Gymnázium, Praha 6, Arabská 14

Programování

ROČNÍKOVÝ PROJEKT

Donkey kong

Duben 2023

Michael Vakula, 2.E

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 …..... dne …..... Michael Vakula .....

Anotace:

Mým ročníkovým projektem bylo vytvoření napodobeniny arkádové hry Donkey kong z 80. let. Ve hře hrajete za ovladatelnou postavičku, která dokáže chodit doleva, doprava, skákat a lézt po žebříku. Cílem je dostat se s postavičkou na poslední plošinu, kde stojí princezna. Hráč si musí dávat pozor na barely, které se na něj neustále kutálejí. Při zásahu hráč ztratí jeden ze tří životů.

Abstract:

The goal of my year project was to create a copy of the famous arcade game Donkey Kong from 80s, created by Nintendo. In the game, you play as a character that can move left, right, jump and climp ladders. Your goal is to make it to the last floar where the princess is. The player must be aware of rolling barrels that constantly fall on them. If the player is hit by a barrel, they lose one of their three hearts.

Zadání:

2D hra, která bude mít několik plošin spojené žebříky, pomocí kterých se postavička dostane nahoru. Z vrchu by se měli kutálet překážky, které by postavička musela přeskočit. Cílem je dostat se k princezně bez toho, aby byl hráč zasáhnut překážkou.

Obsah:

1. Úvod

Zvolené téma jsem si vybral sám na základě touhy po vytvoření jednoduché 2D hry. Předem uvedené zadání jsem dodržel a zrealizoval jak základní funkce programu, tak i obohatil o jiné hravé funkce, které dodávají na hravosti a estetice programu.

Pro vytváření programu jsem využil programovací jazyk Java a vývojové prostředí IntelliJ, které používáme v hodinách předmětu programování, a také se v něm přehledněji pracuje.

2. Historie hry Donkey kong

Původní hra Donkey Kong byla vyvinuta japonskou společností Nintendo v roce 1981 a byla jednou z prvních her této společnosti, která se stala obrovským hitem. Hra se odehrává na staveništi, kde hráč ovládá postavu zvanou Jumpman (později přejmenovaný na Mario), který musí překonávat překážky a uniknout před gorilou jménem Donkey Kong, která hází sudy dolů. Hra se skládá ze čtyř levelů, přičemž každý level se skládá z několika obrazovek. Hráč musí v každém levelu dostat Jumpmana až na vrchol staveniště, kde se nachází Pauline, jeho přítelkyně, kterou musí osvobodit z klece, zatímco Donkey Kong ho pronásleduje.

Hra byla inovativní svým herním stylem a také tím, že nabízela neuvěřitelné vizuální efekty a animace. Donkey Kong se stal kultovní postavou v herním průmyslu a Mario, který byl zpočátku jen vedlejší postavou v Donkey Kongu, se stal hlavní postavou v dalších hrách od Nintenda[1].

[a]Logo společnosti Nintendo

3. Vývojové prostředí

Za prostředí, ve kterém jsem vypracoval tento program jsem si zvolil IntelliJ IDEA Community Edition 2021.3.2 s programovacím jazykem Java 17.

3.1. Java

Java je programovací jazyk a výpočetní platforma vydaná společností Sun Microsystems v roce 1995. Vyvinula se ze skromných začátků, aby poháněla velkou část dnešního digitálního světa tím, že poskytuje spolehlivou platformu, na které je postaveno mnoho služeb a aplikací. Java je zdarma ke stažení pro osobní použití. Nejnovější verzi můžete získat na java.com a všechny vývojové sady a další užitečné nástroje najdete na https://www.oracle.com/javadownload/. [2]

3.2. Java swing

Swing je knihovna uživatelských prvků na platformě Java pro ovládání počítače pomocí grafického rozhraní. Knihovna Swing poskytuje aplikační rozhraní pro tvorbu a obsluhu klasického grafického uživatelského rozhraní. Pomocí Swingu je možno vytvářet okna, dialogy, tlačítka, rámečky, rozbalovací seznamy atd. [3]

3.3. IntelliJ

IntelliJ IDEA je integrované vývojové prostředí (IDE) pro jazyky JVM navržené tak, aby maximalizovalo produktivitu vývojářů. Provádí rutinní a opakující se úkoly za vás tím, že poskytuje chytré dokončování kódu, analýzu statického kódu a umožňuje vám tak soustředit se na světlou stránku vývoje softwaru. [4]

3.4. Pixilart

Pixilart je online software na vytvoření pixelových obrázku a GIFů zdarma. Všechny mnou vytvořené obrázky jsem nakreslil právě v tomto softwaru.

