Java 画板作业

2013599 田佳业

一、设计目标:

使用JFrame 完成一个画板程序

二、程序亮点：  
1. 在样例基础上更改了布局，删除了JMenuBar，增加了专门的setting面板，调节更为简洁细致。

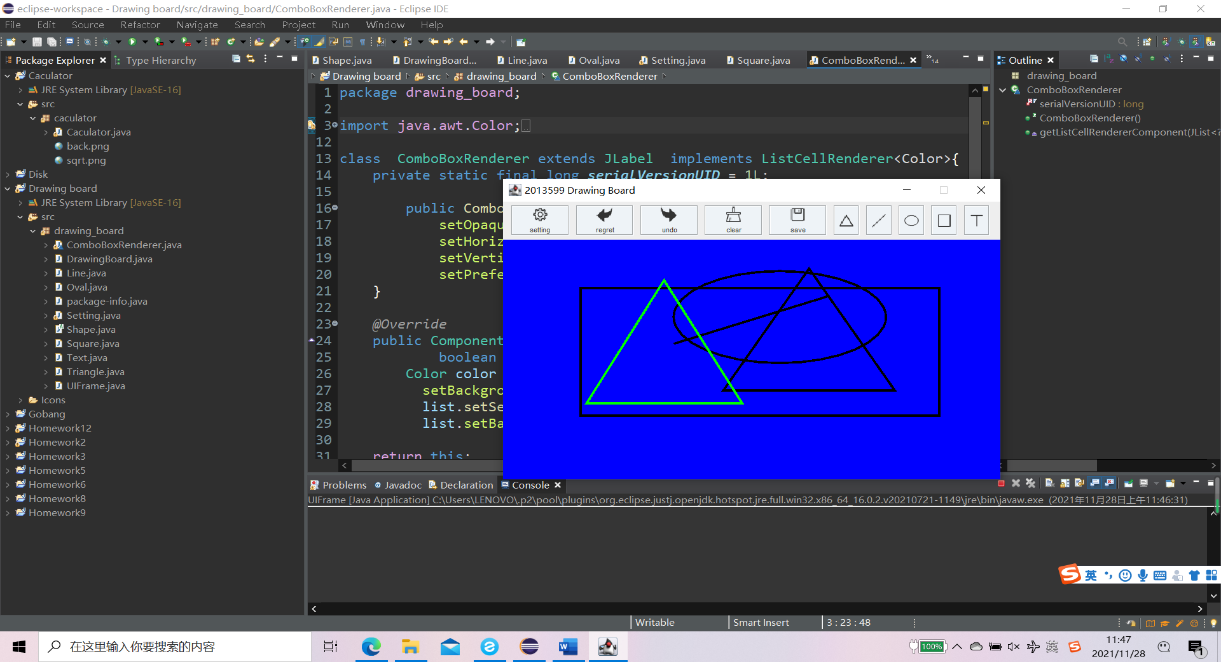
2. 额外实现了回退和清屏的功能。

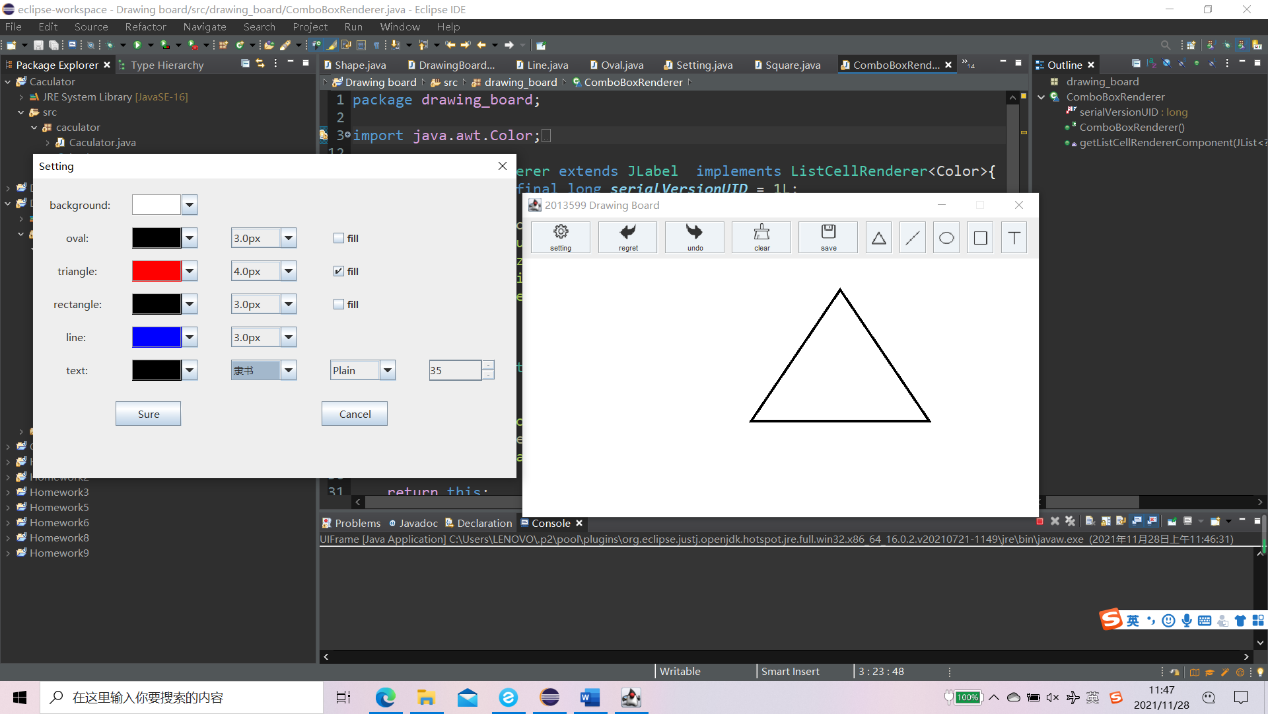
三、运行实例:

完整演示视频已上传哔哩哔哩视频网站，链接：

[Java Swing画板作业程序演示\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/BV1aP4y1V7nV/)

部分实例图片：





四、程序代码:

源码较长，若时间充足，建议直接导入工程文件查看。此处仅作部分展示。

UIFrame.Java

package drawing\_board;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Cursor;

import java.awt.Dimension;

import java.awt.Font;

import java.awt.GridLayout;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.border.Border;

import javax.swing.border.EtchedBorder;

public class UIFrame extends JFrame implements ActionListener{

    public static final Color BG\_BUTTON=new Color(238, 243, 247);

    private static final long serialVersionUID = 1L;

    public static final int DEFAULT\_CURSOR=0;

    public static final int CROSSHAIR\_CURSOR=1;

    public static final int MOVE\_CURSOR=13;

    public static final int TRIANGLE=0;

    public static final int LINE=1;

    public static final int OVAL=2;

    public static final int SQUARE=3;

    public static final int TEXT=4;

    public static final int DRAW\_LENGTH=5;

    private final String[] drawOption={"triangle","line","oval","square","text"};

    private final String[] drawOptionIconPath={"Icons/triangle.png","Icons/line.png","Icons/oval.png","Icons/square.png","Icons/text.png"};

    private final String[] commandOption={"setting","regret","undo","clear","save"};

