování projektu

Pro vypracování projektu jsem použil znalostí získaných studiem výukových tutoriálů o programovacím jazyku .NET/C# umístěných na webové stránce www.itnetwork.cz/csharp [2]. Dalším zdrojem znalostí o tomto jazyku byla pro mě série knih Moderní programování, autorem je Radek Vystavěl, kontakt na webové stránce www.moderniProgramovani.cz [3]. Ke studiu algoritmu Minimax a Alfa-Beta jsem využil text Algoritmy realizující počítačového hráče v jednoduchých deskových hrách, jehož autorem je Tomáš Kühr [4]. Všechny obrázky v mé aplikaci ve formátu png a ikony ve formátu ico jsem stáhnul z webové stránky www.iconfinder.com [5] nebo www.iconarchive.com [6]. Vždy jsem kontroloval aby nebyla porušena autorská práva vlastníků těchto obrázků s ohledem na použití mé aplikace pouze pro akademické a nekomerční účely.

1.4 Volba operačního systému a vývojového prostředí

Hru Bunny War jsem naprogramoval jako samostatně spustitelnou aplikaci pro osobní počítače s nainstalovaným operačním systémem Microsoft Windows. Pro napsání kódu aplikace jsem zvolil programovací jazyk .NET/C# a vývojové prostředí Microsoft Visual Studio Community 2015, jedná se o bezplatnou licenci programu pro vývoj aplikací. Moje aplikace hry využívá podmnožinu rozhraní .NET, která nese název Windows Presentation Foundation (dále v textu jako WPF). WPF využívá značkovací jazyk XAML pro vytváření grafického uživatelského rozhraní. Díky jazyku XAML jsou od sebe odděleny funkčnost a vzhled aplikace. Soubor nápovědy k mé aplikaci jsem vytvořil v nástroji HelpNDoc Personal Edition 4.9.132. Jedná se o program pro vytváření kompilovaných HTML souborů nápovědy (chm souborů) a licence pro nekomerční použití tohoto nástroje je typu freeware.

1.5 Požadavky na spustitelnost programu

Aplikace počítačové hry je vytvořena a odladěna pro fungování v operačním systému Microsoft Windows 8.1 s nainstalovaným rozhraním Microsoft .NET Framework verze 4.5.2.

2 Uživatelská část

Tato část obsahuje popis ovládání aplikace z pohledu uživatele

2.1 Spuštění aplikace

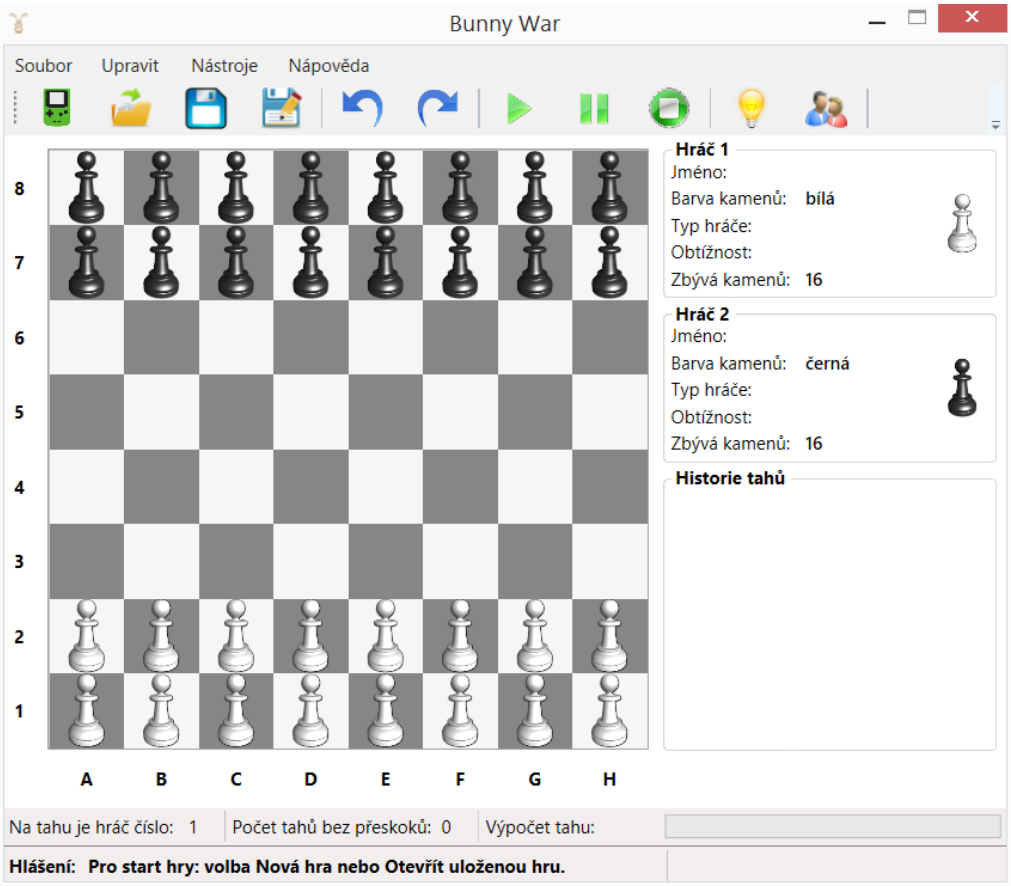
Aplikace počítačové hry Bunny War se spustí otevřením souboru Bunny.exe. Po spuštění aplikace se otevře hlavní okno aplikace s názvem Bunny War, které tvoří jádro ovládání celé aplikace (viz. Obrázek 2). Toto hlavní okno obsahuje ovládací prvky, informační prvky a graficky vytvořenou hrací desku.

