UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2017 Jakub Jakoubek

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Šachové úlohy v Javě FX

Jakub Jakoubek

Bakalářská práce

2017

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2016/2017**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Jakub Jakoubek**

Osobní číslo: **I14106**

Studijní program: **B2646 Informační technologie**

Studijní obor: **Informační technologie**

Název tématu: **Šachové úlohy v Javě FX**

Zadávající katedra: **Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce je v Javě FX vytvořit grafickou aplikaci pro řešení šachových úloh. Součástí programu bude editor pro tvorbu nových úloh.

V teoretické části bude popsán vznik a význam šachových úloh. Dále budou porovnány již existující profesionální i volně dostupné programy pro řešení šachových úloh. Na závěr bude popsán způsob ukládání úloh s využitím technologie XML.

V praktické části bude vytvořena grafická JavaFX aplikace zobrazující šachové úlohy. Úlohy bude možné řešit přímo v aplikaci včetně zobrazení řešení. Také bude možné zadávat nové úlohy a řadit je do různých kategorií, například podle obtížnosti.

Rozsah grafických prací:

Rozsah pracovní zprávy: **cca 35 stran**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

POLGA?R, Zsuzsa a Hoainhan. TRUONG. Chess tactics for champions: a step-by step guide to using tactics and combinations. 1st ed. New York: Random House Puzzles & Games, c2006. ISBN 081293671X.

VOS, Johan. Pro JavaFX 8: a definitive guide to building desktop, mobile, and embedded Java clients / Johan Vos, Weiqi Gao, Stephen Chin, Dean Iverson, James Weaver. Berkeley, CA: Apress, 2014. Expert's voice in Java. ISBN 1430265744.

HAROLD, Elliotte Rusty. a W. Scott. MEANS. XML in a nutshell. 3rd ed. Sebastopol, CA: O'Reilly, c2004. In a nutshell (O'Reilly & Associates). ISBN 0596007647.

Vedoucí bakalářské práce: **Ing. Zdeněk Šilar, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2016**

Termín odevzdání bakalářské práce: **12. května 2017**

Univerzita Pardubice Česká republika

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

Mgr. Josef Horálek, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2017

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 5. 5. 2017

podpis autora
Jakub Jakoubek

ANOTACE

Práce se zabývá tvorbou programu, ve kterém lze řešit šachové úlohy a zadávat nové. Nejprve je řečeno něco o šachových úlohách a jsou představeny některé již existující programy zabývající se stejnou problematikou. Následně jsou popsané použité technologie. Nakonec je rozepsán návrh a implementace programu.

KLÍČOVÁ SLOVA

Šachy, šachové úlohy, Java FX, XML

TITLE

Chess problems in Java FX

ANNOTATION

This work deals with creation of a program which allows solving chess problems and creating new ones. At the begining, there is something about chess problems and there are introduced some already existing similar programs. Subsequently, the used technologies are described. Finally, design and implementation of the program are described.

KEYWORDS

Chess, chess problems, Java FX, XML

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

SDK	Software development kit
JVM	Java virtual machine
XML	Extensible markup language
JDOM	Java document object model
DOM	Document object model
SAX	Simple API for XML
PHP	PHP: Hypertext preprocessor, původně Personal home page
HTML	Hypertext markup language
CSS	Cascading style sheets
DTD	Document type definition

Úvod

Cílem bakalářské práce je za pomocí Javy FX vytvořit program umožňující řešit šachové úlohy. Součástí programu bude také nástroj k zadávání vlastních šachových úloh.

Šachům jsem se věnoval už od dětství. V dnešní době sice nemám čas se jim věnovat natolik, abych se v nich zlepšoval, ale šachové úlohy jsou stále mou nejoblíbenější součástí šachů. Rád si u nich procvičuji logické myšlení a také jsou pro mě dobrým odreagováním. To je důvod, proč jsem si zvolil jako téma bakalářské práce právě tvorbu programu pro řešení šachových úloh. Spojím při tom práci se zábavou a program budu moci poté využívat ve volném čase.

V první kapitole budou šachové úlohy popsané. Uvede se zde jak šachové úlohy vznikají a také jak přispívají šachové komunitě. Na konci kapitoly budou zmíněny ty nejčastější typy šachových úloh, a i pár kuriózních.

Jelikož podobné programy už existují, tak v druhé kapitole budou některé takové programy popsány a srovnány. Uvedený bude jeden profesionální program, jedna aplikace pro mobilní zařízení a jeden volně dostupný web s aplikací pro řešení šachových úloh.

Třetí kapitola shrne technologie a programy, které budou použity pro vývoj aplikace. Chci použít pouze volně dostupný software, to znamená žádné placené knihovny či nástroje.

Čtvrtá kapitola bude rozebírat návrh aplikace. V první části se popíše grafické rozhraní. To se bude skládat ze čtyř částí. Poté se popíše logika dvou nejnáročnějších procesů aplikace, a to řešení úlohy a zadávání nové úlohy.

Pátá kapitola popisuje implementaci jednotlivých částí aplikace. Začne se s grafickým rozhraním. Poté se uvedou jednotlivé třídy aplikace a k nim stručný popis. U větších tříd budou uvedeny ukázky kódu. Na závěr popsána práce se soubory, ve kterých se budou ukládat šachové úlohy.

V poslední kapitole bude popsáno ovládání jednotlivých částí aplikace. Tyto části zde budou také zobrazeny. Z této kapitoly by čtenář měl získat představu, jak se aplikace bude používat a jak vypadá.

1 ŠACHOVÉ ÚLOHY

Když se řekne šachová úloha, myslí se tím šachová pozice, ve které má hráč možnost konkrétní kombinací tahů získat výhodu, a to i když soupeř zahraje nejlepší tahy. Zmíněnou výhodou může být získání figury nebo přímo matová situace [1].

V některých případech hráč obětuje jednu či více figur k dosažení matové pozice. Ve zbylých případech pak hráč dočasně obětuje slabou figuru nebo dobrou pozici, aby získal cennou figuru. Častým nástrojem hráče k získání výhody je tah, který vytvoří více hrozeb najednou, které pak soupeř není chopen všechny pokrýt [1].

1.1 Vznik šachové úlohy

Ve většině případů šachová úloha zobrazuje reálnou pozici, která může během hry nastat. Z toho vyplývá, že mnoho úloh vzniká právě při běžné hře, nejčastěji pak na turnajích, kde jsou partie těch nejlepších hráčů sledovány a mnohdy i rozebírány šachovými mistry či velmistry.

Někteří pokročilý hráči, většinou trenéři, mohou využít svých zkušeností, aby připravili modelovou pozici vystihující konkrétní problém. Tím může být provedení matu, získání figury či dosažení nějakého jiného cíle.

1.2 Význam šachových úloh pro hráče

Šachové úlohy slouží jako cvičební pomůcka. Oproti procvičování hraním šachů, například po internetu či proti počítači, jsou šachové úlohy zaměřeny na konkrétní problematiku. Také jich šachista vyřeší za stejný čas mnohem víc, než by odehrál celých partií. Řešení šachových úloh se tedy dá brát jako koncentrovaný trénink. Jelikož se vždy jedná o konkrétní problémy s jedním řešením, tak se šachové úlohy uplatní i u amatérských hráčů šachu, kteří si pouze chtějí procvičit logické myšlení.

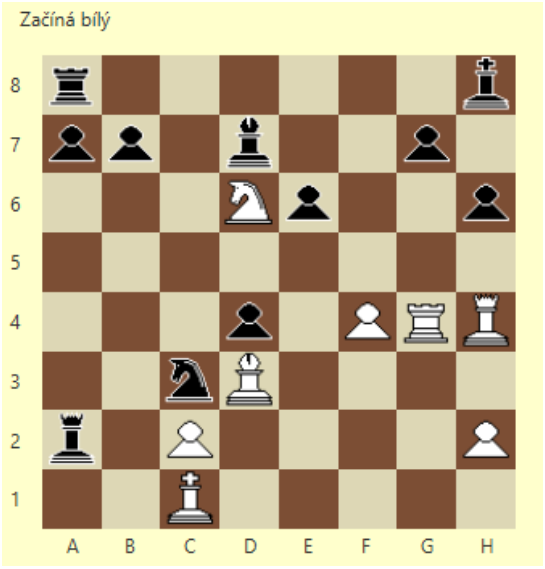
1.3 Typy šachových úloh

Typů šachových úloh existuje celá řada. Dále budou popsané ty nejčastější a také pár kuriózních.