[b] Logo programovacího jazyku java

4. Struktura funkcí programu v Java

V následujících podkapitolách se budu věnovat důležitým třídám a metodám, které v programu hrají klíčovou roli¨

4.1. Klíčové třídy a metody

Základní stavební kámen programu je třída Jframe. Třída JFrame je součástí Java Swing API, která reprezentuje hlavní okno grafického uživatelského rozhraní (GUI) pro desktopové aplikace v Javě. JFrame poskytuje základní funkce pro nastavení vlastností okna, jako jsou jeho velikost, umístění na obrazovce, název, ikona a možnosti pro minimalizaci, maximalizaci a zavření okna. Také umožňuje přidávat další komponenty do okna, jako jsou tlačítka, textová pole, seznamy a další, které jsou známy jako Swing komponenty. Zavolat ji můžeme pomocí extends JFrame.[5]

Další důležitá třída je Jpanel, která je také součástí Java Swing API a slouží k vytvoření prázdného kontejneru, který může být použit pro seskupení a uspořádání dalších komponent v uživatelském rozhraní. JPanel může obsahovat různé komponenty, jako jsou tlačítka, textová pole, seznamy a další, které se mohou nacházet v určitých uspořádáních, například ve sloupcích, řádcích nebo mřížce. Použití třídy JPanel je užitečné zejména v případech, kdy potřebujeme seskupit několik komponent dohromady a chceme je umístit na jednom místě v uživatelském rozhraní. Zavolat ji můžeme pomocí extends JPanel. [6]

Třída Rectangle (čtverec) je základní třídou pro reprezentaci obdélníků v geometrii. Obdélník je definován jako čtyřúhelník s protilehlými stranami rovnoběžnými a se stejnou délkou (šířkou). Třída Rectangle typicky obsahuje data (proměnné) pro délku a šířku obdélníka a metody (funkce), které umožňují manipulaci s těmito daty, jako je například výpočet obvodu a plochy obdélníka nebo změna jeho rozměrů. Zavolat ji můžeme pomocí extends Rectangle. [7]

Třída Runnable je součástí Java API a slouží k definování spustitelných úloh nezávislých na vláknech. Jedná se o rozhraní (interface) obsahující pouze jednu abstraktní metodu "run()", kterou je nutné implementovat. Tato metoda obsahuje kód, který má být vykonán, když je instance Runnable spuštěna. Použití třídy Runnable umožňuje oddělit samotnou úlohu od vlákna, které ji spouští. To znamená, že můžeme vytvořit jednu instanci úlohy a spustit ji v různých vláknech nebo dokonce v rámci jednoho vlákna opakovaně. To nám umožňuje lépe řídit a plánovat běh našeho programu a zlepšit jeho výkon. Pro spuštění instance Runnable můžeme použít třídu Thread a předat jí tuto instanci jako argument do konstruktoru. Zavolat ji můžeme pomocí extends Runnable. [8]

Metoda paint() je jedna z metod definovaných v třídě java.awt.Component, která se používá pro vykreslení grafických prvků na obrazovku nebo na jiný povrch. Tuto metodu můžeme přepsat (override) v potomcích třídy Component, abychom mohli definovat, co se má vykreslit a jaké prvky a vlastnosti se mají použít. Metoda paint() přijímá jako argument objekt Graphics, který reprezentuje plochu, na kterou se má vykreslit. Pomocí tohoto objektu můžeme nastavit různé vlastnosti kreslení, jako jsou barva, font, šířka a další. Poté můžeme použít metody tohoto objektu pro vykreslení různých prvků, obrazců nebo textu na plochu. [9]

Třída KeyAdapter je součástí Java API a slouží k implementaci ovladače událostí klávesnice. Jedná se o adaptér, který implementuje rozhraní KeyListener a poskytuje výchozí implementaci pro všechny jeho metody. Tímto způsobem můžeme snadno přepsat (override) pouze ty metody, které nás zajímají, a nechat ostatní výchozí implementaci. Rozhraní KeyListener definuje tři metody: keyPressed(), keyReleased() a keyTyped(). Metoda keyPressed() je volána, když uživatel stiskne klávesu, keyReleased() je volána, když uživatel uvolní klávesu a keyTyped() je volána, když uživatel stiskne klávesu, která generuje znak. Třída KeyAdapter poskytuje výchozí implementaci pro všechny tyto metody, která nic nedělá. Pro použití třídy KeyAdapter vytvoříme instanci této třídy a předáme ji jako argument do metody addKeyListener() na komponentě, kterou chceme sledovat. Poté přepsáme pouze ty metody, které nás zajímají, a zpracujeme události klávesnice podle našich potřeb.[10]

Metoda TimerTask je třída v Javě, která slouží k plánování opakovaného spouštění určitých úloh nebo kódů v určitých časových intervalech. Tato třída je součástí balíčku java.util a používá se společně s třídou Timer.[11]

4.2. Zpracování funkcí

Třída main v mém programu slouží pouze pro vytvoření objektu JFrame podle konstruktoru třídy HerniRam rozšířenou o třídu Jframe.

```
package com.company;
//Třída main s metodou main
public class Main {

    public static void main(String[] args) {
        // Metoda main vytvoří Nový hrací rám podle konstruktoru ze třídy HraciRam
        HraciRam ram = new HraciRam();

    }
}
```

