STŘEDNÍ PRŮMYSLOVÁ ŠKOLA  
MLADÁ BOLESLAV

**ROČNÍKOVÁ PRÁCE**

Vojtěch Vít

Mladá Boleslav 2022

STŘEDNÍ PRŮMYSLOVÁ ŠKOLA  
MLADÁ BOLESLAV

**ROČNÍKOVÁ PRÁCE**

**Autor: Vojtěch Vít  
Studijní obor: 18-20-M/01 Informační technologie  
Vedoucí práce: Lukáš Mázl**

Mladá Boleslav 2022

# Obsah

[Obsah 3](#_Toc102299635)

[1 Úvod 6](#_Toc102299636)

[1.1 Navržení aplikace 7](#_Toc102299637)

[1.1.1 Test 8](#_Toc102299638)

[1.2 Problematiky a logiky 9](#_Toc102299639)

[1.2.1 Konvoluce 10](#_Toc102299640)

[1.2.2 Úprava barevných kanálů RGB 11](#_Toc102299641)

[1.3 Práce s Java FX 11](#_Toc102299642)

[1.3.1 Vzhled aplikace 12](#_Toc102299643)

[1.3.2 Propojení skrze controller 12](#_Toc102299644)

[2 Závěr 13](#_Toc102299645)

[3 Přílohy 14](#_Toc102299646)

[3.1 Seznam obrázků 14](#_Toc102299647)

[3.2 Zdroje 14](#_Toc102299648)

**Prohlášení**

Prohlašuji, že jsem svou ročníkovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Nemám závažný důvod proti zpřístupňování této ročníkové práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) v platném znění.

V Mladé Boleslavi dne podpis:

# Úvod

Smyslem práce bylo vytvořit aplikaci prostřednictvím jazyku Java s pomocí Java FX v programátorské aplikaci IntelliJ IDEA, která bude umět upravovat s obrázky pomocí filtrů a problematiky konvoluce. Toto téma bylo navrženo vedoucím práce, které autor zpracoval a nadále vytvořil aplikaci, která se do budoucna dá vylepšit a upravit. Aplikace umožňuje uživateli nahrát do aplikace obrázek, vybrat si libovolný filtr a ten nadále aplikovat. Po aplikaci filtru se zavolá konvoluce, která upraví jeho hodnoty a upraví tak barevné kanály obrázku. Na tento upravený obrázek nám aplikace dá náhled. Tento styl úpravy obrázků nenabízí velikou škálu praktických využití, ale umožňuje například na obrázku detekovat hrany, zaostřit, či rozostřit. Pouze při použití správného filtru. Uživatel má v aplikaci možnost vybírat z před vytvořených filtrů.

## Navržení aplikace

Ze začátku bylo třeba vymyslet, jak bude aplikace navržena a zkonstruována. Autor za pomoci vedoucího vytvořili konstrukci, kde je v jednotlivých balíčcích uložena zvlášť logika problematiky, jednotlivé filtry, modely a soubory potřebné pro Java FX. V aplikaci jsou použity rozhraní neboli Interface, který prezentuje jednu celou problematiku, nebo objekt. Zde jsou konkrétně použita rozhraní na prezentaci samotné konvoluce a poté dvě další rozhraní na prezentaci obrázku a filtru. Pomocí rozhraní se na jednotlivé věci lépe odkazuje.

Obsah obrázku text, monitor, snímek obrazovky, černá

Popis byl vytvořen automaticky

### Test

V aplikaci je vytvořený i test, který otestuje zda, zda konvoluce funguje správně. Test nefunguje na barevné kanály, funguje pouze na samostatnou konvoluci, tudíž testuje, zda roznásobujeme obrázek a filtr správně. Test obsahuje připravené hodnoty pro obrázek, filtr a také hodnoty na výstupu po provedení konvoluce.

**private** Image getMockImage() {

**return** **new** Image() {

**@Override**

**public** int[][][] getArray() {

**return** **new** int[][][]{

{

{1, 0, 1},

{1, 0, 1},

{0, 0, 1}

}

};

}

};

}

**private** Filter getMockFilter() {

**return** **new** **Filter**() {

**@Override**

**public** int[][] getArray() {

**return** **new** int[][]{

{1, 0, 1},

{1, 0, 1},

{0, 0, 1}

};

}

};