2.2 Ovládací prvky

Ovládací prvky se nacházejí v horní části hlavního okna aplikace a je to panel nabídek (menu bar) a panel nástrojů (tool bar). Většina nabídek z panelu nabídek je přístupná v panelu nástrojů jako tlačítka. Ovládání aplikace je možné volbami z panelu nabídek, tlačítky z panelu nástrojů a klávesovými zkratkami, které jsou uvedeny v panelu nabídek u názvu volby. V následujícím popisu ovládání budu popisovat jednotlivé volby tak, jak jsou označeny svým názvem v panelu nabídek hlavního okna anebo svými popisy v dialogovém okně Nová hra a Změna nastavení hráčů.

2.3 Informační prvky

Informační prvky se nacházejí v pravé části hlavního okna aplikace a tvoří ji tři oddělené skupiny informačních prvků. V první skupině se zobrazují informace o Hráči číslo 1, ve druhé skupině informace o hráči číslo 2 a ve třetí skupině je přehledně zobrazena Historie tahů. V dolní části okna jsou umístěny 2 stavové řádky (status bar), které informují hráče o tom kdo je na tahu, kolik tahů bylo provedeno bez přeskočení figur protihráče. Ve stavovém řádku se také zobrazuje informace o průběhu výpočtu tahu v případě, kdy hraje počítačový hráč nebo když lidský hráč požádá o nápovědu nejlepšího tahu a zobrazuje se zde také slabá zpětná vazba programu s podtitulkem Hlášení.



Obrázek 2 - Okno aplikace po jejím spuštěním

2.4 Hrací deska

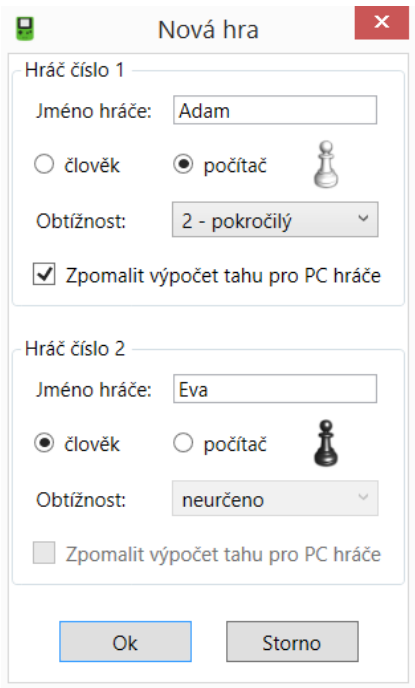
Hrací deska je rozdělena na 8 x 8 políček s umístěnými kameny hráčů. Na začátku každé partie hry jsou kameny umístěny ve výchozích pozicích dle pravidel hry. Označení políček zleva hrací desky je číslicemi 1 až 8 a zespoda hrací desky písmeny A až H.

2.5 Zahájení hry

Po spuštění programu existují dvě možnosti jak začít hru. První volbou je Nová hra, druhou volbou je Otevřít hru.

2.5.1 Volba Nová hra

Po této volbě se otevře dialogové okno s názvem Nová hra (viz. Obrázek 3) a uživatel si může zvolit parametry hry. Na výběr má možnost zda bude hrát jako lidský hráč, označeno volbou člověk nebo zvolí, že hráče bude ovládat počítač. Uživatel může zvolit libovolnou kombinaci lidského a počítačového hráče pro danou partii hry. Pokud uživatel zvolí hru počítačového hráče, musí také zvolit Obtížnost (inteligenci) počítačového hráče a dále volitelně může nastavit Zpomalení výpočtu tahu pro PC hráče. Pokud si uživatel přeje, může nastavit i Jméno hráče pro oba hráče. Pokud nezvolí svá jména, budou použita přednastavená jména Hráč 1 a Hráč 2.



