JAD Project 2: mini CAD

1. 项目简介

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。

2. 设计思路

2.1 背景窗口

```
public class miniCAD extends JFrame
```

miniCAD 类为 CAD 的背景窗口。此类继承 JFrame，创建一个空白窗口，在窗口上显示其他 component，为整个 CAD 系统提供背景和框架。

```
DrawPanel p
JPanel buttons
```

miniCAD 类中包含 DrawPanel 对象和 JPanal 对象，前者实现绘图界面，后者绘制控制按钮栏。这两个对象将在之后详细介绍。

```
this.add(buttons, BorderLayout.NORTH);
this.addMouseListener(p);
this.addMouseMotionListener(p);
this.addKeyListener(p);
this.add(p);
```

在 miniCAD 构造函数中，DrawPanel 对象被当作鼠标监听器、键盘监听器、组件添加到 miniCAD 对象中，JPanel 被当作组件添加到 miniCAD 对象中。

2.2 控制按钮

控制按钮 buttons 为 JPanel 类型的对象，为 miniCAD 类内定义的对象。在 miniCAD 类中有 buttons_init 函数对其进行初始化。

在 buttons_init 函数中逐个定义每一个按钮的名称、按下后执行动作。绘制和改变颜色按钮只需改变 DrawPanel 对象内的数据，保存和恢复按钮需要使用 JFileChooser 类进行文件系统操作，利用 IO 流将
DrawPanel 中的对象保存到文件中或读取到当前 DrawPanel 中。按钮定义完成后将其添加到 buttons 对象中，随后添加到 miniCAD。

2.3 绘制模块

```
class DrawPanel extends JPanel implements MouseMotionListener, MouseListener,
KeyListener
```

绘制函数主要由 DrawPanel 类实现，该类为 JPanel 类的子类，并且实现 MouseMotionListener, MouseListener, KeyListener 以完成鼠标键盘控制。

```
    ArrayList<DrawUnit> patterns = null;
    int draw_choice;
    int pattern_choice;
    int set_color;
    int click_ctrl;
    int x1, y1, x2, y2;
    String input_text;
```

DrawPanel 类中有多个成员变量。patterns 将绘制的所有图案存储到 ArrayList 中，DrawUnit 为每个绘制的图案单元；draw_choice 表示当前绘制什么种类的图案；pattern_choice 表示当前点击选中的图案在 patterns 中的索引；set_color 表示当前画笔颜色；click_ctrl 表示当前鼠标状态，在 miniCAD 中所有图案的绘制过程都通过点击两个点完成，当 click_ctrl 为 1 或 2 时表示当前正在进行图案绘制，此时点击为选择绘制点，当 click_ctrl 为 0 时进行图案选取，以当前点击处是否在该图案的外接矩形范围内判断是否选中； x1,y1,x2,y2 为绘画的位置；input_text 表示绘制文字时输入的字符串。

```
public void paint(Graphics g)
public void mouseDragged(MouseEvent e)
public void mouseClicked(MouseEvent e)
public void keyPressed(KeyEvent e)
```

DrawPanel 中主要的功能函数如上。paint 函数实现绘制过程；mouseDragged 函数监控鼠标拖动事件，实现选中图像的拖动；mouseClicked 函数监控鼠标按键，当在绘制状态下进行记录绘制点坐标，记录完成进行绘制，当在选取状态下尝试选取图案；keyPressed 函数监控键盘事件，当选中一个图案时，按相应按键可实现改变线条粗细或图案大小。

2.4 绘制单元

```
abstract class DrawUnit implements Serializable{
    int x1, y1, x2, y2, brush, color;
    void Drag(int move_x, int move_y)
    abstract void ReShape(int sizeup);
    void BrushChange(int brushup)
    void ColorChange(int new_color)
    void Set(Graphics g)
    void DrawRange(Graphics g)
    abstract void repaint(Graphics g);
}
```

