STŘEDNÍ PRŮMYSLOVÁ ŠKOLA BRNO,
PURKYŇOVA, PŘÍSPĚVKOVÁ ORGANIZACE

Tetris

TOMÁŠ JUKELSON

V4A

Profilová část maturitní zkoušky
MATURITNÍ PRÁCE

BRNO 2021

Prohlášení

Prohlašuji, že jsem maturitní práci Tetris vypracoval samostatně a použil jen zdroje uvedené v seznamu literatury.

Prohlašuji, že:

Beru na vědomí, že zpráva o řešení maturitní práce a základní dokumentace k aplikaci bude uložena v elektronické podobě na intranetu SPŠ Brno, Purkyňova, příspěvková organizace.

Beru na vědomí, že bude má maturitní práce včetně zdrojových kódů uložena v knihovně SPŠ Brno, Purkyňova, příspěvková organizace, dostupná k prezenčnímu nahlédnutí. Škola zajistí, že nebude pro nikoho možné pořizovat kopie jakékoliv části práce.

Beru na vědomí, že SPŠ Brno, Purkyňova, příspěvková organizace, má právo celou moji práci použít k výukovým účelům a po mém souhlasu nevýdělečně moji práci užít ke své vnitřní potřebě.

Beru na vědomí, že pokud je součástí mojí práce jakýkoliv softwarový produkt, považují se za součást práce i zdrojové kódy, které jsou předmětem maturitní práce, případně soubory, ze kterých se práce skládá. Součástí práce není cizí ani vlastní software, který je pouze využíván za přesně definovaných podmínek, a není podstatou maturitní práce.

Tomáš Jukelson

Brechtova 1

V Brně dne 17. 4. 2021 .....

Vedoucí práce: RNDr. Lenka Hrušková

Odborný konzultant:

Anotace

Tato maturitní práce se zabývá vytvořením 2D hry Tetris. V této hře ihned po spuštění se dostane uživatel do hlavního menu, jež dává uživateli možnost zadat svoji přezdívku, pod kterou bude jeho skóre zaznamenané během hraní uloženo; spustit hru; změnit nastavení klávesových zkratek pro ovládání padajících obrazců; podívat se na SB. Se zahájením hry se spouští i klasický soundtrack Tetrisu a obrazce již začínají padat. Hráč má možnost obrazce posouvat vlevo i vpravo, otáčet je a urychlit jejich pád za což je odměněn v podobě přidání bodů do jeho skóre. Pokud hráč naskládá obrazce tak, aby vyplnili celý řádek hrací plochy, tento řádek se smaže a hráči je připočten velký počet bodů. Hra končí v momentě, kdy se nové obrazce nemůžou ihned po vytvoření posunout dolů. Během této hry se může hráč kdykoliv rozhodnout pozastavit hru a následně buďto pokračovat nebo se může vrátit do hlavního menu, a tak předčasně ukončit hru. Pokud se takto rozhodne, jeho progres je uložen a může si jej prohlédnout ve SB, který ukazuje nejlepších 5 zaznamenaných výkonů hráčů na tomto počítači.

Obsah

Teoretický úvod

Tetris je jedna z prvních 2D počítačových her. Hra spočívá v tom, že se na horní straně hrací plochy objevují obraze o objemu čtyř kostek, které postupně padají na spodek hrací plochy. Hráč má možnost během pádu obrazce ovládat posunováním těchto objektů vlevo, vpravo, otáčením či zrychlením pádu. Pokaždé kdy obrazce dopadnou až na dno hrací plochy, či obrazec, který leží v cestě ztratí hráč kontrolu nad tímto objektem a nahoře se objevuje další obrazec, kterým může uživatel hýbat.

V budoucnosti plánuji programovat více počítačových her, a tak jsem si chtěl ozkoušet na vlastní pěst, co vše je třeba udělat a znát ke zhotovení hry s poměrně jednoduchými pravidly.

Existuje více možných řešení, jak zrealizovat Tetris. Je možné užití jazyků Python, C++ nebo programu Unity či popřípadě HTML a PHP, pokud bych si zvolil, že aplikace má být hratelná na internetové stránce a mnoho dalších způsobů.

Mým cílem bylo úplně od základu jen za pomocí Javy vytvořit aplikaci, co nejpodobnější hře Tetris a během toho si zapamatovat svůj postup abych své zkušenosti mohl použít v budoucnosti.

