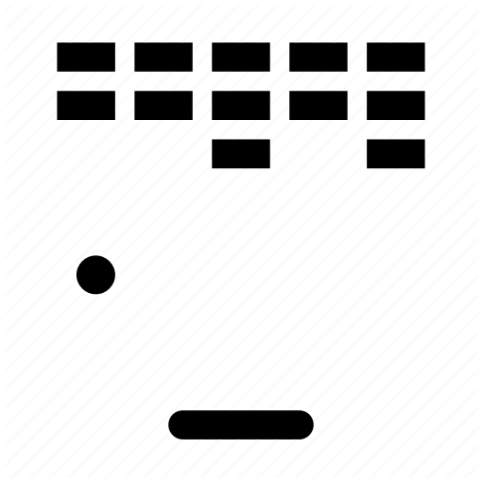
Gymnázium Praha 6, Arabská 14

předmět Programování

Brick Breaker

ročníkový projekt



Anna Šimánková, 1E

květen 2021

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne 3.5.2021 Anna Šimánková

Anotace:

Tato práce obsahuje obecné informace, ale i bližší postup jak při sestavování kódu, tak úprav v grafickém prostředí scenebuilder, které vedly k vytvoření počítačové hry brick breaker. Jsou zde dva hlavní objekty, prvním je spodní deska, kterou hráč ovládá klávesnicí a snaží se zachytit a odrazit druhý hlavní objekt kuličku. Kulička lítá v prostoru a vráží do horních obdélníků tak, že když se nějakého dotkne, tak daný obdélník zmizí. Pokud se nám povede „sestřelit“ všechny horní objekty tak je hra vyhrána, když vám ale kulička spadne na spodní okraj 3x a vy ji nezachytíte posuvným obdélníkem, pak jste prohráli.

Název práce: brick breaker

Autor: Anna Šimánková

Zadání projektu:

Počítačová hra, kdy se hráč, pomocí kuličky odražené z desky, kterou posouvá v horizontálním směru, snaží zasáhnout jednotlivé cihly různé barvy. Hráč začíná se třemi životy, život ztratí, když kulička skončí na spodním okraji obrazovky. Hra končí úspěšně, pokud kulička dosáhne horního okraje obrazovky. Hra bude jednoúrovňová. Barvy cihel budou různé v rozsahu standardních RGB barev (max 4 barvy). Hra začne tlačítkem Start new game z nabídkového menu, další funkce bude restart hry a Exit, ukončení.