利用抽象类 DrawUnit 作为绘制的所有图案的父类。该类中实现一些图案共有的功能，具体的绘制过程则由子类具体图案实现。

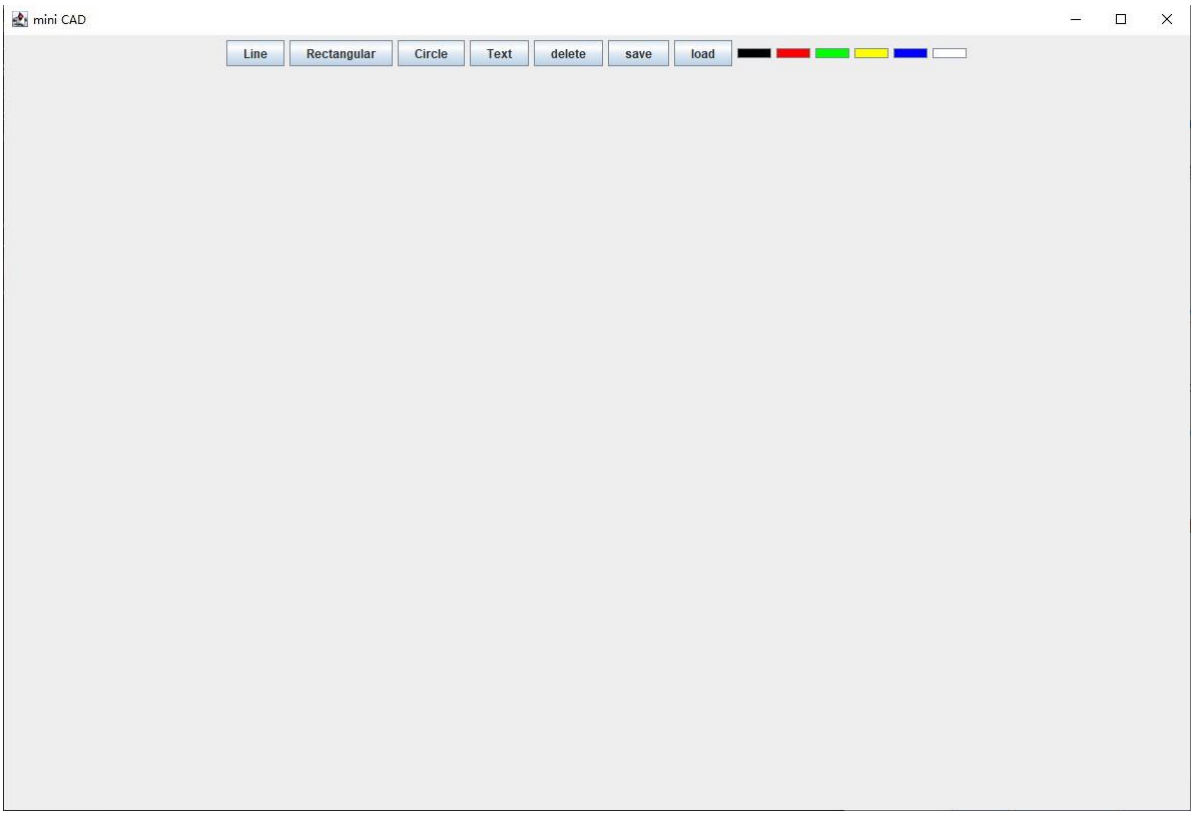
x1,y1,x2,y2 表示规定该图案的两点位置信息；brush 表示图案线条粗细；color 表示图案颜色。

Drag 函数实现拖动功能；BrushChange 函数改变线条；ColorChange 改变颜色；Set 利用类的成员变量对当前绘制参数进行设定；DrawRange 绘制当该图案被选中时的提示图案。抽象函数 ReShape 实现图案的大小改变，repaint 进行具体绘制。