[c] main metoda

Třída HerniRam je rozšířena o třídu Jframe, jeho konstruktor vytváří objekt typu JPanel podle konstruktoru třídy HerniPanel. V konstruktoru dále přidá objekt herniPanel do konponentu JFrame a nastavuje název okna, nastavuje pozadí na černé, volá metodu pack(), která se používá v grafických knihovnách k uspořádání prvků v okně. Tato metoda "balí" prvky do jejich přirozené velikosti a umístí je na nejbližší volnou pozici v okně.[12] Dále přidá tlačítko na zavření aplikace a nastaví, aby se okno nedalo zvětšit.

```
package com.company;

import java.awt.*;
import javax.swing.*;
//Třída HraciRam je rozšířena o třídu JFrame ze knihovny java swing
public class HraciRam extends JFrame {
    //Vytvoření objektu
    HerniPanel herniPanel;

    HraciRam(){
        //Objekt postaven podle konstruktoru třídy HerniPanel
            herniPanel = new HerniPanel();
            //nastavení určitých parametrů u vytvořeného objektu
            this.add(herniPanel);
            this.setTitle("Donkey Kong");
            this.setResizable(false);
            this.setBackground(Color.BLACK);
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            this.pack();
            this.setVisible(true);
            this.setLocationRelativeTo(null);
    }
}
```

[d] Třída HraciRam

Třída HerniPanel je rozšířena o třídu JPanel a také implementuje třídu Runnable. V třídě jsou dány neměnitelné proměnné jako například šířka a výška okna a dalších objektů. Důležitou proměnnou je hodna panel typu int. Podle ní se jiné metody orientují a určují co mají vykreslit do panelu a co ne.

```
package com.company;

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
//Třída HerniPanel je rozšířena o třídu Jpanel ze knihovny java swing, třída je také
public class HerniPanel extends JPanel implements Runnable{
    //Velmi důležitá proměnná, podle ní se rozhoduje co se namaluje na panel a co ne
    public static int panel = 1;
    //Předepsané proměnné, které nelze měnit
    static final int HERNI_SIRKA = 1000;
    static final int HERNI_VYSKA = 750;
    static final Dimension SCREEN_SIZE = new Dimension(HERNI_SIRKA, HERNI_VYSKA);
    static final int SIRKA_PLOSINY = 850;
    static final int VYSKA_PLOSINY = 20;
    static final int SIRKA_ZEBRIKU = 20;
    static final int VYSKA_ZEBRIKU = 70;
    static final int MARIO_VELIKOST_XY = 25;
    static final int BARREL_VELIKOST_XY = 25;
    static final int OPICAL_VELIKOST_XY = 50;
    static final int PRINCEZNA_VELIKOST_XY = 25;
```

[e] proměnné třídy HerniPanel

Dále jsou v třídě předepsané použité objekty.

```
//Vytvořené objekty
    Thread gameThread;
    Image image;
    Graphics graphics;
    Plosina plosina0a;
    Plosina plosina0b;
    Plosina plosina1a;
    Plosina plosina1b;
    Plosina plosina2a;
    Plosina plosina2b;
    Plosina plosina3a;
    Plosina plosina3b;
    Plosina plosina4a;
    Plosina plosina4b;
    Plosina plosina5a;
    Plosina plosina5b;
    Plosina plosina6a;
    Plosina plosina6b;
    Plosina plosina7;
    Plosina plosina8a;
    Plosina plosina8b;
    Mario mario;
    Mario fakeMario1;
    Mario fakeMario2;
    Mario fakeMario3;
    Mario fakeMario4;
    Mario fakeMario5;
    Zebrik zebrik1;
    Zebrik zebrik2;
    Zebrik zebrik3;
    Zebrik zebrik4;
    Zebrik zebrik5;
    Zebrik zebrik6;
    Zebrik zebrik7;
    Zebrik fakZebrik;
    Barrel barrel1;
    Barrel barrel2;
    Barrel barrel3;
    Barrel barrel4;
    Barrel barrel5;
    Barrel barrel6;
    Barrel barrel7;
    Barrel fakeBarrel;
    Opicak opicak;
    Princezna princezna;
    Princezna fakePrincezna;
    Cas cas;
    HromadaBarelu hromadaBarelu;
    Logo logo;
    Start start;
    Ovladani ovladani;
    OvladaniList ovladaniList;
    VyhralJsi vyhralJsi;
    Srdicko fakeSrdicko;
    ZpetDoMenu zpetDoMenu;
    ProhralJsi prohralJsi;
    ZkusitZnovu zkusitZnovu;
    Zivoty zivotyVelke;
    Zivoty zivotyMale;
```

[f] Objekty

V samotném konstruktoru třídy jsou volány metody, podle kterých se určené objekty tvoří. Dále konstruktor tvoří komponent KeyListener, který je postaven podle konstruktoru třídy KeyAdapter, nastaví velikost okna, a spustí nově vytvořené vlákno.

```
HerniPanel(){
        novaPlosina();
        novyMario();
        novyZebrik();
        novyBarrel();
        novyOpicak();
        novaPrincezna();
        novyCas();
        novyHromadaBarelu();
        novyLogo();
        novyStart();
        novyOvladani();
        novyOvladaniList();
        novyVyhralJsi();
        novySrdicko();
        novyZpetDoMenu();
        novyProhralJsi();
        novyZkusitZnovu();
        novyZivoty();


        this.setFocusable(true);
        this.addKeyListener(new AL());
        this.setPreferredSize(SCREEN_SIZE);

        gameThread = new Thread( target: this);
        gameThread.start();

}
```