Seznam použitých zkratek

SB – score board

GUI – Graphical User Interface

UI – User Interface

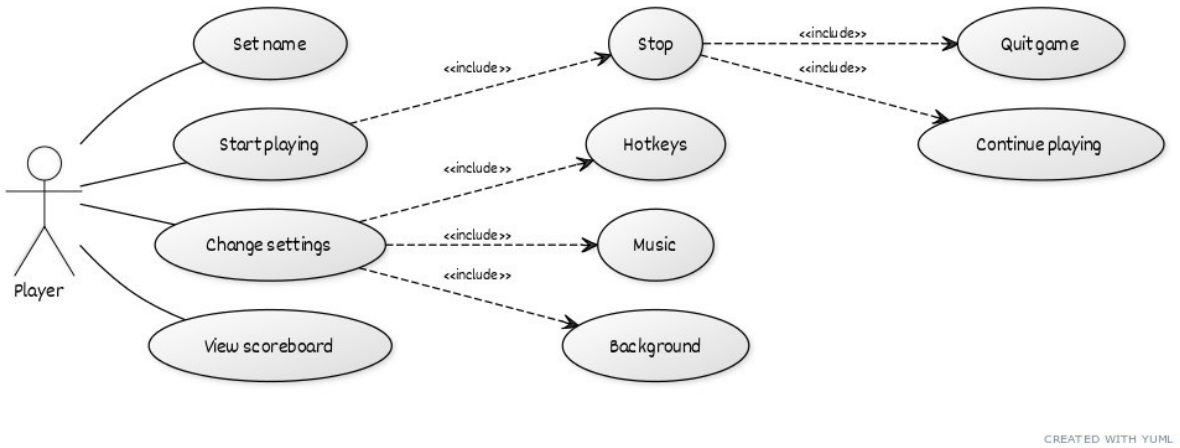
CSS – Cascading Style Sheets

UML – Unified Modeling Language

1 Rozbor řešení

Je nutné vytvořit uživatelsky přívětivé a intuitivní GUI. Stejný gameplay Tetris, možnost změny nastavení pro ovládání obrazců a možnost se následně podívat na své nejlepší dosažené skóre.

1 Návrh použití



Ilustrace 1: UML diagram

1.1 Hlavní menu

První, co uživatel uvidí po spuštění jakékoliv by vždy mělo být hlavní menu, kde má uživatel nastavení hotkeys a nickname, před spuštěním hry. Zároveň by mu neměla být odepřena možnost bez jakýchkoliv prodlev hru spustit. Pro přehlednost a intuitivitu dává největší smysl umístit možnost pohledu na top skóre právě zde – v hlavním menu

1.1.1 Tlačítko 'Play'

Velké nepřehlédnutelné tlačítko Play by zpravidla mělo být umístěno tam, kam se oči uživatele upnou hned po zapnutí programu. Toto tlačítko přenese hráče rovnou do hry Tetris.

1.1.2 Tlačítko 'Change settings'

Změna nastavení je druhá nejdůležitější funkce (hned po tlačítku 'Play'), proto jsem jej umístil hned pod tlačítko 'Play'. Pokud jej bude uživatel hledat, najde ho téměř ihned. Toto tlačítko umožní uživateli změnu hudby, barvy pozadí a hotkeys pro ovládání obrazců.

1.1.3 Tlačítko 'View scoreboard'

Pohled na své skóre není klíčové pro hru, proto jsem jej umístil jako nejnižší z tlačítek. Toto tlačítko umožní uživateli se podívat na svých 5 nejvyšších dosažených bodů během času stráveném v této hře.

1.1.4 Pole pro vepsání jména

Textové vstupní pole s průhledným textem „Your name:“, který zmizí po započnutí psaní do tohoto pole má být intuitivní způsob, jak si uživatel uloží své jméno, pod kterým si přeje mít uložené skóre po skončení jeho hry. Pro přehlednost jsem toto textové vstupní pole umístil právě nad tlačítko 'Play'

1.2 Gameplayová část

Hlavní část je hrací pole 12 na 30. Vpravo od hracího pole může být čárou oddělená část, kde se zobrazuje, jaké má zrovna hráč skóre. Zároveň je třeba umístit Stop button tak, aby neblokoval pohled na hrací plochu a zbytečně ztěžoval hraní.

Po ukončení hry způsobem prohry mi přišlo vhodné zajistit, aby se přes velkou část obrazovky zobrazilo velké písmo s textem “Game Over”.