由 DrawUnit 拓展的实体类如下，分别具体实现绘制过程

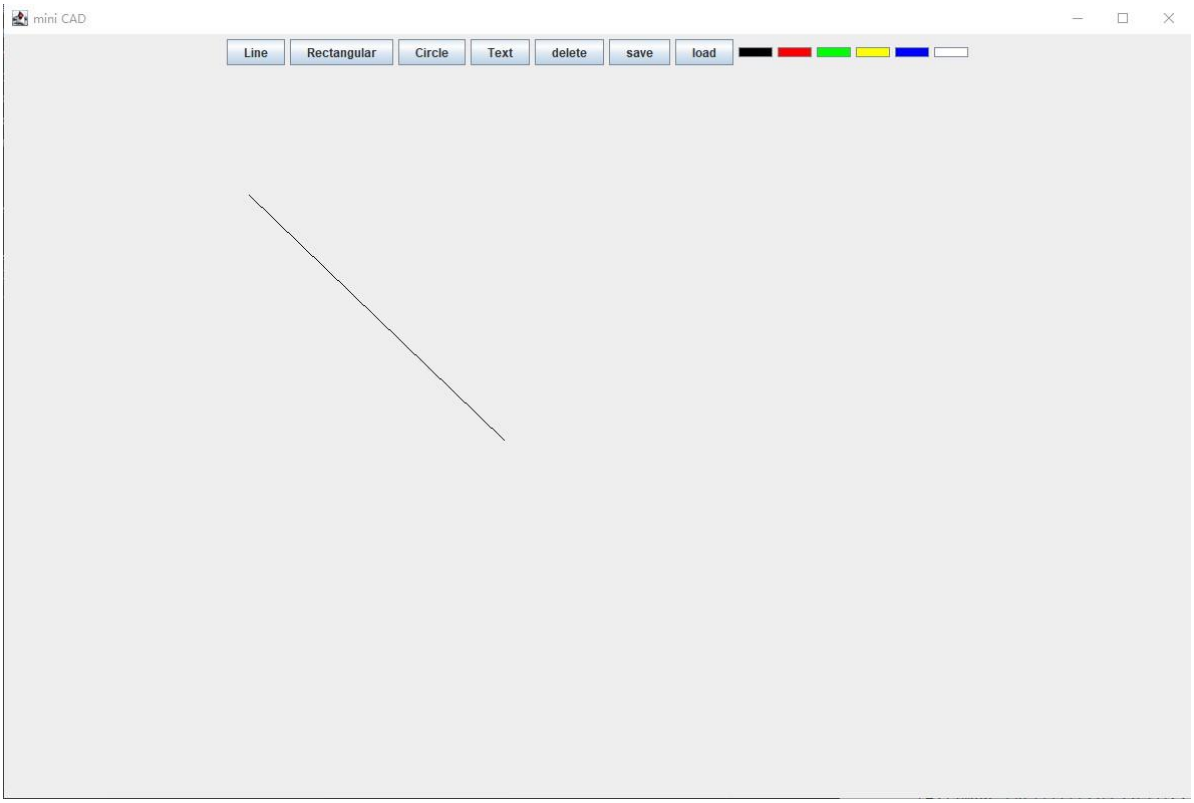
```
class DrawLine extends DrawUnit
class DrawRect extends DrawUnit
class DrawCircle extends DrawUnit
class DrawText extends DrawUnit
```

