Willis Allstead

CS326

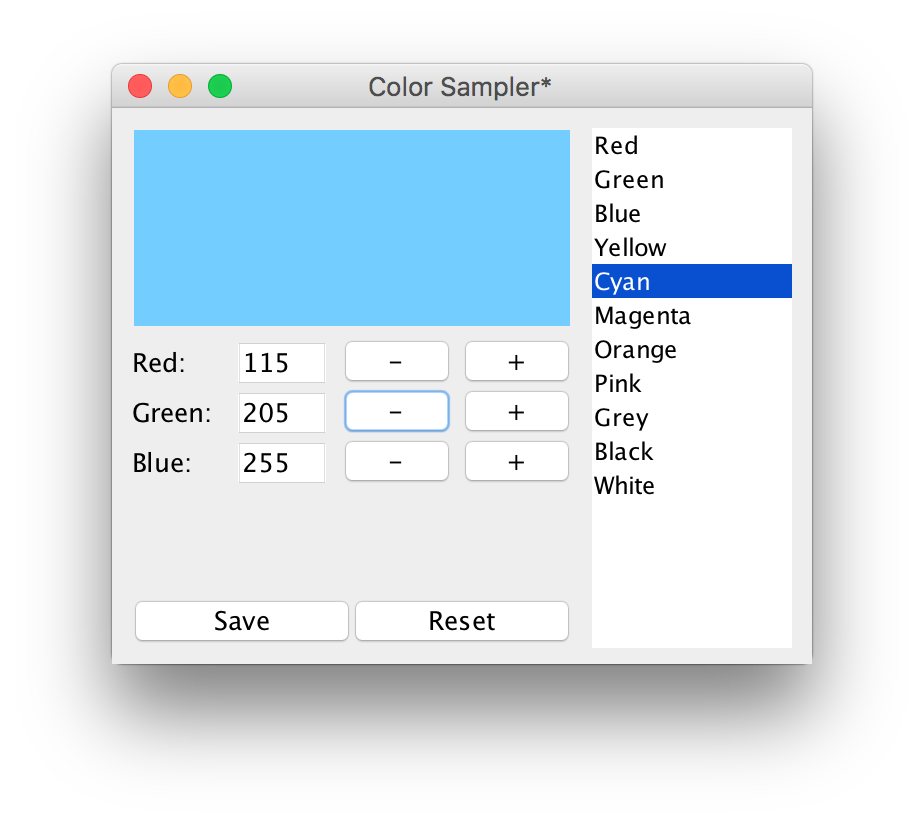
5/9/17

HW8

I made these assumptions:

1. I could use any layout method as long as the finished product looked like the screenshots.
2. I use an arbitrary “input.txt” filename for input, which I could have instead prompted the user for.
3. I decided that after a color was saved, if another color was clicked and then the modified one was clicked again it would show the saved value for that color.

Screenshot of program:



The code for the program:

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import javax.swing.event.\*;

import java.io.\*;

public class WindowApplication extends JFrame {

protected DrawColor colorDrawn;

protected MyColor[] colors = new MyColor[30]; // 30 is arbitrary

protected int colorCount;

protected JList listColors;

protected MyColor currentColor;

protected MyColor modifiedColor; // where I will store unsaved colors

protected JTextField tfRed;

protected JTextField tfGreen;

protected JTextField tfBlue;

protected JButton redUp;

protected JButton redDown;

protected JButton greenUp;

protected JButton greenDown;

protected JButton blueUp;

protected JButton blueDown;

protected JButton save;

protected JButton reset;

public static void main(String argv[]) {

new WindowApplication("Color Sampler");

}

public WindowApplication(String title) {

super(title);

try {

readFile("input.txt");

} catch (IOException e) {

System.out.println("Failed to read in colors: " + e);

}

setBounds(100, 100, 350, 300);

addWindowListener(new WindowDestroyer());

colorDrawn = new DrawColor();

MyColor firstColor = colors[0];

currentColor = colors[0];

colorDrawn.paintColor = new Color(firstColor.red, firstColor.green, firstColor.blue);

JLabel redLabel = new JLabel("Red:");

JLabel greenLabel = new JLabel("Green:");

