**JLTPaint**

# Úvod

JLTPaint je jednoduchý malovací program implementovaný v jazyce Java. Program poskytuje uživateli grafické uživatelské rozhraní (GUI) pro kreslení, mazání, aplikaci filtrů na obrázky a další operace spojené s malováním. Projekt je určen pro lokální použití a slouží jako demonstrační příklad práce s knihovnou Swing a manipulací s obrázky.

# Tým

Tvoříme tým o 3 osobách, Lukáš Markel, Jaromír Mynarčík a Tomáš Rosa, z toho vznikl také název naší aplikace, JLTPaint. Každý v týmu pracoval na jiné části programu nebo projektu.

**Lukáš Markel**

- Vedoucí týmu, rozdělení prací, dokumentace, prezentace projektu

- Programování: GUI

**Jaromír Mynarčík**

- Programování: Práce se soubory, vkládání a ukládání obrázků, filtry obrázků

**Tomáš Rosa**

- Programování: Malování, paleta barev, guma, generování obrázků

# Hlavní funkce

## Kreslení a mazání

* Uživatel může kreslit pomocí nástroje "pero" a mazat pomocí "gumovacího" nástroje.
* Podporuje různé tloušťky nástroje a změnu barvy pera.

## Práce s obrázky

* Načtení obrázku z disku.
* Uložení aktuálního plátna jako obrázku.
* Obnovení původního načteného obrázku.

## Filtry

* Aplikace negativního filtru na načtený obrázek.
* Prahový filtr s nastavitelným parametrem prahu.
* Nastavení intenzity jednotlivých barevných kanálů (RGB).

## Generování obrázků

* Vygenerování náhodného barevného pozadí (wallpaper).
* Vytvoření jednoduchého obrázku slunce.

## Zpětné operace

* Možnost vrácení poslední akce pomocí funkce "Undo".
* Vymazání celého plátna.

# Struktura projektu

Projekt je rozdělen do několika tříd:

## PaintApp

* Hlavní třída programu, která inicializuje grafické uživatelské rozhraní.

## PaintFrame

* Hlavní okno aplikace, které obsahuje menu, tlačítka a panel pro kreslení.
* Obsahuje implementaci menu pro načítání, ukládání a práci s filtry.

## PaintCanvas

* Hlavní panel pro kreslení.
* Obsahuje logiku pro kreslení, mazání, aplikaci filtrů a manipulaci s obrázky.
* Podporuje různé režimy nástroje (pero, guma).

## Line

* Reprezentuje jednotlivé čáry na plátně.
* Obsahuje informace o počátečním a koncovém bodu, barvě a tloušťce čáry.

## Operation

* Uchovává informace o provedených operacích (např. kreslení, mazání).
* Používá se pro implementaci funkce "Undo".

## OperationType

* Výčtový typ definující různé typy operací (DRAW, ERASE, LOAD\_IMAGE).

# Použité technologie

## Programovací jazyk

* Java

## Vývojové prostředí

* Visual Studio Code
* IntelliJ IDEA

## Komunikační prostředky

* Discord
* GitHub
* Instagram

## Knihovny

**Java Swing**

* Používá se pro vytvoření GUI.

**Java AWT**

* Používá se pro manipulaci s grafikou a obrázky.

**Java ImageIO:**

* Slouží pro načítání a ukládání obrázků.

# Návod k použití

**Spuštění programu**

* Spusťte program pomocí připraveného JAR souboru:

**Kreslení**

* Vyberte nástroj (pero nebo gumu) a kreslete na plátno.
* Nastavte barvu pera pomocí výběru barvy.

**Práce s obrázky**

* Načtěte obrázek přes menu "File > Load Image".
* Uložte obrázek přes menu "File > Save Image".
* Obnovte původní obrázek přes "File > Restore Original Image".

**Filtry**

* Aplikujte filtry z menu "Filters" (negativ, prahování, RGB úpravy).