Nově vytvořené obrazce se vždy objeví na uprostřed nejvyššího řádku hrací plochy. Tyto obrazce budou padat na spodek plochy tak, že se budou posunovat vždy o vzdálenost 1 čtverečku hrací plochy každý časový interval, který můžu buďto zůstat v průběhu celé hry stejný nebo jak to bývá u většiny verzí Tetrisu – čas v každém intervalu se bude postupně zmenšovat, aby se tak hra stávala těžší s každou sekundou hraní. Další velmi důležitá pravidlo Tetrisu je, že pokud hráč naskládá obrazce tak, aby utvořili plný řádek bez mezer, daný řádek zmizí. Hráč může obrazce posouvat vlevo a vpravo, otáčet je a urychlit jejich pád. Kontrolu nad obrazcem hráč ztrácí v momentě, kdy se obrazec dotkne dna hrací plochy nebo obrazce, který leží v cestě. Zároveň tak také dostává kontrolu nad nově vytvořeným obrazcem na vrcholu hrací plochy. Během celého průběhu gameplaye by měl hrát Tetris soundtrack.

1.3 Změna nastavení

Podstatná část, kterou by měla mít každá verze Tetrisu, jež nabízí možnost změny nastavení, je změna hotkeys pro ovládání obrazců. Další možné funkce, které toto submenu může nabízet, je výběr hudby, která bude v průběhu gameplaye hrát nebo změna pozadí.

1.4 Scoreboard

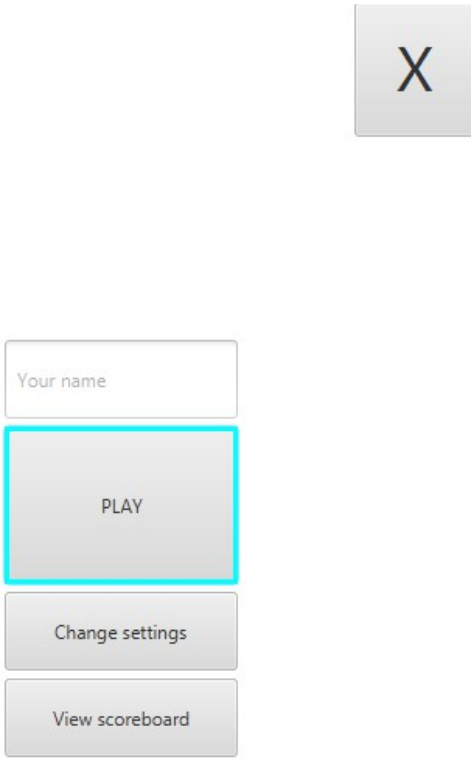
Scoreboard je část programu, která umožní uživateli pohled na jeho dosavadní úspěchy v této hře. Často bývají informace na SB zapsané ve formátu pořadí/nickname/skóre. Je možné dát uživateli možnost si prohlédnout **všechny** zapsané průběhy jeho her za pomoci například scroll baru, ale mě přišlo rozumnější zobrazit jenom **prvních pět** nejlepších průběhů, neboť si myslím, že málokoho budou zajímat úspěchy sotva o pět bodů větší než 0. Navíc mi to přijde takto přehlednější.

2 Realizace

Program jsem se rozhodl vytvářit v JavaFX - Javou implementované GUI

2.1 Hlavní menu

Ihned po spuštění programu se vygeneruje hlavní menu. To zahrnuje tlačítka 'Play', 'Change settings', 'View scoreboard', vstupní pole s textem “Your name“ a tlačítko ukončující program. Jak jsem již zmínil, tlačítko 'Play' je největší a zvýrazněné (v tomto případě je zvýrazněné tyrkysovým ohraničením.

