java应用技术

Laboratory Projects 2

miniCAD实验

student report

王方懿康 3190101086

Date: 2021/11/12

Chapter 1: 任务

做一个简单的绘图工具，以CAD的方式操作，能放置直线、矩形、圆和文字，能选中图形，修改参数，如颜色等，能拖动图形和调整大小，可以保存和恢复。功能请参考视频演示。@

要求上传：

1. 源码；
2. 实验报告；
3. 可执行的jar文件。

演示视频下载链接: https://pan.baidu.com/s/1dFaZ2Ml 密码: d3xg

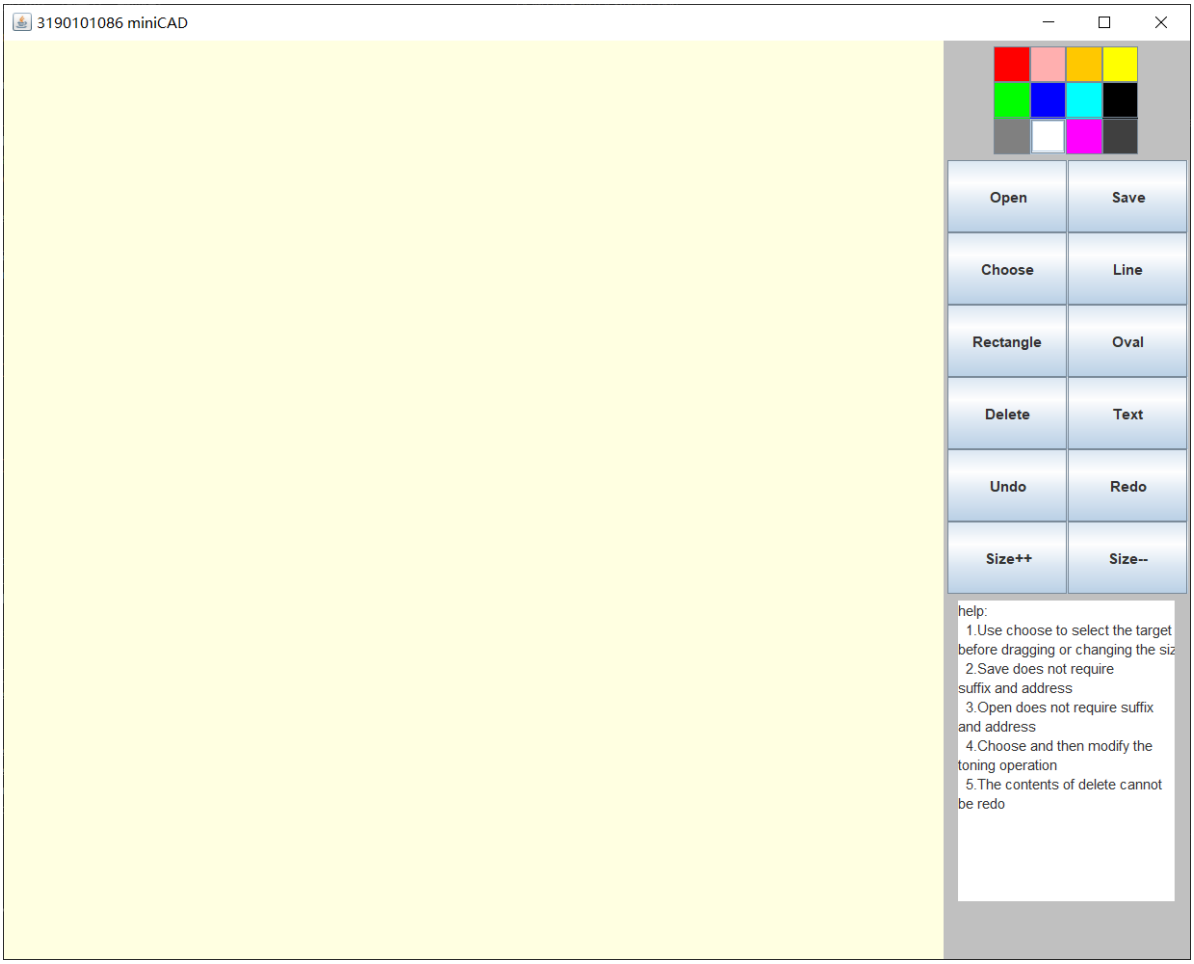
Chapter 2: 关键代码实现

2.1 屏幕绘制逻辑

```
        myMouse = new DrawListener();
        this.addMouseListener(myMouse);
        this.addMouseMotionListener(myMouse);
        this.addKeyListener(myMouse);

        JPanel board = new JPanel();
        JPanel toolBar = new JPanel();
        this.add(board, BorderLayout.CENTER);
        this.add(toolBar, BorderLayout.EAST);
        board.setBackground(new Color(255, 255, 225));

        toolBar.setPreferredSize(new Dimension(205, HEIGHT));
        toolBar.setBackground(Color.LIGHT_GRAY);

        JPanel ColorBar = new JPanel();
        JPanel ButtonBar = new JPanel();
        ColorBar.setLayout(new GridLayout(3, 4));
        ButtonBar.setLayout(new GridLayout(6, 2));

        String[] shapeName = { "Open", "Save", "Choose", "Line", "Rectangle",
"Oval", "Delete", "Text", "Undo", "Redo",
                "Size++", "Size--" };
        for (int i = 0; i < shapeName.length; i++) {
            JButton nowButton = new JButton(shapeName[i]);
            nowButton.setPreferredSize(new Dimension(100, 60));
            nowButton.addActionListener(myMouse);
            nowButton.setToolTipText(shapeName[i]);
            ButtonBar.add(nowButton);
        }

        Color[] colorArray = { Color.red, Color.pink, Color.orange,
Color.yellow, Color.green, Color.blue, Color.cyan,
                Color.black, Color.gray, Color.white, Color.magenta,
Color.DARK_GRAY };
        for (int i = 0; i < colorArray.length; i++) {
            JButton nowButton = new JButton();
            nowButton.setBackground(colorArray[i]);
            nowButton.setPreferredSize(new Dimension(30, 30));
            nowButton.addActionListener(myMouse);
            nowButton.setToolTipText(colorArray[i].toString());
            ColorBar.add(nowButton);
        }

        // toolBar.add(ColorBar, BorderLayout.SOUTH);
        // toolBar.add(ButtonBar, BorderLayout.CENTER);