3. 演示与结果



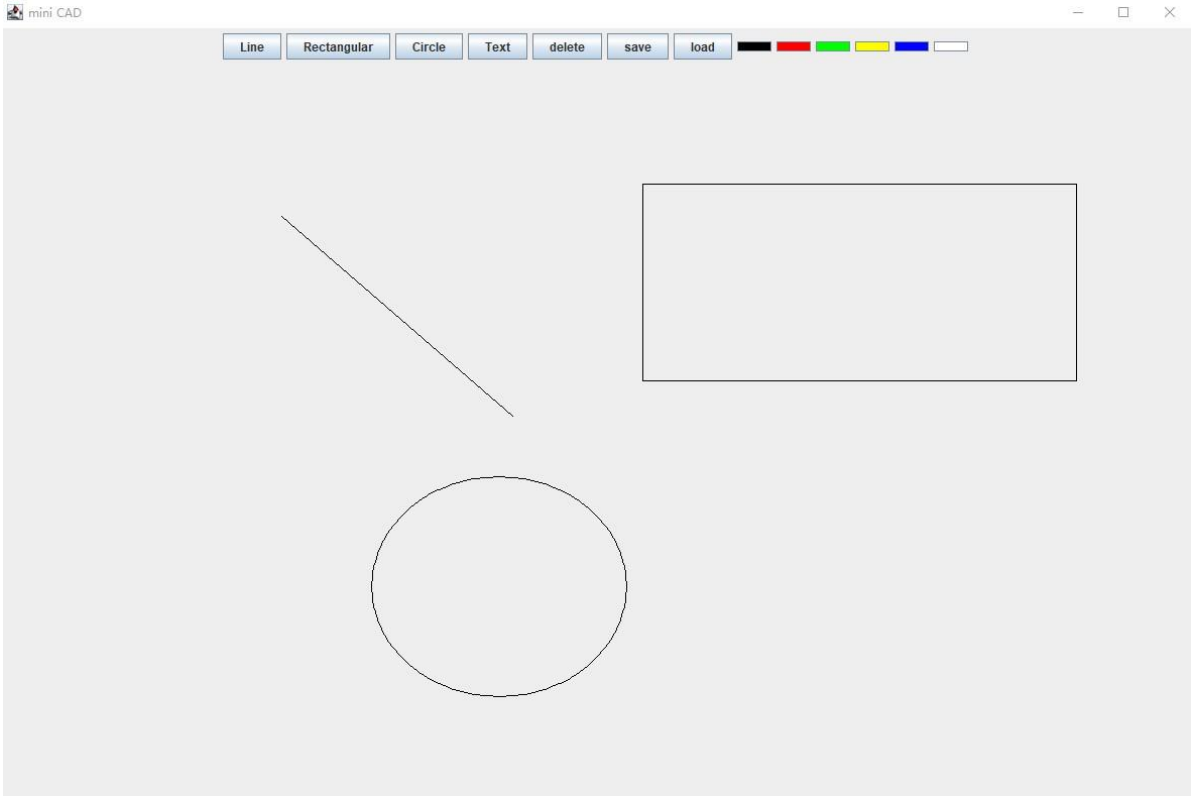
进入程序界面如下，上方有控制按钮栏，整个窗口为绘图界面。

控制按钮共 13 个，前 5 个按钮控制绘图：Line 画直线，Rectangular 画矩形，CIrcle 画圆，Text 画文字， delete 删除图案。之后的两个按钮实现保存和恢复功能，save 选择位置进行保存，load 选择文件恢复图像。最后六个按钮控制图案颜色，选择颜色与按钮颜色一致。

