JAD-miniCAD

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。

- 基本操作

右侧 Button 功能

- 1. 选中模式
- 2. 画线
- 3. 画矩形
- 4. 输入文字
- 5. 画圆形
- 6. 调色板（9种颜色供选择）

鼠标控制绘制参数

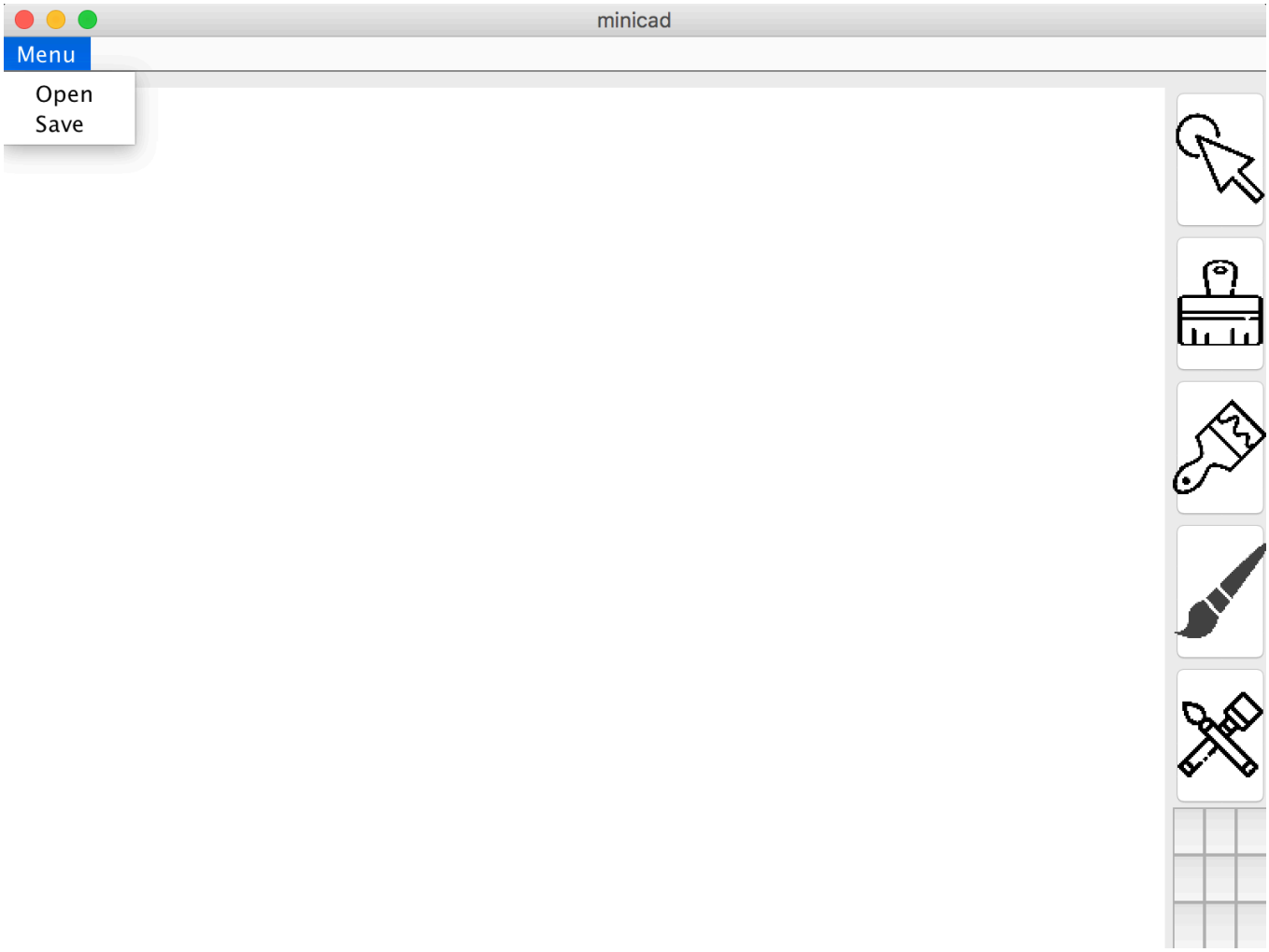
- 1.左键 （选中模式下）： 可以拖拽物体
- 2.右键 （选中模式下）： 可以右击删除物体
- 3.滑轮 （选中模式下）： 通过滑前滑后，改变物体的大小

键盘控制绘制参数(选中模式下)

- key code 61 ‘+’ : 加粗
- key code 45 ‘-‘ : 变细
- key code 44 ‘<‘ : 变大
- key code 46 ‘>’ : 变小

- 基本界面布局
    - 顶部菜单栏Menu，包含两个选项
        - Open： 实现加载图片包括cad格式，png等图片格式，加载后可以编辑
        - Save ： 实现保存图片包括cad格式，png等图片格式，保存后可以加载
    - 右侧工具栏
        - 第一个：选中模式
        - 第二个：画线
        - 第三个：画矩形
        - 第四个：输入文字
        - 第五个：画圆

