



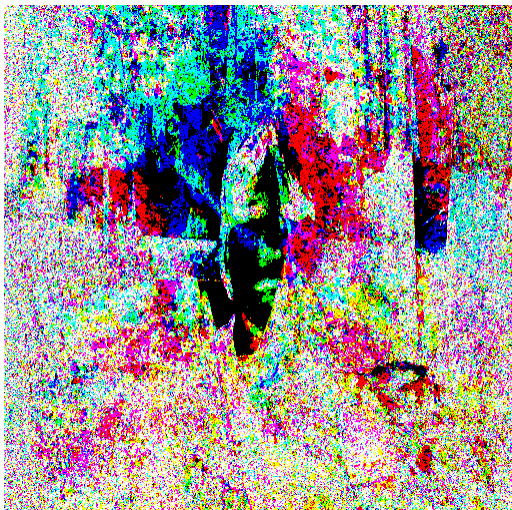
Leevi Virtanen

## Kuvaus

ImageMangler on kuvankäsittelyohjelma. Sen tarkoitus on tarjota taiteilijoille nopea ja tietokoneelle kevyt tapa käsitellä kuvia ainutlaatuisilla tavoilla. Se on hyvä tuottamaan muun muassa levynkansia ja tekstuureja. Keskimääräinen ihminen voisi käyttää sitä esimerkiksi taiteellisten kuvakollaasien tekemiseen.

ImageMangler hyödyntää boolean algebraa ja bittien siirtelyä kuvien käsittelyyn. Bittien siirtelystä johtuva artefaktien muodostuminen tarjoaa uuden tavan muuttaa kuvan kirkkautta epätasaisemmin ja taiteellisemmin. Ohjelman pääominaisuus on kuitenkin kuvien väriarvojen epätavallinen prosessointi niiden vertauksilla ja muilla laskutoimituksilla.

## Esimerkkejä



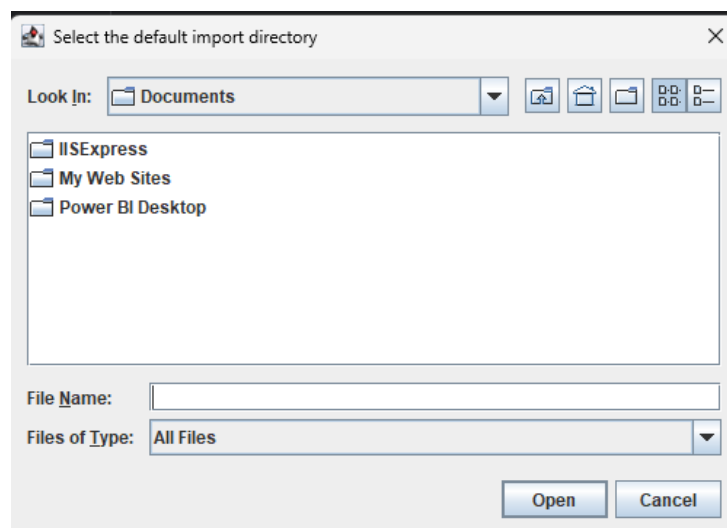
# Sisällysluettelo

<b>Kuvaus</b>	<b>1</b>
<b>Käyttöliittymä</b>	<b>3</b>
Pääkäyttöliittymä:	3
<b>Luokkien kuvaus</b>	<b>4</b>
ImageTool	4
FileTool	5
saveFile	5
PixelProcessor	6
ImageMangler	7
<b>Projektin kehitys ja mahdollinen jatkokehitys</b>	<b>9</b>

