Übung 1: Color Picker: Swing Solution

```java
import java.awt.Color;
import java.util.LinkedList;
import java.util.List;

public class ColorModel {
    private Color color;
    private final List<ColorListener> listeners = new LinkedList<>();

    public void addColorListener(ColorListener l) {
        listeners.add(l);
    }

    public void removeColorListener(ColorListener l) {
        listeners.remove(l);
    }

    public Color getColor() {
        return color;
    }

    public void setColor(Color color) {
        if (!color.equals(this.color)) {
            this.color = color;

            for (ColorListener l : listeners) {
                l.colorValueChanged(color);
            }
        }
    }
}
```

```java
import java.awt.Color;

@FunctionalInterface
public interface ColorListener {
    void colorValueChanged (Color c);
}
```

```java
import java.awt.Color;
import javax.swing.JRadioButton;

class ColorRadioButton extends JRadioButton {

    ColorRadioButton(ColorModel model, String label, Color color) {
        super(label, false);

        addActionListener(e -> model.setColor(color));
        model.addColorListener(c -> setSelected(c.equals(color)));
    }

}
```

```java
import java.awt.Color;
import javax.swing.JCheckBoxMenuItem;

class ColorMenuItem extends JCheckBoxMenuItem {

    ColorMenuItem(ColorModel model, String label, Color color) {
        super(label);

        addActionListener(e -> model.setColor(color));
        model.addColorListener(c -> setSelected(c.equals(color)));
    }

}
```

```java
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;
import javax.swing.JComponent;

// This ist he background color field which is painted in the selected color
class ColorField extends JComponent {
    private static final int SIZE = 120;
    private Color color;

    ColorField(ColorModel model) {
        color = model.getColor();
        model.addColorListener(c -> {color = c; repaint();});
        setPreferredSize(new Dimension(SIZE, SIZE));
    }

    @Override
    public void paint(Graphics g) {
        Dimension d = getSize();
        g.setColor(color);
        g.fillRect(0, 0, d.width, d.height);
        g.setColor(Color.black);
        g.drawRect(0, 0, d.width - 1, d.height - 1);
    }

}
```

```java
import java.awt.Color;

public enum ColorChannel {
    RED(Color.RED) {
        @Override
        public int getValue(Color color) { return color.getRed(); }

        @Override
        public Color modifiedColor(Color color, int value) {
            return new Color(value, color.getGreen(), color.getBlue());
        }
    },
    GREEN(Color.GREEN) {
        @Override
        public int getValue(Color color) { return color.getGreen(); }

        @Override
        public Color modifiedColor(Color color, int value) {
            return new Color(color.getRed(), value, color.getBlue());
        }
    },
    BLUE(Color.BLUE) {
        @Override
        public int getValue(Color color) { return color.getBlue(); }

        @Override
        public Color modifiedColor(Color color, int value) {
            return new Color(color.getRed(), color.getGreen(), value);
        }
    };

    ColorChannel(Color color) { this.color = color; }

    private Color color;
    public Color getColor() { return color; }

    public abstract int getValue(Color color);
    public abstract Color modifiedColor(Color color, int value);
}
```

```java
import javax.swing.JScrollBar;

class ColorScrollBar extends JScrollBar {

    ColorScrollBar(ColorModel model, ColorChannel channel, int orientation, int val){
        super(orientation, val, 0, 0, 255);
        setBackground(channel.getColor());

        addAdjustmentListener(e -> model.setColor(
                    channel.modifiedColor(model.getColor(), getValue())));
        model.addColorListener(c -> setValue(channel.getValue(c)));
    }

}
```

```
import java.awt.Color;
import java.awt.event.FocusEvent;
import java.awt.event.FocusListener;
import javax.swing.JTextField;
import javax.swing.event.DocumentEvent;
import javax.swing.event.DocumentListener;

class ColorTextDecField extends JTextField
                        implements DocumentListener, FocusListener, ColorListener {
    private ColorModel model;
    private ColorChannel channel;

    ColorTextDecField(ColorModel model, ColorChannel channel) {
        super("", 5);
        this.model = model;
        this.channel = channel;
        getDocument().addDocumentListener(this);
        addFocusListener(this);
        model.addColorListener(this);
    }

    @Override
    public void insertUpdate(DocumentEvent e) { textChangeNotification(); }
    @Override
    public void removeUpdate(DocumentEvent e) { textChangeNotification(); }
    @Override
    public void changedUpdate(DocumentEvent e) { }

    private void textChangeNotification() {
        try {
            int value = Integer.parseInt(getText());
            if (value >= 0 && value < 256) {
                model.setColor(channel.modifiedColor(model.getColor(), value));
            }
        } catch (Exception x) {
            x.printStackTrace();
        }
    }

    @Override
    public void focusGained(FocusEvent e) { }

    @Override
    public void focusLost(FocusEvent e) {
        try {
            ColorTextDecField.super.setText("" + Integer.parseInt(getText()));
        } catch (Exception ex) {
            super.setText("" + channel.getValue(model.getColor()));
        }
    }

    @Override
    public void colorValueChanged(Color color) {
        setText("" + channel.getValue(color));
    }

}
```

```
import javax.swing.JTextField;

class ColorTextHexField extends JTextField {

    ColorTextHexField(ColorModel model, ColorChannel channel) {
        super("", 3);
        setEditable(false);
        model.addColorListener(c -> setText(Integer.toHexString(channel.getVa-
lue(c))));
    }

}
```

```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JButton;

class ColorButton extends JButton implements ColorListener, ActionListener {
    enum Type { BRIGHTER, DARKER }

    private ColorModel model;
    private Type type;

    ColorButton(ColorModel model, Type type, String label){
        super(label);

        this.type  = type;
        this.model = model;
        addActionListener(this);
        model.addColorListener(this);
    }

    @Override
    public void actionPerformed(ActionEvent e){
        Color c = model.getColor();
        switch(type) {
            case BRIGHTER: model.setColor(c.brighter()); break;
            case DARKER:   model.setColor(c.darker());   break;
        }
    }

    @Override
    public void colorValueChanged(Color c){
        switch(type) {
            case BRIGHTER: setEnabled(!c.equals(c.brighter())); break;
            case DARKER:   setEnabled(!c.equals(c.darker()));   break;
        }
    }
}
```