        - 第六个：色彩选择框，包含九个小框分别代表 9 种颜色，在选中模式下，选中一个物体后，点击相应颜色框可以修改选中物体的颜色参数
    - 中间部分：画板



具体布局代码大致如下：

- MenuBar

```
jFrame.setLayout(new BorderLayout(5, 10));
/**
 * Menu Bar
 */
JMenuBar menuBar = new JMenuBar();
JMenu fileMenu = new JMenu("Menu");
menuBar.add(fileMenu);

JMenuItem openFile = new JMenuItem("Open");
fileMenu.add(openFile);
JMenuItem saveFile = new JMenuItem("Save");
fileMenu.add(saveFile);
...
    jFrame.add(menuBar, BorderLayout.NORTH);     //容器上方放置菜单栏
```

- ToolBar

```
JButton drawLine = new JButton();
ImageIcon icon_line = new ImageIcon("icon/line.png");
```

```
drawLine.setPreferredSize(new Dimension(60, 60));
Image lineImg = icon_line.getImage().getScaledInstance(60, 60,
icon_line.getImage().SCALE_DEFAULT);
icon_line = new ImageIcon(lineImg);
drawLine.setIcon(icon_line);//添加按钮icon
drawLine.addActionListener(
    ...
);
//其他button类似
```

- 颜色板 Button
    - 9个小button采用GridLayout方式布局在Panel中

```
JPanel colorPanel = new JPanel();
      colorPanel.setLayout(new GridLayout(0, 3)); //固定3列
      colorPanel.add(black);
      ...
      colorPanel.add(yellow);
```

- 画板
    - 以BorderLayout方式布局在中央
    - jFrame设置标题，尺寸，默认对齐参数
    - 设置为可见

```
panel = new PaintPanel();
jFrame.add(panel, BorderLayout.CENTER);
/**
* jFrame设置
*/
jFrame.add(paintTool, BorderLayout.EAST);
jFrame.setTitle("minicad");
jFrame.setSize(800, 600);
jFrame.setLocationRelativeTo(null);
jFrame.setDefaultCloseOperation(jFrame.EXIT_ON_CLOSE);
jFrame.setVisible(true);
```

基本几何物体

- Geometric
    - Line
    - Circle
    - Rectangle

```
abstract class Geometric {
    Font font;
    int LocX;
    int LocY;
    int width;
    int height;
    int size;
    Point pointFirst;
```

```
    Point pointSecond;
    Color defaultColor;
    abstract void drawPanel(Graphics g);
    abstract boolean clickTest(Point p);
}
```

所有需要绘制的基本问题需要的共同参数

- 点的位置
- 绘制颜色
- 尺寸大小
- 绘制方法
- 检测选中模式

事件监听

- 鼠标监听器
  - mouseRelease
  - mousePressed
  - mouseClicked
  - ……
  - KeyListener
    - key code 61 '+' : 加粗
    - key code 45 '-' : 变细
    - key code 44 '<' : 变大
    - key code 46 '>' : 变小

```
KeyListener keylistener = new KeyListener() {
        @Override
        public void keyTyped(KeyEvent e) {

        }

        @Override
        public void keyPressed(KeyEvent e) {
            //super.keyPressed(e);
            //System.out.println("key>>>"+e.getKeyCode());
            switch (e.getKeyCode())
            {
                case 61:{
                    geoObj.get(selection).strokeWid++;
              //      System.out.println("++>>>");
                }break;
                case 45:{
                    geoObj.get(selection).strokeWid--;
              //      System.out.println("-->>>");
                }
                case 46: {
                    if ((geoObj.get(selection) instanceof PaintPanel.Text))
                    {
```

```
                    geoObj.get(selection).size++;
                } else
                {
                    geoObj.get(selection).pointSecond.x++;
                    geoObj.get(selection).pointSecond.y++;
                    geoObj.get(selection).height++;
                    geoObj.get(selection).width++;
                }
            }break;
            case 44:{
                if ((geoObj.get(selection) instanceof PaintPanel.Text))
                {
                    geoObj.get(selection).size--;
                } else
                {
                    geoObj.get(selection).pointSecond.x--;
                    geoObj.get(selection).pointSecond.y--;
                    geoObj.get(selection).height--;
                    geoObj.get(selection).width--;
                }
            }break;
            default:break;
        }
        panel.repaint();
    }

    @Override
    public void keyReleased(KeyEvent e) {

    }
};
```