[g] Konstruktor třídy HerniPanel

Dále jsou v třídě HerniPanel vytvořeny metody, které jsme mohli vidět v konstruktoru. Metody vytvoří objekt podle konstruktoru určité třídy. V dokumentaci nebudu ukazovat všechny metody, ale pro ukázku tu mám metodu pro vytvoření objektu typu plosina.

```
//Metoda pro vytvoření objektů podle konstruktoru ze třídy Plosina
    public void novaPlosina() {
        plosina0a = new Plosina( x: 0, (HERNI_VYSKA - VYSKA_PLOSINY), (HERNI_SIRKA/2), VYSKA_PLOSINY);
        plosina0b = new Plosina((HERNI_SIRKA/2), (HERNI_VYSKA - VYSKA_PLOSINY), (HERNI_SIRKA/2), VYSKA_PLOSINY);
        plosina1a = new Plosina( x: 0,(HERNI_VYSKA - VYSKA_PLOSINY - 90), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina1b = new Plosina((SIRKA_PLOSINY/2),(HERNI_VYSKA - VYSKA_PLOSINY - 90), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina2a = new Plosina((HERNI_SIRKA - SIRKA_PLOSINY), (HERNI_VYSKA - VYSKA_PLOSINY - 180), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina2b = new Plosina((HERNI_SIRKA - (SIRKA_PLOSINY/2)), (HERNI_VYSKA - VYSKA_PLOSINY - 180), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina3a = new Plosina( x: 0, (HERNI_VYSKA - VYSKA_PLOSINY - 270), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina3b = new Plosina((SIRKA_PLOSINY/2), (HERNI_VYSKA - VYSKA_PLOSINY - 270), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina4a = new Plosina((HERNI_SIRKA - SIRKA_PLOSINY), (HERNI_VYSKA - VYSKA_PLOSINY - 360), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina4b = new Plosina((HERNI_SIRKA - (SIRKA_PLOSINY/2)), (HERNI_VYSKA - VYSKA_PLOSINY - 360), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina5a = new Plosina( x: 0, (HERNI_VYSKA - VYSKA_PLOSINY - 450), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina5b = new Plosina((SIRKA_PLOSINY/2), (HERNI_VYSKA - VYSKA_PLOSINY - 450), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina6a = new Plosina((HERNI_SIRKA - SIRKA_PLOSINY), (HERNI_VYSKA - VYSKA_PLOSINY - 540), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina6b = new Plosina((HERNI_SIRKA - (SIRKA_PLOSINY/2)), (HERNI_VYSKA - VYSKA_PLOSINY - 540), (SIRKA_PLOSINY/2), VYSKA_PLOSINY);
        plosina7 = new Plosina( x: 600, (HERNI_VYSKA - VYSKA_PLOSINY - 630), SIRKA_PLOSINY: 400, VYSKA_PLOSINY);
        plosina8a = new Plosina( x: 0, y: 500, (HERNI_SIRKA/2), VYSKA_PLOSINY);
        plosina8b = new Plosina((HERNI_SIRKA/2), y: 500, (HERNI_SIRKA/2), VYSKA_PLOSINY);
```

[h] Metoda pro vytvoření objektů podle konstruktoru ze třídy Plosina

Metoda paint()

```
//Metoda z knihovny java swing, která podle objektu Graphis konfiguruje příslušné komponenty
    public void paint(Graphics g) {
        image = createImage(getWidth(),getHeight());
        graphics = image.getGraphics();
        nakresli(graphics);
        g.drawImage(image, x: 0, y: 0, observer: this);

    }
```

[i] Metoda paint()

Dále je v třídě metoda nakresli, která využívá komponent Graphics g a podle něj kreslí objekty na panel. Třídy, podle kterých jsou objekty tvořeny, mají vlastní metodu nakresli(), tudíž ví jak se mají nakreslit.

```
//Metoda která kreslí všechny vytvořené objekty
public void nakresli(Graphics g) {
    if (panel == 2){
plosina0a.nakresli(g);
plosina0b.nakresli(g);
plosina1a.nakresli(g);
```

[j] Metoda nakresli()

Metoda move() slouží k nastavení souřadnicové polohy x, y na panelu aplikace. Tuto metodu využívají objekty, které se v aplikaci nějakým způsobem hýbají, takže například postavička, za kterou se hraje.

```
    //metoda, která zobrazuje objekty na určité souřadnicové poloze a umožnuje tak jejich pohyb
    public void move() {
if (panel == 2){
    mario.move();

    barrel1.move();
    if (barrel2.povoleniKPohybu == 1){
        barrel2.move();
    }
    if (barrel3.povoleniKPohybu == 1){
        barrel3.move();
    }
```

[k] Metoda move()

Metoda zkontrolujKolize() je jedna z nejdůležitějších pro správné fungování aplikace. Hlídá, jestli nějaký objekt interaguje s jiným objektem a určuje jeho chování, když k interakci dojde. Například u hratelné postavy (mario) se hlídá, pokud se dotýká objektu plošina a posouvá jeho x a y souřadnici tak, aby se do plošiny nepropadl. Metoda u maria dále hlídá, jestli se dotýká, barelu, žebříku, princezny nebo hlídá, aby x a y nebylo větší než velikost okna a mario tak nemohl chodit za rám.