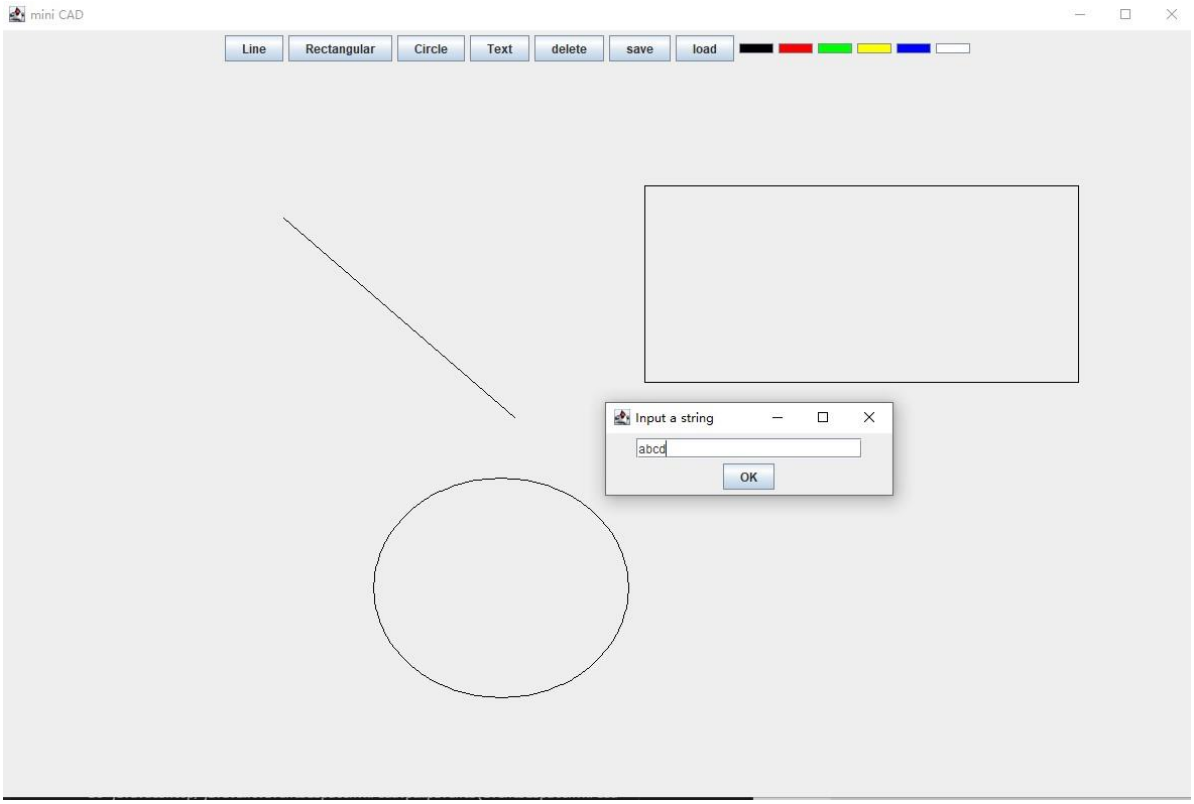


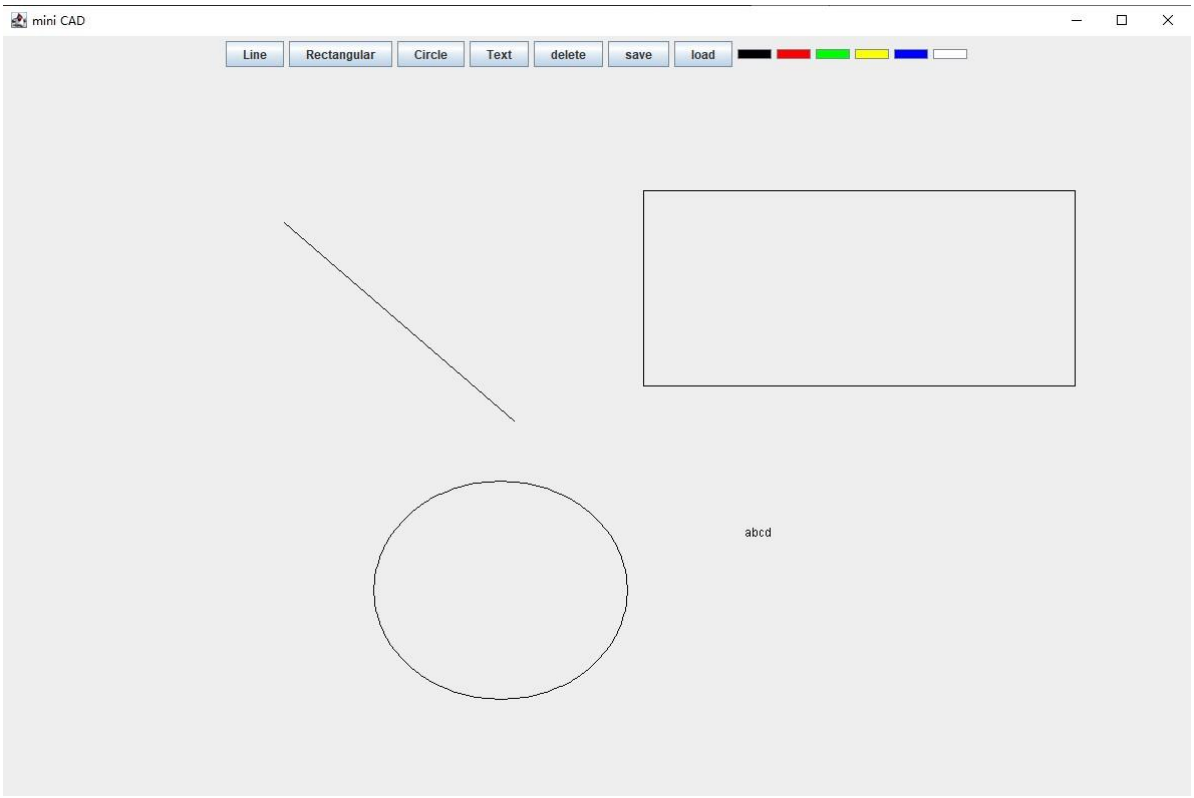
先点击 Line 按钮，随后在屏幕上点击两点，绘制直线。

同理可以绘制矩形和圆形。