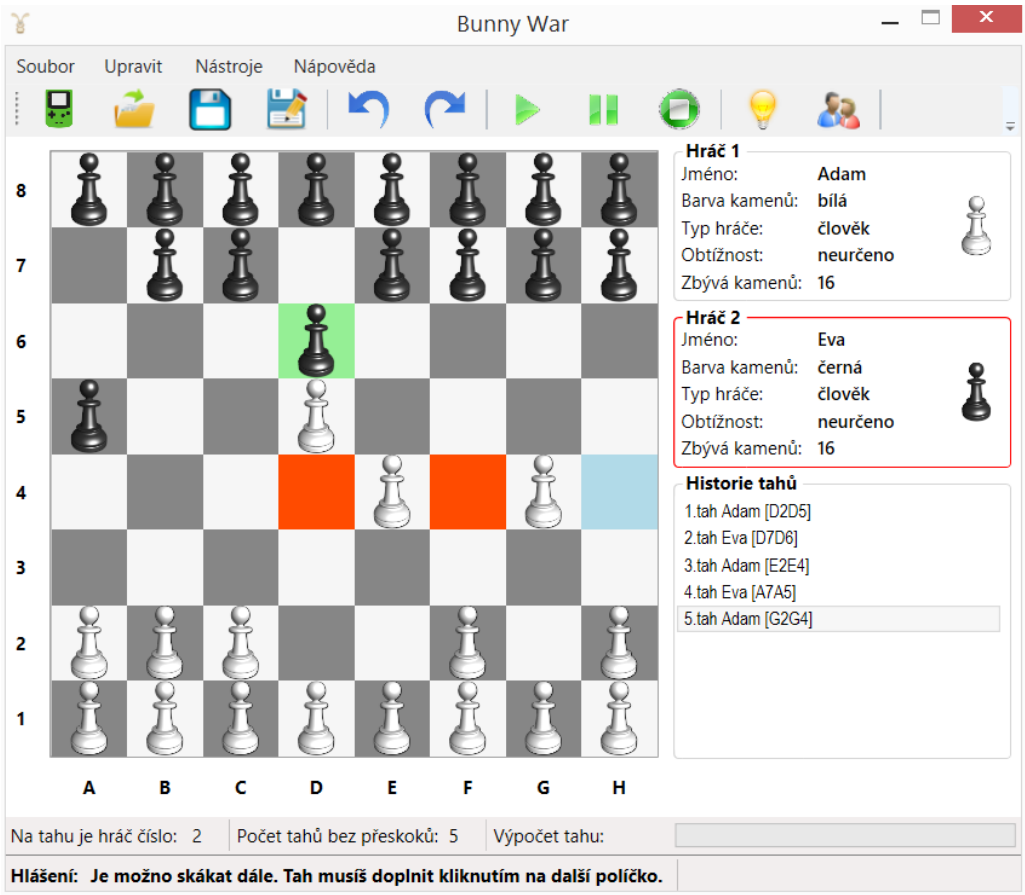
Obrázek 3 - Dialogové okno Nová hra

2.5.2 Volba Otevřít uloženou hru

Po této volbě bude hráč vyzván novým dialogovým oknem s názvem Otevřít k výběru rozehrané nebo ukončené partie hry, kterou si uložil v jiné instanci hry. Hra potom začíná z místa, kde ji uložil.

2.6 Průběh hry

Hráč ovládá pohyb kamenů na desce pomocí počítačové myši. Levým tlačítkem myši vybírá kámen, kterým chce pohybovat a dále políčko na desce, na které se chce přesunout. Pravým tlačítkem zruší svůj výběr tahu. Označení kamenu se provede zeleným zbarvením pozadí políčka. Políčka, na které chce táhnout se podbarvují modrým pozadím. Pokud je tah dle pravidel, ihned se provede a okamžitě začíná hrát druhý hráč. Pokud hráč skáče přes soupeřovy kameny a existuje možnost sebrat další soupeřův kámen, je pozadí políčka zbarveno červeně a uživatel je o tom informován ve stavovém řádku (viz. Obrázek 4). Nápověda nejlepšího tahu se zobrazuje graficky žlutým podbarvením políček a slabou zpětnou vazbou ve stavovém řádku. Odehrané tahy se zobrazují v informačním prvku Historie tahů. V této Historii tahů se může uživatel pohybovat volbou Tah zpět nebo Tah vpřed. Informace o počtu tahů bez přeskoku a číslo hráče na tahu se také zobrazují ve stavovém řádku programu. Hra končí dle pravidel hry výhrou jednoho z hráčů nebo remízou. O výsledku hry je uživatel informován dialogovým oknem s názvem Konec partie hry. Poté je vyzván k případnému uložení dohrané partie hry. Volbou Storno může stejnou partii hry hrát znova třeba z jiného místa, procházet si odehrané tahy nebo začít úplně novou partii hry.