Kontrola kolize u objektu mario.

```
    //Velmi obsáhlá a důležitá metoda, slouží k rozpoznání zda li některé objety mezi sebou interagují
    public void zkontrolujKolizi() {
//podmínka aby objekt mario nemohl za rám aplikace
            if(mario.x<=0)
                mario.x=0;
            if(mario.x >= (HERNI_SIRKA-MARIO_VELIKOST_XY))
                mario.x = HERNI_SIRKA-MARIO_VELIKOST_XY;
//Umožňuje objektu mario chození po plošinách a další interakce
            if (mario.intersects(plosina0a)) {
                mario.y = (HERNI_VYSKA - VYSKA_PLOSINY - MARIO_VELIKOST_XY);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina0b)){
                mario.y = (HERNI_VYSKA-VYSKA_PLOSINY-MARIO_VELIKOST_XY);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina1a)){
                mario.y = (HERNI_VYSKA-VYSKA_PLOSINY-MARIO_VELIKOST_XY-90);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina1b)){
                mario.y = (HERNI_VYSKA-VYSKA_PLOSINY-MARIO_VELIKOST_XY-90);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina2a)){
                mario.y = (HERNI_VYSKA-VYSKA_PLOSINY-MARIO_VELIKOST_XY-180);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina2b)){
                mario.y = (HERNI_VYSKA-VYSKA_PLOSINY-MARIO_VELIKOST_XY-180);
                mario.povoleniVyskocit = 1;
            } else if(mario.intersects(plosina3a)){
```

[l] Kontrola kolizí u objektu mario

Kontrola kolizí u objektu barel

```
//Umožňuje objektům barrel kutálení po plošinách
            if(barrel1.intersects(plosina0a)) {
                barrel1.y = (HERNI_VYSKA - VYSKA_PLOSINY - BARREL_VELIKOST_XY);
                barrel1.rychlost = -5;
            }else if(barrel1.intersects(plosina0b)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY);
                barrel1.rychlost = -5;
            }else if(barrel1.intersects(plosina1a)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-90);
                barrel1.rychlost = 5;
            }else if(barrel1.intersects(plosina1b)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-90);
                barrel1.rychlost = 5;
            }else if(barrel1.intersects(plosina2a)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-180);
                barrel1.rychlost = -5;
            }else if(barrel1.intersects(plosina2b)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-180);
                barrel1.rychlost = -5;
            }else if(barrel1.intersects(plosina3a)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-270);
                barrel1.rychlost = 5;
            }else if(barrel1.intersects(plosina3b)){
                barrel1.y = (HERNI_VYSKA-VYSKA_PLOSINY-BARREL_VELIKOST_XY-270);
                barrel1.rychlost = 5;
            }else if(barrel1.intersects(plosina4a)){
```

[m] Kontrola kolizí u objektu barel

Metoda také podle kontrole souřadnic daných objektů říká, jak se má objekt nakresli. Ukázkou je kontrola souřadnic barelů, aby objekt opicak věděl jaký obrázek má nakreslit a tím je tvořena animace

```
//díky podmínkám je objekt opičák se schopen se chovat podle souřadnicích ostatních objektů
        if (barrel1.x >= 600 && barrel1.x <= 780 && barrel1.y == 75) {
            opicak.polohaOpicaka = 1;
        }else if (barrel2.x >= 600 && barrel2.x <= 780 && barrel2.y == 75){
            opicak.polohaOpicaka = 1;
        }else if (barrel3.x >= 600 && barrel3.x <= 780 && barrel3.y == 75){
            opicak.polohaOpicaka = 1;
        }else if (barrel4.x >= 600 && barrel4.x <= 780 && barrel4.y == 75){
            opicak.polohaOpicaka = 1;
        }else if (barrel5.x >= 600 && barrel5.x <= 780 && barrel5.y == 75){
            opicak.polohaOpicaka = 1;
        }else if (barrel6.x >= 600 && barrel6.x <= 780 && barrel6.y == 75){
            opicak.polohaOpicaka = 1;
        }else if (barrel7.x >= 600 && barrel7.x <= 780 && barrel7.y == 75){
            opicak.polohaOpicaka = 1;
        } else opicak.polohaOpicaka = 0;
```

[n] Kontrola souřadnic objektu barrel

Podmínky pro lezení po žebříku.

```
//Podmínky umožňujou lezení po žebříku
        if(mario.intersects(zebrik1)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik2)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik3)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik3)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik4)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik5)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else if (mario.intersects(zebrik6)){
            mario.povoleniKLezeni = 1;
            mario.gravitace =0;
        }else{
            mario.povoleniKLezeni = 0;
            mario.gravitace =5;
        }
```

[o] Podmínky pro lezení po žebříku

Dále v metodě najdeme podmínky pro akci, když se barrel dotkne mária, mário se dotkne princezny a podmínky, které umožňují animaci určitých objektů.