Obsah

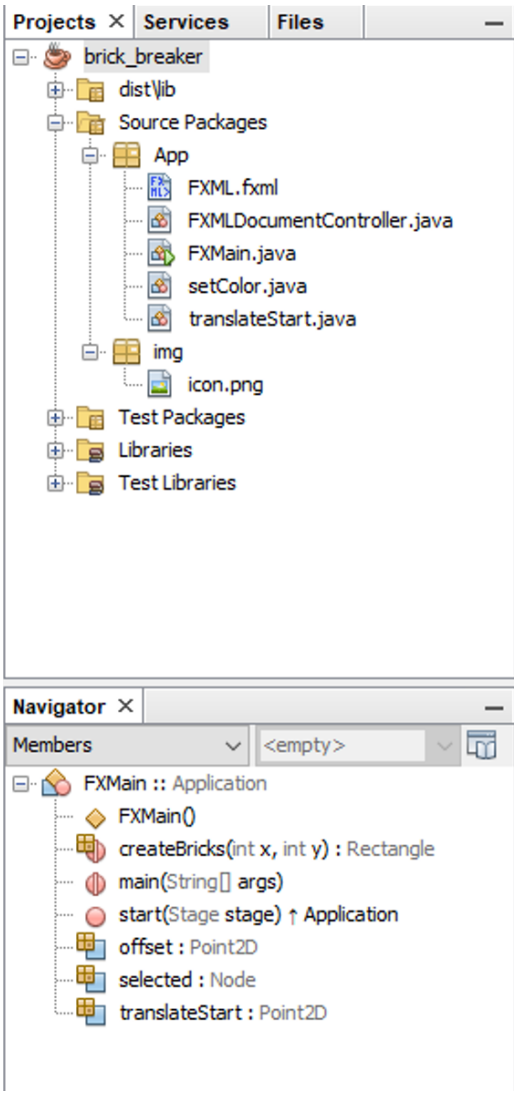
1. Úvod

Jako téma jsem si pro ročníkový projekt vybrala jednoduchou, ale přitom zábavnou hru Brick Breaker, vyvinutou v roce 1999 Ali Asariem, kdy právě díky své jednoduchosti a vysoké míře zábavy se těšila velkému zájmu hráčů. Grafický vzhled je jednoduchý, s možností výběru 4 barev cihel. Je zajímavé, že pomocí Java FX lze vytvářet aplikace nebo hry v moderním designu, ale zároveň se přenést i do doby minulé a takové aplikace vtisknout potřebného ducha i atmosféru.

2. Stuktura aplikace v IDE NetBeans

Celý kód je uložen v projektu nazvaném brick_breaker , v Source Packages jsou soubory FXML ve kterém je definována **scene** a **pane**. V FXMain.java se nachází hlavní kód aplikace. Pomocí knihovny setColor.java lze v aplikaci nastavit jednu ze čtyř různých barev pro cihly, viz. 2**.2. Možnost volby barvy.**

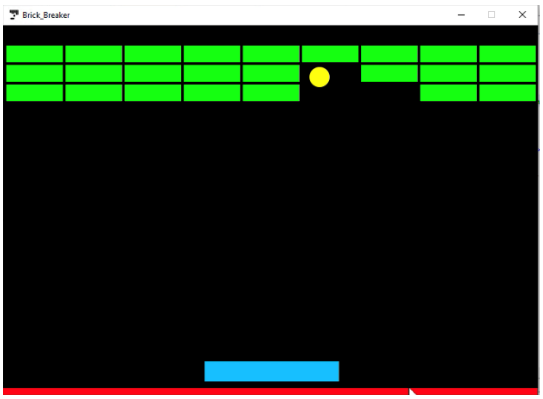
V img je uložen obrázek icon.png, což je ikona aplikace, která se zobrazí v levém horním, rohu. Tuto ikonu jsem vytvořila v grafickém editoru PhotoShop 7.0.