        toolBar.add(ColorBar);
        toolBar.add(ButtonBar);
        JTextArea TextBar = new JTextArea(
                "help: \n  1.Use choose to select the target icon \nbefore
dragging or changing the size \n  2.Save does not require \nsuffix and address\n
3.Open does not require suffix \nand address\n  4.Choose and then modify the
\ntoning operation\n  5.The contents of delete cannot \nbe redo\n ");
        TextBar.setPreferredSize(new Dimension(180, 250));
        // TextBar.setSize(30, 40);
        toolBar.add(TextBar);
```

```
        this.setVisible(true);
```

绘制结果：



2.2 Event处理逻辑

2.2.1鼠标逻辑如下

```
public void mouseReleased(MouseEvent e) {
        Dragging = false;
    }

    public void mouseEntered(MouseEvent e) {

    }

    public void mouseExited(MouseEvent e) {
    }

    public void mouseDragged(MouseEvent e) {
        int edX = e.getX(), edY = e.getY();
        int x1 = Math.min(stX, edX), x2 = Math.max(stX, edX);
        int y1 = Math.min(stY, edY), y2 = Math.max(stY, edY);

        if (nowSelect.equals("Choose")) {
            if (nowShape != null) {
                nowShape.move(edX - curX, edY - curY);
                DoList.remove(nowShape);
                DoList.push(nowShape);
```

```
                myFrame.paint(myGraph);
            }
        } else if (nowSelect.equals("Line") || nowSelect.equals("Rectangle") ||
nowSelect.equals("Oval")) {
            if (Dragging) {
                if (!DoList.isEmpty())
                    DoList.pop();
            } else {
                Dragging = true;
            }
            if (nowSelect.equals("Line"))
                nowShape = DoList.push(new Line("Line", stX, stY, edX, edY,
nowColor));
            else if (nowSelect.equals("Rectangle"))
                nowShape = DoList.push(new Rectangle("Rectangle", x1, y1, x2 -
x1, y2 - y1, nowColor));
            else if (nowSelect.equals("Oval")) {
                nowShape = DoList.push(new Oval("Oval", x1, y1, x2 - x1, y2 -
y1, nowColor));
            }
            myFrame.paint(myGraph);
        }
        curX = edX;
        curY = edY;
    }

    @Override
    public void mouseMoved(MouseEvent e) {
    }
```

