import java.awt.event.\*;

import java.awt.\*;

import java.applet.\*;

import java.util.Vector;

public class Drawtest extends Applet{

DrawPanel panel;

DrawControls controls;

public void init() {

setLayout(new BorderLayout());

panel = new DrawPanel();

controls = new DrawControls(panel);

add("Center", panel);

add("South",controls);

}

public void destroy() {

remove(panel);

remove(controls);

}

public static void main(String args[]) {

Frame f = new Frame("DrawTest");

Drawtest drawTest = new Drawtest();

drawTest.init();

drawTest.start();

f.add("Center", drawTest);

f.setSize(300, 300);

f.show();

}

public String getAppletInfo() {

return "A simple drawing program.";

}

}

class DrawPanel extends Panel implements MouseListener, MouseMotionListener {

public static final int LINES = 0;

public static final int POINTS = 1;

int mode = LINES;

Vector lines = new Vector();

Vector colors = new Vector();

int x1,y1;

int x2,y2;

public DrawPanel() {

setBackground(Color.white);

addMouseMotionListener(this);

addMouseListener(this);

}

public void setDrawMode(int mode) {

switch (mode) {

case LINES:

case POINTS:

this.mode = mode;

break;

default:

throw new IllegalArgumentException();

}

}

public void mouseDragged(MouseEvent e) {

e.consume();

switch (mode) {

case LINES:

x2 = e.getX();

y2 = e.getY();

break;

case POINTS:

default:

colors.addElement(getForeground());

lines.addElement(new Rectangle(x1, y1, e.getX(), e.getY()));

x1 = e.getX();

y1 = e.getY();

break;

}

repaint();

}

public void mouseMoved(MouseEvent e) {

}

public void mousePressed(MouseEvent e) {

e.consume();

switch (mode) {

case LINES:

x1 = e.getX();

y1 = e.getY();

x2 = -1;

break;

case POINTS:

default:

colors.addElement(getForeground());

lines.addElement(new Rectangle(e.getX(), e.getY(), -1, -1));

x1 = e.getX();

y1 = e.getY();

repaint();

break;

}

}

public void mouseReleased(MouseEvent e) {

e.consume();

switch (mode) {

case LINES:

colors.addElement(getForeground());

lines.addElement(new Rectangle(x1, y1, e.getX(), e.getY()));

x2 = -1;

break;

case POINTS:

default:

break;

}

repaint();

}

public void mouseEntered(MouseEvent e) {

}

public void mouseExited(MouseEvent e) {

}

public void mouseClicked(MouseEvent e) {

}

public void paint(Graphics g) {

int np = lines.size();

/\* draw the current lines \*/

g.setColor(getForeground());

for (int i=0; i < np; i++) {

Rectangle p = (Rectangle)lines.elementAt(i);

g.setColor((Color)colors.elementAt(i));

if (p.width != -1) {

g.drawLine(p.x, p.y, p.width, p.height);

} else {

g.drawLine(p.x, p.y, p.x, p.y);

}

}

if (mode == LINES) {

g.setColor(getForeground());

if (x2 != -1) {

g.drawLine(x1, y1, x2, y2);

}

}

}

}

class DrawControls extends Panel implements ItemListener {

DrawPanel target;

public DrawControls(DrawPanel target) {

this.target = target;

setLayout(new FlowLayout());

setBackground(Color.lightGray);

target.setForeground(Color.red);

CheckboxGroup group = new CheckboxGroup();

Checkbox b;

add(b = new Checkbox(null, group, false));

b.addItemListener(this);

b.setForeground(Color.red);

add(b = new Checkbox(null, group, false));

b.addItemListener(this);

b.setForeground(Color.green);

add(b = new Checkbox(null, group, false));

b.addItemListener(this);

b.setForeground(Color.blue);

add(b = new Checkbox(null, group, false));

b.addItemListener(this);

b.setForeground(Color.pink);

add(b = new Checkbox(null, group, false));

b.addItemListener(this);

b.setForeground(Color.orange);

add(b = new Checkbox(null, group, true));

b.addItemListener(this);

b.setForeground(Color.black);

target.setForeground(b.getForeground());

Choice shapes = new Choice();

shapes.addItemListener(this);

shapes.addItem("Lines");

shapes.addItem("Points");

shapes.setBackground(Color.lightGray);

add(shapes);

}

public void paint(Graphics g) {

Rectangle r = getBounds();

g.setColor(Color.lightGray);

g.draw3DRect(0, 0, r.width, r.height, false);

int n = getComponentCount();

for(int i=0; i<n; i++) {

Component comp = getComponent(i);

if (comp instanceof Checkbox) {

Point loc = comp.getLocation();

Dimension d = comp.getSize();

g.setColor(comp.getForeground());

g.drawRect(loc.x-1, loc.y-1, d.width+1, d.height+1);

}

}

}

public void itemStateChanged(ItemEvent e) {

if (e.getSource() instanceof Checkbox) {

target.setForeground(((Component)e.getSource()).getForeground());

} else if (e.getSource() instanceof Choice) {

String choice = (String) e.getItem();

if (choice.equals("Lines")) {

target.setDrawMode(DrawPanel.LINES);

} else if (choice.equals("Points")) {

target.setDrawMode(DrawPanel.POINTS);

}

}

}

}