Dále máme v třídě HerniPanel metodu run(), která je nezbytná v případě, že je třída rozšířena o třídu Runnable. Tato metoda run() funguje jako smyčka a volá v ní důležité metody jako nakresli(), move() a zkontrolujKolize() aby se objekty pořád do kolo updatovali a vykreslovali na správném místě.

```
//Metoda run(), neboli v her často nazívána game loop, velmi důležitá metoda, která neustále kontroluje podmínky a vykresluje podmínky
    //Metodu run() jsem převzal z videa: https://youtu.be/oLirZqJFKPE
    public void run() {
        //game loop
        long lastTime = System.nanoTime();
        double amountOfTicks =60.0;
        double ns = 1000000000 / amountOfTicks;
        double delta = 0;
        while(true) {
            long now = System.nanoTime();
            delta += (now -lastTime)/ns;
            lastTime = now;
            if(delta >=1) {
                move();
                zkontrolujKolizi();
                repaint();
                delta--;

            }
        }
    }
```

[p] Metoda run()

Poslední metoda slouží k zaznamenání akcí z klávesnice, metoda v sobě nese všechny objekty, které se při určité akci z klávesnice orientují.

```
    //Třída rozšířená o třídu z knihovny pro čtení z klávesnice
    public class AL extends KeyAdapter{
        //Metoda pro zaznamenávání zmáčknutí klávesy
        public void keyPressed(KeyEvent e) {
            mario.keyPressed(e);
            start.keyPressedStart(e);
            ovladani.keyPressedOvladani(e);
            ovladaniList.keyPressedOvladaniList(e);
            zpetDoMenu.keyPressedDoMenu(e);
            zkusitZnovu.keyPressedZkusitZnovu(e);
        }
        //Metoda pro zaznamenávání puštění klávesy
        public void keyReleased(KeyEvent e) {
            mario.keyReleased(e);
            zkusitZnovu.keyReleasedZkusitZnovu(e);
        }
    }
}
```

[q] Čtení z klávesnice

Dále si ukážeme třídy a jejich metody podle kterých se objekty vytvoří. Většina těchto tříd jsou skoro identické, a proto se nebudu věnovat všem ale pouze těm, ve kterých jsou neukázané metody.

Třída Mario je rozšířena o třidu Rectangle, tudíž její objekty lze chápat jako pravidelné čtyřúhelníky, délkou dvou různých stran a souřadnicemi x a y. Třída Mario má mnoho proměnných hodnot, jelikož mario je objekt, který má mnoho typů, jak se vykreslit. Metody ve třídě mario je konstruktor, pro snadné vytvoření objektu, metodu nakresli, která podle parametrů říká jaký obrázek se má u objektu nakreslit, metodu pro zaznamenávání určitých kláves, a metodu move(), která mění souřadnice objektu mario.

Konstrukor.

```
//Konstruktor
    Mario(int x, int y, int SIRKA_MARIA,int VYSKA_MARIA) {
        super(x, y, SIRKA_MARIA, VYSKA_MARIA);
        skace = false;
        povoleniVyskocit = 0;
```

[r] Konstruktor třídy Mario

Metoda nakresli()

```
//Metoda pro nakreslení
public void nakresli(Graphics g) {
    if (kamchodi == 0){
        g.drawImage(doLeva, x, y, observer: null);
    }
    if (kamchodi == 1){
        g.drawImage(doPrava, x,y,  observer: null);
    }
    if (lezeAnoNe == 1){
        g.drawImage(leze, x, y ,  observer: null);
    }
    if (jeFake == 1){
        g.drawImage(mario50, x,y, observer: null);
    }
    if (jeFake == 2){
        g.drawImage(marioJump50, x,y, observer: null);
    }
    if (jeFake == 3){
        g.drawImage(marioLeze50, x,y, observer: null);
    }
```

[s] Metoda nakresli třídy Mario

Metoda pro zaznamenání, jestli byla klávesa zmáčknuta.

```
//Metoda pro zaznamenání pokud byla určitá klávesa zmáčknuta
public void keyPressed(KeyEvent e) {
    if(e.getKeyCode()==KeyEvent.VK_A) {
        nastavSmerX(-rychlostX);
        kamchodi = 0;
    }
    if(e.getKeyCode()== KeyEvent.VK_D) {
        nastavSmerX(rychlostX);
        kamchodi = 1;
    }
    if(e.getKeyCode()==KeyEvent.VK_SPACE){
        if(povoleniVyskocit==1){
            skace = true;
        }
    }
    if(e.getKeyCode()== KeyEvent.VK_W) {
        if(povoleniKLezeni == 1) {
            rychlostPoZebriku= -2;
        }
    }
    if(e.getKeyCode()== KeyEvent.VK_S) {
        if(povoleniKLezeni ==1) {
            rychlostPoZebriku= 2;
        }
```

[t] Mario keyPressed

Metoda pro zaznamenání, jestli byla klávesa puštěna.

```
//Metoda pro zaznamenání jestli byla určitá klávesa puštěna
    public void keyReleased(KeyEvent e) {
        if (e.getKeyCode() == KeyEvent.VK_A) {
            nastavSmerX(0);
        }
        if (e.getKeyCode() == KeyEvent.VK_D) {
            nastavSmerX(0);
        }
        if(e.getKeyCode()==KeyEvent.VK_SPACE){

        }
        if(e.getKeyCode()== KeyEvent.VK_W) {
            rychlostPoZebriku = 0;
        }
        if(e.getKeyCode()== KeyEvent.VK_S) {
            rychlostPoZebriku = 0;
        }
    }
```