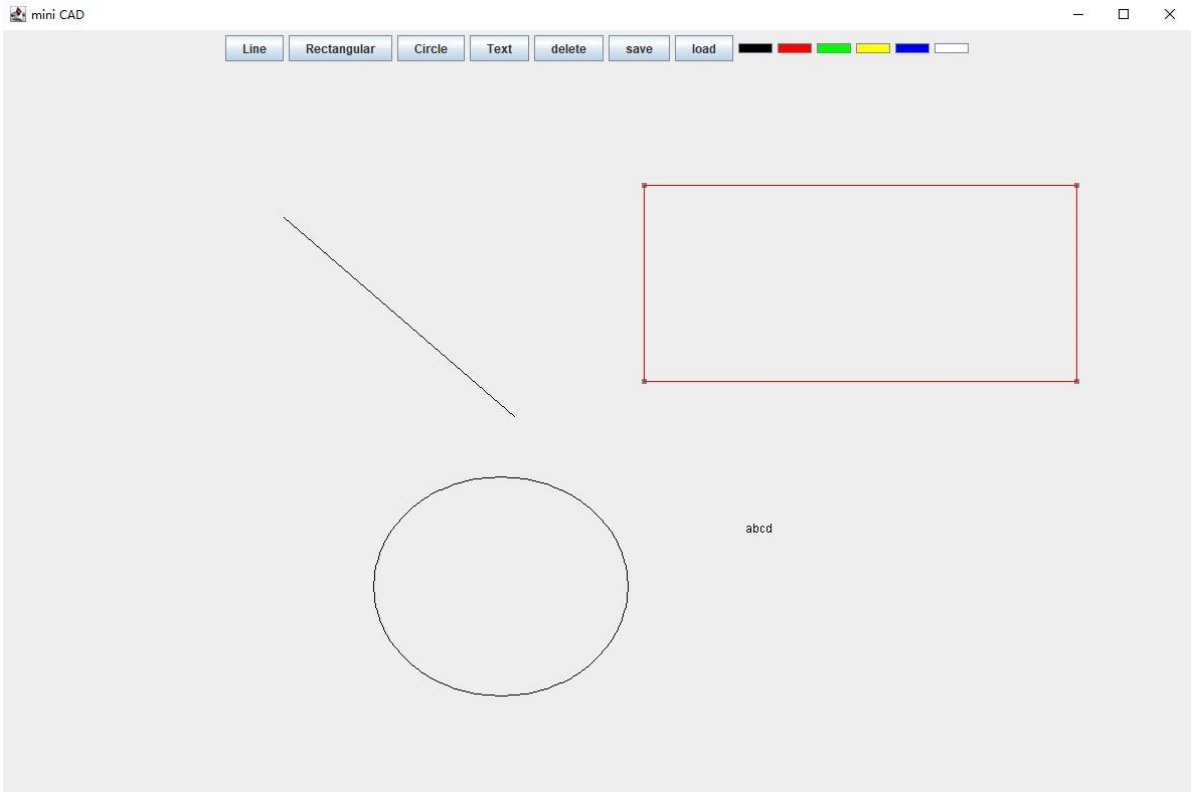


点击 text 按钮后出现对话框，输入字符串点击 OK，之后再选取两点，完成绘制。

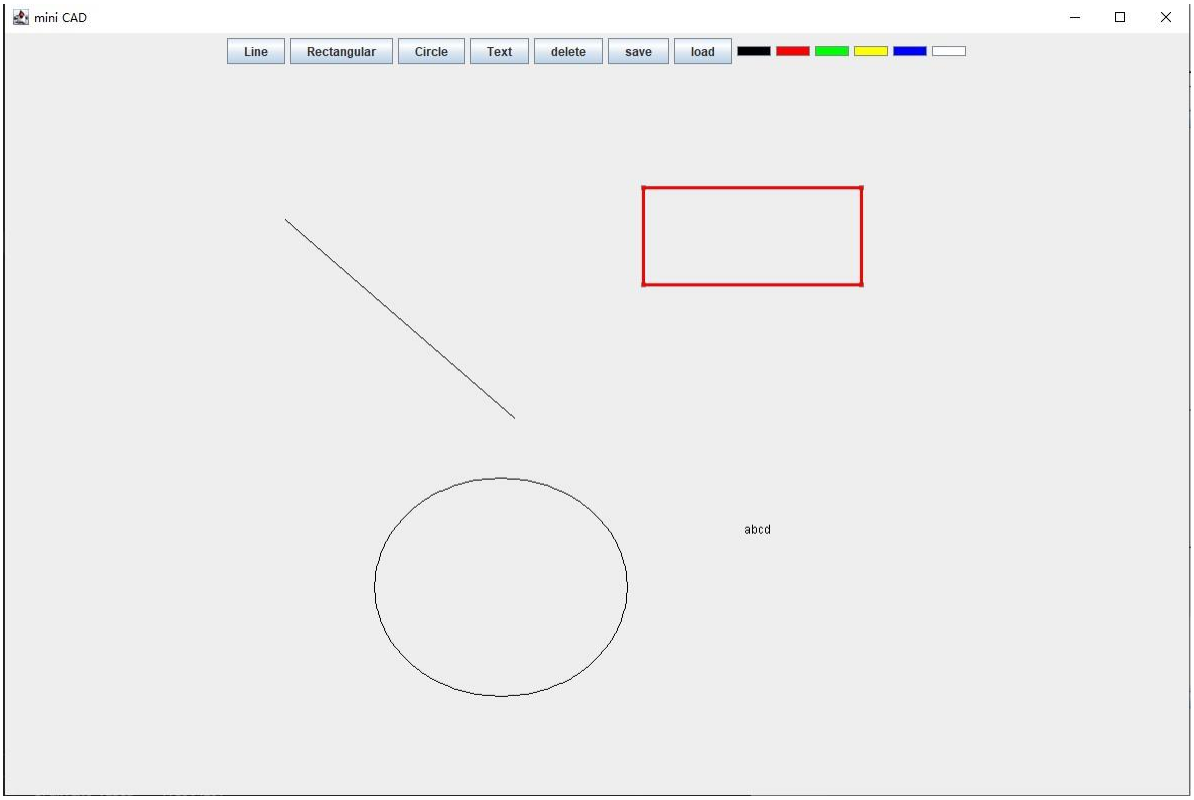