JLabel blueLabel = new JLabel("Blue:");

tfRed = new JTextField(Integer.toString(firstColor.red));

tfRed.setEditable(false); // project spec says only editable by buttons

tfGreen = new JTextField(Integer.toString(firstColor.green));

tfGreen.setEditable(false);

tfBlue = new JTextField(Integer.toString(firstColor.blue));

tfBlue.setEditable(false);

redUp = new JButton("+");

redUp.addActionListener(new ActionHandler());

redDown = new JButton("-");

redDown.addActionListener(new ActionHandler());

greenUp = new JButton("+");

greenUp.addActionListener(new ActionHandler());

greenDown = new JButton("-");

greenDown.addActionListener(new ActionHandler());

blueUp = new JButton("+");

blueUp.addActionListener(new ActionHandler());

blueDown = new JButton("-");

blueDown.addActionListener(new ActionHandler());

save = new JButton("Save");

save.addActionListener(new ActionHandler());

reset = new JButton("Reset");

reset.addActionListener(new ActionHandler());

getContentPane().setLayout(null); // row,col

getContentPane().add(colorDrawn);

getContentPane().add(listColors);

getContentPane().add(redLabel);

getContentPane().add(greenLabel);

getContentPane().add(blueLabel);

getContentPane().add(tfRed);

getContentPane().add(tfGreen);

getContentPane().add(tfBlue);

getContentPane().add(redUp);

getContentPane().add(redDown);

getContentPane().add(greenUp);

getContentPane().add(greenDown);

getContentPane().add(blueUp);

getContentPane().add(blueDown);

getContentPane().add(save);

getContentPane().add(reset);

colorDrawn.setBounds(10, 10, 220, 100);

listColors.setBounds(10 + 220 + 10, 10, 100, 260);

redLabel.setBounds(10, 115, 50, 25);

greenLabel.setBounds(10, 140, 50, 25);

blueLabel.setBounds(10, 165, 50, 25);

tfRed.setBounds(60, 115, 50, 25);

tfGreen.setBounds(60, 140, 50, 25);

tfBlue.setBounds(60, 165, 50, 25);

redDown.setBounds(110, 115, 65, 25);

redUp.setBounds(170, 115, 65, 25);

greenDown.setBounds(110, 140, 65, 25);

greenUp.setBounds(170, 140, 65, 25);

blueDown.setBounds(110, 165, 65, 25);

blueUp.setBounds(170, 165, 65, 25);

save.setBounds(5, 245, 120, 25);

reset.setBounds(115, 245, 120, 25);

setVisible(true);

}

public void readFile(String filename) throws IOException {

FileInputStream stream = new FileInputStream(filename);

InputStreamReader reader = new InputStreamReader(stream);

StreamTokenizer tokens = new StreamTokenizer(reader);

String colorName;

int red;

int green;

int blue;

int colorCounter = 0; // count of colors given

while (tokens.nextToken() != tokens.TT\_EOF) {

colorName = (String) tokens.sval;

tokens.nextToken();

red = (int) tokens.nval;

tokens.nextToken();

green = (int) tokens.nval;

tokens.nextToken();

blue = (int) tokens.nval;

MyColor color = new MyColor(colorName, red, green, blue);

colors[colorCounter] = color; // will be at 0 index for first color

colorCounter++;

System.out.println(colorName + " " + red + " " + blue + " " + green);

}

colorCount = colorCounter;

listColors = new JList();

listColors.addListSelectionListener(new ListHandler());

listColors.setListData(colors);

stream.close();

}

private class ListHandler implements ListSelectionListener {

public void valueChanged(ListSelectionEvent e) {

if (e.getSource() == listColors) {

if (!e.getValueIsAdjusting()) {

if (modifiedColor != null) {

setTitle("Color Sampler");

modifiedColor = null;

}

int i = listColors.getSelectedIndex();

MyColor chosenColor = (MyColor) listColors.getSelectedValue();

colorDrawn.paintColor = new Color(chosenColor.red, chosenColor.green, chosenColor.blue);

colorDrawn.repaint();

tfRed.setText(Integer.toString(chosenColor.red));

tfGreen.setText(Integer.toString(chosenColor.green));

tfBlue.setText(Integer.toString(chosenColor.blue));

currentColor = chosenColor;