Obrázek 4 - Průběh hry

2.7 Změna nastavení hry

Uživatel má možnost volbou Nastavení hráčů změnit kdykoliv v průběhu rozehrané partie hry všechny parametry hry, které si nastavil při zahájení partie hry. Po zvolení této volby se otevře dialogové okno s názvem Změna nastavení hráčů. V tomto okně mění stejné parametry hry jaké nastavoval v okně s názvem Nová hra (viz. Obrázek č.3).

2.8 Nápověda nejlepšího tahu

Uživatel, který hraje hru jako lidský hráč, může kdykoliv v průběhu hry požádat o nápovědu nejlepšího tahu volbou Nápověda nejlepšího tahu.

2.9 Ukládání a otevírání uložené hry

Uživatel může kdykoliv v průběhu hry uložit rozehranou nebo dokončenou hru volbou Uložit hru nebo Uložit hru jako ... Po této volbě bude uživatel vyzván novým dialogovým oknem s názvem Uložit jako k uložení hry a může zadat název souboru pod jakým chce mít stávající stav hry uložen nebo se použije přednastavené jméno souboru Bunnywar1.bin. Pokud byla rozehraná hra již jednou takto uložena, při příští volbě Uložit hru bude pouze přepsána starší verze souboru novým souborem se stejným názvem. Otevírání uložené hry je popsáno výše v kapitole č.2.5.2 Volba Otevřít uloženou hru. Všechny uložené partie hry mají libovolný název souboru do kterého se ukládají, ale pevně stanovenou příponu souboru označenou *bin*.

2.10 Tah zpět a tah vpřed

Uživatel může procházet již odehrané tahy v Historii tahů volbou Tah zpět a Tah vpřed. Pokud se uživatel posune v Historii tahů na jinou pozici než na poslední odehraný tah a chce pokračovat od tohoto předchozího tahu, může tak udělat a hra tak pokračuje od tohoto místa. V Historii tahů se lze také pohybovat kliknutím levým tlačítkem myši na zvolené číslo tahu.

2.11 Další volby v aplikaci

2.11.1 Volba Pokračovat (výpočet tahu)

Volba Pokračovat (výpočet tahu) slouží k pokračování ve výpočtu tahu pro počítačového hráče nebo k výpočtu nejlepšího tahu pro lidského hráče. K této volbě bývá uživatel vyzván slabou zpětnou vazbou ve stavovém řádku v případě, kdy byl výpočet tahu přerušen z důvodu zvolení jiné volby v aplikaci, např. volbou Změna nastavení hráčů, Uložit hru, Otevřít hru, apod.

2.11.2 Volba Pauza (výpočet tahu)

Volbou Pauza (výpočet tahu) může uživatel kdykoliv přerušit výpočet tahu ať již pro počítačového hráče nebo pro lidského hráče při výpočtu nejlepšího tahu.

2.11.3 Volba Ukončit partii

Pokud si uživatel přeje opustit rozehranou partii hry, může kdykoliv zvolit Ukončit partii. Po této volbě se aplikace hry nastaví do počátečního stavu v jakém byla po spuštění aplikace.

2.11.4 Volba Prohození stran

Standardní nastavení kamenů při startu aplikace hry Bunny War je hráč s bílými kameny v dolní části hrací desky a hráč s černými kameny v horní části desky. Volba Prohození stran umožňuje toto nastavení měnit před startem nové partie hry.

2.11.5 Volba Nápověda

Kdykoliv v průběhu hry může uživatel požádat o nápovědu k aplikaci hry volbou Nápověda. V této nápovědě jsou popsána pravidla hry a ovládání aplikace.

2.12 Ukončení hry

Pokud si uživatel přeje ukončit běh aplikace může k tomu použít volbu Ukončit program nebo může zavřít hlavní okno programu křížkem v pravém horním rohu hlavního okna. Pokud uživatel ukončuje program z rozehrané partie hry, je dotázán zda-li si přeje uložit tuto partii hry před ukončením programu.

3 Programátorská část

Tato část obsahuje popis struktury kódu programu a algoritmus hry.

3.1 Popis struktury kódu

Kód počítačové hry je rozdělený na část, která se stará o logiku hry (logická část) a část, která obstarává grafické uživatelské rozhraní hry (prezentační část). Logická část je napsaná v jazyce C#. Prezentační část aplikace je napsaná v jazyce XAML.