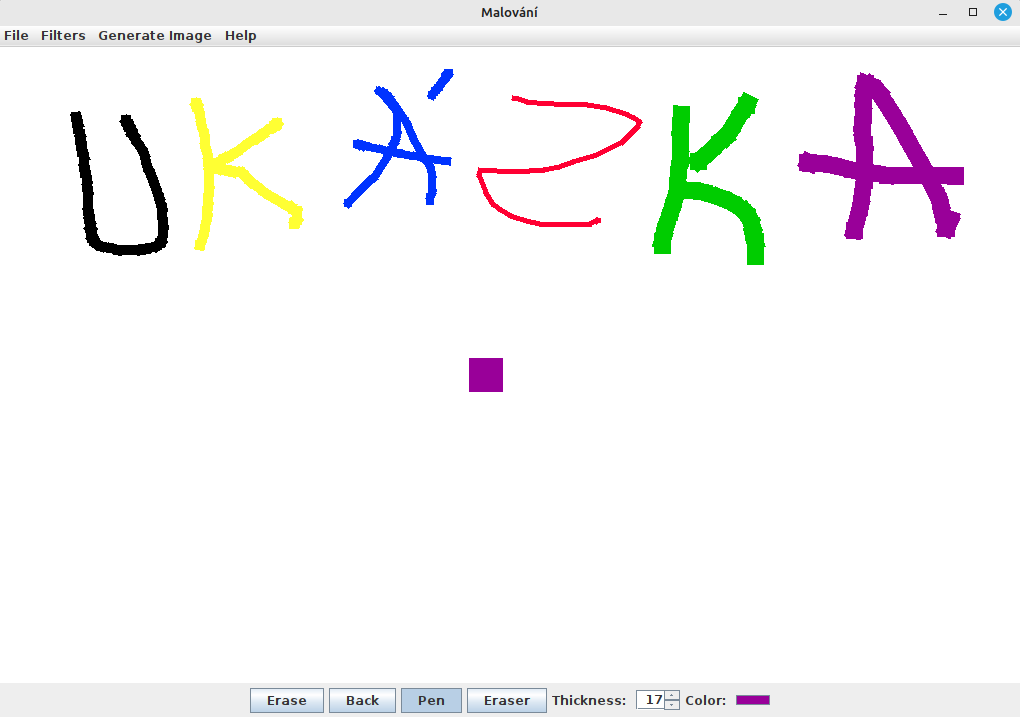
**Generování obrázků**

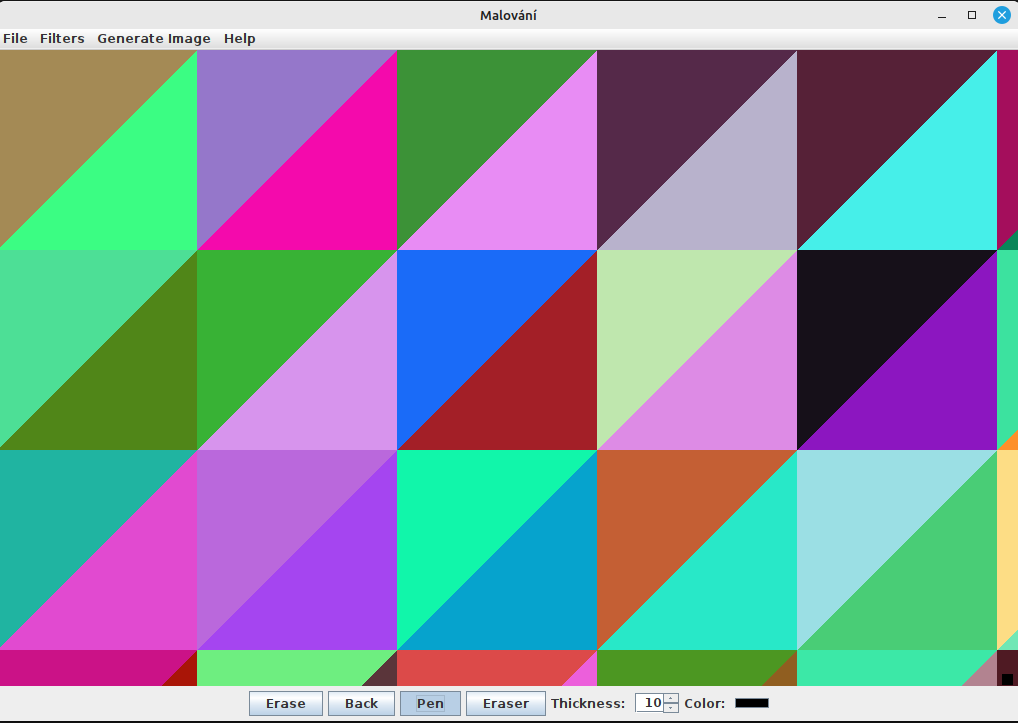
* Vygenerujte obrázek přes menu "Generate Image".