// System.out.println(currentColor);

}

}

}

}

private class ActionHandler implements ActionListener {

public boolean inRange(int val) {

if (val <= 255 && val >= 0) {

return true;

} else {

return false;

}

}

public void actionPerformed(ActionEvent e) {

/\* rename window \*/

if (modifiedColor == null) {

setTitle("Color Sampler\*");

modifiedColor = new MyColor(currentColor);

System.out.println("was null");

}

/\* please look away as I copy and paste something which could have been abstracted \*/

if (e.getSource() == redUp) {

int currentValue = Integer.parseInt(tfRed.getText());

int requestedValue = currentValue + 5;

if (inRange(requestedValue)) {

tfRed.setText(Integer.toString(requestedValue));

modifiedColor.red = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == redDown) {

int currentValue = Integer.parseInt(tfRed.getText());

int requestedValue = currentValue - 5;

if (inRange(requestedValue)) {

tfRed.setText(Integer.toString(requestedValue));

modifiedColor.red = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == greenUp) {

int currentValue = Integer.parseInt(tfGreen.getText());

int requestedValue = currentValue + 5;

if (inRange(requestedValue)) {

tfGreen.setText(Integer.toString(requestedValue));

modifiedColor.green = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == greenDown) {

int currentValue = Integer.parseInt(tfGreen.getText());

int requestedValue = currentValue - 5;

if (inRange(requestedValue)) {

tfGreen.setText(Integer.toString(requestedValue));

modifiedColor.green = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == blueUp) {

int currentValue = Integer.parseInt(tfBlue.getText());

int requestedValue = currentValue + 5;

if (inRange(requestedValue)) {

tfBlue.setText(Integer.toString(requestedValue));

modifiedColor.blue = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == blueDown) {

int currentValue = Integer.parseInt(tfBlue.getText());

int requestedValue = currentValue - 5;

if (inRange(requestedValue)) {

tfBlue.setText(Integer.toString(requestedValue));

modifiedColor.blue = requestedValue;

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

}

} else if (e.getSource() == save) {

System.out.println("\*save");

currentColor.red = modifiedColor.red;

currentColor.green = modifiedColor.green;

currentColor.blue = modifiedColor.blue;

} else if (e.getSource() == reset) {

System.out.println("\*reset");

modifiedColor = null;

modifiedColor = new MyColor(currentColor); // reset to current color

colorDrawn.paintColor = new Color(modifiedColor.red, modifiedColor.green, modifiedColor.blue);

colorDrawn.repaint();

tfRed.setText(Integer.toString(modifiedColor.red));

tfGreen.setText(Integer.toString(modifiedColor.green));

tfBlue.setText(Integer.toString(modifiedColor.blue));

}

}

}

public void writeFile(String filename) throws IOException {

FileOutputStream ostream = new FileOutputStream(filename);

PrintWriter writer = new PrintWriter(ostream);

int i = 0;

while (i < colorCount) {

MyColor thisColor = colors[i];

writer.println(thisColor.name + " " + thisColor.red + " " + thisColor.green + " " + thisColor.blue);

i++;

}

writer.flush();

ostream.close();

}

private class WindowDestroyer extends WindowAdapter {

public void windowClosing(WindowEvent e) {

/\* save the file \*/

try {

writeFile("output.txt");

} catch (IOException io) {

System.out.println("Failed to save: " + io);

}

System.exit(0);

}

}

}

class MyColor {

String name;

int red;

int green;

int blue;

public MyColor(String name, int red, int green, int blue) {

this.name = name;

this.red = red;

this.green = green;

this.blue = blue;

}

public MyColor(MyColor another) { // copy constructor

this.name = another.name;

this.red = another.red;

this.green = another.green;

this.blue = another.blue;

}

public String toString() {

return name;

}

}

class DrawColor extends JComponent {

public Color paintColor = Color.green; // default for debug purposes

public void paint(Graphics g) {

Dimension d = getSize();

g.setColor(paintColor);

g.fillRect(1, 1, d.width - 2, d.height - 2);

}

}