- 鼠标右键监听 （ 删除物体功能实现 ）

```
public void mouseClicked(MouseEvent e) {
    if(geoObj.size() > 1 && e.getButton() == MouseEvent.BUTTON3)
    {
        geoObj.get(selection).pointFirst.x = 0;
        geoObj.get(selection).pointFirst.y = 0;
        geoObj.get(selection).width = 0;
        geoObj.get(selection).height = 0;
        geoObj.get(selection).size = 0;
        geoObj.get(selection).pointSecond.x = 0;
        geoObj.get(selection).pointSecond.y = 0;
        repaint();

    }
}
```

- 鼠标拖拽监听

```
public void mouseDragged(MouseEvent e) {
    if (clickWhich == 0 && isSel) {
        int X = e.getX();
```

```
        int Y = e.getY();
        /**
         * 选中模式下的重绘制
         */
        repaint();
    }
}
```

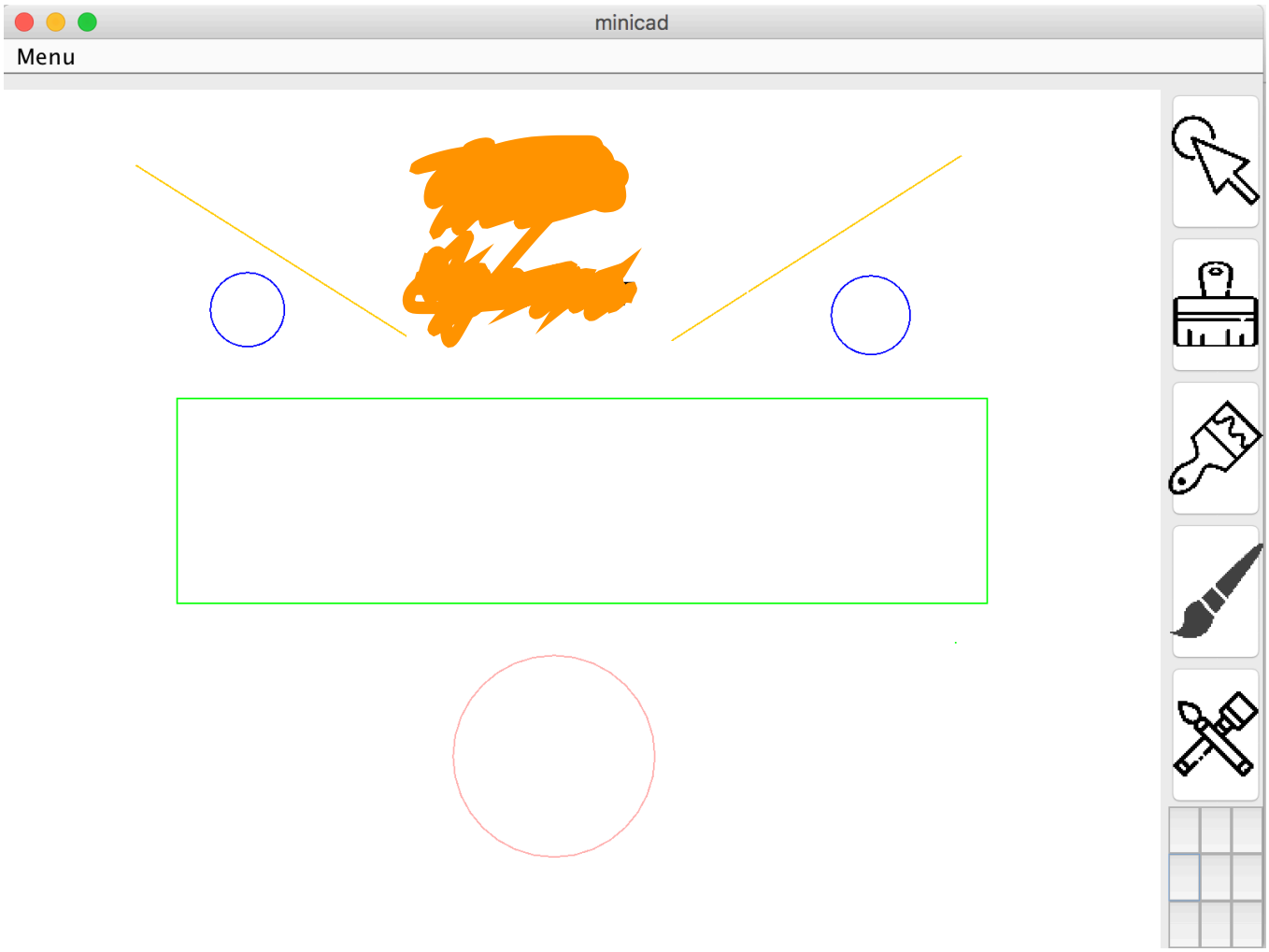
- 鼠标滑轮监听

```
this.addMouseWheelListener(new MouseWheelListener() {
    @Override
    public void mouseWheelMoved(MouseWheelEvent e) {
        if (clickWhich == 0 && isSel) {
            ... //处理变大变小，变粗变细绘制参数的改变
        }
        repaint();
    }
});
```

