**Filters.java**

Zawiera metody nakładania filtrów:

***Szaro-odcieniowego:***

public BufferedImage GrayFilter(BufferedImage originalImage);

originalImage – obrazek do nakładania filtru

**Sepii:**

public BufferedImage SepiaFilter(BufferedImage originalImage);

originalImage – obrazek do nakładania filtru

**Negatywu:**

public BufferedImage NegativeFilter(BufferedImage originalImage);

originalImage – obrazek do nakładania filtru

**Zamiany koloru na wybrany:**

public BufferedImage ChangeFilter(BufferedImage originalImage, int fromColor, int toColor);

originalImage – obrazek do nakładania filtru

fromColor – wartość piksela który będziemy zamieniać

fromColor – wartość piksela na którą będziemy zamieniać

Oraz metodę tworzenia argb piksela zapodannymi wartościami :

public int MakePixel(int A, int R, int G, int B)

A – wartość Alpha

R – wartość Red

G – wartość Green

B – wartość Blue

**MainFrame.java**

**Inicjalizacja:**

public BufferedImage originalPicture = null; //oryginalny obrazek

public BufferedImage myPicture = null; //zmieniony obrazek

public int color; //zmienna do wyliczania koloru rysowania

public int fromColor; //zmienna do wyliczania koloru oryginalnego do zamiany

public int toColor; //zmienna do wyliczania koloru nowego do zamiany

Boolean Pressed = false; //indykator naciśnięcia na obszar do rysowania

Boolean Mode = true; //indykator trybu wskażnika myszy(true - rysowanie, false - zamiana koloru)

Filters filters = new Filters(); //inicjalizacja filtrów

**Metody:**

public void GetColor()

Pobiera wartości sliderów i zmienia kolor tła ColorPanel

public void GetColorChanged()

Pobiera wartości sliderów i zmienia kolor tła ColorPanelChanged

public BufferedImage resizeImage(BufferedImage originalImage, int type)

Przyjmuje BufferedImage. Zmienia jego rozmiary żeby obrazek nie był zbyt rozciągnięty

Jest to zrobione specjalnie dla Icon w Label, zwraca BufferedImage

originalImage – obrazek oryginalny

type – typ piksela (RGB,ARGB,etc.)

public void openPicture()

Tworzy i otwiera okno dialogowe dla otwarcia obrazu I zapisuje jedo w *myPicture*

public void savePicture()

Tworzy i otwiera okno dialogowe dla zapisywania obrazu

private void OptOpenActionPerformed(java.awt.event.ActionEvent evt)

Przycisk otwierania obrazku

private void OptGrayActionPerformed,

private void OptSepiaActionPerformed(java.awt.event.ActionEvent evt),

private void OptNegativeActionPerformed(java.awt.event.ActionEvent evt),

private void SwapButtonActionPerformed(java.awt.event.ActionEvent evt)

Uruchamiają Filtr oraz wyświetlają komunikat jeżeli nie było otwarto obrazku

private void OptSaveActionPerformed(java.awt.event.ActionEvent evt)

Przycisk zapisywania obrazku

private void ImageHolderPanelMousePressed(java.awt.event.MouseEvent evt)

Zmienia Pressed na true po naciskaniu na obszar dla rysowania

private void ImageHolderPanelMouseMoved(java.awt.event.MouseEvent evt)

Wyświetla koordynaty oraz zamienia kursor

private void ImageHolderPanelMouseDragged(java.awt.event.MouseEvent evt)

Po ruchaniu wciśniętą myszą wstawia piksel i odrazu wyświetla zmieniony obrazek

private void RSliderStateChanged(javax.swing.event.ChangeEvent evt),

private void GSliderStateChanged(javax.swing.event.ChangeEvent evt),

private void BSliderStateChanged(javax.swing.event.ChangeEvent evt)

Obsługa sliderów do zmiany koloru dla rysowania

private void ImageHolderPanelMouseExited(java.awt.event.MouseEvent evt)

Po wyjściu za obszar dla rysowania zmienia kursor na defoltowy

private void RChooseSliderStateChanged(javax.swing.event.ChangeEvent evt),

private void GChooseSliderStateChanged(javax.swing.event.ChangeEvent evt),

private void BChooseSliderStateChanged(javax.swing.event.ChangeEvent evt)

Obsługa sliderów do wyboru koloru zamiany

private void SwapButtonActionPerformed(java.awt.event.ActionEvent evt)