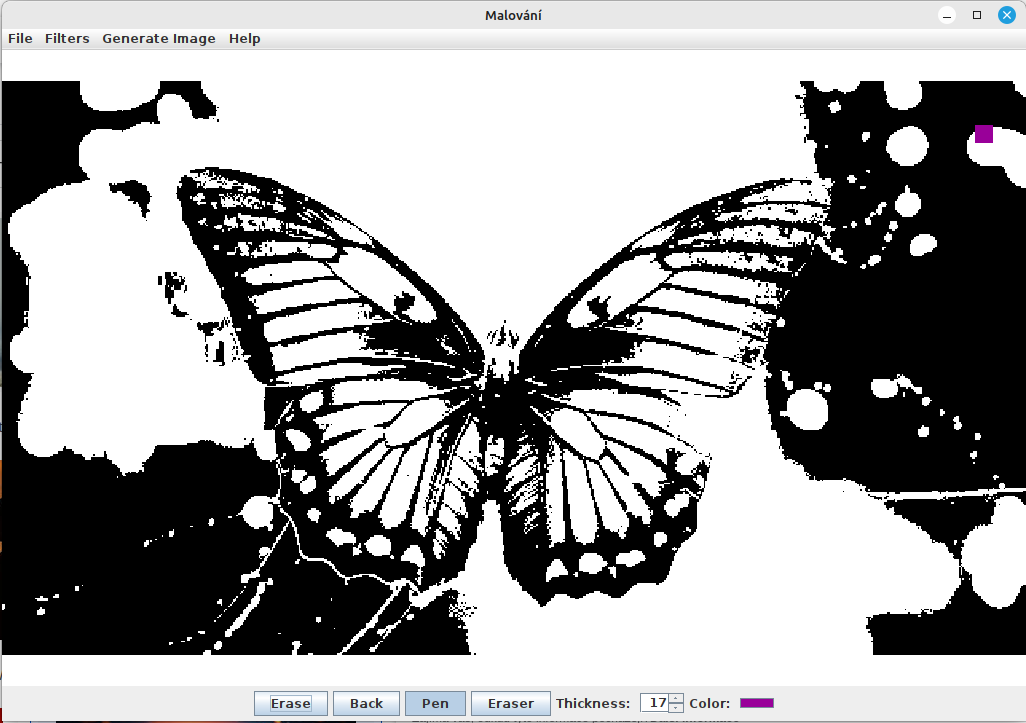
**Mazání a zpětné kroky**

* Použijte tlačítka "Erase" a "Back" pro vymazání a vrácení poslední akce.

# Ukázka







# Použitá literatura

K vypracování projektu jsme použili nespočet zdrojů. Mezi naše nejdůležitější zdroje patřila platforma YouTube a fórum Stack Overflow. Využili jsme také oficiální dokumentace přímo programovacího jazyku, nebo použitých knihoven.

# Závěr

Z role vedoucího hodnotím projekt úspěšně. V týmu jsme všichni spolupracovali pomalu jako jeden. Komunikace šla hladce a každý nalezený problém jsme vzápětí společně vyřešili. Asi můžu za všechny říct, že na projekt budeme rádi vzpomínat.