3.1.1 Code Behind

Každé okno vytvořené ve WPF má kromě XAML kódu ještě Code Behind, který obsahuje volání logiky. Hlavní okno mé aplikace je instance třídy MainWindow, která je potomkem třídy Window. Z Code Behind okna MainWindow volám logiku mé aplikace.

3.2 Prezentační část aplikace

Okna ve WPF jsou samostatné třídy, které mají své grafické uživatelské rozhraní (dále v textu jen GUI) napsané v jazyce XAML. Jako ovládací prvky v mých oknech WPF jsem zvolil buď standardní prvky, které nabízí vývojové prostředí Microsoft Visual Studio Community 2015 nebo mé vlastní ovládací prvky. Hlavní okno MainWindow obstarává hlavní interakci uživatele s aplikací. Pro zadání nové hry a změnu nastavení hry používám okno s názvem NastaveniHracuNabidka. Dále jsem použil dialogová okna s názvem Oaplikaci a Vitezství. Jak už název napovídá jedná se o informační okna, která zobrazí informaci o autorovi počítačové aplikace a informaci kdo vyhrál hru, případně informaci o remíze. Všechna má okna jsou reprezentována třídou, kterou dědí ze třídy Window.

Jednotlivá políčka na hrací desce jsou instancemi mého uživatelského ovládacího prvku s názvem UzivPrvekPolicko, který je potomkem třídy UserControl. Tento uživatelský prvek umožňuje reagovat na změnu vlastností a umožňuje zasílat standardní události, zejména ty, které se týkají počítačové myši. Využil jsem tedy možností, které rozhraní WPF nabízí a poskládal desku ze 64 takovýchto uživatelských prvků v hlavním okně MainWindow. Ovládání těchto prvků je umožněno metodami, které jsem vytvořil v Code Behind hlavního okna. Pro vykreslování figurek hráčů využívám vlastnost Canvas uživatelského prvku, kterou jsem nazval platno. Pozadí prvku UzivPrvekPolicko je vlastnost s názvem Background. Jako pozadí jsem zvolil několik barev pro různé situace, které se mohou vyskytnout v průběhu hraní hry (vlastnost třídy Brushes s označením LightBlue, LightGreen, OrangeRed, Yellow). Pozadí prvku nemá vliv na platno a na obrázky v něm vykreslené.

3.2.1 Obrázky v aplikaci

GUI obsahuje řadu grafických prvků. Ty standardní jsou součástí rozhraní WPF. Mezi nestandardní patří obrázky na pozadí tlačítek v panelu nástrojů a v hlavním menu. Dále jsou to ikony hlavního okna a ikony vedlejších oken. Některé z těchto obrázků a ikon používám i v ploše otevřených oken. Dále používám obrázky figur bílého a černého pěšce v hrací ploše jako figurky kamenů pro hru Bunny War [5], [6].

3.3 Logická část aplikace

3.3.1 Třída DispecerHry

Pro správný běh aplikace bylo potřebné vytvořit třídu DispecerHry. Instanci této třídy si vyrábím v Code Behind hlavního okna MainWindow při spuštění programu. Třída DispecerHry je kořenovou třídou mé stromové struktury tříd. Jednotlivé uzly, čili instance podřízených tříd, vytvářím v konstruktoru hlavního okna při jeho inicializaci. Kořenová třída mi obstarává komunikaci s dalšími třídami. Na instanci této třídy kladu dotazy o hře pomocí vlastností typu Get a nebo nastavuji parametry hry vlastnostmi typu Set.

3.3.2 Třída Hra

Ve třídě Hra si hlídám především kdo vyhrál, jestli není možnost kam táhnout dál, jestli je to remíza.

3.3.3 Třída Hrac

Ve třídě Hrac si nastavuji jednotlivé hráče hry. Nastavuji jim jméno, obtížnost hry, zda-li se jedná o počítačového hráče. Dále mi třída vrací počet kamenů hráče a jakým kamenem hráč hraje.

3.3.4 Třída Deska

Důležitou proměnnou (datovou složkou) ve třídě Deska je proměnná s názvem hraciDeska, což je dvourozměrné pole 8x8 čísel typu int, které reprezentuje hrací desku podobnou jako ve stolní variantě hry Bunny War. V tomto poli si uchovávám hodnoty 1, 2 a 0. Pokud je na indexu v poli číslo 1, označuje mi tak políčko na desce obsazené hráčem číslo 1, který hraje bílými kameny. Pokud je na indexu v poli číslo 2, označuje mi tak políčko na desce obsazené hráčem číslo 2, který hraje černými kameny. Pokud je v poli na některém z indexů hodnota 0, značí to, že políčko je prázdné. V duchu tohoto návrhu hrací desky jsem napsal metody, které jsem potřeboval k inicializaci hry do počátečního stavu, k posunu kamenů na desce, ke kontrole oprávněnosti tahů jednotlivých hráčů. V této třídě si hlídám pravidla hry.