    private final String[] commandOptionIconPath={"Icons/setting.png","Icons/regret.png","Icons/undo.png","Icons/clear.png","Icons/save.png"};

    private JButton drawButton[]=new JButton[drawOption.length];

    private JButton commandButton[]=new JButton[commandOption.length];

    private final int[] drawOptionIconSize={20,20,20,20,20};

    private final int[] commandOptionIconSize={20,20,20,20,20};

    private DrawingBoard board;

    public UIFrame() {

            super();

            initUI();

            this.setTitle("2013599 Drawing Board");

            this.setLocationRelativeTo(null);

            this.setSize(640, 400);

            this.setResizable(false);

        }

    public static void main(String args[]) {

        UIFrame mainFrame=new UIFrame();

        mainFrame.setVisible(true);

        mainFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

    }

    public void initUI() {

        JPanel drawPanel = new JPanel();

        drawPanel.setLayout(new GridLayout(1,5,8,8));

        drawPanel.setPreferredSize(new Dimension(200,40));

        for (int i = 0; i < drawOption.length; i++) {

            ImageIcon drawIcon=new ImageIcon(drawOptionIconPath[i]);

            drawIcon.setImage(drawIcon.getImage().getScaledInstance(drawOptionIconSize[i], drawOptionIconSize[i], Image.SCALE\_SMOOTH));

            drawButton[i] = new JButton(drawOption[i]);

            drawButton[i].setIcon(drawIcon);

            drawButton[i].setFont(new Font("Arial", Font.PLAIN, 0));

            drawButton[i].setVerticalTextPosition(JButton.CENTER);

            drawButton[i].setHorizontalTextPosition(JButton.CENTER);