3. Návod

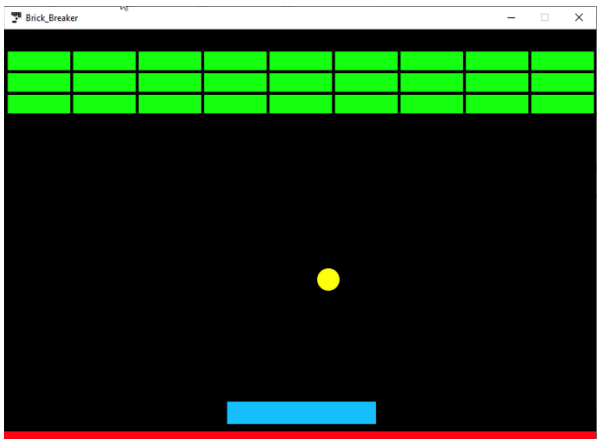
3.1 Jak hru hrát

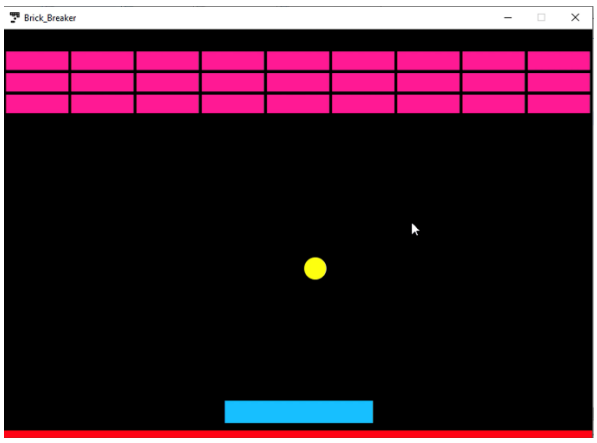
Základem hry je posouvání modrého jezdce-paddle, který ovládáme klávesnicovými šipkami **left** a **right**, kdy se jezdcem snažíme odrazit žlutou kuličku, která poté narazí do jedné nebo více cihel, které odstraní (*brickPane.getChildren().remove(brick)*). Vyhráváte, pokud odstraníte všechny cihly a pokud kulička nenarazí na červený pruh na spodním okraji. Konec hry se oznámí zobrazením nápisu xxxxxx GAME OVER xxxxxx.

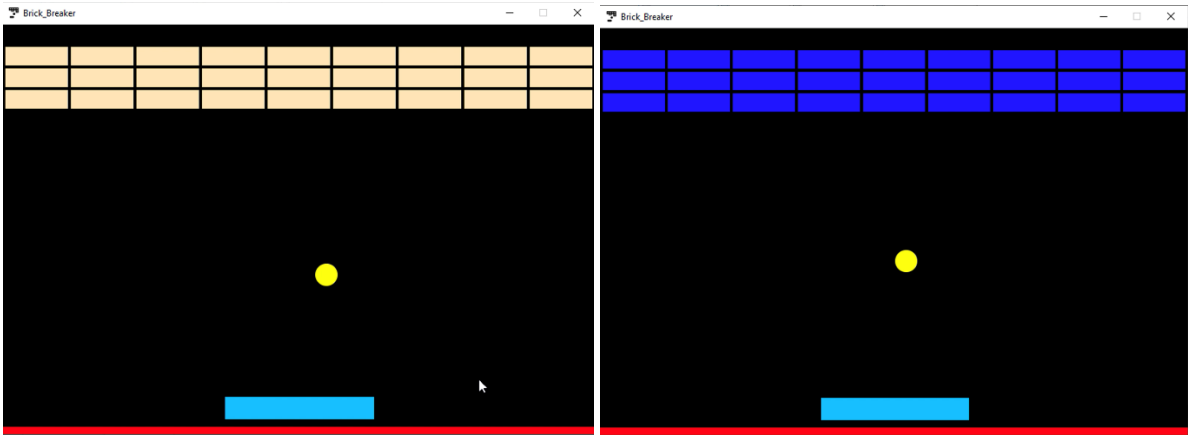


3.2 Možnost volby barvy

Barva u jezdce a kuličky je sice stálá a nedá se změnit, každopádně je tu možnost volby barvy cihel. Celkem si hráč může vybrat ze 4 barev, je mezi nimi zelená, béžová, modrá a růžová, kdy zelená *(Color.LIME*) je výchozí.







4. Popis jednotlivých částí kódu

4.1 Metoda vytváření cihel

Pomocí metody createBricks vytvářím objekt typu Rectangle s rozměry 84 x 25, kde hodnoty x a y souřadnic jsou generovány for cyklem. Podmínka if pro volbu barevného schématu z knihovny setColor.

```
//------------------------
//Metoda vytváření cihel

    static Rectangle createBricks(int x, int y) {
        Rectangle brick = new Rectangle(x, y, 84, 25);

        setColor col = new setColor();  //inicializace tridy setColor

        int colorScheme = 1;            //int volby barvy
        Paint barva = null;             //proměnná typu Paint

        if (colorScheme ==1)            //podmínka výběru barevného schématu
        {
            barva = col.brickColorScheme1;
        }
        if (colorScheme ==2)
        {
            barva= col.brickColorScheme2;
        }
         if (colorScheme ==3)
        {
            barva= col.brickColorScheme3;
        }
          if (colorScheme ==4)
        {
            barva= col.brickColorScheme4;
        }

        brick.setFill(barva);               //vyplnění cihel barvou
        return brick;                       //návratová hodnota metody

    }
//---------------------------------------
```