1.3.1 Mat několikátým tahem

Cílem je zmatovat soupeře za použití co nejmenšího počtu tahů. Jako u všech úloh se i zde předpokládá, že soupeř zahraje vždy nejlepší možný tah. Některé tyto úlohy mohou být

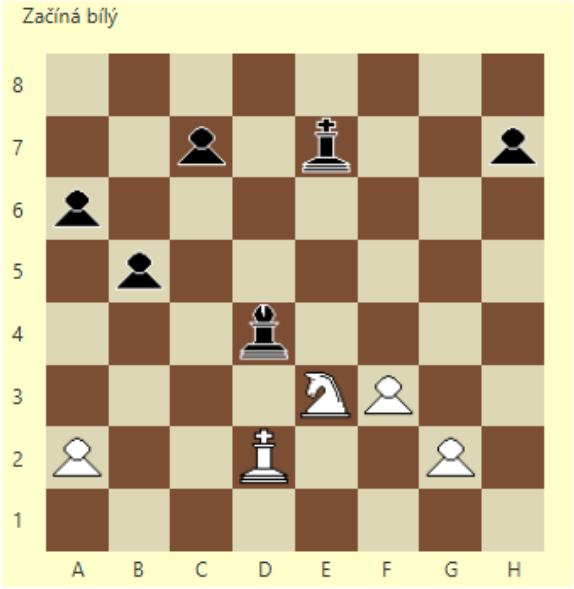
zaměřeny na konkrétní typ matu. Tím může být například dušený mat, kde jezdec matuje krále, který se zablokoval vlastními figurami.



Obrázek 1 - Šachová úloha: Mat 2. tahem

1.3.2 Zisk figury

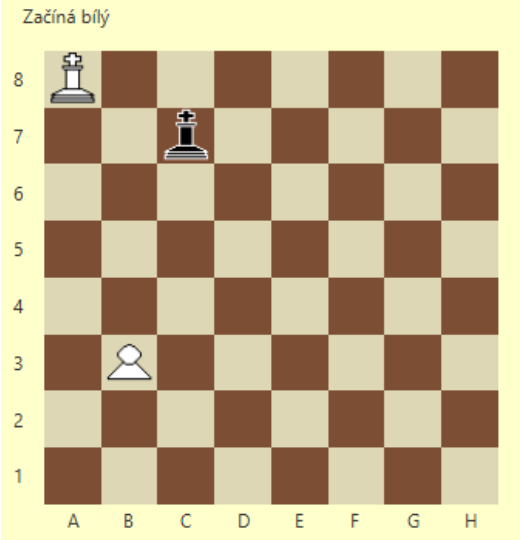
Dalším typickým zástupcem šachových úloh je úloha, kde se hráč snaží získat výhodu v podobě figury. Typicky se jedná o sebrání figury bez ztráty vlastní anebo s obětí nějaké slabší.



Obrázek 2 - Šachová úloha: Zisk figury

1.3.3 Pěšcová koncovka

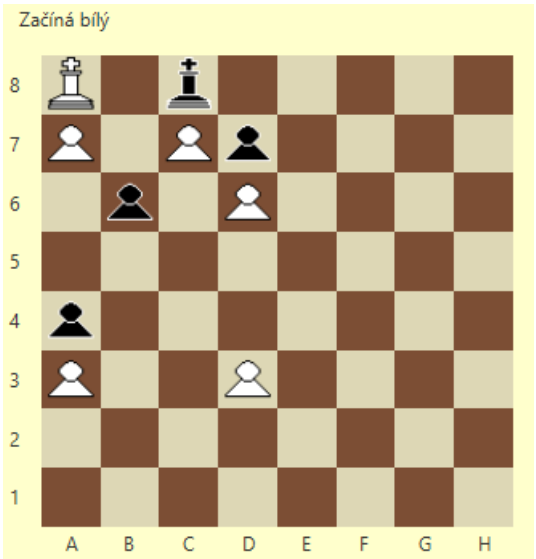
Pozice, ve které obě strany ztratily většinu nebo všechny silné figury. Cílem bývá protlačit pěšce na poslední řadu šachovnice, aby se mohl povýšit na silnější figuru. To v těchto případech zpravidla vede k výhře. Stejně jako u zisku figury zde již není potřeba dohrávat pozici do matu.



Obrázek 3 - Šachová úloha: Koncovka

1.3.4 Samo řešitelná úloha

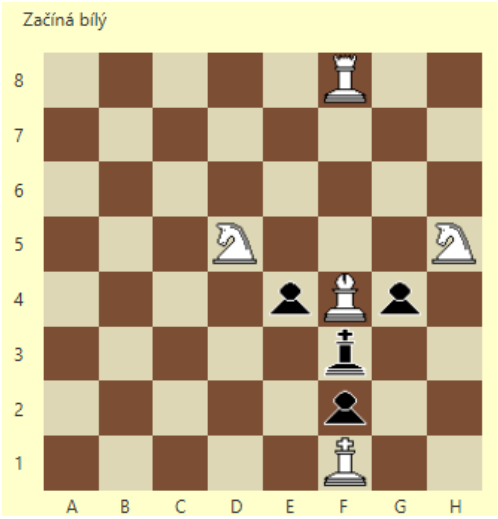
Jedná se o kuriózní pozici, ve které může hráč podle pravidel zahrát pouze jediný tah. Soupeř pak odehraje nejlepší (a většinou také jediný) možný tah. Ten hráče postaví do situace, ve které má zase pouze jediný možný tah. Řešením samo řešitelné úlohy je tedy sekvence vynucených tahů, které vedou k matu nebo patu.



Obrázek 4 - Šachová úloha: samo řešitelná

1.3.5 Samomat

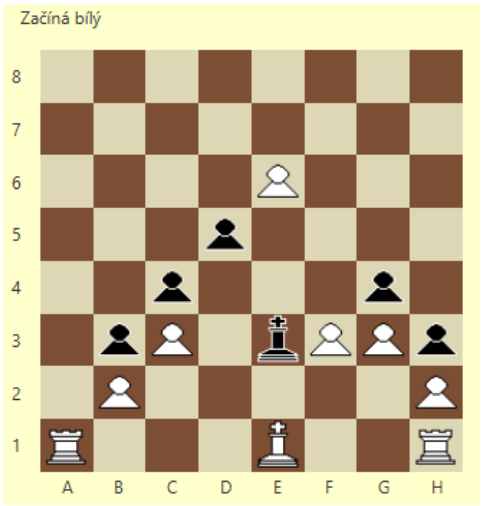
Další spíš kuriózní úlohou je takzvaný samomat. Cílem není vyhrát či získat výhodu nad soupeřem. Hráč má za úkol přinutit soupeře, aby vyhrál. Hráč tedy soupeře staví do pozice, ve které může provést vždy pouze jediný tah. Sekvence těchto tahů vedou k situaci, kdy soupeř dává hráči mat.



Obrázek 5 - Šachová úloha: Samomat

1.3.6 Úlohy využívající nedostatky v pravidlech

Tyto šachové úlohy se již neobjevují, ale kdysi vznikaly, aby poukázaly na nedostatky pravidel. V minulosti se například dalo provést rošádu za předpokladu, že král ani věž se nepohnuli. To vedlo k možnosti provést vertikální rošádu s věží, která vznikla povýšením pěšce na stejném sloupci jako je král. Dalším příkladem byla možnost povýšit svého pěšce na figuru soupeřovi barvy. Oba tyto nedostatky již byly ošetřeny.



Obrázek 6 - Šachová úloha: Využití vertikální rošády

2 VYBRANÝ SOFTWARE PRO ŘEŠENÍ ŠACHOVÝCH ÚLOH

V dnešní době existuje několik různých programů, ve kterých lze šachové úlohy řešit. V této části budou některé programy popsány a srovnány.