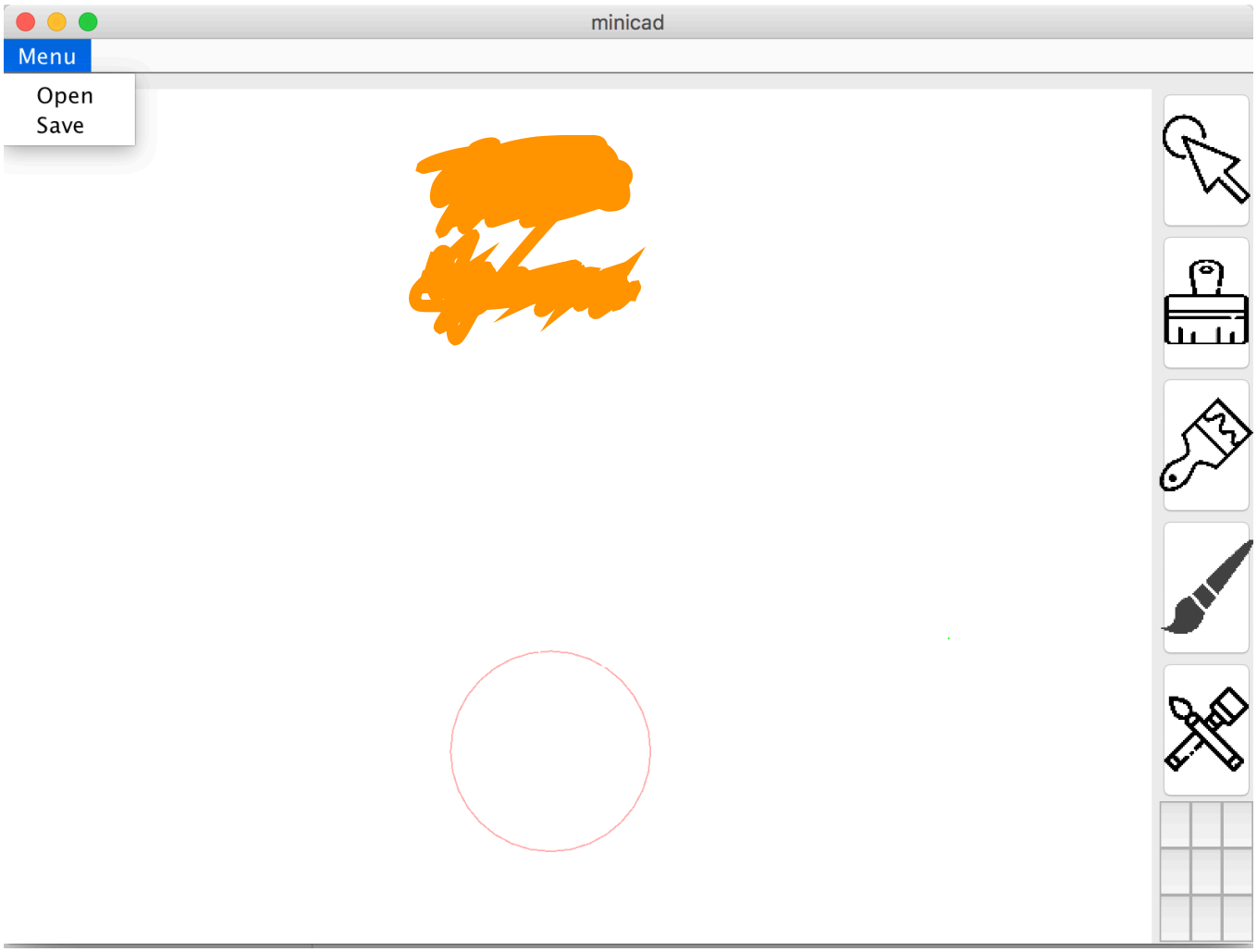
- 键盘监听
    - 需要获取焦点
    - panel.setFocusable(true);
    - panel.requestFocusInWindow();