}

**@Test**

**public** void testConvolution() {

int[][][] convolutionResult = convolutionService.convolution(getMockImage(), getMockFilter());

assertArrayEquals(convolutionResult, **new** int[][][]{

{

{0, 3, 0},

{0, 5, 0},

{0, 3, 0}

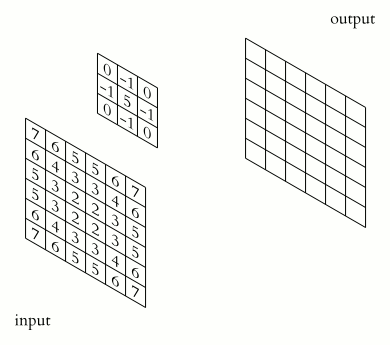
}

});

}

## Problematiky a logiky

Tuto aplikaci provází několik problematik. Základní a hlavní problematika je konvoluce viz. obrázek. Kde jde zejména o roznásobování obrázku a flitru do outputu. Další problematika je převádění obrázku na int pole a zase zpět kvůli barevným kanálům (RGB). Další problematika, která je spíše problém jazyka a toť jest, že aplikace v jedné části pracuje s komponentou image, který je součástí Java FX a další část pracuje s komponentou image z balíčku awt a aplikace má poté problém s oběma najednou pracovat, tudíž je třeba metoda, která je bude převádět.



### Konvoluce

Konvoluce, která představuje hlavní problematiku této práce zobecňuje násobení matice. U vybraného filtru vypočítáme středové souřadnice, pokud bychom chtěli filtr, který není 3x3, což v tomto případě není potřeba, ale pokud by autor chtěl aplikaci v budoucnu upravit, toto mu usnadní práci. Vypočítaný střed filtru poté postupně projíždí matici a na každém indexu se provede roznásobení hodnot v rozsahu filtru od jeho prvního indexu, pokud filtr rozsahem přesahuje obrázek hodnota se násobí nulou, nebo vedlejším indexem, který je v rozsahu. Poté co se roznásobí všechny hodnoty, tak se sečtou a výsledkem je jedno číslo, které se zapíše do nové výsledné matice.

**for** (**int** i = 0; i < image.length; i++) {

**for** (**int** j = 0; j < image[0].length; j++) {

**for** (**int** k = 0; k < filterArray.length; k++) {

**for** (**int** l = 0; l < filterArray[0].length; l++) {

**if**(k + i - xMiddleCoordinations < 0 || l + j - yMiddleCoordinations < 0 || k + i - xMiddleCoordinations > (image.length - 1) || l + j - yMiddleCoordinations > (image[0].length - 1)){

**continue**;

}

**else**{

output[i][j] += image[i + k - xMiddleCoordinations][j + l - yMiddleCoordinations] \* filterArray[k][l];

}

}

}

}

Zdrojový kód č. 1

### Úprava barevných kanálů RGB

Součástí aplikace je převod obrázku na int pole. Obrázek se zobrazuje jako trojrozměrné pole neboli [][][]. Kde první dimenze představuje barevné kanály (RGB = red, green, blue), druhá dimenze šířku a třetí výšku. Pomocí cyklu dvou vnořených cyklů for projdeme obrázek a získáme si hodnotu rgb z jednotlivých pixelů (1 pixel = 1 index pole).

int pixelIntValue = bufferedImage.getRGB(i, j);

int r = (pixelIntValue >> 16 << 24 >>> 24);

int g = (pixelIntValue >> 8 << 24 >>> 24);

int b = (pixelIntValue << 24 >>> 24);

RGB[0][i][j] = r;

RGB[1][i][j] = g;

RGB[2][i][j] = b;