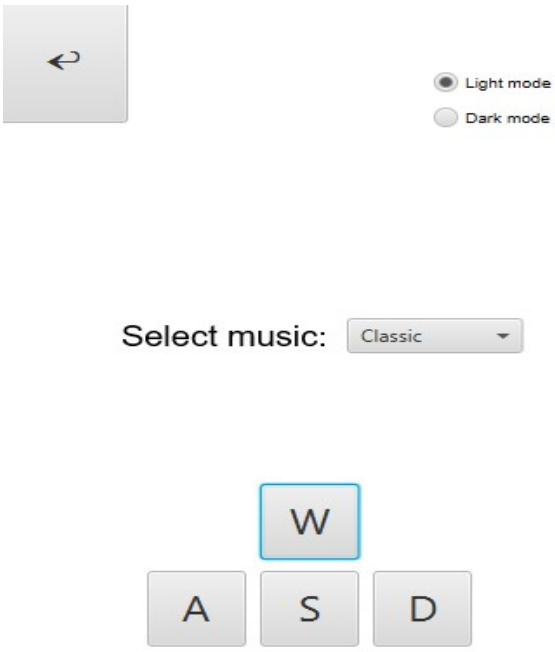


Ilustrace 2: Main menu

2.2 Changing settings

Po stisknutí tlačítka 'Change settings' umístěném v hlavním menu se z obrazovky smažou nynější textová pole a tlačítko a na místo toho se vygeneruje submenu pro změnu nastavení obsahující tlačítko 'Return to main menu', které opět smaže vše co toto submenu obsahuje a znovu se vygeneruje hlavní menu. Toto submenu obsahuje 4 klávesy, kterýma

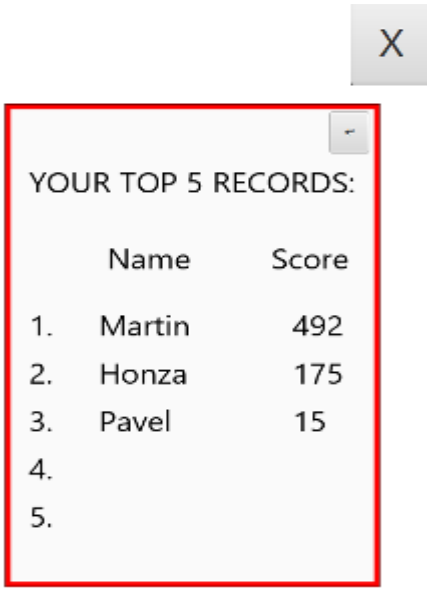
může hráč ovládat padající obrazce. Po stisknutí těchto kláves na obrazovce dostane uživatel možnost změnu daného hotkey. Uživatel si zde může také zvolit barvu pozadí a typ Tetris soundtracku.



Ilustrace 3: Changing settings

2.3 View scoreboard

Po stisknutí tlačítka 'Change settings' umístěném v hlavním menu se vygeneruje rámeček s rudým ohraničením obsahují dosavadní záznamy her hráčů a tlačítko pro opětovné schování SB. Jak jsem již zmiňoval, SB zobrazí pouze nejlepších pět záznamů.



Ilustrace 4: Scoreboard

2.3.1 Způsob uložení a čtení skóre

Po ukončení hry (ať už vlastnoručním navrácením do hlavního manu nebo prohrou) se do jednoho arraye uloží dosažené skóre a do druhého arraye se uloží nick pod kterým hráč hrál. Array se skórem se ihned začne řadit. Existuje nespočet řadících algoritmů. Mě přišlo nejlogičtější použít Bubble sort, neboť potřebuji správně seřadit pouze prvních **pět** míst. A Bubble sort řadí tak, že jeho první seřazená čísla jsou právě ta nejvyšší (nebo nejnižší – podle toho jakým směrem řadíme). Z tohoto důvodu můžeme cyklus řazení přerušit kdykoliv potřebujeme a ta důležitá místa budou mít správnou hodnotu.

```
private static void Sort() {
    System.out.println("Sorting started");
    int helper = 0;
    int bottom = 0;
    for (int i=0; i<5; i++) { //only top 5 needed
        for (int j=Main.users; j > 0+bottom; j--) {
            System.out.println("one mini cycle started");
            if (Main.savedScore_numbers[j] > Main.savedScore_numbers[j - 1]) {
                helper = Main.savedScore_numbers[j];
                Main.savedScore_numbers[j] = Main.savedScore_numbers[j - 1];
                Main.savedScore_numbers[j - 1] = helper;
                nameSwap(j);
            }
        }
        bottom++;
        if (i+1 > Main.users) {
            break;
        }
    }
}

private static void nameSwap(int index){
    String temp;

    temp = Main.savedScore_names[index];
    Main.savedScore_names[index] = Main.savedScore_names[index-1];
    Main.savedScore_names[index-1] = temp;
}
```

Ilustrace 5: Bubble sort

Pokaždé když prohodím místy hodnoty arraye skóre, prohodíme stejná místa i v arrayi jmen. Po seřazení arrayí, přepisuji informaci do textových souborů. Takto docílím zachování informací i po vypnutí programu.