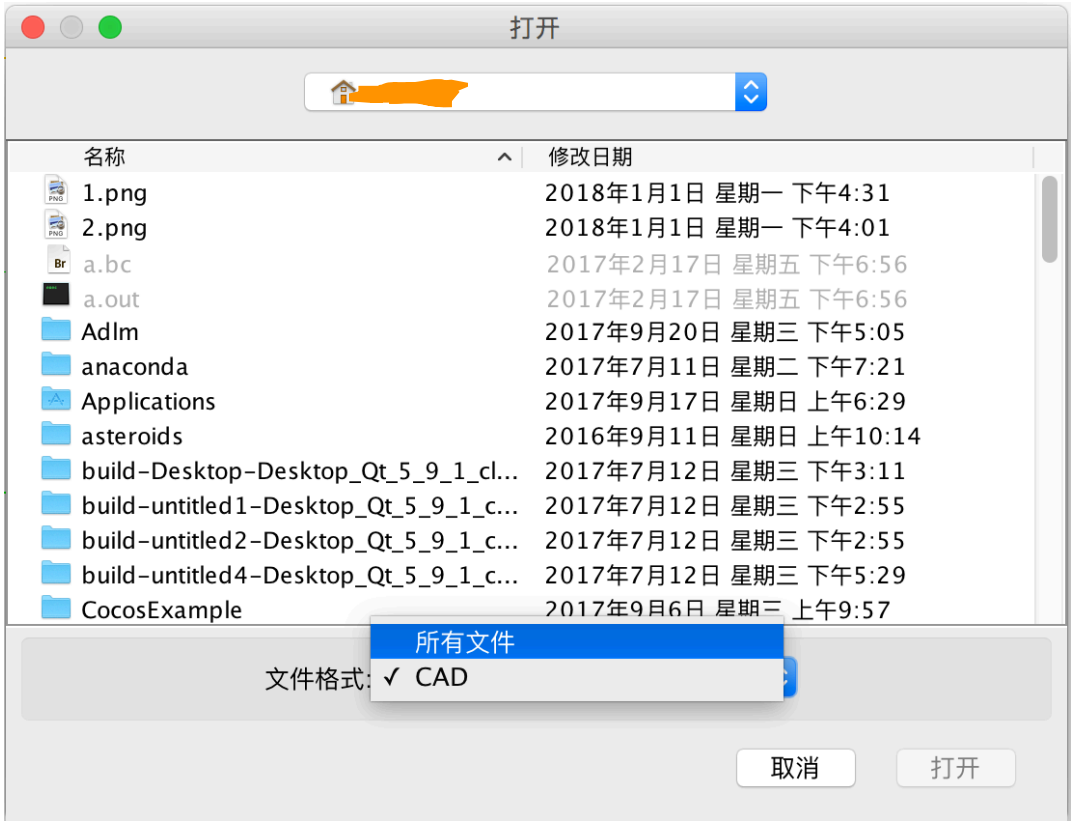
```
panel.addKeyListener(new KeyListener() {
    @Override
    public void keyTyped(KeyEvent e) {

    }

    @Override
    public void keyPressed(KeyEvent e) {
        switch (e.getKeyCode())
        {
        case KeyEvent.VK_DOWN:...;break;
        case KeyEvent.VK_UP:...;break;
        case KeyEvent.VK_GREATER:...;break;
        case KeyEvent.VK_GREATER:...;break;
        case KeyEvent.VK_2:...;break;
        default:keychar = e.getKeyChar();
        }
        panel.repaint();
    }

    @Override
    public void keyReleased(KeyEvent e) {

    }
});
panel.setFocusable(true);
panel.requestFocusInWindow();
```