Zamiana koloru oraz wyświetla komunikat jeżeli nie było otwarto obrazku

private void ChooseButtonActionPerformed(java.awt.event.ActionEvent evt)

Zmiana trybu przy naciśnięciu na wybór colour

private void DrawButtonActionPerformed(java.awt.event.ActionEvent evt)

Zmiana trybu przy naciśnięciu na rysowanie

**Kod MainFrame.java:**

package myproject;

import java.awt.image.BufferedImage;

import java.awt.\*;

import java.io.File;

import java.io.IOException;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.imageio.ImageIO;

import javax.swing.\*;

import javax.swing.filechooser.FileNameExtensionFilter;

/\*\*

\*

\* @author Persei

\*/

public class MainFrame extends javax.swing.JFrame {

/\*\*

\* Creates new form MainFrame

\*/

public BufferedImage originalPicture = null; //oryginalny obrazek

public BufferedImage myPicture = null; //zmieniony obrazek

public int color; //zmienna do wyliczania koloru rysowania

public int fromColor; //zmienna do wyliczania koloru oryginalnego do zamiany

public int toColor; //zmienna do wyliczania koloru nowego do zamiany

Boolean Pressed = false; //indykator naciśnięcia na obszar do rysowania

Boolean Mode = true; //indykator trybu wskażnika myszy(true - rysowanie, false - zamiana koloru)

Filters filters = new Filters(); //inicjalizacja filtrów

public MainFrame() {

initComponents();

//wyłaczenie zmiany rozmiaru

setResizable(false);

}

/\*

Pobiera wartości sliderów i zmienia kolor tła ColorPanel

\*/

public void GetColor(){

color = filters.MakePixel(255, RSlider.getValue(), GSlider.getValue(), BSlider.getValue());

ColorPanel.setBackground(new Color(color));

}

/\*

Pobiera wartości sliderów i zmienia kolor tła ColorPanelChanged

\*/

public void GetColorChanged(){

toColor = filters.MakePixel(255, RChooseSlider.getValue(), GChooseSlider.getValue(), BChooseSlider.getValue());

ColorPanelChanged.setBackground(new Color(toColor));

}

/\*

Przyjmuje BufferedImage

Zmienia jego rozmiary żeby obrazek nie był zbyt rozciągnięty

Jest to zrobione specjalnie dla Icon w Label

Zwraca BufferedImage

\*/

public BufferedImage resizeImage(BufferedImage originalImage, int type){

BufferedImage resizedImage = new BufferedImage(PictureBoxLabel.getWidth(), PictureBoxLabel.getHeight(), type);

Graphics2D g = resizedImage.createGraphics();

g.drawImage(originalImage,0,0,PictureBoxLabel.getWidth(),PictureBoxLabel.getHeight(),null);

g.dispose();

return resizedImage;

}

/\*

Tworzy i otwiera okno dialogowe dla otwarcia obrazu

\*/

public void openPicture() throws IOException{

JFileChooser fopen = new JFileChooser();

//Ustawia rozmiary okna dialogowego

fopen.setPreferredSize(new Dimension(600,500));

//Ustawiam filtr

fopen.setFileFilter(new FileNameExtensionFilter("Images", "jpg", "png","bmp"));

int ret = fopen.showDialog(null, "Open");

if(ret == JFileChooser.APPROVE\_OPTION){

//inicjalizuje originalPicture z podanego pliku

originalPicture = ImageIO.read(fopen.getSelectedFile().getAbsoluteFile());

//typ dla pixeli ustalam na TYPE\_INT\_ARGB jako defoltowy jeżeli niema albo taki jak w oryginalnym obrazku originalPicture

int type = originalPicture.getType() == 0? BufferedImage.TYPE\_INT\_ARGB : originalPicture.getType();

//zmiana rozmiarów

myPicture = resizeImage(originalPicture, type);

//Wstawiam obraz jako Icon w Label

PictureBoxLabel.setIcon(new ImageIcon(myPicture));

}

}

/\*

Tworzy i otwiera okno dialogowe dla zapisywania obrazu

\*/

public void savePicture() throws IOException {

JFileChooser fsave = new JFileChooser();

fsave.setPreferredSize(new Dimension(600, 500));

int ret = fsave.showSaveDialog(null);

if (ret == JFileChooser.APPROVE\_OPTION) {

File file = fsave.getSelectedFile();

ImageIO.write(myPicture, "png", file);

}

}

/\*\*

\* This method is called from within the constructor to initialize the form.