2.1 CT-ART 6.0

Profesionální světový program k řešení šachových úloh. Úlohy jsou řazeny do různých obtížností od úloh pro začátečníky až po úlohy vhodné pro mistry a velmistry šachu. Databáze programu obsahuje celkem přes 10 tisíc různých šachových úloh. Program se dodává v několika různých světových jazycích a cena se pohybuje okolo 30 $ [2].

Obrázek 7 - Okno programu CT-ART

2.2 iChess Pro - Chess Puzzles

Aplikace pro zařízení s operačním systémem android. Obsahuje přes tisíc šachových úloh a umožňuje přidávat vlastní šachové úlohy ve formátu PGN, který je rozšířeným formátem pro šachové úlohy na internetu. Cena programu je 1.99 $. K dispozici je i verze zdarma obsahující reklamy třetích stran [3].

Obrázek 8 - aplikace iChess Pro - Chess Puzzles

2.3 Chess.com

Chess.com je jedna s největších a nejpopulárnějších webových stránek zaměřených na šachy. Aktuálně má přes 17 a půl miliónů registrovaných uživatelů. Stránka nabízí velkou škálu nástrojů pro zlepšování dovedností v šachách, mimo jiné i řešení šachových úloh, kterých je momentálně na stránce přes 50 tisíc [4].

Obrázek 9 - Webová aplikace chess.com

2.4 Srovnání

Nejprofesionálněji je proveden program CT-ART, který je také díky tomu doporučován právě pro profesionální hráče šachu jako doplněk k tréninku. Na druhou stranu web chess.com je díky své dostupnosti, obrovské komunitě a stále se rozšiřujícímu počtu úloh nejlepší volbou jak pro začínající hráče, tak i pro pokročilé šachisty. Chess.com také nabízí aplikaci pro telefonní zařízení a je tedy mnohem lepší volbou, než iChess Pro - Chess Puzzles.

3 POUŽITÉ TECHNOLOGIE A SOFTWARE

K vývoji aplikace byly využity pouze volně dostupné technologie a nástroje, které jsou v následujících podkapitolách popsány.

3.1 Java FX

Java FX je soubor grafických a mediálních balíčků, které umožňují vývojáři navrhovat, vytvářet, testovat a ladit bohaté klientské aplikace. Grafika se pak ukládá pomocí FXML souboru. Ten lze snadno vytvářet například pomocí programu JavaFX Scene Builder [5].

V roce 2008 vyšlo Java FX 1.0 SDK. O rok později vyšla verze 1.2 a v dubnu 2010 pak vyšla verze 1.3. Verze 2.0, která vyšla v roce 2011, umožňovala využití v libovolném jazyce, který běží na JVM. Největším milníkem byl příchod Javy 8, do které je již Java FX přímo integrovaná [6].

JavaFX vznikla jako nástupce pro jiné, již existující, grafické balíčky Javy, a to Swing. JavaFX nabízí lepší zpracování událostí, možnost stylování pomocí CSS, podporu moderních dotykových zařízení (včetně multi-dotykových displejů), integrování webového prohlížeče a mnoho dalšího. Od Javy 8 je také možné rozšířit existující swing aplikace o Java FX prvky [5].

3.2 XML

XML je rozšiřovatelný značkovací jazyk. Vznikl pro ukládání a přesouvání dat. Hlavní výhodou je, že je takzvaně samo popisující. S tím souvisí i to, že nemá pevně dané tagy. Je tedy možné si vytvořit libovolně pojmenovaný tag. Data se ukládají jako čistý text, takže soubor typu xml je možné snadno editovat pomocí libovolného textového editoru [7].

Díky jednoduchosti ukládání je velmi snadné xml soubory číst, a to jak uživatelem, tak libovolným programem. Většina moderních programovacích jazyků již nabízí nástroje pro usnadnění práce s xml soubory a pro programátora je tedy méně časově náročné implementovat čtení či zápis daného xml dokumentu. Proto se také XML využívá často právě k přesunu dat mezi jednotlivými aplikacemi [7].

Snadná možnost editovat xml soubor ručně uživatelem může vést k problémům s validací souboru. Uživatel může například udělat překlep v názvu tagu, vynechat nějaký atribut nebo přidat tag, který není pro daný program pochopitelný. Tomu se dá předejít pomocí DTD nebo XML Schématu. V obou dvou případech se jedná o soubor definující strukturu xml souboru,

tedy názvy a strukturu tagů, názvy atributů a datové typy hodnot atributů. Rozdíl mezi nimi je, že XML Schéma je psané ve formátu xml zatímco DTD využívá vlastní formát [7].

3.2.1 JDOM

JDOM je volně dostupná Java knihovna zaměřená na práci s XML dokumenty. Nabízí nástroje pro efektivní čtení, manipulaci a zápis XML dokumentů. Jedná se o alternativu k DOM a SAX knihovnám. JDOM je aktuálně k dispozici ve verzi 2.0.6 [8].

3.3 FXML

FXML je značkovací jazyk vytvořený na základě jazyka XML. Tento jazyk vznikl pro usnadnění tvorby grafického prostředí pro JavaFX aplikace. Napsáním grafického prostředí do fxml souboru se oddělí od aplikační logiky, čímž zpřehlední kód. O interakci mezi grafickým rozhraním a jádrem aplikace se pak stará speciální Java třída implementující javafx.fxml.Initializable [9].

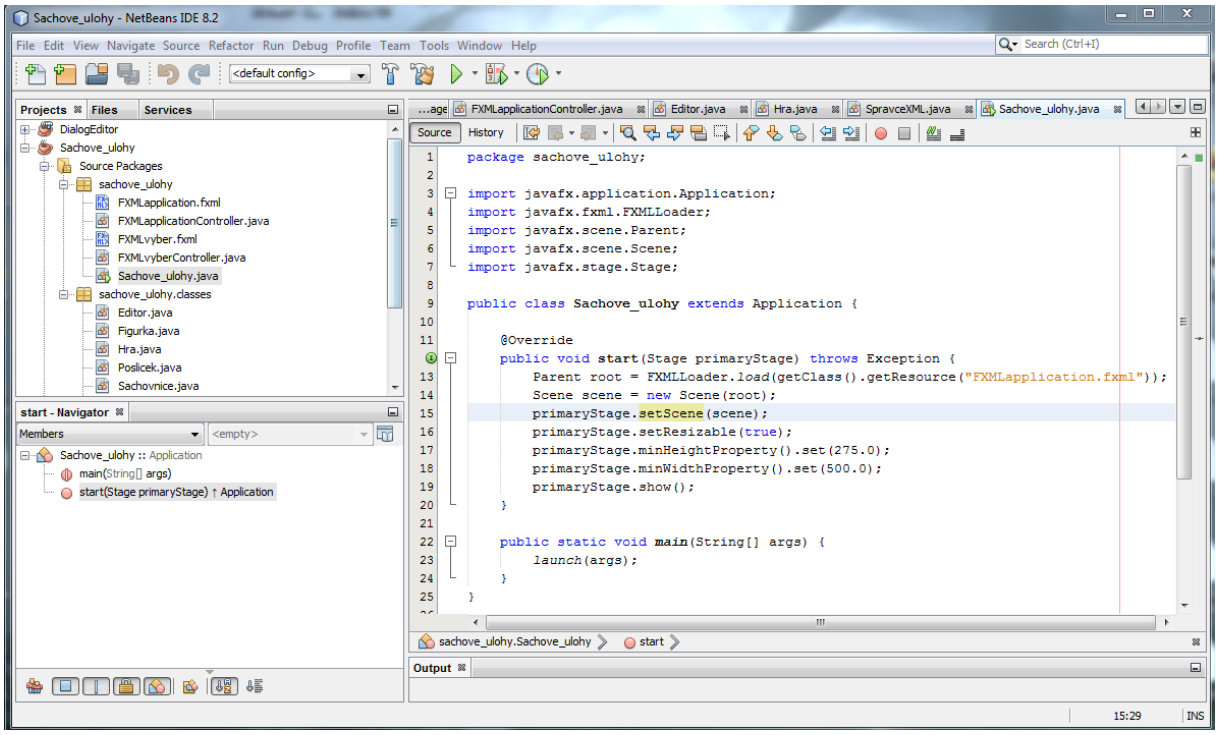
3.4 NetBeans

NetBeans je volně dostupné vývojové prostředí se zveřejněnými zdrojovými kódy. Umožňuje vytvářet aplikace i webové stránky v mnoha různých jazycích, například: C, C++, HTML, JavaScript, PHP. NetBeans je také oficiálním vývojovým prostředím pro Javu 8 [10].