3.3.5 Třída Pozice

Datovými složkami třídy Pozice jsou proměnné typu int s názvem vyska a sirka. Každý kámen na hrací desce je definován svou pozicí na hrací desce, kterou jsem nazval vyska a sirka. S instancemi třídy Pozice dále pracuji v jiných třídách. Metody, které jsem vytvořil, mi umožňují převádět zadání pozice kamene v textové formě (např. "A2") do přijatelného formátu dvou čísel typu int a naopak.

3.3.6 Třída Tah

Ve třídě Tah mám datové složky seznamPozic a preskoceneKameny, což jsou indexované seznamy instancí tříd Pozice. Ve hře Bunny War může být tah kamene vícenásobný. Tudíž si musím počítat v každém tahu také přeskočené kameny. K tomu používám vlastností typu Get a Set a metody, které pracují s datovými složkami této třídy.

3.3.7 Třída HistorieTahu

Instance třídy HistorieTahu je využita pro aktualizaci odehraných tahů v ovládacím prvku s nazvem historieTahuListBox hlavního okna aplikace MainWindow. Metody v této třídě pracují se seznamem odehraných tahů instance třídy ObservableCollection<Tah> a s podrobným výpisem těchto tahů jako instance třídy ObservableCollection<string>.

3.3.8 Třída GeneratorTahu

Pro výpočet tahu počítačového hráče a pro výpočet tahu nápovědy nejlepšího tahu lidského hráče využívám instanci třídy GeneratorTahu. V této třídě mám definovány metody, které přebírají již hotové instance třídy DispecerHry, Deska, Hrac, Tah, Pozice. Protože výpočty v mých metodách zabírají velký výpočetní čas hlavního vlákna aplikace, ve kterém běží GUI, rozhodl jsem se volat tyto metody z instance třídy BackgroundWorker hlavního okna aplikace MainWindow, která mi dovoluje provádět operace výpočtů ve vedlejším vlákně. Třída BackgroundWorker mi také umožňuje zpětnou vazbu na ovládací prvek s názvem mujProgressBar v hlavním okně aplikace a dále mi také umožňuje jednoduše zastavovat a obnovovat výpočet tahu na pozadí.

3.3.9 Průběh výpočtu tahu

Třída BackgroundWorker využívá bazén vláken (thread pool), sama spravuje vytvořená vlákna a ukončuje jejich práci. K činnosti této třídy jsem musel vytvořit ovladač události DoWork, RunWorkerCompleted a ProgressChanged. V ovladači pro událost DoWork nechávám vypočítat nejlepší tah podle zadané hloubky tím, že volám metodu GenerujNejlepsiTah z instance třídy GeneratorTahu. Ovladač pro událost RunWorkerCompleted mi ošetřuje případy, kdy byl výpočet zrušen nebo došlo k jiné chybě. V případě, že byl výpočet dokončen bez přerušení, jsou následně provedeny další kontroly na stav hry a vypočtený tah je vykreslen na hrací desce jako přesun kamenů počítačového hráče nebo jako nápověda nejlepšího tahu žlutým zbarvením pozadí políček hrací desky. Ovladač pro událost ProgressChanged mi aktualizuje grafický ovládací prvek mujProgressBar (potomek třídy ProgressBar) umístěný v hlavním okně aplikace.

3.3.10 Výpočet nejlepšího tahu pro počítačového hráče

Volání nejlepšího tahu pro počítačového hráče provádím v obsluze události DoWork pro instanci třídy BackgroundWorker s názvem bw voláním metody GenerujNejlepsiTah. Tato metoda přebírá jako své parametry aktuální hrací desku, hloubku procházení herního stromu a hráče na tahu a vrací nejlepší tah pro tyto vstupy jako instanci třídy Tah. Metoda GenerujNejlepsiTah volá jinou metodu ve své třídě s názvem GenerujTahy a metodu Alfabeta, což je má implementace stěžejní části algoritmu Alfa-beta.