\* WARNING: Do NOT modify this code. The content of this method is always

\* regenerated by the Form Editor.

\*/

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

XLabel = new javax.swing.JLabel();

YLabel = new javax.swing.JLabel();

ImageHolderPanel = new javax.swing.JPanel();

PictureBoxLabel = new javax.swing.JLabel();

RSlider = new javax.swing.JSlider();

GSlider = new javax.swing.JSlider();

BSlider = new javax.swing.JSlider();

ColorPanel = new javax.swing.JPanel();

RtextHolder = new javax.swing.JLabel();

GtextHolder = new javax.swing.JLabel();

BtextHolder = new javax.swing.JLabel();

RChooseSlider = new javax.swing.JSlider();

ColorPanelChanged = new javax.swing.JPanel();

ColorPanelOriginal = new javax.swing.JPanel();

GChooseSlider = new javax.swing.JSlider();

BChooseSlider = new javax.swing.JSlider();

DrawButton = new javax.swing.JButton();

ChooseButton = new javax.swing.JButton();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jLabel3 = new javax.swing.JLabel();

SwapButton = new javax.swing.JButton();

RtextHolder1 = new javax.swing.JLabel();

RtextHolder2 = new javax.swing.JLabel();

RtextHolder3 = new javax.swing.JLabel();

jMenuBar1 = new javax.swing.JMenuBar();

jMenu1 = new javax.swing.JMenu();

OptOpen = new javax.swing.JMenuItem();

OptSave = new javax.swing.JMenuItem();

jMenu2 = new javax.swing.JMenu();

OptGray = new javax.swing.JMenuItem();

OptSepia = new javax.swing.JMenuItem();

OptNegative = new javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setBackground(javax.swing.UIManager.getDefaults().getColor("Button.disabledForeground"));

setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT\_CURSOR));

setMinimumSize(new java.awt.Dimension(1239, 846));

XLabel.setText("X: 0");

YLabel.setText("Y :0");

YLabel.setToolTipText("");

ImageHolderPanel.setBackground(new java.awt.Color(255, 255, 255));

ImageHolderPanel.setPreferredSize(new java.awt.Dimension(1000, 800));

ImageHolderPanel.addMouseMotionListener(new java.awt.event.MouseMotionAdapter() {

public void mouseDragged(java.awt.event.MouseEvent evt) {

ImageHolderPanelMouseDragged(evt);

}

public void mouseMoved(java.awt.event.MouseEvent evt) {

ImageHolderPanelMouseMoved(evt);

}

});

ImageHolderPanel.addMouseListener(new java.awt.event.MouseAdapter() {

public void mouseExited(java.awt.event.MouseEvent evt) {

ImageHolderPanelMouseExited(evt);

}

public void mousePressed(java.awt.event.MouseEvent evt) {

ImageHolderPanelMousePressed(evt);

}

public void mouseReleased(java.awt.event.MouseEvent evt) {

ImageHolderPanelMouseReleased(evt);

}

});

PictureBoxLabel.setBackground(new java.awt.Color(153, 153, 153));

PictureBoxLabel.setName(""); // NOI18N

javax.swing.GroupLayout ImageHolderPanelLayout = new javax.swing.GroupLayout(ImageHolderPanel);

ImageHolderPanel.setLayout(ImageHolderPanelLayout);

ImageHolderPanelLayout.setHorizontalGroup(

ImageHolderPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(PictureBoxLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, 1000, Short.MAX\_VALUE)

);

ImageHolderPanelLayout.setVerticalGroup(

ImageHolderPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(PictureBoxLabel, javax.swing.GroupLayout.DEFAULT\_SIZE, 800, Short.MAX\_VALUE)

);

RSlider.setMaximum(255);

RSlider.setValue(255);

RSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

RSliderStateChanged(evt);

}

});

GSlider.setMaximum(255);

GSlider.setValue(255);

GSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

GSliderStateChanged(evt);

}

});

BSlider.setMaximum(255);

BSlider.setValue(255);

BSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

BSliderStateChanged(evt);

}

});

ColorPanel.setBackground(new java.awt.Color(255, 255, 255));

javax.swing.GroupLayout ColorPanelLayout = new javax.swing.GroupLayout(ColorPanel);