Vývojové prostředí usnadňuje programování v mnoha ohledech. Zvýrazňuje zdrojový kód jak syntakticky, tak sémanticky. Také zvýrazňuje páry závorek a umožňuje snadno zformátovat kód, aby byl čitelnější. Další pomůckou je automatické napovídání a popřípadě i doplňování kódu. V neposlední řadě průběžně kontroluje kód, jestli se v něm nenachází chyba znemožňující kompilaci a také uživateli zobrazuje užitečné tipy, například že má v kódu nevyužitý import [10].

V NetBeans se také dá snadno orientovat v struktuře programu. V panelu navigace se dá vyobrazit jak struktura balíčků a tříd, tak i všechny soubory v daném projektu. Dalším nástrojem pro správu programu je vestavěná podpora systémů Git, Subversion a dalších. Díky tomu je možné vyvíjet program i ve větším počtu lidí [10].

Pro vývoj aplikace byl použit NetBeans ve verzi 8.2. Na oficiálních stránkách je k dispozici mnoho různých balíčků a doplňků, ale k vývoji postačil základní balíček Java SE, který podporuje technologie Java SE a Java FX.



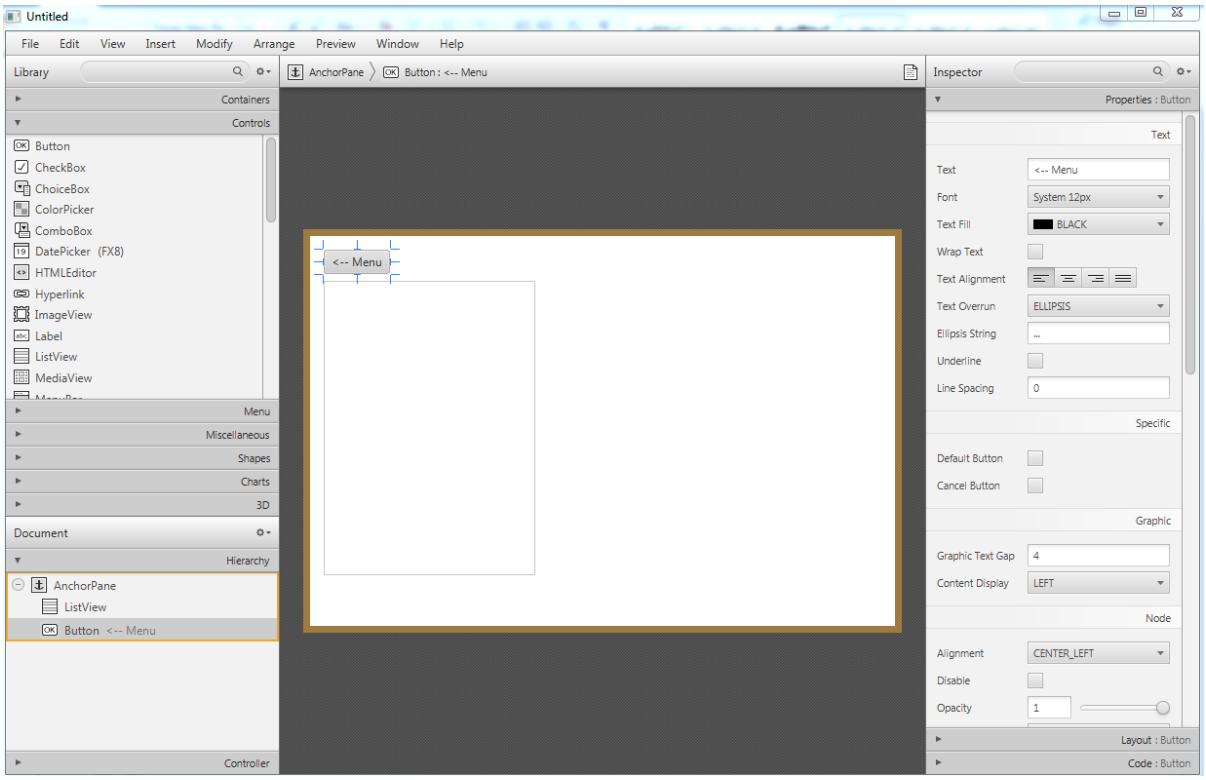
Obrázek 10 - Vývojové prostředí NetBeans

3.5 JavaFX SceneBuilder

JavaFX SceneBuilder je visuální nástroj, který umožňuje snadné navrhování grafických rozhraní pro JavaFX aplikace. Uživatel pomocí metody „uchop a polož“ rozmístí grafické prvky a poté jim může upravit nastavení. Také jim může přiřadit název, pod kterým budou prvky dostupné v příslušném kontroléru. Výstupem tohoto nástroje je fxml dokument [11].

JavaFX SceneBuilder je napsán v jazyce Java, za použití Java FX, a je tedy spustitelný jak na operačních systémech windows, tak na Mac OS X nebo Linuxu. Další výhodou je propojení s vývojovým prostředím NetBeans, které umožňuje snadné provázání uživatelského rozhraní se zdrojovým kódem aplikace [11].

K vývoji aplikace byla použita verze JavaFX SceneBuilder 2.0. Vývoj tohoto nástroje se zastavil, ale v dnešní době se vyvíjí téměř totožný nástroj zvaný Gluon Scene Builder, který je také volně dostupný a má zveřejněné i své zdrojové kódy. Nejnovější verze je 8.3.0, které vyšla 16. prosince 2016 [12].



Obrázek 11 – JavaFX SceneBuilder 2.0

4 NÁVRH APLIKACE

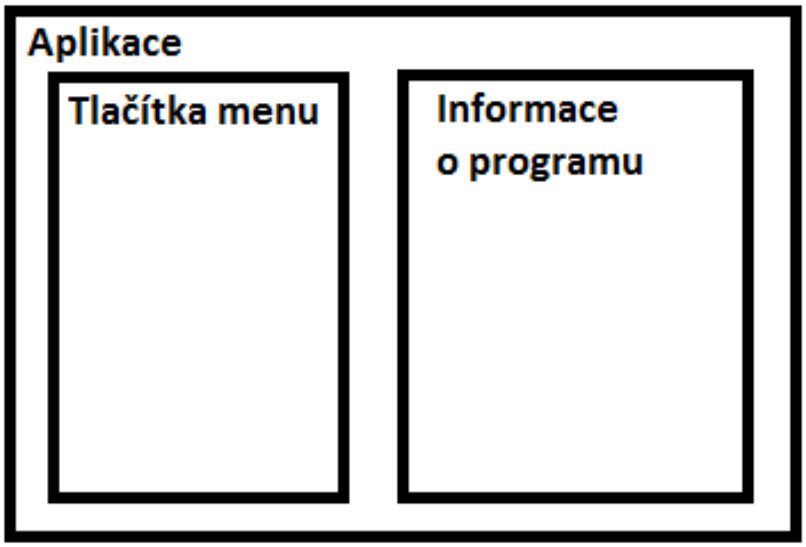
Než se aplikace vytvoří, je třeba si promyslet, jak bude vypadat a jak se bude přibližně chovat. V této kapitole bude obojí podrobněji popsáno.

4.1 Uživatelské rozhraní

Uživatelské rozhraní bude rozděleno do čtyř částí: menu, řešitelská část, zadávací část a výběr úlohy.