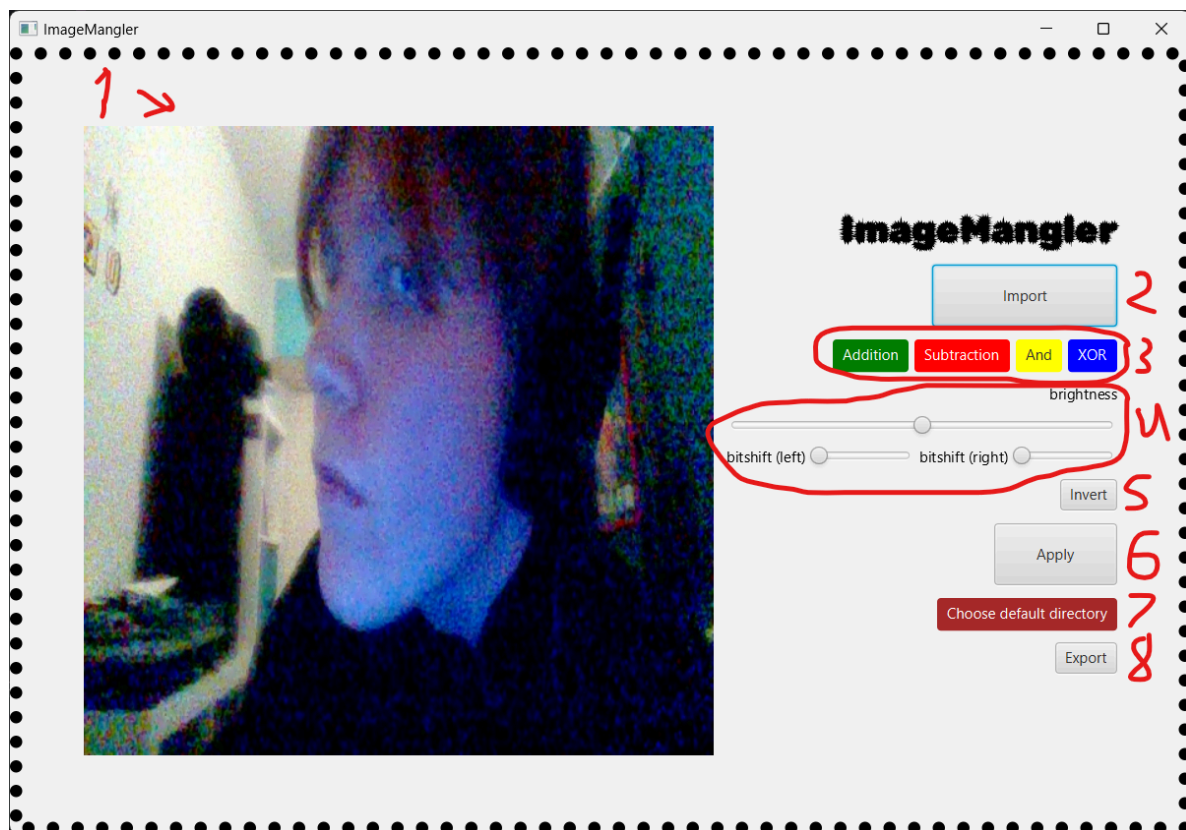
## Käyttöliittymä

Ohjelma avautuu ensin vain tiedostoselaimena ja kehottaa käyttäjää valitsemaan oletushakemiston.

Valitse hakemisto, josta aiot tuoda kuvia eniten. Suosittelena varsinkin kuvia luonnosta tai selfieitä.



## Pääkäyttöliittymä



1. Näyttää kuvan, jota käsitellään. Tämä on myös mitä kuvan vieminen tuottaa ulostulona.
2. Tuo uuden kuvan. Poistaa nykyisen kuvan.
3. Kuvan tuonti ja sen vertaus ja matemaattiset toiminnot. Näiden teknisestä toiminnasta lisää luokkien kuvauksessa. (HUOM! Muuttavat heti kuvan kun toinen valitaan!)
4. Kirkkaus ja bittien liikutus toiminnot, jotka laitetaan käyttöön toimeenpano napista, koska ne voivat vähentää kuvan bitti syvyyttä merkittävästi.
5. Kääntää kuvan värit.
6. Toimeenpano nappi, joka toimeenpanee kirkkaus ja bitshift asetukset.
7. Valitsee oletushakemiston, josta ohjelma hakee oletuksella kuvia
8. Vie kuvan samaan kansioon, jossa ohjelma on.

# Luokkien kuvaus

## ImageTool

Abstrakti luokka, joka sisältää funktioita, joita käytetään kuvien tietotyyppien muuttamiseen, kokojen muuttamiseen ja tuomiseen. Datatyyppien ja kokojen muutoksiin käytetyt funktiot ovat StackOverFlow käyttäjiltä [Kevin Bähre](#) ja [Ocracoke](#). Tämä siksi, että Javan natiivit funktiot samoihin toimintoihin veisivät merkittävästi enemmän aikaa, joka on merkityksellistä näin nopeatempoisessa ohjelmassa.

`BufferedImage resize(BufferedImage, int, int)`

Määrittää BufferedImagen koon uudelleen int parametrien perusteella ja palauttaa sen.

`Image getImage(BufferedImage)`

Muuntaa annetun BufferedImagen JavaFX Imageen ja palauttaa sen.

`BufferedImage pickImage((File))`

Avaa tiedostoselaimen ja palauttaa kuvan, jonka käyttäjä valitsee sieltä. Ylikuormitetussa versiossa methodia avaa valmiiksi parametrin antaman hakemiston.