[v] Mario keyReleased

Metoda move().

```
//Metoda pro updatování polohy objektu mario
    public void move () {
        x = x + xZrychleni;
        y = y + rychlostPoZebriku;
        y = y + gravitace;

        if (povoleniVyskocit == 1){
            maxVyskok = y - 50;
        }

        if (skace) {


            y -= yZrychleni;

            if (y <= maxVyskok) {
                skace = false;
            }
        }
    }
}
```

[w] Mario move()

Zajímavou třídou je třída Cas(), která má metodu timerTask(), ta díky své funkci je schopna každou sekundu přičíst jedničku k proměnné a udělat tak časovač. Do objektu typu Rectangle pak tento čas lze zapisovat pomocí metody drawString().

```
//Metoda pro počítání času na sekundy a minuty
        int sekundy = 0;
        int minuty = 0;
       Timer timer = new Timer();

       TimerTask task = () -> {
              if (HerniPanel.panel == 2){
                  sekundy++;
                  if (sekundy == 60) {
                      sekundy = 0;
                      minuty++;
                  }
              }
              if (HerniPanel.panel == 5){
                  sekundy = 0;
                  minuty = 0;
              }

       };

       public void startTimer() { timer.scheduleAtFixedRate(task,  delay: 1000,  period: 1000); }
       //Metoda pro nakreslení
```

[x] Timer

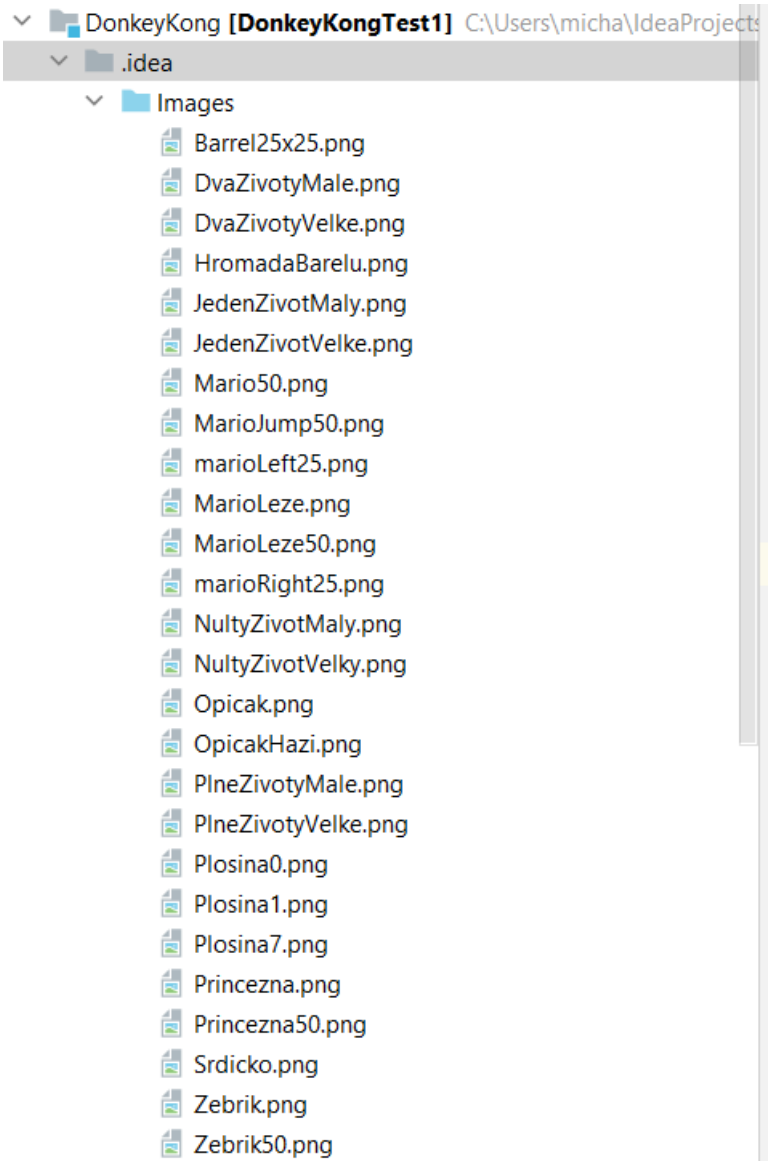
```
public void nakresli(Graphics g) {
g.setColor(Color.WHITE);
    g.setFont(new Font( name: "Consolas",Font.PLAIN, size: 60));
    g.drawString(format("%02d:%02d", minuty, sekundy),  x: 50, y: 50);
```

[y] Metoda drawString

5. Popis grafiky a její stylizace

Grafické zpracování je tvořeno metodou drawString a drawImage. Pro drawImage jsou u jednotlivých tříd vytvořeny proměnné typu BufferedImage, které v sobě dokážou nést obrázek.

Složka obrázků.