            //not visible

            Border broder=new EtchedBorder();

               drawButton[i].setBorder(broder);

               drawButton[i].setBackground(BG\_BUTTON);

               drawButton[i].setFocusPainted(false);

            drawButton[i].setPreferredSize(new Dimension(40, 40));

            drawPanel.add(drawButton[i]);

            drawButton[i].addActionListener(this);

        }

        board= new DrawingBoard();

        Dimension dimensionMenu=new Dimension(800,450);

        board.setPreferredSize(dimensionMenu);

        JPanel  commandPanel= new JPanel();

        commandPanel.setLayout(new GridLayout(1,5,8,8));

        Dimension dimensionSetting=new Dimension(400,40);

        commandPanel.setPreferredSize(dimensionSetting);

        for (int i = 0; i < commandOption.length; i++) {

            ImageIcon commandIcon=new ImageIcon(commandOptionIconPath[i]);

            commandIcon.setImage(commandIcon.getImage().getScaledInstance(commandOptionIconSize[i], commandOptionIconSize[i], Image.SCALE\_SMOOTH));

            commandButton[i] = new JButton(commandOption[i], commandIcon);

            commandButton[i].setIcon(commandIcon);

            commandButton[i].setFont(new Font("Arial", Font.PLAIN, 9));

            commandButton[i].setVerticalTextPosition(JButton.EAST);

            commandButton[i].setHorizontalTextPosition(JButton.CENTER);

             Border broder=new EtchedBorder();

            commandButton[i].setBorder(broder);

            commandButton[i].setBackground(BG\_BUTTON);

            commandButton[i].setFocusPainted(false);

            commandButton[i].setPreferredSize(new Dimension(80, 40));

            commandPanel.add(commandButton[i]);

            commandButton[i].addActionListener(this);

        }

        JPanel  topPanel= new JPanel();

        topPanel.setPreferredSize(new Dimension(800,50));

        topPanel.add(commandPanel,BorderLayout.EAST);

        topPanel.add(drawPanel,BorderLayout.WEST);

        getContentPane().add(board, BorderLayout.CENTER);

        getContentPane().add(topPanel, BorderLayout.NORTH);

        this.setBackground(Color.WHITE);

 }

    private void drawHandler(int type) {

        DrawingBoard.cursor = CROSSHAIR\_CURSOR;

        board.setTool(type);

        board.setCursor(new Cursor(CROSSHAIR\_CURSOR));

    }

    @Override

    public void actionPerformed(ActionEvent e) {

        String curButton=e.getActionCommand();

        if (curButton.equals("triangle")) {

            drawHandler(TRIANGLE);

        } else if (curButton.equals("line")) {

            drawHandler(LINE);

        } else if (curButton.equals("oval")) {

            drawHandler(OVAL);

        }

        else if (curButton.equals("square")) {

            drawHandler(SQUARE);

        }

        else if (curButton.equals("text")) {

            String textName = JOptionPane.showInputDialog("Please enter text");

            board.setTextName(textName);

            drawHandler(TEXT);

        }

        else if (curButton.equals("setting")) {

            Setting setting = new Setting(board);

            setting.setLocationRelativeTo(null);

            setting.setVisible(true);

        }

        else if (curButton.equals("regret")) {

            if (board.shapes.size() < 1) {

                JOptionPane.showMessageDialog(null, "No element on the board!");

                return;

            }

            int index = board.shapes.indexOf(board.currentShape);

            Shape before = index > 0 ? board.shapes.get(index - 1) : null;

            if (!board.cancelShapes.contains(board.currentShape)) {

                board.shapes.remove(board.currentShape);

                board.cancelShapes.add(board.currentShape);

                board.currentShape = before;

            }

            board.repaint();

        }

        else if (curButton.equals("undo")) {

            if (board.cancelShapes.size() < 1) {

                JOptionPane.showMessageDialog(null, "No element to undo!");

                return;

            }

            Shape shape = board.cancelShapes.get(board.cancelShapes.size()-1);

            if (!board.shapes.contains(shape)) {

                board.cancelShapes.remove(shape);

                board.shapes.add(shape);

                board.currentShape = shape;

