Allgemeines Programmierpraktikum **Sommersemester 2023**
Dr. Henrik Brosenne Georg-August-Universität Göttingen
Institut für Informatik

Übung 04

Das tagging muss vor dem Abgabetermin Mi., 31.05., 8 Uhr erfolgen.
Testate ab Mi., 31.05.

Vorbereitung

Prüfsumme und Testat

Bitte verwenden Sie ihr GitLab Projekt, das Sie für die vorherigen Übungen erstellt haben, auch für Übung 04 und alle weiteren Übungen. Folgen Sie der gleichen Abgabeprozedur (*tagging*).

Java

Informieren Sie sich im Moduk `java.desktop` der *Java Platform, Standard Edition 11 API Specification* `https://docs.oracle.com/en/java/javase/11/docs/api/java.desktop/module-summary.html` über die Swing-Komponenten `javax.swing.JButton` und `javax.swing.Box`, sowie die Layout-Manager `java.awt.BorderLayout` (default bei `JFrame`) und `java.awt.FlowLayout` (default bei `JPanel`). Informieren Sie sich weiterhin über `javax.imageio.ImageIO` und `java.awt.image.BufferedImage`.

Aufgabe 1

Grafik

Programmieren Sie eine Anwendung, die auf der Kommandozeile den Namen eine Grafikdatei übergeben bekommt und ein Fenster öffnet, um die Datei darzustellen.

Im oberen Teil des Fensters befinden sich drei Schaltflächen *Original*, *Grayscale*, *Pattern*. In der Mitte befindet sich ein Bereich, in dem die Grafik angezeigt wird, deren Darstellung über die Schaltflächen verändert werden kann. **Unten rechts** ist eine Schaltfläche um die Anwendung zu beenden.

Das Fenster könnte wie folgt aussehen mit der im GitLab unter *uebung→uebung04-data* hinterlegten Grafik `kandinsky.jpg`.

Die Anzeigefläche für die Grafik ist so groß, dass ein Pixel aus der Grafikdatei durch ein Rechteck der Größe 2x2 Pixel dargestellt werden kann.

Button *Original*/Standard

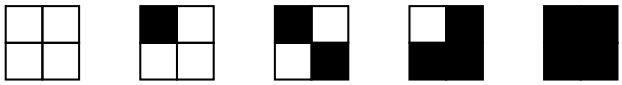
Die Farbe (RGB/rot-grün-blau Wert) jedes Pixel der Grafik wird gelesen und ein Rechteck in dieser Farbe gezeichnet.

Button *Grayscale*

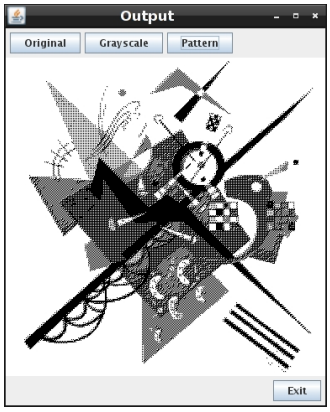
Die Farbe (RGB-/Rot-Grün-Blau-Wert) jedes Pixel der Grafik wird gelesen. Es wird jeweils der Rot-, Grün und Blau-Wert ermittelt und deren Mittelwert berechnet, dann wird ein Rechteck gezeichnet, wobei der Rot-, Grün- und Blau-Wert der Zeichenfarbe der berechnete Mittelwert ist.

Button *Pattern*

Der Mittelwert für jeden Pixel wird wie bei *Grayscale* berechnet. Abhängig davon welche RGB-Werte verwendet werden ergibt sich ein Wertebereich für den Mittelwert, der in 5 Abschnitt aufgeteilt wird (z.B. kann bei Werten 0 bis 255 die Aufteilung mit Division durch 52 erfolgen). Jedem Abschnitt wird anschließend das passenden der folgenden Muster (von schwarzen und weißen Pixeln) zugeordnet und gezeichnet.



Für das obige Beispiel liefert die Pattern-Darstellung folgendes Ergebnis.



Allgemein

- Bei auftretenden Fehlern wird eine aussagekräftige Fehlermeldung geliefert.
- Verwenden Sie Ant zum automatisierten Übersetzen des Quelltext. Nach dem Übersetzen befinden sich die `java`- und die `class`-Dateien in unterschiedlichen Verzeichnissen.
- Erzeugen Sie eine ausführbare Version der Anwendung in einem `jar`-Archiv mit Ant
- Kommentieren Sie Ihren Programmtext ausführlich. Erstellen Sie für die Klassen, Schnittstellen und alle Attribute Dokumentationskommentare für `javadoc`. Erzeugen Sie mit Ant eine API-Dokumentation. Spärliche und/oder schlechte Kommentierung führt zu Punktabzug.

Hinweis

Als Grundlage können Sie folgendes Programm verwenden, das im Git unter *uebung→uebung04-data* hinterlegt ist.

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4
5  class AppFrame extends JFrame {
6      public AppFrame(String title) {
7          super(title);
8          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9  }   }
10
11 class AppDrawPanel extends JPanel {
12     public Dimension getPreferredSize() {
13         return new Dimension(500, 200);
14     }
15
16     protected void paintComponent(Graphics g) {
17         super.paintComponent(g);
18         g.drawLine(10, 10, 490, 190);
19         g.drawString("Ein einfaches Panel", 50, 100);
20 }   }
21
22 class AppMouseAdapter extends MouseAdapter {
23     public void mouseClicked(MouseEvent e) {
24         if (e.getClickCount() > 1)
25             System.exit(0);
26 }   }
27
28 public class AppDrawEvent
29 {
30     public static void main( String[] args ) {
31         JFrame frame = new AppFrame("Allgemeines Programmierpraktikum");
32         JPanel panel = new JPanel();
33         frame.add(panel);
34
35         JPanel draw  = new AppDrawPanel();
36         JLabel label = new JLabel("Doppelklicken zum Beenden");
37         panel.add(draw);
38         panel.add(label);
39
40         AppMouseAdapter m = new AppMouseAdapter();
41         label.addMouseListener(m);
42
43         frame.pack();
44         frame.setVisible(true);
45 }   }
```