Metoda GenerujTahy vygeneruje seznam tahů instance třídy List<Tah>, ve kterém jsou všechny možné tahy hráče na tahu na dané desce složené z *legálních* tahů (ortogonální a diagonální pohyb kamenů po desce) a z tahů, které jsou *přeskokem* kamene protihráče. Metoda GenerujTahy předává tento seznam metodě Alfabeta, ale ještě předtím použije 2 úpravy. První úprava spočívá v tom, že zamíchá *legální* tahy do náhodného pořadí ve vygenerovaném seznamu tahů, ale připomínám, že se stále jedná o tytéž *legální* tahy. Druhá úprava spočívá v tom, že tahy, které jsou *přeskokem* kamene protihráče posune na začátek seznamu možných tahů. Tyto 2 úpravy jsem zvolil ze dvou důvodů. Prvním je ten, že Alfa-beta ořezávání je účinnější, pokud se nejprve zkoumají nejsilnější tahy, v mém případě za ně

považuji tahy *přeskoku* kamene protihráče. Druhým důvodem je ten, že algoritmus Alfa-beta postupuje při ořezávání zleva doprava, nebo-li od prvního tahu k poslednímu v mém seznamu tahů a při oceňování se díky zamíchání legálních tahů může v metodě GenerujNejlepsiTah výpočet zastavit u jiného prvního tahu, který vedl k nejlepšímu ohodnocení. Tímto způsobem jsem přiměl počítačového hráče k větší aktivitě v pohybu na hrací desce.

Metoda Alfabeta je rekurzí, která ve svém těle volá již zmíněnou metodu GenerujTahy a pokud je dosaženo hloubky 0, vrací ohodnocení pozice metodou OhodnotPozici, což je má verze ohodnocovací funkce. Tato funkce používá pouze materiální složku, kterou je rozdíl v počtu kamenů hráče na tahu a kamenů protihráče. Ohodnocovací funkci jsem zvolil jednoduššího typu z důvodu, aby nezpomalovala časově náročný výpočet nejlepších tahů pro hru Bunny War, která mívá herní strom o poměrně velké šířce, zejména pro hloubky 3 a více.

3.3.11 Výpočet nápovědy nejlepšího tahu pro lidského hráče

Výpočet nápovědy nejlepšího tahu pro lidského hráče provádím podobným způsobem jako výpočet nejlepšího tahu pro počítačového hráče. Rozdíl spočívá v tom, že nejlepší tah pro lidského hráče volám z obsluhy události DoWork z instance třídy BackGroundWorker s názvem bwNapoveda a mám napevno nastavenu hloubku procházení herního stromu na velikost 3.

3.4 Algoritmus hry

3.4.1 Otevření hlavního okna aplikace

Spuštění aplikace Bunny War začíná vytvořením hlavního okna aplikace, což je instance třídy MainWindow, která je potomkem třídy Window. Konstruktor této třídy vytvoří instanci kořenové třídy mé stromové struktury tříd. Tato kořenová třída má název DispecerHry. Této kořenové třídě vytvořím v konstruktoru MainWindow instance uzlů této třídy, jedná se o instance tříd Hra, Deska, GeneratorTahu, HistorieTahu a tři instance třídy Hrac s názvem hrac1, hrac2, hracNaTahu. V konstruktoru hlavního okna si také vytvářím instance třídy BackgroundWorker s názvem bw a bwNapoveda. Hlavní okno zahájí vlastní smyčku událostí v operačním systému Windows. WPF umožňuje pro většinu ovládacích prvků v okně vytvořit metody pro obsluhu událostí, které se týkají počítačové myši. Uživatel proto ovládá hru jednoduchým kliknutím levým tlačítkem myši na ovládací prvky hry (popis ovládacích prvků v části č.2 Uživatelská část). Událost kliknutí pravým tlačítkem myši používám ke zrušení zadávání tahu lidským hráčem.

3.4.2 Zadávání tahu

Políčka na desce jsou reprezentována mým uživatelským prvkem s názvem UzivatelskyPrvekPolicko, který je potomkem třídy UserControl. Principem zadávání tahu a provádění tahu je obsluha událostí a vlastností tohoto ovládacího prvku. Z událostí jsem vybral MouseLeftButtonDown, MouseRightButtonDown, MouseEnter, MouseLeave a pro tyto jsem napsal obslužné metody v Code Behind hlavního okna. Z vlastností uživatelského prvku UzivatelskyPrvekPolicko jsem si vybral vlastnost Background, kterou měním zbarvení pozadí políčka a vlastnost Canvas, která mi dovoluje vykreslit do tohoto uživatelského prvku obrázek třídy BitmapImage, čili obrázek kamene hráče. Metody, kterými měním tyto vlastnosti jsem napsal v Code Behind hlavního okna.