ColorPanel.setLayout(ColorPanelLayout);

ColorPanelLayout.setHorizontalGroup(

ColorPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 0, Short.MAX\_VALUE)

);

ColorPanelLayout.setVerticalGroup(

ColorPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 38, Short.MAX\_VALUE)

);

RtextHolder.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

RtextHolder.setText("R");

GtextHolder.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

GtextHolder.setText("G");

BtextHolder.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

BtextHolder.setText("B");

RChooseSlider.setMaximum(255);

RChooseSlider.setValue(255);

RChooseSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

RChooseSliderStateChanged(evt);

}

});

ColorPanelChanged.setBackground(new java.awt.Color(255, 255, 255));

javax.swing.GroupLayout ColorPanelChangedLayout = new javax.swing.GroupLayout(ColorPanelChanged);

ColorPanelChanged.setLayout(ColorPanelChangedLayout);

ColorPanelChangedLayout.setHorizontalGroup(

ColorPanelChangedLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 40, Short.MAX\_VALUE)

);

ColorPanelChangedLayout.setVerticalGroup(

ColorPanelChangedLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 38, Short.MAX\_VALUE)

);

ColorPanelOriginal.setBackground(new java.awt.Color(255, 255, 255));

javax.swing.GroupLayout ColorPanelOriginalLayout = new javax.swing.GroupLayout(ColorPanelOriginal);

ColorPanelOriginal.setLayout(ColorPanelOriginalLayout);

ColorPanelOriginalLayout.setHorizontalGroup(

ColorPanelOriginalLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 40, Short.MAX\_VALUE)

);

ColorPanelOriginalLayout.setVerticalGroup(

ColorPanelOriginalLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 38, Short.MAX\_VALUE)

);

GChooseSlider.setMaximum(255);

GChooseSlider.setValue(255);

GChooseSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

GChooseSliderStateChanged(evt);

}

});

BChooseSlider.setMaximum(255);

BChooseSlider.setValue(255);

BChooseSlider.addChangeListener(new javax.swing.event.ChangeListener() {

public void stateChanged(javax.swing.event.ChangeEvent evt) {

BChooseSliderStateChanged(evt);

}

});

DrawButton.setIcon(new javax.swing.ImageIcon(getClass().getResource("/myproject/pointer\_click\_arrow\_mouse\_cursor-512.png"))); // NOI18N

DrawButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

DrawButtonActionPerformed(evt);

}

});

ChooseButton.setIcon(new javax.swing.ImageIcon(getClass().getResource("/myproject/ColorSwap.jpg"))); // NOI18N

ChooseButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

ChooseButtonActionPerformed(evt);

}

});

jLabel1.setText("Rysowanie");

jLabel2.setText("Zamiana koloru");

jLabel3.setText("----->");

jLabel3.setToolTipText("");

SwapButton.setText("Change Color");

SwapButton.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

SwapButtonActionPerformed(evt);

}

});

RtextHolder1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

RtextHolder1.setText("R");

RtextHolder2.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

RtextHolder2.setText("G");

RtextHolder3.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N

RtextHolder3.setText("B");

jMenu1.setText("File");

OptOpen.setText("Open");

OptOpen.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OptOpenActionPerformed(evt);

}

});

jMenu1.add(OptOpen);

OptSave.setText("Save");

OptSave.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OptSaveActionPerformed(evt);

}

});

jMenu1.add(OptSave);

jMenuBar1.add(jMenu1);

jMenu2.setText("Filters");

OptGray.setText("Gray");

OptGray.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OptGrayActionPerformed(evt);

}

});

jMenu2.add(OptGray);

OptSepia.setText("Sepia");

OptSepia.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OptSepiaActionPerformed(evt);

}

});

jMenu2.add(OptSepia);

OptNegative.setText("Negative");

OptNegative.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

OptNegativeActionPerformed(evt);

}

});

jMenu2.add(OptNegative);

jMenuBar1.add(jMenu2);

setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(YLabel, javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(XLabel, javax.swing.GroupLayout.Alignment.TRAILING))

.addGap(10, 10, 10))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(RSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, 175, Short.MAX\_VALUE)

.addComponent(GSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE)

.addComponent(BSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE)

.addComponent(ColorPanel, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addGap(30, 30, 30)

.addComponent(ColorPanelOriginal, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addComponent(ColorPanelChanged, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(GtextHolder)

.addComponent(RtextHolder))