- 功能实现

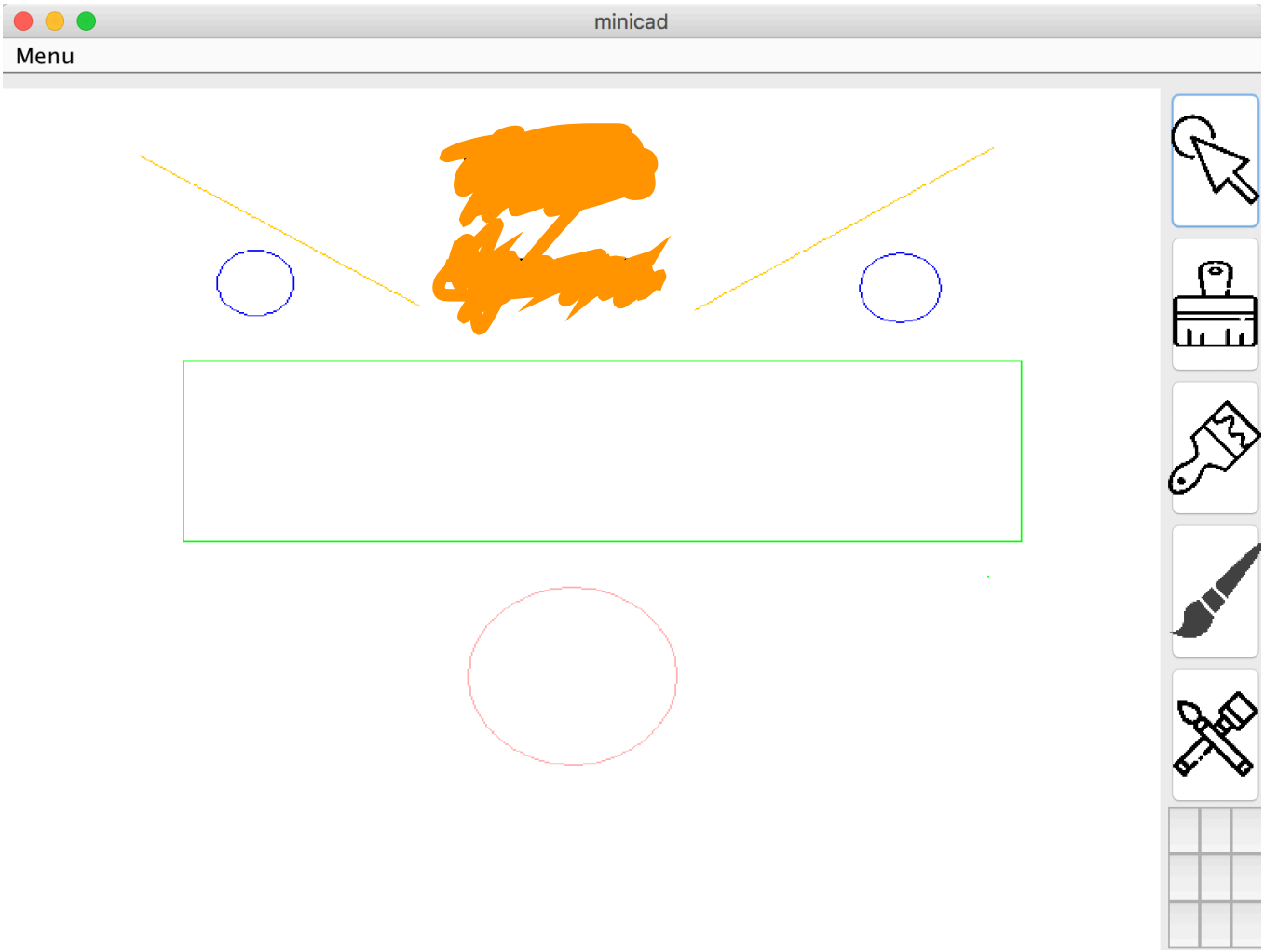


点击工具栏第一个**Button**进入选中模式，选中物体，点击鼠标右键实现删除
删除后**Menu->Open**可加载之前保存的图片

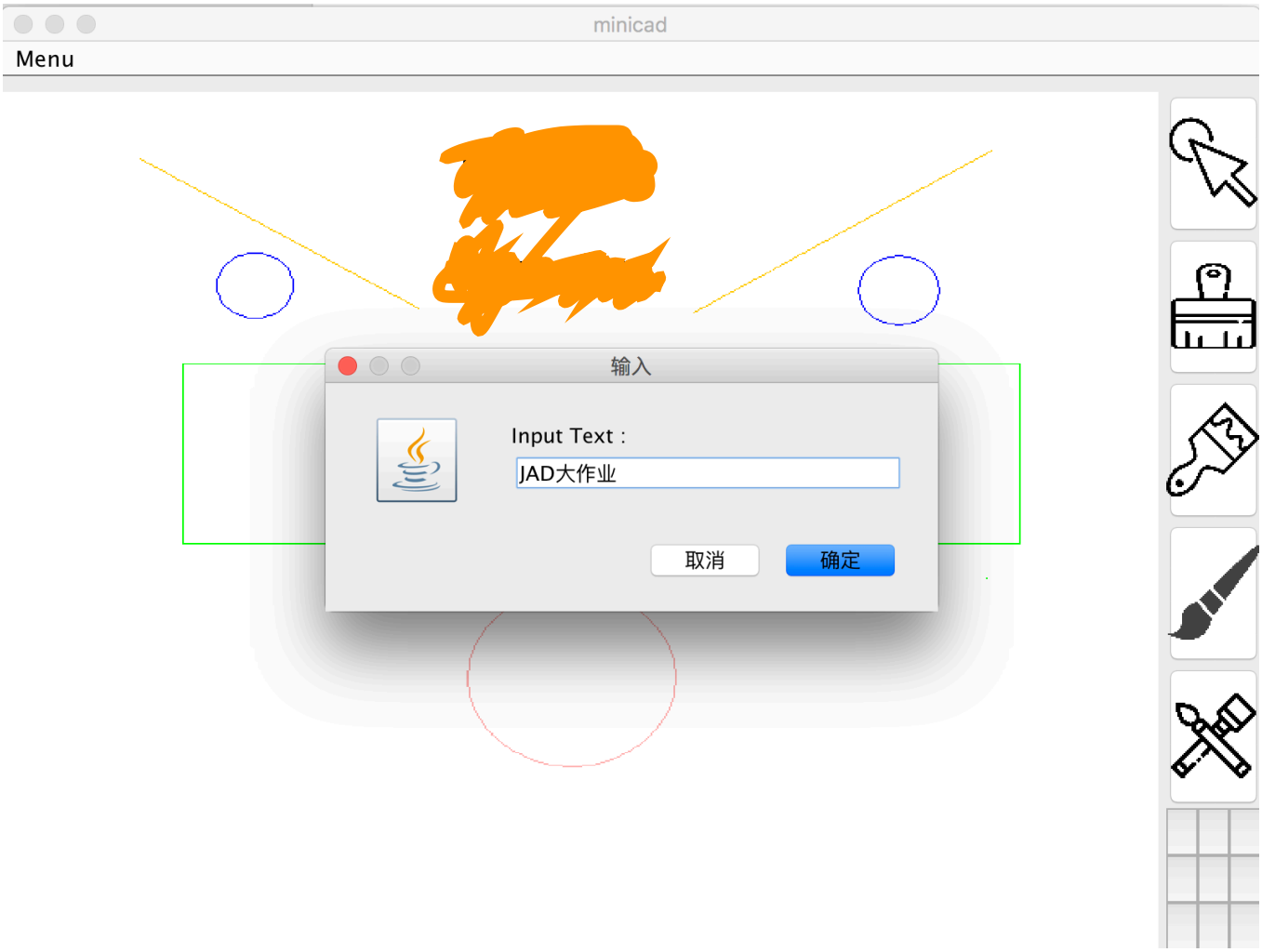

minicad
Menu
Open
Save


打开
名称
修改日期
1.png
2018年1月1日 星期一 下午4:31
2.png
2018年1月1日 星期一 下午4:01
a.bc
2017年2月17日 星期五 下午6:56
a.out
2017年2月17日 星期五 下午6:56
Adlm
2017年9月20日 星期三 下午5:05
anaconda
2017年7月11日 星期二 下午7:21
Applications
2017年9月17日 星期日 上午6:29
asteroids
2016年9月11日 星期日 上午10:14
build-Desktop-Desktop_Qt_5_9_1_cl...
2017年7月12日 星期三 下午3:11
build-untitled1-Desktop_Qt_5_9_1_c...
2017年7月12日 星期三 下午2:55
build-untitled2-Desktop_Qt_5_9_1_c...
2017年7月12日 星期三 下午2:55
build-untitled4-Desktop_Qt_5_9_1_c...
2017年7月12日 星期三 下午5:29
CocosExample
所有文件
CAD
取消
打开