4.2 Nastavení objektů ball, rect, xlabel a base

Kulička v aplikaci je objekt Circle, jezdec a spodní červený okraj je Rectangle. Nastavením .relocate(x,y) definujeme výchozí umístění v okně aplikace.

```
Circle ball = new Circle(15, Color.YELLOW);  //nastavení barvy kuličky
Rectangle rect = new Rectangle(200,505, 200, 30);  //nastavení parametrů objektu Rectangle - v tomto případě paddle
Rectangle base = new Rectangle (0,540, 800, 15);
Label xlabel = new Label ("xxxxxx GAME OVER xxxxxx");
xlabel.setTextFill(Color.web("#FF76a3"));
xlabel.setStyle("-fx-font-family: Arial; -fx-font-size: 25");
xlabel.setLayoutX(230);
xlabel.setLayoutY(250);
xlabel.setVisible(false);

rect.relocate (300,500);//umístění objektu rect na obrazovce
rect.setFill(Color.DEEPSKYBLUE);    //nastavení barvy objektu rect - paddle
base.setFill(Color.RED);    //nastavení barvy objektu base
ball.relocate(400,300);  //umístění objektu base na obrazovce
base.setVisible(true);
```

4.3 Generování pole cihel a naplnění cyklem FOR

Vytvoření pole *brick* typu *Rectangle*, kdy se následně pomocí cyklu *for* nastaví hodnoty *x* a *y* souřadnic a zároveň se přidá do *brickpane* pro zobrazení ve scéně.

```
//Pole cihel

    Rectangle brick [] = new Rectangle[30];  //vytvoření pole brick s poctem
    int i=0;   //int i - vychoz hodnota counteru pro for cyklus
    int x = 0; //výchozí hodnota x souřadnice
    int y = 30;  //výchozí hodnota y souřadnice
    for(i=0, x = 5; x<=797; x=x+88,i=i+1 ){
        brick[i] = createBricks(x,y);
        brickPane.getChildren().add(brick[i]);

        if(x==797){
            for( x = 5; x<=797; x=x+88,i=i+1 ) {
                brick[i] = createBricks(x,y+29);
                brickPane.getChildren().add(brick[i]);


            if(x==797){
            for( x = 5; x<=797; x=x+88,i=i+1 ) {
                brick[i] = createBricks(x,y+58);
                brickPane.getChildren().add(brick[i]);


                }
            }
        }


    root.getChildren().addAll(ballPane,brickPane,rect,base,ball, xlabel); //definice členů pro root pane
```

4.4 Ovládání jezdce z klávesnice

Ovládání obbjektu rect, tedy jezdce pomocí šipek Left a Right na klávasnici pomocí setOnKeyPressed s definicí KeyCode, v našem případě stranové šipky a určením rozsahu tak, aby jezdec nevyjížděl mimo okno aplikace. Pohyb jezdce je o hodnotu 80 oběma směry.

```
//------------------------------------------------------------------------
//Pohyb objektu rect po stisknutí klávesy LEFT a RIGHT o hodnotu 80 a nastavením rozsahu pro levý a pravý roh

root.getScene().setOnKeyPressed((final KeyEvent e) -> {
    if (e.getCode() == KeyCode.LEFT && rect.getLayoutX()>=-150)
        rect.setLayoutX( rect.getLayoutX() - 80 );

     if (e.getCode() == KeyCode.RIGHT && rect.getLayoutX()<=350)
        rect.setLayoutX( rect.getLayoutX() + 80 );
  });


//------------------------------------------------------------------------
```