.addGap(14, 14, 14))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addComponent(BtextHolder)

.addGap(17, 17, 17)))))

.addGap(0, 0, Short.MAX\_VALUE))

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)

.addComponent(BChooseSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT\_SIZE, 187, Short.MAX\_VALUE)

.addComponent(GChooseSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addComponent(ChooseButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel2))

.addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.createSequentialGroup()

.addComponent(DrawButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(jLabel1))

.addComponent(RChooseSlider, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.PREFERRED\_SIZE, 0, Short.MAX\_VALUE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(RtextHolder3)

.addComponent(RtextHolder2)

.addComponent(RtextHolder1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))))

.addGroup(layout.createSequentialGroup()

.addGap(55, 55, 55)

.addComponent(SwapButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)))

.addComponent(ImageHolderPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addContainerGap(24, Short.MAX\_VALUE))

);

layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(14, 14, 14)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)

.addComponent(ImageHolderPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(DrawButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addGroup(layout.createSequentialGroup()

.addGap(13, 13, 13)

.addComponent(jLabel1)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(ChooseButton, javax.swing.GroupLayout.PREFERRED\_SIZE, 40, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createSequentialGroup()

.addGap(16, 16, 16)

.addComponent(jLabel2)))

.addGap(50, 50, 50)

.addComponent(ColorPanel, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(RSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(RtextHolder))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(GtextHolder)

.addComponent(GSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(BtextHolder)

.addComponent(BSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addGap(102, 102, 102)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

.addComponent(ColorPanelChanged, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(ColorPanelOriginal, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()

.addGap(44, 44, 44)

.addComponent(jLabel3)

.addGap(16, 16, 16)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(RChooseSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(RtextHolder1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(GChooseSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE)

.addComponent(RtextHolder2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(RtextHolder3)

.addComponent(BChooseSlider, javax.swing.GroupLayout.PREFERRED\_SIZE, javax.swing.GroupLayout.DEFAULT\_SIZE, javax.swing.GroupLayout.PREFERRED\_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(SwapButton)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE)

.addComponent(XLabel)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addComponent(YLabel)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT\_SIZE, Short.MAX\_VALUE))

);

pack();

}// </editor-fold>

/\*

Obsługa menu

\*/

private void OptOpenActionPerformed(java.awt.event.ActionEvent evt) {

try {

openPicture();

} catch (IOException ex) {

Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);

}

}

/\*

Uruchamia Filtr oraz wyświetla komunikat jeżeli nie było otwarto obrazku

\*/

private void OptGrayActionPerformed(java.awt.event.ActionEvent evt) {

if (originalPicture == null) {

JOptionPane.showMessageDialog(null, "Open the image first", "InfoBox: " + "Warning", JOptionPane.INFORMATION\_MESSAGE);

}

else PictureBoxLabel.setIcon(new ImageIcon(filters.GrayFilter(myPicture)));

}

//Zmienia Pressed na true po naciskaniu na obszar dla rysowania

private void ImageHolderPanelMousePressed(java.awt.event.MouseEvent evt) {

Pressed = true;

}

//Wyświetla koordynaty oraz zamienia kursor

private void ImageHolderPanelMouseMoved(java.awt.event.MouseEvent evt) {

XLabel.setText("X: "+String.valueOf(evt.getX()));

YLabel.setText("Y: "+String.valueOf(evt.getY()));

setCursor(Cursor.CROSSHAIR\_CURSOR);

}

//Zmienia Pressed na true po opuszczeniu przycisku oraz wybiera kolor jeżeli był włączony

private void ImageHolderPanelMouseReleased(java.awt.event.MouseEvent evt) {

Pressed = false;

if(!Mode){

fromColor = myPicture.getRGB(evt.getX(), evt.getY());

ColorPanelOriginal.setBackground(new Color(fromColor));

}

}

//Po ruchaniu wciśniętą myszą wstawia piksel i odrazu wyświetla zmieniony obrazek

private void ImageHolderPanelMouseDragged(java.awt.event.MouseEvent evt) {

if (Mode) {

if (Pressed) {

myPicture.setRGB(evt.getX(), evt.getY(), color);

PictureBoxLabel.setIcon(new ImageIcon(myPicture));

}

}

}

//Obsługa sliderów do zmiany koloru dla rysowania

private void RSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColor();

}

private void BSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColor();

}

private void GSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColor();