4.1.1 Menu

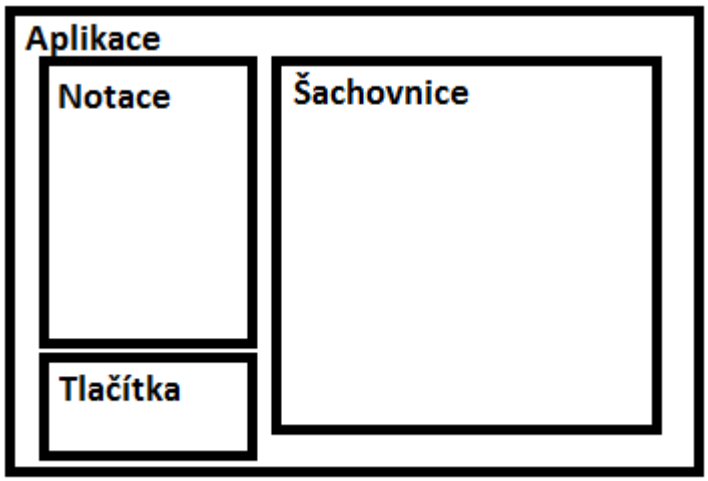
Úvodní okno, které se zobrazí při startu programu. Okno bude malé a nebude možné změnit jeho velikost. V pravé části budou zobrazeny informace o programu: název, rok, autor. V levé části budou funkční tlačítka: otevřít sadu, upravit sadu, smazat sadu, ukončit program.



Obrázek 12 - Rozložení menu

4.1.2 Řešitelská část

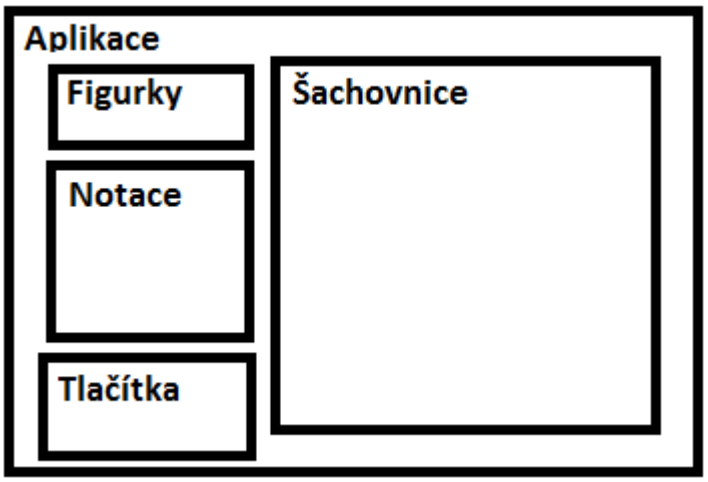
Hlavní okno aplikace, které bude sloužit k řešení načtených úloh. Velikost okna bude možné libovolně měnit. V levé části záznam provedených tahu a pod ním budou funkční tlačítka: restartovat úlohu, další tah, další úloha, vybrat úlohu. V pravé části se bude nacházet interaktivní šachovnice.



Obrázek 13 - Rozložení řešitelské části

4.1.3 Zadávací část

Okno, ve kterém bude možné zadávat nové úlohy. Velikost bude možné libovolně měnit. V pravé části bude interaktivní šachovnice. V levé části budou jednotlivé figurky. Dále tam bude historie provedených tahů s tlačítkem pro vrácení posledního tahu, a nakonec tam budou funkční tlačítka: uložit úlohu a zahodit úlohu.



Obrázek 14 - Rozložení zadávací části

4.1.4 Výběr úlohy

Okno pro výběr úlohy bude zastávat dvě činnosti. Za prvé bude umožňovat výběr konkrétní úlohy k řešení a za druhé bude umožňovat správu upravované sady úloh. Většinu okna zabere list, který bude obsahovat seznam načtených úloh. Ve spodní části pak budou funkční tlačítka: nová úloha, smazat úlohu nebo vybrat úlohu.

Obrázek 15 - Rozložení výběru úloh

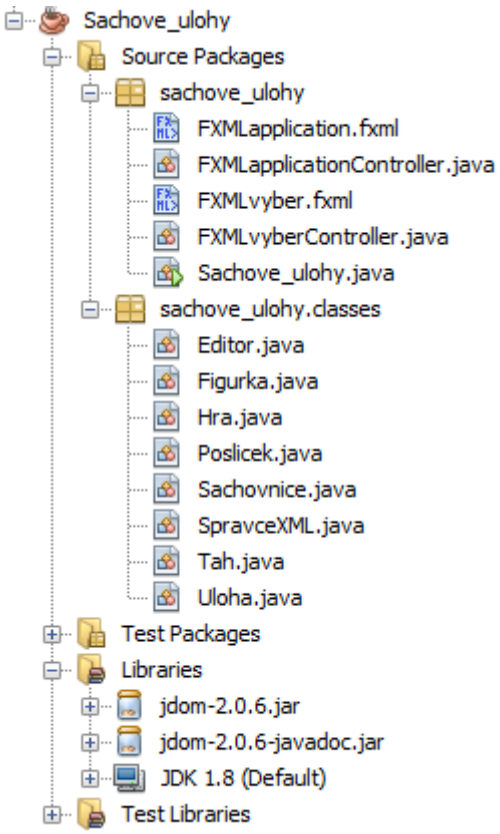
4.2 Logika programu

Úloha se bude řešit pomocí klikání myší na jednotlivá políčka šachovnice či figurky. Logika provedení tahu je znázorněna diagramem v příloze A. Program bude vyžadovat provedení správného tahu uvedeného v úloze, jakýkoliv jiný tah bude označen za špatný a neprovede se. Po provedení úspěšného tahu se buď automaticky provede tah soupeře anebo, pokud se jednalo o poslední tah úlohy, bude uživatel upozorněn na úspěšné dokončení úlohy.

Pro vytváření úlohy budou sloužit tlačítka jednotlivých figurek a šachovnice obdobně jako u řešení úlohy. Diagram v příloze B zobrazuje zpracování kliknutí na šachovnici během vytváření nové úlohy. Nejprve se budou zadávat jednotlivé figurky na šachovnici. Až bude počáteční rozestavení připraveno, tak se začnou vytvářet jednotlivé tahy.

5 IMPLEMENTACE APLIKACE

V této kapitole bude popsána implementace vyvíjené aplikace. To zahrnuje tvorbu grafického prostředí, popis jednotlivých tříd a na závěr práci se soubory. Spouštěcí třída a kontroléry pro uživatelské rozhraní se nachází v hlavním balíčku programu a ostatní třídy se nachází v balíčku *classes*, který je součástí hlavního balíčku.



Obrázek 16 - Struktura programu

5.1 GUI

Grafické rozhraní aplikace je tvořeno dvěma fxml soubory. Tyto soubory jsou pak provázané se speciálními Java třídami, takzvanými kontroléry. Ty zajišťují interakci mezi prvky grafické aplikace (například tlačítky) a jádrem programu. Většina grafických prvků je staticky vytvořena ve výše zmiňovaných souborech v programu JavaFX SceneBuilder.

5.1.1 Dynamicky generované prvky

Součástí grafického rozhraní jsou také dynamicky generované prvky. To jsou prvky, které byly vytvořeny až po spuštění programu. V tomto případě se jedná o šachovnici, která se skládá ze dvou dvourozměrných polí ImageView. První vrstva vykreslí jednotlivá políčka

šachovnice, druhá pak vykresluje jednotlivé figurky. Dále je vygenerován popisek šachovnice (označení řádků a sloupců).

5.1.2 Škálovatelnost okna

Důležitou vlastností grafického rozhraní aplikace je responzivní design. Pokud se aplikace nachází ve stavu řešení úlohy anebo vytváření úlohy, je možné měnit libovolně velikost okna, a to od rozlišení 500×275 až po rozlišení obrazovky. Prvek zobrazující provedené tahy pak přizpůsobuje svou výšku oknu aplikace. Šachovnice také mění svoji velikost, a to na nejvyšší možnou s ohledem na zachování poměru stran.

5.2 Hlavní třídy

Třídy *Hra*, *Editor* a *SpravceXML* tvoří jádro programu. Jedná se o nejrozsáhlejší třídy programu, které zajišťují správu zásadních částí programu: řešení úlohy, vytváření úlohy a práci se soubory.

5.2.1 Třída *Hra*

Úkolem třídy *Hra* je zajištění chodu programu během řešení šachové úlohy. To zahrnuje rozestavení úlohy, kontrolu správnosti tahu, grafické provedení tahů a vypisování historie tahů (notaci). Třída obsahuje ukazatele na grafické prvky (šachovnice, notace a text stavu), stavové flagy (hráč na tahu, figurka vybrána a úloha vyřešena) a instanci aktuálně řešené úlohy.