2.2.2Button逻辑如下

```
public void actionPerformed(ActionEvent e) {
        System.out.println(e.getActionCommand());
        if (e.getActionCommand().equals("")) {
            JButton nowButton = (JButton) e.getSource();
            nowColor = nowButton.getBackground();
            if (nowShape != null) {
                nowShape.setColor(nowColor);
                myFrame.paint(myGraph);
            }
        } else {
            nowSelect = e.getActionCommand();
            if (nowSelect.equals("Choose")) {
                nowShape = null;
            } else if (nowSelect.equals("Delete")) {
                if (nowShape != null) {
                    // TodoList.add(nowShape);
                    DoList.remove(nowShape);
                    myFrame.paint(myGraph);
                }
                nowSelect = "Choose";
            } else if (nowSelect.equals("Save")) {
                operation = Operation.Saving;
                buildFrame("Save as .mycad file", "Please enter your file
name");
                nowSelect = "Choose";
```

```
            } else if (nowSelect.equals("Open")) {
                operation = Operation.Opening;
                buildFrame("Opening .mycad file", "Please enter your file
name");
                nowSelect = "Choose";
            } else if (nowSelect.equals("OK")) { // OK buttons in saving or
opening
                fileFrame.setVisible(false);
                if (!content.getText().isEmpty()) {
                    System.out.println(content.getText());
                    switch (operation) {
                    case Texting:
                        nowShape = DoList.push(new Text("Text", curX, curY,
content.getText(), nowColor));
                        myFrame.paint(myGraph);
                        break;
                    case Saving:
                        ObjectOutputStream oos;
                        try {
                            // save to D, no need for user to input .mycad
                            oos = new ObjectOutputStream(new
FileOutputStream("D:\\" + content.getText() + ".mycad"));
                            oos.writeObject(DoList);
                        } catch (IOException ie) {
                            System.out.println(ie.getMessage());
                        }
                        break;
                    case Opening:
                        ObjectInputStream ios;
                        try {
                            ios = new ObjectInputStream(new
FileInputStream("D:\\" + content.getText() + ".mycad"));
                            Object obj = ios.readObject();
                            DoList = (Stack<Shape>) obj;
                            myFrame.paint(myGraph);
                        } catch (Exception ie) {
                            System.out.println("###" + ie.getMessage());
                        }
                        break;

                    }
                }
                nowSelect = "Choose";
                operation = Operation.OTHER;
            } else if (nowSelect.equals("Undo") && !DoList.isEmpty()) {
                TodoList.push(DoList.pop());
                myFrame.paint(myGraph);
                nowSelect = "Choose";
            } else if (nowSelect.equals("Redo") && !TodoList.isEmpty()) {
                DoList.push(TodoList.pop());
                myFrame.paint(myGraph);
                nowSelect = "Choose";
            } else if (nowSelect.equals("Size++")) {
                if (nowShape != null)
                    nowShape.changeSize(0.5f);
                myFrame.paint(myGraph);
                nowSelect = "Choose";
            } else if (nowSelect.equals("Size--")) {
```

```
                if (nowShape != null)
                    nowShape.changeSize(-0.5f);
                myFrame.paint(myGraph);
                nowSelect = "Choose";
            }
        }
    }
```

2.3 Shape与Shape的继承类

shape抽象类

cx与cy代表图像左上角的顶点坐标

size是线宽

name是该图形类型

```
public abstract class Shape implements Serializable {

    protected int cx, cy; // stand for leftmost up point
    protected Color color;
    protected float size;
    protected String name; // "Line" "Rectangle" etc

    // cons func
    public Shape(String name, int cx, int cy, Color color) {
        this.name = name;
        this.cx = cx;
        this.cy = cy;
        this.color = color;
        size = 3.0f;
    }

    public void setColor(Color color) {
        this.color = color;
    }

    public void move(int deltax, int deltay) {
        cx += deltax;
        cy += deltay;
    }

    public float changeSize(float ds) {
        if (this.size + ds > 10.0f)
            this.size = 10.0f;
        else if (this.size + ds < 1.0f)
            this.size = 1.0f;
        else
            this.size += ds;
        return this.size;
    }

    public abstract void draw(Graphics g);

    public abstract boolean in(int curX, int curY);
```

```
}
```

以Oval为例子，Shape的子类要实现绘制函数、范围判断函数和构造函数。

```
public class Oval extends Shape implements Serializable {
    // parameter width and height stand for 2a and 2b
    private int width, height;

    public Oval(String name, int cx, int cy, int width, int height, Color color)
{
        super(name, cx, cy, color);
        this.width = width;
        this.height = height;
    }

    public boolean in(int x, int y) {
        // actually judge if in the rectangle
        return x >= cx && x <= cx + width && y >= cy && y <= cy + height;
    }

    public void draw(Graphics g) {
        g.setColor(color);
        ((Graphics2D) g).setStroke(new BasicStroke(size));
        // Draws the outline of an oval. The result is a circle or ellipse that
fits
        // within the rectangle specified by the x, y, width, and height
arguments.
        g.drawOval(cx, cy, width, height);
    }
}
```

Ch3 测试结果

可通过jar或者源码运行miniCAD

以下是使用jar运行

测试发现结果与预期相符合。

制作了测试视频Demo。