}

/\*

Uruchamia Filtr oraz wyświetla komunikat jeżeli nie było otwarto obrazku

\*/

private void OptSepiaActionPerformed(java.awt.event.ActionEvent evt) {

if (originalPicture == null) {

JOptionPane.showMessageDialog(null, "Open the image first", "InfoBox: " + "Warning", JOptionPane.INFORMATION\_MESSAGE);

} else PictureBoxLabel.setIcon(new ImageIcon(filters.SepiaFilter(myPicture)));

}

//Po wyjściu za obszar dla rysowania zmienia kursor na defoltowy

private void ImageHolderPanelMouseExited(java.awt.event.MouseEvent evt) {

setCursor(Cursor.DEFAULT\_CURSOR);

}

/\*

Uruchamia Filtr oraz wyświetla komunikat jeżeli nie było otwarto obrazku

\*/

private void OptNegativeActionPerformed(java.awt.event.ActionEvent evt) {

if (originalPicture == null) {

JOptionPane.showMessageDialog(null, "Open the image first", "InfoBox: " + "Warning", JOptionPane.INFORMATION\_MESSAGE);

} else PictureBoxLabel.setIcon(new ImageIcon(filters.NegativeFilter(myPicture)));

}

//Obsługa menu

private void OptSaveActionPerformed(java.awt.event.ActionEvent evt) {

try {

savePicture();

} catch (IOException ex) {

Logger.getLogger(MainFrame.class.getName()).log(Level.SEVERE, null, ex);

}

}

//Obsługa sliderów do wyboru koloru zamiany

private void RChooseSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColorChanged();

}

private void GChooseSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColorChanged();

}

private void BChooseSliderStateChanged(javax.swing.event.ChangeEvent evt) {

GetColorChanged();

}

//Zmiana trybu przy naciśnięciu na rysowanie

private void DrawButtonActionPerformed(java.awt.event.ActionEvent evt) {

Mode = true;

}

//Zmiana trybu przy naciśnięciu na wybór coloru

private void ChooseButtonActionPerformed(java.awt.event.ActionEvent evt) {

Mode = false;

}

//Zamiana koloru oraz wyświetla komunikat jeżeli nie było otwarto obrazku

private void SwapButtonActionPerformed(java.awt.event.ActionEvent evt) {

if (originalPicture == null) {

JOptionPane.showMessageDialog(null, "Open the image first", "InfoBox: " + "Warning", JOptionPane.INFORMATION\_MESSAGE);

} else PictureBoxLabel.setIcon(new ImageIcon(filters.ChangeFilter(myPicture,fromColor,toColor)));

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

//<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

/\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.

\* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html

\*/

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(MainFrame.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new MainFrame().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JSlider BChooseSlider;

private javax.swing.JSlider BSlider;

private javax.swing.JLabel BtextHolder;

private javax.swing.JButton ChooseButton;

private javax.swing.JPanel ColorPanel;

private javax.swing.JPanel ColorPanelChanged;

private javax.swing.JPanel ColorPanelOriginal;

private javax.swing.JButton DrawButton;

private javax.swing.JSlider GChooseSlider;

private javax.swing.JSlider GSlider;

private javax.swing.JLabel GtextHolder;

public javax.swing.JPanel ImageHolderPanel;

private javax.swing.JMenuItem OptGray;

private javax.swing.JMenuItem OptNegative;

private javax.swing.JMenuItem OptOpen;

private javax.swing.JMenuItem OptSave;

private javax.swing.JMenuItem OptSepia;

private javax.swing.JLabel PictureBoxLabel;

private javax.swing.JSlider RChooseSlider;

private javax.swing.JSlider RSlider;

private javax.swing.JLabel RtextHolder;

private javax.swing.JLabel RtextHolder1;

private javax.swing.JLabel RtextHolder2;

private javax.swing.JLabel RtextHolder3;

private javax.swing.JButton SwapButton;

private javax.swing.JLabel XLabel;

private javax.swing.JLabel YLabel;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JMenu jMenu1;

private javax.swing.JMenu jMenu2;

private javax.swing.JMenuBar jMenuBar1;

// End of variables declaration

}

**Kod Filters.java:**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package myproject;

import java.awt.image.BufferedImage;

/\*\*

\*

\* @author Persei

\* Klasa zawiera filtry

\*/

public class Filters {

//Konstruktor klasy

public Filters() {

}

/\*

Nakłada szaro-odcieniowy filtr.