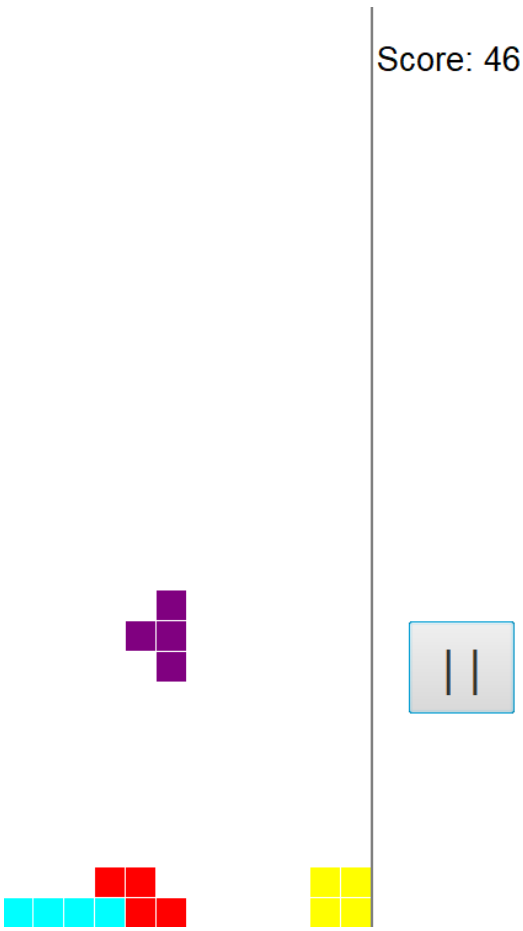
Teď můžu při každém následném spuštění programu načíst minulé záznamy hry a přidávat do arrayí a následně tak můžu pracovat i se staršími daty.

```
public static void recordScore() {

    //dont sort only 1 score
    if (Main.users != 0) {
        try {
            Sort();
        }
        catch (Exception e) {
            System.out.println(e + "\n Skipping the problem");
        }
    }

    try {
        FileWriter myWriter = new FileWriter( fileName: "scoredatabase.txt");
        for (int i = 0; i<Main.savedScore_numbers.length; i++) {
            if (Main.savedScore_names[i] != null) {
                myWriter.write(String.valueOf(Main.savedScore_numbers[i]));
                myWriter.write( str: "\n");
            }
        }
        myWriter.close();
        System.out.println("Recording of your score was successful.");
    } catch (IOException e) {
        System.out.println("Couldn't record your score.");
        e.printStackTrace();
    }

    try {
        FileWriter myWriter = new FileWriter( fileName: "namedatabase.txt");
        for (int i = 0; i<Main.savedScore_names.length; i++) {
            if (Main.savedScore_names[i] != null) {
                myWriter.write(Main.savedScore_names[i]);
                myWriter.write( str: "\n");
            }
        }
        myWriter.close();
        System.out.println("Recording of your name was successful.");
    } catch (IOException e) {
        System.out.println("Couldn't record your name.");
        e.printStackTrace();
    }
}
```

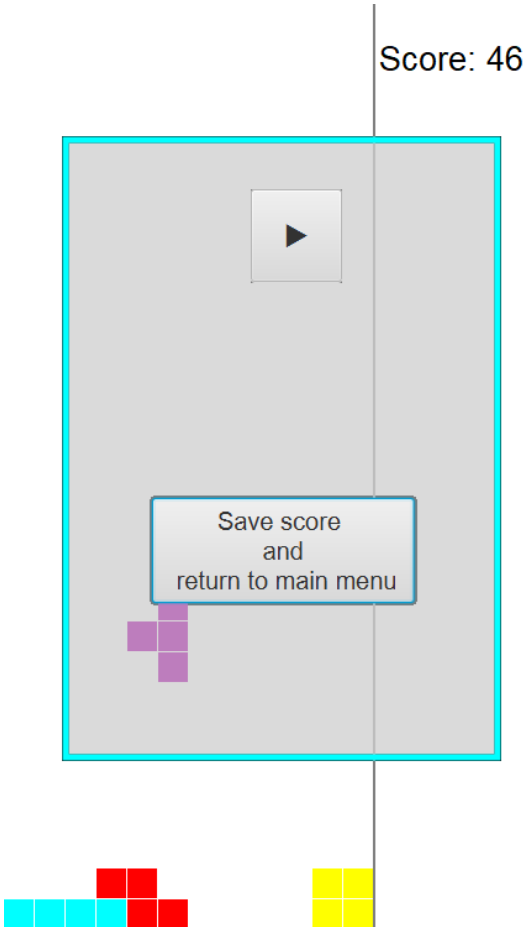
Ilustrace 6: Zapisování do souborů

2.4 Gameplay

Po stisknutí klávesy 'Play' v hlavním menu začíná ta gameplayová část Tetrisu. Vygeneruje se hrací pole 12 na 30 a začíná hrát hudba. Na pravé části obrazovky se zobrazí nynější skóre a pozastavovací stop. Pokud hráč stiskne tlačítko, hra bude pozastavena a hráč si může následně zvolit, zda-li chce pokračovat ve hraní nebo jestli chce hru ukončit.