Mezi veřejné funkce patří nastavení úlohy, restartování úlohy, nápověda tahu a zpracování kliknutí na šachovnici. Funkce kliknutí pak může volat ještě privátní funkce (kontrola tahu, provedení tahu hráče a provedení tahu soupeře).

```
public void nastavUlohu(Uloha uloha) {
    aktualniTah = 0;
    aktualniUloha = uloha;
    cernyNaTahu = uloha.zacinaCerny;
    jeVybranaFigurka = false;
    sachovnice.skliditSachovnici();
    notace.getItems().clear();
    uloha.rozestaveni().forEach((figurka) -> {
        sachovnice.nastavFigurku(figurka.radek, figurka.sloupec,
                figurka.typ, figurka.jeCerna);
    });
    ulohaVyresena = false;
    nazev.setText(aktualniUloha.getNazev());
    status.setText("Začíná " +
            (aktualniUloha.zacinaCerny ? "černý" : "bílý"));
}
```

5.2.2 Třída *Editor*

Třída *Editor* zprostředkovává nástroj pro tvorbu nové úlohy. V první části obstarává přípravu počátečního postavení úlohy. Následně zaznamenává provedené tahy a na konci novou úlohu sestaví. Třída obsahuje ukazatele na grafické prvky (šachovnice, notaci a text stavu), stavové flagy (figurka vybrána, rozestavení uloženo, bílý/černý král postaven a začínající hráč) a kolekce rozestavených figurek, provedených tahů a historii tahů (provedené změny).

```
  public Uloha sestavitUlohu() {
        if (postaveniUlozeno && tahy.size() > 0) {
            List<Figurka> figurkyProUlohu = new ArrayList<>();
            figurkyProUlohu.addAll(figurky);
            List<Tah> tahyProUlohu = new ArrayList<>();
            tahyProUlohu.addAll(tahy);
            return new Uloha(figurkyProUlohu, tahyProUlohu, zacinaCerny,
nazevUlohy);
        } else {
            return null;
        }
    }
```

5.2.3 Třída *SpravceXML*

Třída *SpravceXML* obstarává veškerou práci se soubory xml. Ke své činnosti využívá třídy knihovny *jdom2*. Instance udržuje načtený soubor a kolekci načtených úloh. Mezi hlavní funkce patří načtení, uložení a vytvoření souboru.

```
    public boolean nactiSoubor(File soubor){
        try {
            souborXML=cteni.build(soubor);
            this.soubor=soubor;
            prectiUlohy();
        } catch (JDOMException | IOException ex) {
           return false;
        }
        return ulohy.size()>0;
    }
```

5.3 Vedlejší třídy

Následující třídy usnadňují implementaci jednotlivých částí programu. Jedná se o pomocné třídy reprezentující jednotlivé prvky aplikace (šachová figurka, tah atd.).

5.3.1 Třída *Figurka*

Instance třídy *Figurka* v programu reprezentuje jednu šachovou figurku. *Figurka* obsahuje veřejný výčet *Typ*, který obsahuje všechny typy šachových figurek a k tomu speciální typ „žádná“. *Figurka* o sobě zachovává informace o svém typu, barvě, obrázku a pozici na šachovnici.

5.3.2 Třída *Tah*

Instance třídy *Tah* v programu odpovídá jednomu provedenému tahu, a to jak při řešení úlohy, tak při zadávání nové úlohy. Třída *Tah* obsahuje veřejný výčet *TypPohybu*, který obsahuje následující položky: pohyb, braní, braní mimochodem, malá rošáda a velká rošáda. Tah o sobě zachovává svůj typ, typ figurky, počáteční pozici, cílovou pozici a případně typ figurky pro povýšení.

5.3.3 Třída *Uloha*

Instance třídy *Uloha* odpovídá jedné šachové úloze. *Úloha* obsahuje kolekci figurek odpovídající počátečnímu rozestavení úlohy, kolekci tahů odpovídající správnému řešení úlohy, barvu hráče, který začíná a informaci o svém názvu. *Úloha* také umožňuje získat konkrétní *Tah* podle jeho indexu, tato funkce se využívá při řešení úlohy.

5.3.4 Třída *Sachovnice*

Třída *Sachovnice* slouží k správě grafického zobrazení šachovnice. Uvnitř třídy *Sachovnice* je privátní třída *Policko*, které obsahuje ukazatel na *ImageView* šachového políčka, typ figurky a její barvu. Šachovnice pak obsahuje dvourozměrné pole 8 × 8 políček.

S šachovnicí jde manipulovat pomocí funkcí přidání figurky na políčko, odstranění figurky a sklizení celé šachovnice. Dále je možné získat typ figurky na daném políčku anebo zda je na daném políčku figurka dané barvy. Pro účely editoru je také možné získat kolekci všech figurek ležících na šachovnici.

5.3.5 Třída *Poslicek*

Instance třídy *Poslicek* slouží ke komunikaci mezi hlavním oknem aplikace a oknem pro výběr úlohy. Hlavní okno předává název souboru načtené sady, instanci třídy *SpravceXML* a informaci, zda se má výběrové okno otevřít pro výběr úlohy nebo pro správu sady. Výběrové okno pak předává index vybrané úlohy nebo upozornění, že uživatel chce vytvořit novou úlohu.

5.4 Práce se souborem úloh

Úlohy se zapisují do souborů xml. Jeden soubor (takzvaná sada úloh) může obsahovat libovolné množství šachových úloh.

5.4.1 Struktura XML souboru

Základem dokumentu je párový tag root. Ten obsahuje párové tagy uloha, které v atributech udržují informaci o svém názvu a o začínajícím hráči. Dále obsahuje párové tagy postaveni

a tahy. Tag postaveni udržuje nepárové tagy figurka, které mají atributy typ, barva a pozice. Tag tahy obsahuje nepárové tagy tah, které v atributech udržují informace o typu figurky, počáteční pozici, cílové pozici, typu pohybu a nepovinou informaci o povýšení pěšce.