[z] Složka obrázků

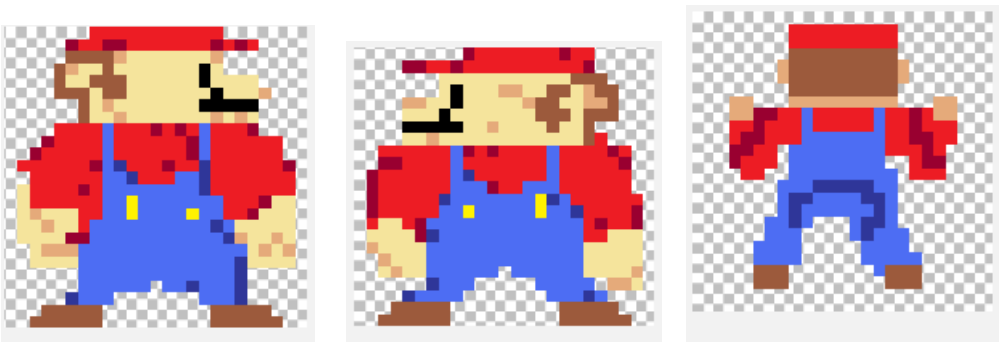
Určení proměnné typu BufferedImage o daný obrázek

```
try {
    doPrava = ImageIO.read(new File( pathname: "C:\\Users\\micha\\IdeaProjects\\DonkeyKong\\.idea\\Images\\marioRight25.png"));
} catch (IOException e) {
    e.printStackTrace();
}

try {
    doLeva = ImageIO.read(new File( pathname: "C:\\Users\\micha\\IdeaProjects\\DonkeyKong\\.idea\\Images\\marioLeft25.png"));
} catch (IOException e) {
    e.printStackTrace();
}
try {
    leze = ImageIO.read(new File( pathname: "C:\\Users\\micha\\IdeaProjects\\DonkeyKong\\.idea\\Images\\MarioLeze.png"));
} catch (IOException e) {
    e.printStackTrace();
}
```

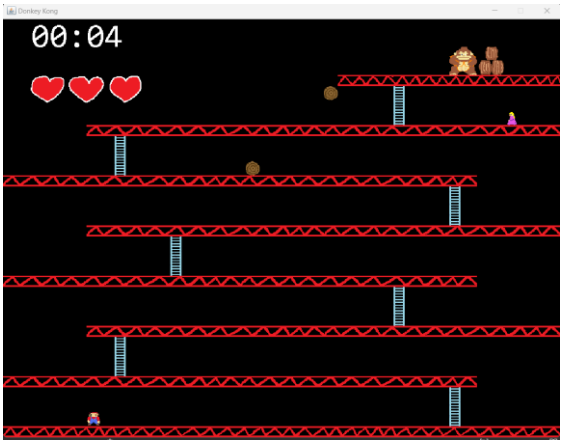
[a1] Určení proměnné typu BufferedImage

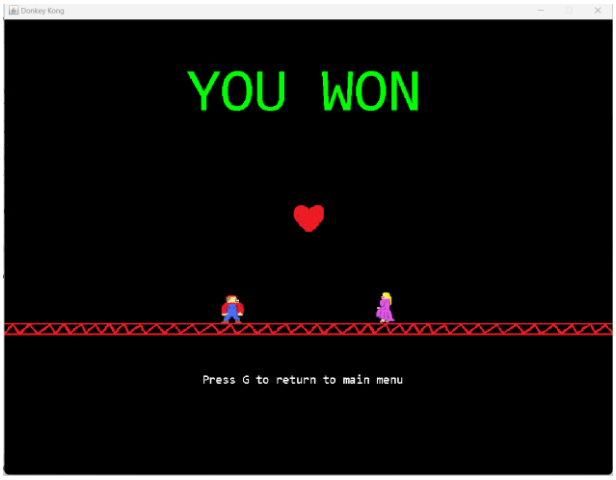
Ukázka jednotlivých obrázků.



[b1] Obrazky maria

Jednotlivé snímky aplikace





[c1] Snimky aplikace

6. Závěr

Celkový pohled z mé strany na finální verzi ročníkového projektu je takový, že se práce vyvedla. Zadání jsem úspěšně splnil a program bezchybně funguje. Program jde ale určitě dále rozvíjet a obohacovat o další funkce.

Při práci jsem projevil opravdový zápal do dané problematiky a získal jsem mnoho užitečných zkušeností a vědomostí zejména v oblasti objektově orientovaného programování, které v budoucích projektech využiji.

99. Zdroje

[1] https://en.wikipedia.org/wiki/Donkey_Kong

[2] Oracle. (n.d.). *What is Java technology and why do I need it?* Java.com. Retrieved May 1, 2022, from https://www.java.com/en/download/help/whatis_java.html

[3] https://cs.wikipedia.org/wiki/Swing_(Java)

[4] 2000–2023 JetBrains s.r.o. (n.d.). *IntelliJ IDEA overview: Intellij idea*. IntelliJ IDEA Help. Retrieved May 1, 2022, from https://www.jetbrains.com/help/idea/discover-intellij-idea.html

[5] https://docs.oracle.com/javase/7/docs/api/javax/swing/JFrame.html

[6] https://docs.oracle.com/javase/7/docs/api/javax/swing/JPanel.html

[7] https://docs.oracle.com/javase/7/docs/api/java/awt/Rectangle.html

[8] https://docs.oracle.com/javase/7/docs/api/java/lang/Runnable.html

[9] https://www.oracle.com/java/technologies/painting.html

[10] https://docs.oracle.com/javase/7/docs/api/java/awt/event/KeyAdapter.html

[11] https://docs.oracle.com/javase/7/docs/api/java/util/TimerTask.html

[12] https://stackoverflow.com/questions/22982295/what-does-pack-do

99. Seznam obrázků