## FileTool

Abstrakti luokka, joka sisältää funktioita, joita käytetään tiedostoselaimesta hakemistojen etsintään, tiedostojen löytämiseen ja käyttäjätiedon tallentamiseen ja lukemiseen.

`String Find(String)`

Etsii resources kansioista tiedoston ja palauttaa sen polun stringinä.

Boolean Save(BufferedImage)

Tallentaa BufferedImagen projektikansioon PNG:nä. Nimi on aina "img"+satunnainen integer 0-9999. Jos tiedoston tallennus ei onnistu palauttaa false, muuten true.

File pickDirectory()

Avaa tiedostoselaimen ja palauttaa käyttäjän valitseman hakemiston File tietotyyppinä.

void writeSave(saveFile)

Serialisoi ja tallentaa saveFilen projektikansioon nimellä "ImgMng.mangle".

saveFile readSave()

Yrittää lukea projektikansiosta tiedoston "ImgMng.mangle", deserialisoi ja palauttaa sen.

## saveFile

Käyttäjätieto luokka, joka käyttää Serializable luokkaa. Sisältää tiedon siitä milloin ohjelma on viimeksi avattu ja oletushakemiston.

File defaultDir

Oletushakemisto. Getterit ja setterit.

String saveDate

Päivämäärä siitä milloin viimeksi ohjelmisto avattiin. Getter

saveFile((saveFile)|(File))

Constructori, joka joko ottaa argumentteina Filen, joka tallennetaan oletushakemistoksi tai saveFilen, jolloin constructori tallentaa annetun saveFilen tiedot tehtyyn olioon ja tallentaa tiedoston kiintolevylle FileTool.writeSave:lla.

`setSaveData(save)`

Tallentaa annetun `saveFile` olion tiedot omistavaan `saveFile` olioön ja tallentaa tiedot kiintolevylle `FileToll.writeSave:lla`.

## **PixelProcessor**

Abstrakti luokka, jota käytetään kuvien muokkaukseen.

Pääperiaate on vain jokaiselle pikselille valitun muokkauksen teko tai kahden kuvan pikseleiden vertaus tai matemaattinen laskelma.

`Int colorClamp(int color)`

Rajoittaa `int` arvon -1 ja 256 välille ja palauttaa sen.

`BufferedImage changeAll(BufferedImage, double, int, int)`

Lisää annetun `BufferedImage`n pikseleihin väriarvoihin `double`n.

Bitshifttaa `int1` kertaa vasemmalle ja `int2` kertaa oikealle.

Palauttaa kaikkien operaatioiden jälkeen `BufferedImage`n.

`BufferedImage imgSubstract(BufferedImage, BufferedImage)`

Iteroi `BufferedImage1`:en pixelien väriarvojen läpi ja vähentää niistä `BufferedImage2`:en samassa kohdassa olevien pixelien väriarvot. Palauttaa `BufferedImage1`:en.

`BufferedImage imgAdd(BufferedImage, BufferedImage)`

Iteroi `BufferedImage1`:en pixelien väriarvojen läpi ja lisää niihin `BufferedImage2`:en samassa kohdassa olevien pixelien väriarvot. Palauttaa `BufferedImage1`:en.

BufferedImage imgAnd(BufferedImage, BufferedImage)  
Iteroi BufferedImage1:en pixelien väriarvojen läpi ja vertaa niiden bittejä BufferedImage2:en pixelien väriarvojen bitteihin AND(&) operaattorilla ja palauttaa lopputuloksen.

BufferedImage imgXor(BufferedImage, BufferedImage)  
Iteroi BufferedImage1:en pixelien väriarvojen läpi ja vertaa niiden bittejä BufferedImage2:en pixelien väriarvojen bitteihin XOR(^) operaattorilla ja palauttaa lopputuloksen.

BufferedImage invert(BufferedImage)  
Vähentää 255:stä annetun BufferedImagen pixelien väriarvot ja palauttaa lopputuloksen.

## ImageMangler

Pääluokka jossa määritellään ohjelman käyttöliittymä ja nappien tapahtumakohtaiset metodit.

Image img  
JavaFX versio bufimg:sta, joka näytetään mainviewissä. Ns. pääkuva.

BufferedImage bufimg  
Kuva joka on ohjelmiston ulostulo ja jota muokataan ohjelmistossa.

ImageView mainview  
Näyttää img:n.

Slider bitshift  
Määrittelee montako kertaa toimeenpano nappi liikuttaa img:en bittejä vasemmalle.

Slider bitshiftright  
Määrittelee montako kertaa toimeenpano nappi liikuttaa img:en bittejä oikealle.

saveFile save

Käyttäjätiedon tallennus olio.

int canvasSizeX

Määrittelee mihin leveyteen kaikki kuvat määritellään.