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <Uloha nazev="Mat 3. tahem" zacina="b">
    <postaveni>
      <figurka typ="S" barva="b" pozice="e6" />
      <figurka typ="K" barva="b" pozice="c5" />
      <figurka typ="V" barva="b" pozice="h4" />
      <figurka typ="S" barva="w" pozice="d3" />
      <figurka typ="J" barva="w" pozice="f3" />
      <figurka typ="J" barva="b" pozice="a2" />
      <figurka typ="P" barva="w" pozice="b2" />
      <figurka typ="P" barva="w" pozice="c2" />
      <figurka typ="V" barva="w" pozice="a1" />
      <figurka typ="K" barva="w" pozice="b1" />
    </postaveni>
    <tahy>
      <tah fig="V" od="h4" do="h1" typ="pohyb" />
      <tah fig="J" od="f3" do="g1" typ="pohyb" />
      <tah fig="V" od="h1" do="g1" typ="brani" />
      <tah fig="S" od="d3" do="f1" typ="pohyb" />
      <tah fig="V" od="g1" do="f1" typ="brani" />
    </tahy>
  </Uloha>
</root>
```

5.4.2 Vytvoření sady úloh

Při vytváření souboru je nejprve uživatel vyzván pro zadání názvu. Následně je vytvořen soubor xml se zadaným názvem ve složce ulohy. V souboru je vytvořen základní tag root. Následně je soubor připraven pro ukládání nových úloh.

5.4.3 Načítání sady úloh

Z otevřeného souboru se pomocí třídy *jdom2.SAXBuilder* vytvoří *jdom2.Dokument*, což je stromová struktura elementů. Následně se spustí funkce pro přeložení dokumentu do kolekce úloh. S tou pak program může pracovat (číst úlohy pro řešení nebo přidávat nové z editoru).

Při otevření dialogového okna pro výběr souboru se zobrazí složka /ulohy nacházející se ve stejném adresáři jako aplikace. To však neomezuje uživatele pouze pro tento adresář, může si tedy otevřít libovolný xml soubor ve svém počítači či na výměnném médiu. Pokud se pro účely řešení úloh načte soubor, který bude poškozen nebo nebude obsahovat žádné řešitelné úlohy, tak bude uživatel upozorněn, že daný soubor nelze načíst.

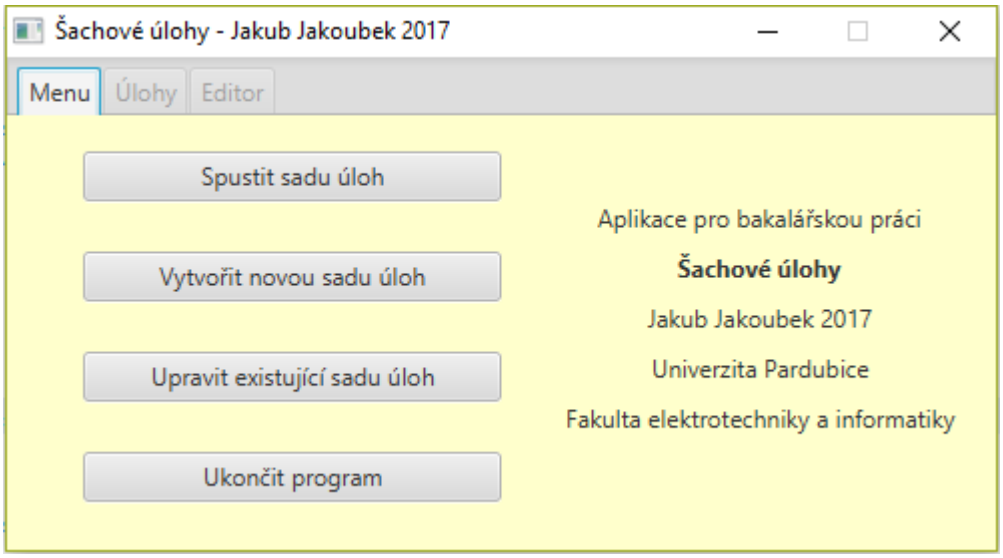
5.4.4 Ukládání sady úloh

O ukládání úloh do souboru se stará třída *jdom2.XMLOutputter*. Nejprve se přeloží celá kolekce úloh do *jdom2.Dokumentu* a poté se pomocí výše zmíněné třídy uloží do otevřeného xml souboru. Ukládání je voláno při každé změně v kolekci úloh, tedy při smazání úlohy nebo při přidání nové.

Ukládání probíhá metodou přepsání souboru. Soubor se tedy pokaždé smaže a znovu se do něj uloží veškeré informace. Soubor se neudržuje otevřený po celou dobu programu, ale otevírá se, jen když je potřeba. Může tedy nastat, že bude daný soubor změněn třetí stranou během doby, kdy se bude upravovat v aplikaci. Pro aplikaci to problém nebude, protože při přidání nebo smazání úlohy se do souboru uloží všechny úlohy uchovávané v aplikaci. Díky tomu se ztratí veškeré změny provedené třetí stranou.

6 OVLÁDÁNÍ APLIKACE

Po spuštění aplikace se otevře okno s úvodní nabídkou. Tlačítko *Spustit sadu úloh* otevře dialogové okno pro výběr xml souboru a při úspěšném načtení přepne aplikaci do okna pro řešení úlohy. Tlačítko *Vytvořit novou sadu úloh* otevře malé dialogové okno pro zadání názvu sady. Poté se otevře okno se seznamem úloh (okno výběru). Tlačítko *Upravit existující úlohu* otevře dialogové okno pro výběr xml souboru. Po úspěšném načtení se otevře okno se seznamem úloh v dané sadě. Poslední tlačítko *Ukončit program* ukončí celou aplikaci.



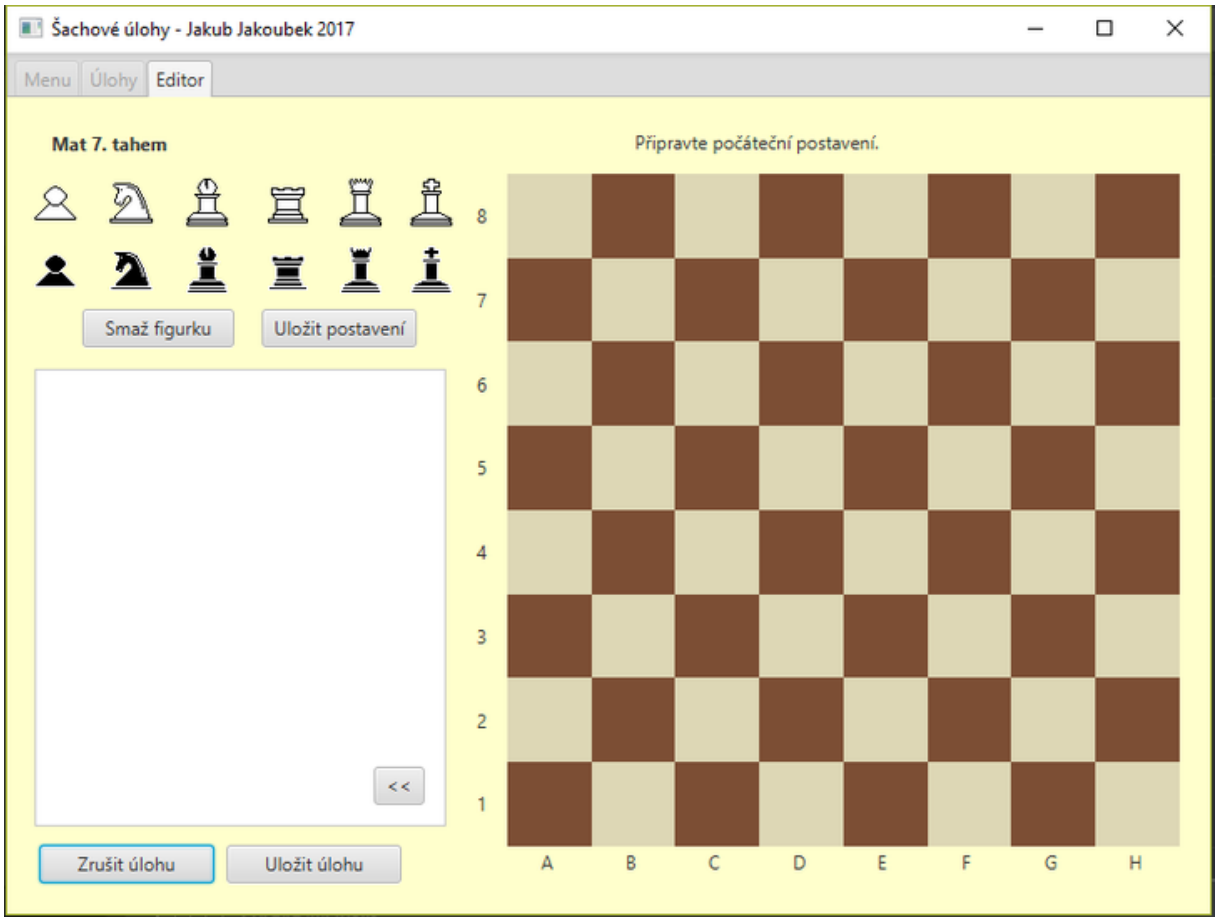
Obrázek 17 - Okno aplikace: úvodní nabídka

Okno pro řešení úlohy obsahuje tlačítko pro navrácení zpátky do hlavní nabídky, dvě tlačítka pro navigaci mezi tahy (*Restart* a *Další tah*) a dvě tlačítka pro navigaci mezi úlohami (*Další úloha* a *Vybrat úlohu*). Tlačítko *Vybrat úlohu* otevře okno se seznamem úloh. V pravé části okna se nachází šachovnice s načtenou úlohou. Tahy se provádějí kliknutím na políčko s figurkou a následným kliknutím na cílové políčko. Tah se provede pouze tehdy, pokud takto uživatel zadá správný tah definovaný v úloze.

Obrázek 18 - Okno aplikace: řešení úlohy

Okno pro zadávání nové úlohy obsahuje paletu figurek. Kliknutím na figurku se daná figurka zvolí a následným klikáním na políčka šachovnice se daná figurka pokládá. V případě, že se na daném políčku nachází figurka krále, je uživatel upozorněn, že krále nelze odstranit, pouze přesunout. Po stisknutí tlačítka *Smaž figurku* se zapne mód mazání. Klikáním na figurky na šachovnici budou tyto figurky odstraněny. Tlačítko Ulož postavení uloží rozestavěné figurky a umožní začít zadávat tahy.

Tahy se zadávají obdobně, jako se řeší úloha. Rozdíl je, že je zde potřeba zadávat tahy i soupeře. Barvu hráče aplikace detekuje automaticky podle prvního provedeného tahu. Aplikace nekontroluje nemožné tahy, proto je u listu provedených tahů tlačítko, které umožňuje vrátit poslední provedený tah. Úlohu lze poté uložit tlačítkem *Uložit úlohu* anebo zahodit tlačítkem *Zrušit úlohu*. Obě možnosti otevřou opět okno se seznamem úloh.