Bierzę piksel po pikselu podanego obrazka i wylicza wartość każdego kanału oprócz Alpha.

Każdy kanał zajmuje 8 bitów Alpha z indeksu 24 do indeksu 31, Red z 15 do 23, Green z 8 do 15 i Blue z 0 do 7.

I teraz róbmy right shift 32 bity na 16 pozycij z operacjej ADD 0xff(255 w hex) dla Red, 32 bity na 8 pozycij ADD 0xff dla Green i róbmy tylko ADD 0xff dla Blue

Każdy kanał jest mnożony przez stałe wartości dla otrzymania szaro-odcieniowych pikseli

\*/

public BufferedImage GrayFilter(BufferedImage originalImage){

BufferedImage newImage = originalImage;

int Width = originalImage.getWidth();

int Height = originalImage.getHeight();

int pixel,r,g,b;

int gray;

for (int x = 0; x < Width; x++) {

for (int y = 0; y < Height; y++) {

pixel = originalImage.getRGB(x,y);

//a = (pixel>>24) & 0xff;

r = (pixel>>16) & 0xff;

g = (pixel>>8) & 0xff;

b = pixel & 0xff;

gray = (int)(r\*0.3+g\*0.59+b\*0.11);

//Zabiezpieczenie

if (gray<0) {

gray =0;

}

if (gray>255) {

gray = 255;

}

//Wstawianie pikseli

newImage.setRGB(x, y, MakePixel(255, gray, gray, gray));

}

}

return newImage;

}

/\*

Nakłada filtr Sepia

\*/

public BufferedImage SepiaFilter(BufferedImage originalImage){

BufferedImage newImage = originalImage;

int Width = originalImage.getWidth();

int Height = originalImage.getHeight();

int pixel,r,g,b;

int Sr,Sg,Sb;

for (int x = 0; x < Width; x++) {

for (int y = 0; y < Height; y++) {

pixel = originalImage.getRGB(x,y);

//A = (pixel>>24) & 0xff;

r = (pixel>>16) & 0xff;

g = (pixel>>8) & 0xff;

b = pixel & 0xff;

Sr = (int)(0.393\*r + 0.769\*g + 0.189\*b);

Sg = (int)(0.349\*r + 0.686\*g + 0.168\*b);

Sb = (int)(0.272\*r + 0.534\*g + 0.131\*b);

if (Sr>255) {

r = 255;

} else r = Sr;

if (Sg>255) {

g = 255;

} else g = Sg;

if (Sb>255) {

b = 255;

} else b = Sb;

newImage.setRGB(x, y, MakePixel(255, r, g, b));

}

}

return newImage;

}

/\*

Nakłada filtr negatywnych kolorów

\*/

public BufferedImage NegativeFilter(BufferedImage originalImage){

BufferedImage newImage = originalImage;

int Width = originalImage.getWidth();

int Height = originalImage.getHeight();

int pixel,r,g,b;

for (int x = 0; x < Width; x++) {

for (int y = 0; y < Height; y++) {

pixel = originalImage.getRGB(x,y);

//A = (pixel>>24) & 0xff;

r = (pixel>>16) & 0xff;

g = (pixel>>8) & 0xff;

b = pixel & 0xff;

newImage.setRGB(x, y, MakePixel(255, 255-r, 255-g, 255-b));

}

}

return newImage;

}

/\*

Nakłada Change Filtr

\*/

public BufferedImage ChangeFilter(BufferedImage originalImage, int fromColor, int toColor){

BufferedImage newImage = originalImage;

int Width = originalImage.getWidth();

int Height = originalImage.getHeight();

int pixel,r,g,b;

for (int x = 0; x < Width; x++) {

for (int y = 0; y < Height; y++) {

pixel = originalImage.getRGB(x,y);

if (pixel == fromColor) {

pixel = toColor;

r = (pixel>>16) & 0xff;

g = (pixel>>8) & 0xff;

b = pixel & 0xff;

newImage.setRGB(x, y, MakePixel(255, r, g, b));

}

}

}

return newImage;

}

/\*

Służy do otrzymania ARGB wartości.

Shift Alpha na 24 pozycji oraz OR z Red przeniesioną o 16 pozycji, Green o 8 i Blue

\*/

public int MakePixel(int A, int R, int G, int B){

return (A<<24) | (R<<16) | (G<<8) | B;

}

}