import java.awt.
import java.awt.event.

/**
* A TrafficLight has three lenses: red, yellow and green.
* It can be set to signal Go, Caution, Stop or Walk.
*
* @version 1
*/

public class TrafficLight extends Panel
{
// Three Lenses and a Button

private Lens red          = new Lens( Color.red );
private Lens yellow       = new Lens( Color.yellow );
private Lens green        = new Lens( Color.green );
private Button nextButton = new Button("Next");

/**
* Construct a traffic light.
*/

public TrafficLight()
{
this.setLayout(new BorderLayout());

// create a Panel for the Lenses
Panel lensPanel = new Panel();
lensPanel.setLayout( new GridLayout( 3, 1 ) );
lensPanel.add( red );
lensPanel.add( yellow );
lensPanel.add( green );
this.add( BorderLayout.NORTH, lensPanel );

// configure the "Next" button
Sequencer sequencer = new Sequencer( this );
NextButtonListener payAttention =
new NextButtonListener( sequencer );
nextButton.addActionListener( payAttention );
this.add( BorderLayout.CENTER, nextButton);
}

// Methods that change the light

/**
* Set the light to stop (red).
*/

public void setStop()
{
red.turnOn();
yellow.turnOff();
green.turnOff();
}

/**
* Set the light to caution (yellow).
*/

public void setCaution()
{
red.turnOff();
yellow.turnOn();
green.turnOff();
}

/**
* Set the light to go (green).
*/

public void setGo()
{
red.turnOff();
yellow.turnOff();
green.turnOn();
}

/**
* Set the light to walk.
* (In Boston, red and yellow signal walk.)
*/

public void setWalk()
{
red.turnOn();
yellow.turnOn();
green.turnOff();
}

/**
* The traffic light simulation starts at main.
* @param args ignored.
*/

public static void main( String[] args )
{
Frame frame         = new Frame();
TrafficLight light  = new TrafficLight();
frame.add( light );
frame.addWindowListener( new ShutDownLight() );
frame.pack();
frame.show();
}
// A ShutDownLight instance handles close events generated
// by the underlying window system with its windowClosing
// method.

// This is an inner class, declared inside the
// TrafficLight class since it's used only here.

private static class ShutDownLight extends WindowAdapter
{
    // Close the window by shutting down the light.
    public void windowClosing (WindowEvent e)
    {
        System.exit(0);
    }
}
import java.awt.event.*;

/**
 * A NextButtonListener sends a "next" message to its
 * Sequencer each time a button to which it is listening
 * is pressed.
 *
 * @version 1
 */

public class NextButtonListener implements ActionListener {
   private Sequencer sequencer;

   /**
    * Construct a listener that "listens for" a user's
    * pressing the "Next" button.
    *
    * @param sequencer the Sequencer for the TrafficLight.
    */
   public NextButtonListener(Sequencer sequencer) {
      this.sequencer = sequencer;
   }

   /**
    * The action performed when a push of the button is detected:
    * send a next message to the Sequencer to advance it to
    * the action performed when a push of the button is detected.
    */
   public void actionPerformed(ActionEvent event) {
      this.sequencer.next();
   }
}
/**
 * A Sequencer controls a TrafficLight. It maintains fields
 * for the light itself and the current state of the light.
 *
 * Each time it receives a "next" message, it advances to the
 * next state and sends the light an appropriate message.
 *
 * @version 1
 */

public class Sequencer
{
    // the TrafficLight this Sequencer controls
    private TrafficLight light;

    // represent the states by ints
    private final static int GO       = 0;
    private final static int CAUTION  = 1;
    private final static int STOP     = 2;

    private int currentState;

    /**
     * Construct a sequencer to control a TrafficLight.
     *
     * @param light the TrafficLight we wish to control.
     */
    public Sequencer( TrafficLight light )
    {
        this.light = light;
        this.currentState = GO;
        this.light.setGo();
    }

    /**
     * How the light changes when a next Button is pressed
     * depends on the current state. The sequence is
     * GO -> CAUTION -> STOP -> GO.
     */
    public void next()
    {
        switch ( currentState ) {
            case GO:
                this.currentState = CAUTION;
                this.light.setCaution();
                break;

            case CAUTION:
                this.currentState = STOP;
                this.light.setStop();
                break;

            case STOP:
                this.currentState = GO;
                this.light.setGo();
                break;

            default: // This will never happen
                System.err.println("What color is the light?!\n");
        }
    }
}
// Copyright 2003 Bill Campbell and Ethan Bolker

import java.awt.*;

/**
 * A Lens has a certain color and can either be turned on
 * (the color) or turned off (black).
 *
 * @version 1
 */

public class Lens extends Canvas{

private Color onColor;                // color on
private Color offColor = Color.black; // color off
private Color currentColor;           // color the lens is now

private final static int SIZE = 100;  // how big is this Lens?
private final static int OFFSET = 20; // offset of Lens in Canvas

/**
 * Construct a Lens to display a given color.
 *
 * The lens is black when it's turned off.
 *
 * @param color the color of the lens when it is turned on.
 */

public Lens( Color color ){

this.setBackground( Color.black );
this.onColor = color;
this.setSize( SIZE , SIZE );
this.turnOff();
}

/**
 * How this Lens paints itself.
 *
 * @param g a Graphics object to manage brush and color information.
 */

public void paint( Graphics g ){

this.currentColor = onColor;
this.repaint();
}

/**
 * Have this Lens display its color.
 */

public void turnOn()
{

this.currentColor = onColor;
this.repaint();
}

/**
 * Darken this lens.
 */

public void turnOff()
{

this.currentColor = offColor;
this.repaint();
}