Obrázek 19 - Okno aplikace: zadávání nové úlohy

Okno se seznamem úloh je možné spustit ve dvou režimech. Režim výběru slouží pouze k vybrání úlohy pro řešení. V tomto režimu je aktivní tlačítko *Vybrat* pro potvrzení výběru. Druhý režim slouží k úpravě sady a jsou zde aktivní tlačítka *Nová* a *Smazat*. Tlačítko *Nová* spustí okno pro zadávání nové úlohy. Tlačítkem *Smazat* je možné odstranit vybranou úlohu ze seznamu. V obou režimech se v levém horním rohu nachází tlačítko pro návrat do hlavní nabídky.

Obrázek 20 - Okno aplikace: seznam úloh

Závěr

V rámci bakalářské práce jsem vytvořil program umožňující řešit šachové úlohy. Program umožňuje šachové úlohy načítat ze souboru, řešit je, i zadávat nové. Program tedy splňuje základní požadavky uvedené v zadání práce.

Program má intuitivní ovládání a je celý v češtině. Řešit v něm šachové úlohy tedy může každý, kdo zná pravidla šachů. Teoreticky je ani nemusí znát dokonale, ale v takovém případě by se mu správné tahy hledaly hůř. Pro zadávání úloh už je znalost pravidel nutná, protože program kontroluje dodržení pouze některých pravidel.

Obrázky šachových figur jsem si vytvořil sám ve volně dostupném rastrovém grafickém editoru Gimp. Rád pracuji s rastrovou grafikou, ale příště bych pro tento účel zvolil vektorovou grafiku. Program jsem totiž vytvořil tak, že se dá libovolně měnit velikost okna, a při velkém zvětšení figurky nevypadají tak dobře.

Prostoru pro zlepšení je víc. První, co bych vylepšil, je okno s řešením úlohy. Konkrétně by se dala přidat animace pohybu figurky a zvýraznit políčka, na která daná figurka může. Další část, kterou bych ještě rád zlepšil, je zadávání úloh. Zde by bylo vhodné implementovat důkladnější kontrolu pravidel. Poslední, co bych přidal, je podpora některých rozšířených formátů pro ukládání šachových úloh. Všechny tyto vylepšení mě napadly již při vytváření programu, ale vzhledem k časové náročnosti jsem je nebyl schopen implementovat v rámci této práce.

Během tvorby programu jsem si vyzkoušel, jaké to je pracovat sám na praktickém projektu. To zahrnuje vytvoření návrhu, naprogramování všech požadovaných funkcí i vytvoření vlastní grafiky. Také jsem se naučil něco nového o práci s Javou FX a o práci se soubory XML.

S výsledkem práce jsem velmi spokojen. Už během vývoje někteří mí přátelé projevili o tento program zájem. Rád bych si tedy našel čas program zlepšit a poté bych jim ho rád dal k dispozici.

Použitá literatura

[1] POLGAR, Zsuzsa a Hoainhan, TRUONG. *Chess tactics for champions: a step-by step guide to using tactics and combinations.* 1st ed. New York: Random House Puzzles & Games, c2006. ISBN 081293671X.

[2] Buy or Upgrade to CT-ART 6.0 with Peshka 2.0!. *ChessOK* [online]. Moscow: Convekta, ©2016 [cit. 2017-04-02]. Dostupné z: http://chessok.com/?p=29369

[3] IChess Pro - Chess Puzzles. *AppCrawlr* [online]. Barcelona: SOFTONIC INTERNATIONAL S.A., ©2015 [cit. 2017-04-02]. Dostupné z: http://appcrawlr.com/android/ichess-tactics-ad-free

[4] *Chess.com* [online]. Stanford: Erik and Jay, ©2017 [cit. 2017-04-04]. Dostupné z: https://www.chess.com

[5] What Is JavaFX? *ORACLE* [online]. California: Oracle Corporation, ©2013 [cit. 2017-04-09]. Dostupné z: http://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm

[6] VOS, Johan. *Pro JavaFX 8: a definitive guide to building desktop, mobile, and embedded Java* CA: Apress, 2014. Expert's voice in Java. ISBN 1430265744.

[7] Introduction to XML. *W3schools.com* [online]. Sandnes: Refsnes Data, ©2017 [cit. 2017-04-09]. Dostupné z: https://www.w3schools.com/xml/xml_whatis.asp

[8] Frequently Asked Questions. *Jdom.org* [online]. JDOM project, ©2017 [cit. 2017-04-10]. Dostupné z: http://www.jdom.org/docs/faq.html

[9] Why Use FXML. *ORACLE* [online]. California: Oracle Corporation, ©2014 [cit. 2017-04-15]. Dostupné z: https://docs.oracle.com/javase/8/javafx/fxml-tutorial/why_use_fxml.htm

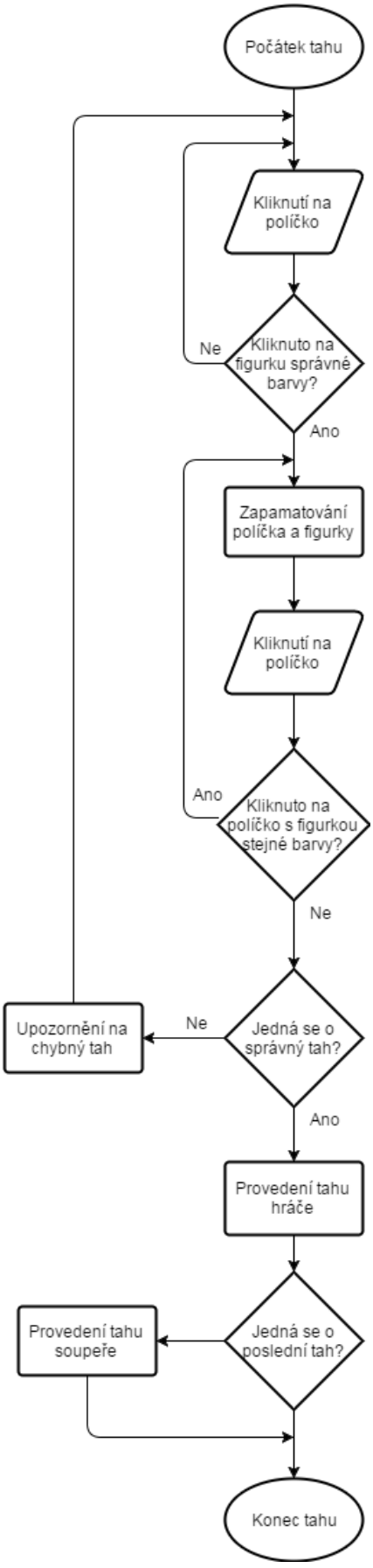
[10] NetBeans IDE - The Smarter and Faster Way to Code. *NetBeans* [online]. California: Oracle Corporation, ©2017 [cit. 2017-04-24]. Dostupné z: https://netbeans.org/features/index.html

[11] JavaFX Scene Builder. *ORACLE* [online]. California: Oracle Corporation, ©2016 [cit. 2017-04-27]. Dostupné z: http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html

[12] Scene Builder. *Gluon* [online]. Gluon, ©2017 [cit. 2017-04-29]. Dostupné z: http://gluonhq.com/products/scene-builder/

Přílohy

Příloha A – *Diagram provedení tahu při řešení úlohy*



Příloha B – *Diagram zpracování kliknutí při zadávání úlohy*