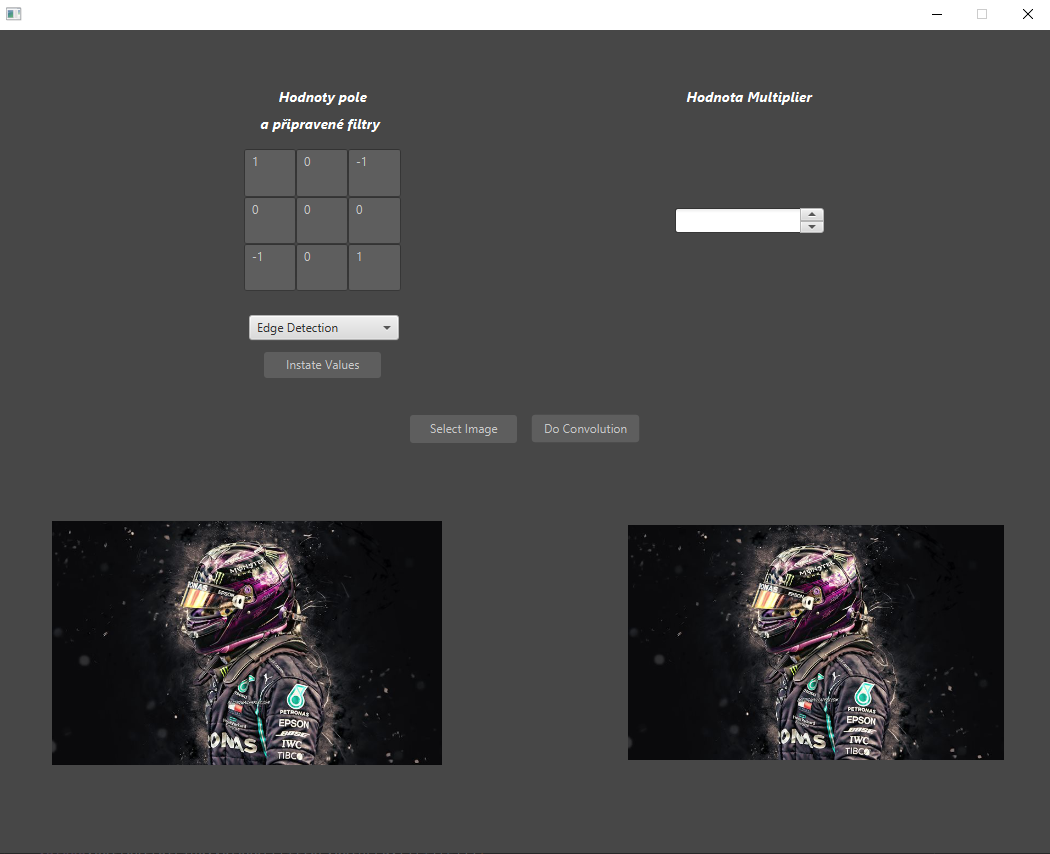
Zdrojový kód č. 1

Poté aplikace potřebuje metodu, kde tuto operaci provede nazpět, tedy pole int převede zpět na obrázek. Tato metoda je sestrojena prakticky identicky jako první.

## Práce s Java FX

Aplikace byla navržena pomocí Java FX scene builder, byla navržena jednoduše a minimalisticky, přes to umožňuje vylepšení do budoucna, jako je například hodnota multiplier. Nebo možnost vytvoření vlastního filtru. Aplikace nám umožňuje náhled na dva obrázky, a to před a po použití filtru. Aplikace má také k dispozici dvě tlačítka, pro nahrání obrázku a pro použití a aplikace konvoluce. Aplikace umožňuje uživateli vybírat z před vytvořených filtrů a pomocí tlačítka dosadit jeho hodnoty do grafické matice. Na aplikaci je napojený soubor css, který se stará o vzhled aplikace.

### Vzhled aplikace

Aplikace byla vzhledově navržena, aby připomínala aplikaci Photoshop. Jelikož se v aplikaci upravují obrázky, tak to autorovi přišlo tematické. Barevné schéma je tedy do šedé barvy a text do bílé, kontrast dobře sedí a nic neztrácí čitelnost a ni viditelnost.

### Propojení skrze controller

Každý komponent v aplikaci je propojený do controlleru, kde je řešena jeho funkčnost. Například u tlačítka nahrání soboru je přes funkci onMouseClick řešena logika otevření průzkumníku, nebo nahrání obrázku do komponentu ImageView.

# Závěr

Aplikaci jsem navrhl za pomoci svého vedoucího. Cílem bylo vytvořit aplikaci, která bude umožňovat úpravu obrázků za pomocí konvoluce. Myslím si, že cíl jsem splnil a obsah naplněn v některých ohledech jsem udělal i něco navíc. V aplikaci jsem zanechal možnost pro možné budoucí rozšíření z důvodů ubývajícího času a také z důvodu, že bych si tuto aplikaci chtěl vzít jako dlouhodobou maturitní práci. Osobně jsem z půle spokojen, aplikace funguje dobře, a i dle mého dobře vypadá, ale nejsem spokojen s věcmi, které jsem nestihl a které aplikace neumí, do budoucna se aplikaci určitě budu ještě věnovat a tyto věci doladím.

# Přílohy

## Seznam obrázků

Obrázky jsem používal pouze ze svého zdroje ve formě screenshotu z mé práce.

## Zdroje

[*https://gist.github.com/alexpahom/1d41a8d4148eb778963edf1ef8c25087*](https://gist.github.com/alexpahom/1d41a8d4148eb778963edf1ef8c25087)

https://docs.oracle.com/javase/tutorial/uiswing/components/filechooser.html