            }

            board.repaint();

        }

        else if (curButton.equals("clear")) {

            board.clearBoard();

            board.shapes.clear();

            board.cancelShapes.clear();

        }

        else if (curButton.equals("save")) {

            board.saveImage();

        }

    }

}

DrawingBoard.java

package drawing\_board;

import java.awt.Color;

import java.awt.Cursor;

import java.awt.Graphics;

import java.awt.Graphics2D;

import java.awt.event.MouseEvent;

import java.awt.event.MouseListener;

import java.awt.event.MouseMotionListener;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import java.util.ArrayList;

import java.util.HashMap;

import javax.imageio.ImageIO;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

public class DrawingBoard extends JPanel implements MouseListener, MouseMotionListener {

    public static final int TRIANGLE=0;

    public static final int LINE=1;

    public static final int OVAL=2;

    public static final int SQUARE=3;

    public static final int TEXT=4;

    private static final long serialVersionUID = 1L;

    private int tool = LINE;

    public static int cursor = UIFrame.DEFAULT\_CURSOR;

    public Shape currentShape;

    public ArrayList<Shape> shapes;

    public ArrayList<Shape> cancelShapes;

    public HashMap<Integer, Color> colorMap = new HashMap<Integer, Color>();

    public HashMap<Integer, Float> strokeMap = new HashMap<Integer, Float>();

    public HashMap<Integer, Boolean> fillMap = new HashMap<Integer, Boolean>();

    public float stroke = 10.0F;

    private String textName;

    public String textFace = "微软雅黑";

    public int textStyle = 0;

    public int textSize = 20;

    public DrawingBoard() {

        colorMap.put(TRIANGLE,Color.BLACK);

        colorMap.put(LINE,Color.BLACK);

        colorMap.put(OVAL,Color.BLACK);

        colorMap.put(SQUARE,Color.BLACK);

        colorMap.put(TEXT,Color.BLACK);

        strokeMap.put(TRIANGLE, 3.0f);

        strokeMap.put(LINE,3.0f);

        strokeMap.put(OVAL,3.0f);

        strokeMap.put(SQUARE,3.0f);

        fillMap.put(TRIANGLE, false);

        fillMap.put(LINE,false);

        fillMap.put(OVAL,false);

        fillMap.put(SQUARE,false);

        setBackground(Color.WHITE);

        shapes = new ArrayList<Shape>();

        cancelShapes = new ArrayList<Shape>();

        addMouseListener(this);

        addMouseMotionListener(this);

    }

    @Override

    protected void paintComponent(Graphics g) {

        super.paintComponent(g);

        Graphics2D graphics2d = (Graphics2D) g;

        if (shapes!=null  && shapes.size() > 0) {

            for (int i = 0; i < shapes.size(); i++) {

                ((Shape) shapes.get(i)).draw(graphics2d);

            }

        }

    }

    @Override

    public void mousePressed(MouseEvent e) {

        if (cursor == 1) {

            if (e.getButton() == 1) {

                switch (tool) {

                case OVAL:

                    currentShape = new Oval(colorMap.get(OVAL), strokeMap.get(OVAL), OVAL, e.getX(), e.getY(),fillMap.get(OVAL));

                    break;

                case TRIANGLE:

                    currentShape = new Triangle(colorMap.get(TRIANGLE), strokeMap.get(TRIANGLE), TRIANGLE, e.getX(), e.getY(),fillMap.get(TRIANGLE));

                    break;

                case LINE:

                    currentShape = new Line(colorMap.get(LINE), strokeMap.get(LINE), LINE,e.getX(), e.getY());

                    break;

                case SQUARE:

                    currentShape = new Square(colorMap.get(SQUARE), strokeMap.get(SQUARE), SQUARE,e.getX(), e.getY(),fillMap.get(SQUARE));

                    break;

                case TEXT:

                    currentShape = new Text(colorMap.get(TEXT), TEXT, e.getX(), e.getY(), getTextName(), textFace, textStyle, textSize);

                    break;

                default:

                    break;

                }

                shapes.add(currentShape);

                repaint();

            }

        }

    }

    @Override

    public void mouseDragged(MouseEvent e) {

            if (currentShape!= null) {

                currentShape.mouseDragged(e);

                repaint();