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.Scrollbar;

import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JPanel;
import javax.swing.SwingUtilities;

// Java Design, ColorPicker Application (Swing version)
// Autor: D. Gruntz
public class ColorApplication extends JFrame {

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new ColorApplication();
            frame.pack();
            frame.setVisible(true);
        });
    }

    private ColorModel model = new ColorModel();

    ColorApplication(){
        setTitle("Color Picker Swing");
        setBackground(Color.lightGray);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        Container c = getContentPane();
        c.setLayout(new BorderLayout());

        JPanel top = new JPanel(new GridLayout(1, 2, 5, 5));
        c.add(top, BorderLayout.NORTH);
        JPanel bottom = new JPanel(new FlowLayout());
        c.add(bottom, BorderLayout.CENTER);
        JPanel p;

        // Scrollbar panel
        p = new JPanel(new GridLayout(3,1,3,3));
        top.add(p);
        p.add(new ColorScrollBar(model, ColorChannel.RED, Scrollbar.HORIZONTAL, 0));
        p.add(new ColorScrollBar(model, ColorChannel.GREEN, Scrollbar.HORIZONTAL,0));
        p.add(new ColorScrollBar(model, ColorChannel.BLUE, Scrollbar.HORIZONTAL, 0));

        // Textfield panel
        p = new JPanel(new GridLayout(3, 2));
        top.add(p);
        p.add(new ColorTextDecField(model, ColorChannel.RED));
        p.add(new ColorTextHexField(model, ColorChannel.RED));
        p.add(new ColorTextDecField(model, ColorChannel.GREEN));
        p.add(new ColorTextHexField(model, ColorChannel.GREEN));
        p.add(new ColorTextDecField(model, ColorChannel.BLUE));
        p.add(new ColorTextHexField(model, ColorChannel.BLUE));
```

```
        // Color Field
        bottom.add(new ColorField(model));

        // CheckBox panel
        p = new JPanel(new GridLayout(0,1));
        bottom.add(p);

        p.add(new ColorRadioButton(model, "red",    Color.red));
        p.add(new ColorRadioButton(model, "blue",   Color.blue));
        p.add(new ColorRadioButton(model, "green",  Color.green));
        p.add(new ColorRadioButton(model, "yellow", Color.yellow));
        p.add(new ColorRadioButton(model, "cyan",   Color.cyan));
        p.add(new ColorRadioButton(model, "orange", Color.orange));


        // Button panel
        p = new JPanel(new GridLayout(2, 1, 5, 5));
        bottom.add(p);
        p.add(new ColorButton(model, ColorButton.Type.DARKER, "Darker"));
        p.add(new ColorButton(model, ColorButton.Type.BRIGHTER, "Brighter"));

        JMenuBar bar = new JMenuBar();
        setJMenuBar(bar);

        JMenu file = new JMenu("File");
        bar.add(file);

        JMenuItem exit = new JMenuItem("Exit");
        file.add(exit);
        exit.addActionListener(e -> System.exit(0));

        JMenu attr = new JMenu("Attributes");
        bar.add(attr);
        attr.add(new ColorMenuItem(model, "red",    Color.red));
        attr.add(new ColorMenuItem(model, "blue",   Color.blue));
        attr.add(new ColorMenuItem(model, "green",  Color.green));
        attr.add(new ColorMenuItem(model, "cyan",   Color.cyan));
        attr.add(new ColorMenuItem(model, "pink",   Color.pink));
        attr.add(new ColorMenuItem(model, "orange", Color.orange));
        attr.add(new ColorMenuItem(model, "magenta",Color.magenta));
        attr.add(new ColorMenuItem(model, "gray",   Color.gray));
        attr.add(new ColorMenuItem(model, "black",  Color.black));

        model.setColor(Color.black);  // update all controls
    }
}
```