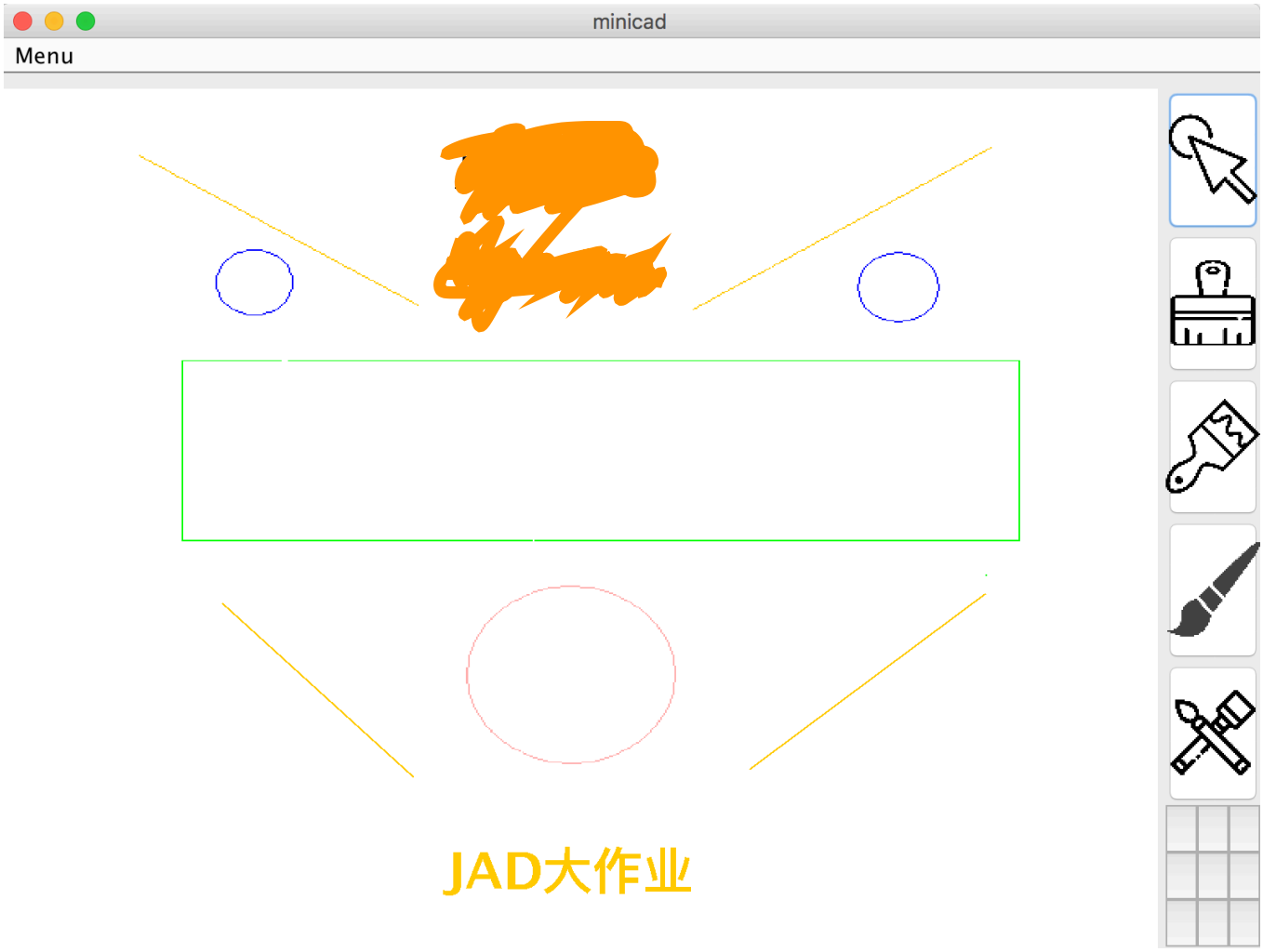
加载之前保存的图片



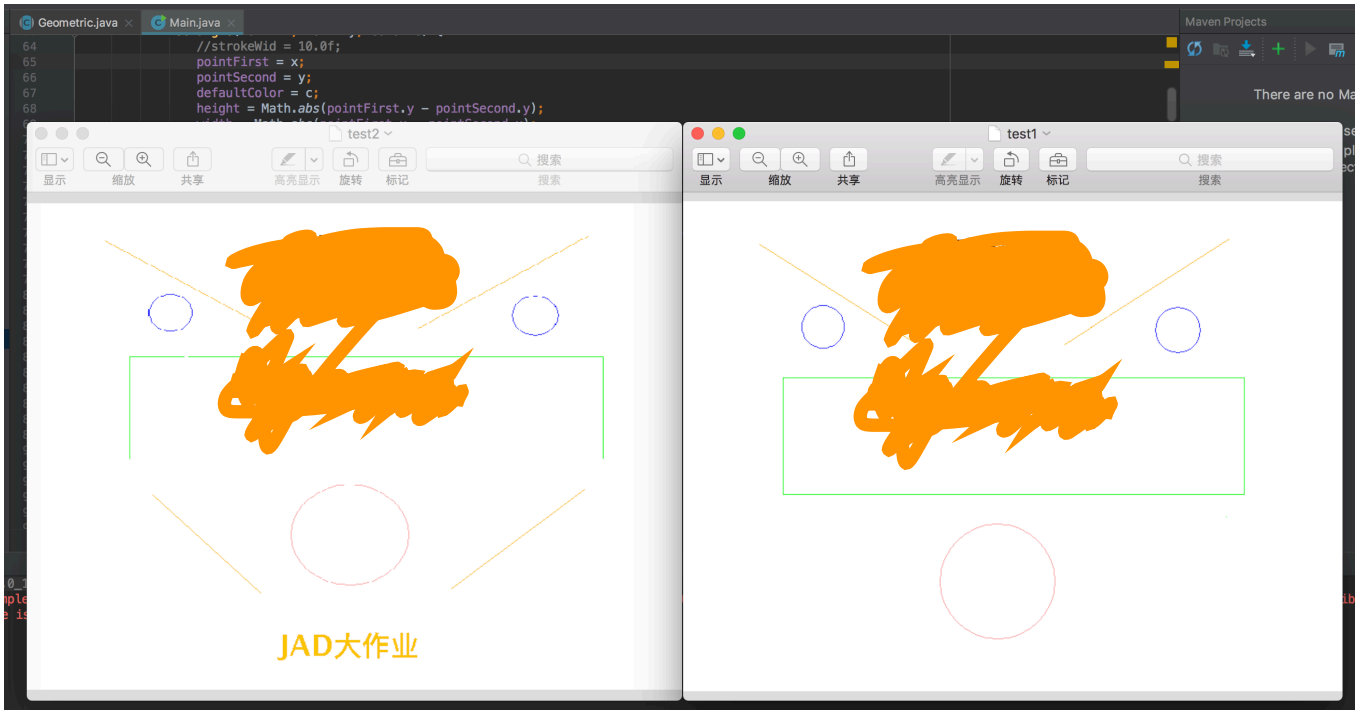
继续进行编辑



成功编辑，点击保存



以下为保存的**test1** 和 **test2** 文件



- 总结

此处大作业，对于JAVA的GUI有了更熟练的运用，对于各类监听事件，也有了更多的理解与运用。

整体来说作业不难，但需要合理安排数据结构，例如线，圆，矩形的存储结构，以及鼠标点击物体的检测，如何提高检测效率，使得绘制软件绘制流畅是需要加以思考的问题。

其此是同时多个监听事件的setfocus相关函数，也有了更深的理解。

整个Project进行下来后，对于JAVA的编程也更加熟悉，但对于JAVA的进阶编程知识还未涉及太多，后续需要加以学习。