            }

    }

    @Override

    public void mouseReleased(MouseEvent e) {

        cursor=UIFrame.MOVE\_CURSOR;

        setCursor(new Cursor(UIFrame.MOVE\_CURSOR));

    }

    @Override

    public void mouseClicked(MouseEvent e) {

            if (null != currentShape) {

                currentShape.mouseDragged(e);

                repaint();

            }

    }

    public void saveImage() {

        try {

            BufferedImage bimg = new BufferedImage(getWidth(), getHeight(), BufferedImage.TYPE\_INT\_RGB);

            Graphics2D graphics2d = bimg.createGraphics();

            paint(graphics2d);

            File file = new File("D:/masterpiece.jpg");

            ImageIO.write(bimg, "jpg", file);

            JOptionPane.showMessageDialog(this, "the image is in D:/masterpiece.jpg");

        } catch (IOException e) {

            JOptionPane.showMessageDialog(this, "error");

            e.printStackTrace();

        }

    }

    public void clearBoard() {

        shapes.clear();

        repaint();

    }

    public void setTool(int i) {

            tool = i;

    }

    public void setStrokeSize(float size, int type) {

        if ( currentShape==null ) {

            throw new NullPointerException("currentShape is Null!");

        }

        if (size < 0 || size > 10) {

            throw new IllegalArgumentException("Invaild Weight Specified!");

        }

        switch (type) {

        case OVAL:

            strokeMap.replace(OVAL, size);

            break;

        case TRIANGLE:

            strokeMap.replace(TRIANGLE, size);

            break;

        case LINE:

            strokeMap.replace(LINE, size);

            break;

        case TEXT:

            break;

        default:

            break;

        }

        currentShape.stroke = size;

        repaint();

    }

    public void setCurrentX(int x) {

        currentShape.currentX = x;

        repaint();

    }

    public void setCurrentY(int y) {

        currentShape.currentY = y;

        repaint();

    }

    public void setCurrentD(int d) {

        currentShape.currentD = d;

        repaint();

    }

    @Override

    public void mouseMoved(MouseEvent e) {

        // TODO Auto-generated method stub

    }

    @Override

    public void mouseEntered(MouseEvent e) {

        // TODO Auto-generated method stub

    }

    @Override

    public void mouseExited(MouseEvent e) {

        // TODO Auto-generated method stub

    }

    public String getTextName() {

        return textName;

    }

    public void setTextName(String textName) {

        this.textName = textName;

    }

}

Shape抽象类

package drawing\_board;

import java.awt.Color;

import java.awt.Graphics2D;

import java.awt.event.MouseEvent;

import java.awt.event.MouseMotionListener;

import javax.swing.JLabel;

public abstract class Shape extends JLabel implements MouseMotionListener {

    private static final long serialVersionUID = 1L;

    protected float stroke;

    protected Color color;

    protected int type;

    protected int startX = 0;

    protected int startY = 0;

    protected int endX = 0;

    protected int endY = 0;

    protected int currentX = 0;

    protected int currentY = 0;

    protected int currentD = 0;

    protected int currentdX = 0;

    protected int currentdY = 0;

    protected Shape(Color color1, float stroke1, int type1, int x, int y) {

        type = type1;

        color = color1;

        stroke = stroke1;

        startX = endX = currentX = x;

        startY = endY = currentY = y;

        addMouseMotionListener(this);

    }

    public void mouseDragged(MouseEvent mouseEvent) {

        if (DrawingBoard.cursor == UIFrame.CROSSHAIR\_CURSOR) {

            endX = mouseEvent.getX();

            endY = mouseEvent.getY();

            currentdX = Math.abs(startX - endX);

            currentdY = Math.abs(startY - endY);

            if (startX > endX) {

                currentX = endX;

            }

            if (startY > endY) {

                currentY = endY;

            }

        } else {

            currentX = mouseEvent.getX()-currentdX/2;

            currentY = mouseEvent.getY()-currentdY/2;

        }

    }

    public void mouseMoved(MouseEvent mouseevent) {

    }

    public abstract void draw(Graphics2D graphics2d);

    public int getType() {

        return type;

    }

    public void setType(int type) {

        this.type = type;

    }

}

五、素材来源

图标素材均来源于iconfont。

同时，在上学期曾使用js+html+css独立完成画板微信小程序。由于没有repaint函数，所以这一次的作业相比微信小程序更好的实现了矢量化的功能。

当然，小程序中也有很多特色功能，二维码附下。