Int canvasSizeY

Määrittelee mihin pituuteen kaikki kuvat määritellään.

EventHandler<ActionEvent> importImg(VBox)

Avaa tiedostoselain ja muuttaa mainviewin kuvan valituksi kuvaksi. Jos defaultDir on olemassa avaa sen oletuksena. Tekee VBoxista näkymättömän.

EventHandler<ActionEvent> setDefaultDirectory()

Avaa tiedostoselain ja laittaa valitun hakemiston oletushakemistoksi.

EventHandler<ActionEvent> addition()

Avaa tiedostoselain ja käyttää PixelProcessor.imgAdd:in argumentteina nykyistä kuvaa ja valittua kuvaa. Vaihtaa pääkuvan lopputulokseksi. Jos defaultDir on olemassa avaa sen oletuksena.

EventHandler<ActionEvent> subtraction()

Avaa tiedostoselain ja käyttää PixelProcessor.imgSubtract:in argumentteina nykyistä kuvaa ja valittua kuvaa. Vaihtaa pääkuvan lopputulokseksi. Jos defaultDir on olemassa avaa sen oletuksena.

EventHandler<ActionEvent> and()

Avaa tiedostoselain ja käyttää PixelProcessor.imgAnd:in argumentteina nykyistä kuvaa ja valittua kuvaa. Vaihtaa pääkuvan lopputulokseksi. Jos defaultDir on olemassa avaa sen oletuksena.

EventHandler<ActionEvent> xor()

Avaa tiedostoselain ja käyttää PixelProcessor.imgXor:in argumentteina nykyistä kuvaa ja valittua kuvaa. Vaihtaa

pääkuvan lopputulokseksi. Jos defaultDir on olemassa avaa sen oletuksena.

```
EventHandler<ActionEvent> invert()
```

Käyttää PixelProcessor.invert nykyiseen kuvaan ja muuttaa pääkuvan lopputulokseksi.

```
EventHandler<ActionEvent> updateImage()
```

Käyttää PixelProcessor.changeAll argumenteilla bufimg, brightnessin arvo, bitshiftin arvo muunnettuna kokonaisluvuksi ja bitshiftRightin arvon muunnettuna kokonaisluvuksi. Asettaa lopputuloksen pääkuvaksi.

```
EventHandler<ActionEvent> exportImage()
```

Tallentaa kiintolevyllä bufimg:n FileTool.Save:lla.

## **Projektin kehitys ja mahdollinen jatkokehitys**

ImageManglerin alkukehitys oli vaivalloista. Ensin opiskelin sitä miten Java käsittelee kuvia ja mikä on optimaalinen tapa saada eri kuvien tietotyypit muutettua toisiinsa, jotta voisin käyttää helposti muokattavinta, mutta samalla tehokasta tietotyyppiä. Käyttöliittymän kehityksessä kului eniten aikaa standardi koodin tuottoon, jota joutuu tuottamaan aina JavaFX:llä paljon.

Projektin kehityksen aikana huomasin miten hyödyllisiä kirjastoja Javassa on kuvien käsittelyyn. Jos olisin lähtenyt tekemään samaa projektia C++:lla, olisi kulunut varmaankin kolminkertaisesti aikaa jos ei enempää. Tunsin itseni ystävystyvän tuotannon aikana Javan ja JavaFX:än kanssa.

Sain ideani tällaiseen kuvan käsittelyyn FM synteesistä, jossa toisesta ääniaallosta poistetaan toinen ääniaalto. Tämän idean sitten laajensin bitti operaatioihin, koska bittien kanssa leikkiminen on ihan kivaa ja lisäys ja poisto ovat jo laajalti käytössä muissa kuvan käsittely työkaluissa.

Hauskin osa tuotantoa oli keksiä erilaisia tapoja muokata kuvia ohjelman rakenteiden tuottamissa rajoituksissa. Varmaankin jatkokehitys pohjautuisi juuri laajasti eri kuvan muokkaus työkaluihin kuten kohinan poistoon, eri proceduaalisten algoritmien perusteella kuvien vertailuun tai free mode jossa kuvakoot voi päättää itse. Jatkokehityksen voisi mahdollisesti myös ulkoistaa tekemällä projektista avointa lähdekoodia ja mainostamalla projektia.

ImageManglerin merkittävin huono puoli on ohjelman avaamisen aikana tulevat vaihtoehdot, jotka saattavat hämmentää ihmisiä. Mielestäni ImageMangler on kokonaisuudessaan kuitenkin toimiva ja yksinkertainen toteutus toisten eri alojen toimivien ja yksinkertaisten ideoiden yhdistyksestä.