Ilustrace 8: Gameplay



Ilustrace 7: Gameplay stopped

Na vrcholu hrací plochy se také objeví obrazec o objemu čtyř čtverečků. Pokaždé když se obrazec posune o jeden čtvereček níže, tak se zjišťuji, jestli se nějaké řady nenaplnila. Pokud se nějaké řady naplnili, tak se smažou všechny čtverečky, které tuto řadu utvářely. A přičte se hráčovi 100 bodů za každou odstraněnou řadu. Jestli že na té řadě ležely nějaké čtverečky, budou všechny přesunuty níže tak, aby se dotýkali buďto dna pole nebo jiného čtverečku. Hra přidává hráči 1 bod za každé vlastnoruční posunutí obrazce níže. Tetris tak odměňuje hráče s rychlým myšlením. Pokud se obrazce naskládají tak, že se dotýkají vrcholu hracího pole a další obrazce se pak nemají kde objevit – hra končí.

```
playing = true;

//repeated refreshing of game status
TimerTask task = () -> {
        Platform.runLater(new Runnable() {
            public void run() {

                if (object.a.getY() == 0 || object.b.getY() == 0 || object.c.getY() == 0 || object.d.getY() == 0)
                    top++;
                else
                    top = 0;

                if (top == 2) {
                    // GAME OVER
                    Text over = new Text("GAME OVER");
                    over.setFill(Color.RED);
                    over.setStyle("-fx-font: 70 arial;");
                    over.setY(250);
                    over.setX(10);
                    group.getChildren().add(over);
                    playing = false;
                    cancel();
                }

                if (playing) {
                    group.getChildren().removeAll(scoretext);
                    group.getChildren().addAll(scoretext);
                    Moving.MoveDown(object);
                    scoretext.setText("Score: " + Integer.toString(score));
                }
            }
        });
};
//repeating task in a set period
if (!threadIsRunning) {
    threadIsRunning = true;
    Timer fall = new Timer();
    fall.schedule(task, delay: 0, period: 300);
}
```

Ilustrace 9: Thread

Zde je ukázano jak vypadá vlákno, které neustále kontroluje, zda-li se nějaký obrazec nedotýká vrcholu hrací plochy, které což by vedlo k vynucenému ukončení hry. Zároveň s každým uběhlým intervalem, kdy se obrazec posuny o jeden čtvereček níže, se refreshuje ukazatel skóre.

Závěr

Jediný nedostatek, který má mnou vytvořený Tetris oproti jiným verzím je, že se mi nepodařilo vymyslet instantní pád obrazce (obvykle pomocí mezerníku). Kromě toho se mi podařilo dodržet zásadní funkce jako je například pohyb obrazců, mazání plných řad, výpočet skóre a změnu nastavení pro lepší pohodlnost uživatele.

Během své práce jsem si rozšířil znalosti ohledná práce s JavaFX a zároveň jsem využil většinu svých dosavadních zkušeností jako jsou způsoby řazení, vlákna a užívání externích souborů.

Zároveň jsem si dobře uvědomil postup při vytváření her. To zahrnuje například: kterou část hry programovat prvně, časté možné chyby a nejlepší způsoby řešení těchto chyb a v neposlední řadě přehlednost a intuitivitu programu.

Seznam ilustrací

Zdroje

[1] *JavaFX 2D Shapes* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[2] *JavaFX Text* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[3] *JavaFX Transformation* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[4] *JavaFX UI* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[5] *JavaFX CSS* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[6] *Media with JavaFX* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[7] *JavaFX Event Handling* [online]. Dostupné z: https://www.javatpoint.com/javafx-tutorial

[8] *Java Multithreading* [online]. Dostupné z: https://www.tutorialspoint.com/java/

[9] *Java Arrays* [online]. Dostupné z: https://www.tutorialspoint.com/java/