4.5 Pohyb kuličky

Pohyb kuličky je zajištěn pomocí animace Timeline s nastavením rychlosti 20ms a hodnotami dx a dy =7. Změnou toho parametru dojde ke změně pohybu kličky.

```
Timeline timeline = null;
    timeline = new Timeline(new KeyFrame(Duration.millis(20),
            new EventHandler<ActionEvent>() {

                double dx = 7; //Step on x or velocity
                double dy = 7; //Step on y
```

Nastavení nekonečného cyklu pohybu je pomocí

timeline.setCycleCount(Timeline.INDEFINITE);

Vlastní spuštění potom *timeline.play();*

Vlastní chování kuličky při kontaktu s „okraji“

```
public void handle(ActionEvent t) {


    //Pohyb kuličky
    ball.setLayoutX(ball.getLayoutX() + dx);
    ball.setLayoutY(ball.getLayoutY() + dy);

    Bounds bounds = ballPane.getLayoutBounds();

    if ( rect.getBoundsInParent().intersects(ball.getBoundsInParent()) ) dy = -dy;


    //Pokud klička dosáhne levého nebo pravého okraje, změní směr.
    if(ball.getLayoutX() <= (bounds.getMinX() + ball.getRadius()) ||
            ball.getLayoutX() >= (bounds.getMaxX() - ball.getRadius()) ){

        dx = -dx;
    }
    //Pokud klička dosáhne horního nebo dolního okraje, změní směr.
    if((ball.getLayoutY() >= (bounds.getMaxY() - ball.getRadius())) ||
            ball.getLayoutY() <= (bounds.getMinY() + ball.getRadius()))
    {
        dy = -dy;
    }
```

4.6 Odstranění cihly

Zničení/odstranění cihly je řešeno pomocí odstranění z brickPane, tedy pokud kulička

pronikne do okrajů brick, dojde k odstranění.

if(ball.getBoundsInParent().intersects(brick.getBoundsInParent())) {
brickPane.getChildren().remove(brick) }

5. Použité technologie

5.1 knihovny

JavaFX 16 IDE

5.2 nástroje

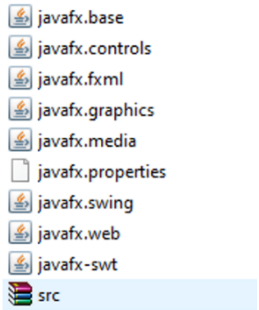
NetBeans IDE 12.2, JavaFX Scene Builder 16.0 v prostředí Windows 10

6. Instalace a požadavky na systém

K běhu aplikace je potřeba instalovaná Java

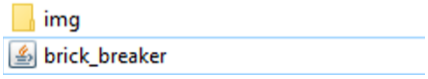
Požadavky na prostředí a vlastní spuštění aplikace:

V uvedené cestě C:\Program Files\Java\javafx-sdk-16\lib musí být obsah knihovny



Lze stáhnout na https://gluonhq.com/download/javafx-16-sdk-windows/

Na lokálním disku C je nutné vytvořit adresář brick_breaker, tedy c:\brick_breaker, kde obsahem budou následující soubory



Tedy, vlastní .jar soubor s aplikací a img složka s ikonou projektu.

Spuštění aplikace z příkazového řádku cmd, kdy přejdemne do složky c:\brick_breaker a spustíme následující příkaz:

java -jar --module-path "C:\Program Files\Java\javafx-sdk-16\lib" --add-modules=javafx.controls,javafx.fxml brick_breaker.jar

Pokud máme nastaveno dle výše uvedeného, pustí se aplikace a můžete začít hrát.

7. Závěr

Ačkoliv se mi nepovedlo splnit všechny body mého zadání ročníkového projektu, tak mám z odvedené práce dobrý dojem. Aplikace je funkční a vzhledově pěkná. I díky tomu, že nebylo zcela snadné tuto aplikaci vyrobit jsem více pronikla do prostředí javaFX a pochopila některé zákonitosti a postupy o kterých jsem předtím nevěděla.