3.4.3 Aktualizace historie tahů

Instance třídy HistorieTahu má datové složky cistySeznamOdehranychTahu a tahyVypsaneDetailne. Instance třídy Hra má datovou složku tahy. V průběhu hry si ukládám jednotlivé tahy jako seznam List<Tah> ve třídě Hra a jako seznam ObservableCollection<Tah> ve třídě HistorieTahu. Pro přehledné zobrazení tahu se všemi potřebnými údaji typu kolikátý tah je to v pořadí, kdo ho hrál a kolik kamenů přeskočil používám v hlavním okně aplikace uživatelský prvek historieTahuListBox (potomek třídy ListBox), který má vlastnost ItemsSource nastavenu na datovou složku třídy HistorieTahu s nazvem tahyVypsaneDetailne, což je seznam ObservableCollection<string>. Pohyb mezi položkami v ovládacím prvku historieTahuListbox mám ošetřen ovladačem události SelectionChanged. Metoda v tomto ovladači porovnává mezi sebou historii tahů uloženou v instanci třídy Hra a historií tahů uloženou v ovládacím prvku historieTahuListbox. Pokud indexy prvků v seznamech tahů nesouhlasí, volám další metody, které posunou hru do stavu, který jsem označil výběrem tahu.

3.4.4 Ukládání a otevírání hry

Pro ukládání aktuálně rozehrané nebo dokončené hry jsem zvolil serializaci kořenové instance třídy DispecerHry do binárního souboru. K otevírání uložené hry používám deserializaci uložené instance třídy DispecerHry. Serializací kořenové třídy mám zaručeno, že se mi serializuje (uchovají se stavy objektů) celá kolekce uzlů (instancí mých tříd) ve stromové struktuře mých tříd. Opětovnou deserializací se mi obnoví stavy objektů mé stromové struktury tak, jak jsem je uložil. Ukládání a otevírání partií hry mám ošetřeno v blocích try-catch-finally.

3.4.5 Vedlejší dialogová okna

Vedlejšími dialogovými okny jsou instance třídy Oaplikaci, třídy Vitezstvi a třídy NastaveniHracuNabidka, všechno to jsou potomci třídy Window. Instance těchto tříd se vytvářejí v průběhu hry v závislosti na volbách uživatele, kterému je dovoleno měnit za běhu hry parametry rozehrané partie hry, může si přát vědět kdo vytvořil aplikaci hry a většinou bývá zvědavý na konec hry kdo vyhrál nebo jestli byla remíza.

3.4.6 Klávesové zkratky

Volby hry jsou přístupné nejen pomocí počítačové myši, ale také pomocí klávesových zkratek. Hlavnímu oknu jsem tedy napsal ovladač události KeyDown, který snímá vstupy uživatele z klávesnice počítače.

3.4.7 Nápověda aplikace

Aplikace používá vestavěnou nápovědu, která je vyvolána kdykoliv v průběhu spuštění aplikace stiskem klávesy F1. Soubor s nápovědou nese název Bunnyhelp.chm a dodává se spolu se spouštěcím souborem aplikace Bunny.exe. Nápovědu spouštím jako samostatný proces, kterým je komponenta operačního systému Windows s názvem hh.exe, která si bere při spuštění jako parametr název souboru Bunnyhelp.chm. Tím se spustí program Nápověda HTML a otevře se soubor s nápovědou k mé aplikaci. Jednotlivá témata jsou v nápovědě přehledně rozdělena a uživatel si z nich může rychle vybrat co ho zajímá.

3.4.8 Ukončení aplikace

K ukončení aplikace obsluhuji událost hlavního okna MainWindow s názvem Closing. V ovladači této události mám ošetřenu možnost uložit rozehranou hru nebo vrátit se zpět. Pouze pokud nebyla zahájena žádná partie hry, provede se opuštění programu okamžitě. Při ukončování běhu programu vždy ukončuji případný externí proces, kterým může být otevřené okno s nápovědou ke hře.

4 Postup pro sestavení aplikace

Aplikaci hry Bunny War dodávám v jednom adresáři s názvem BunnyWar, který obsahuje soubor s názvem BunnyWar.exe, soubor s názvem Bunnyhelp.chm a dokumentaci ke hře v souboru s názvem BunnyWar dokumentace.pdf. Hra se spouští otevřením souboru BunnyWar.exe. Soubor Bunnyhelp.chm obsahuje nápovědu ke hře. Adresář BunnyWar mohu uživateli zaslat elektronickou cestou nebo na médiu CD.

Reference

[1] http://www.deskovehry.info/pravidla/bunnywar.htm, webová stránka s informacemi o deskové variantě hry Bunny War, 2016

[2] http://www.itnetwork.cz/csharp, výukové tutoriály o jazyku .NET/C#, 2016

[3] Radek Vystavěl, série učebnic Moderní programování, nakladatelství moderniProgramovani, s.r.o., 2007, 2008, 2009, 2015

[4] Tomáš Kühr, Algoritmy realizující počítačového hráče v jednoduchých deskových hrách, výukový text, KI PřF UPOL, 2011

[5] http://www.iconfinder.com, webová stránka, ze které lze stáhnout obrázky a ikony, 2016

[6] http://www.iconarchive.com, webová stránka, ze které lze stáhnout obrázky a ikony, 2016