mini CAD
Line
Rectangular
Circle
Text
delete
save
load
Input a string
abcd
OK


mini CAD
Line
Rectangular
Circle
Text
delete
save
load
abcd

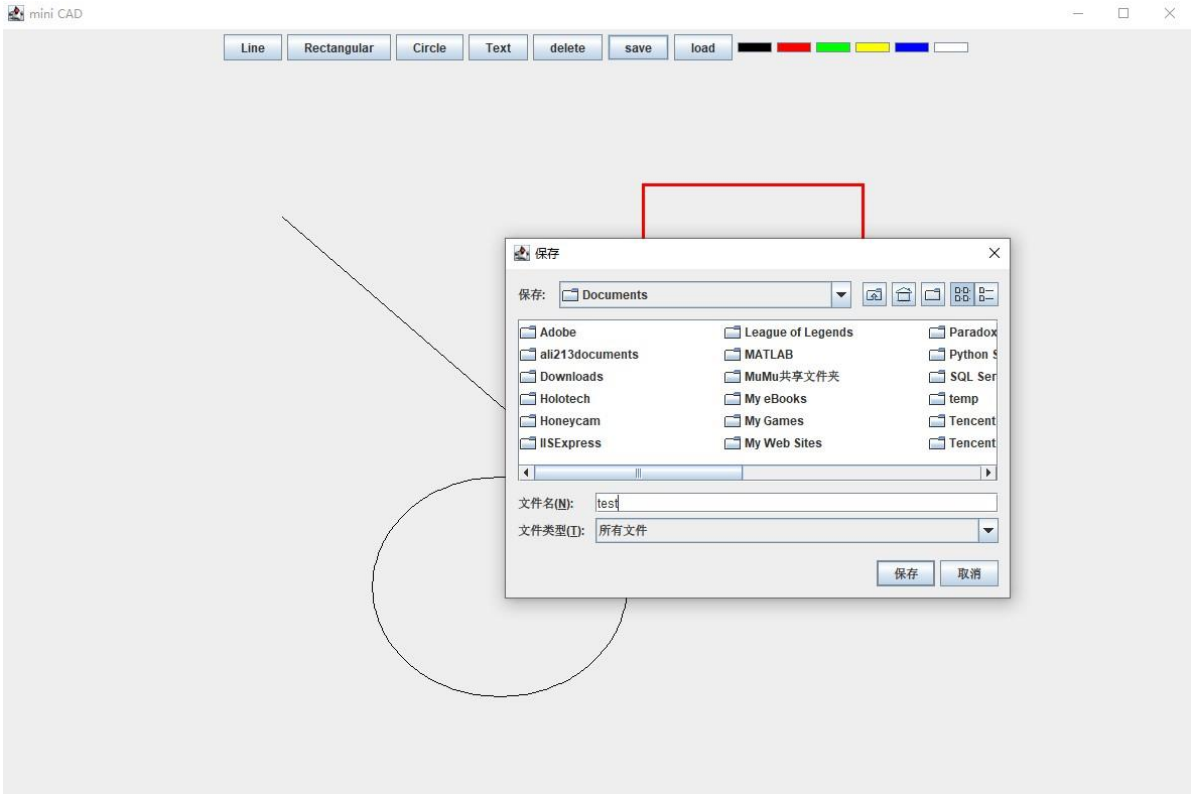
选中图案改变颜色



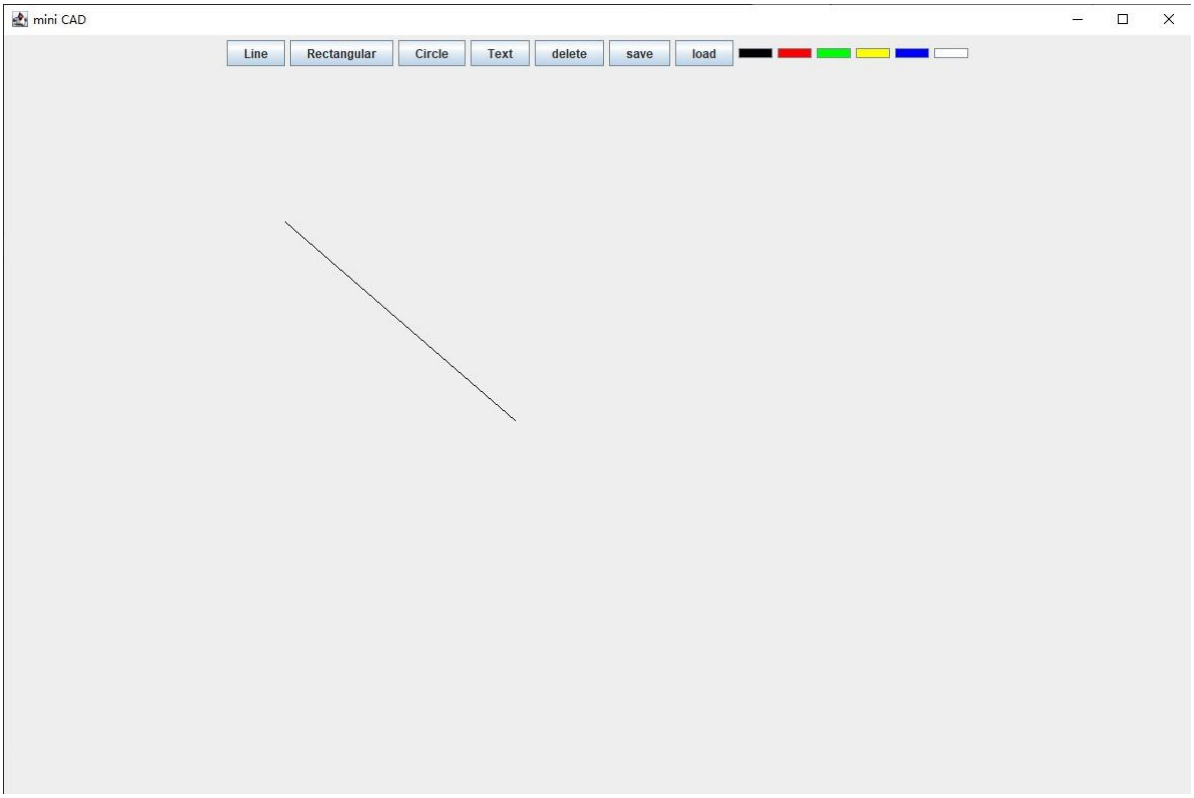
改变大小和线条粗细



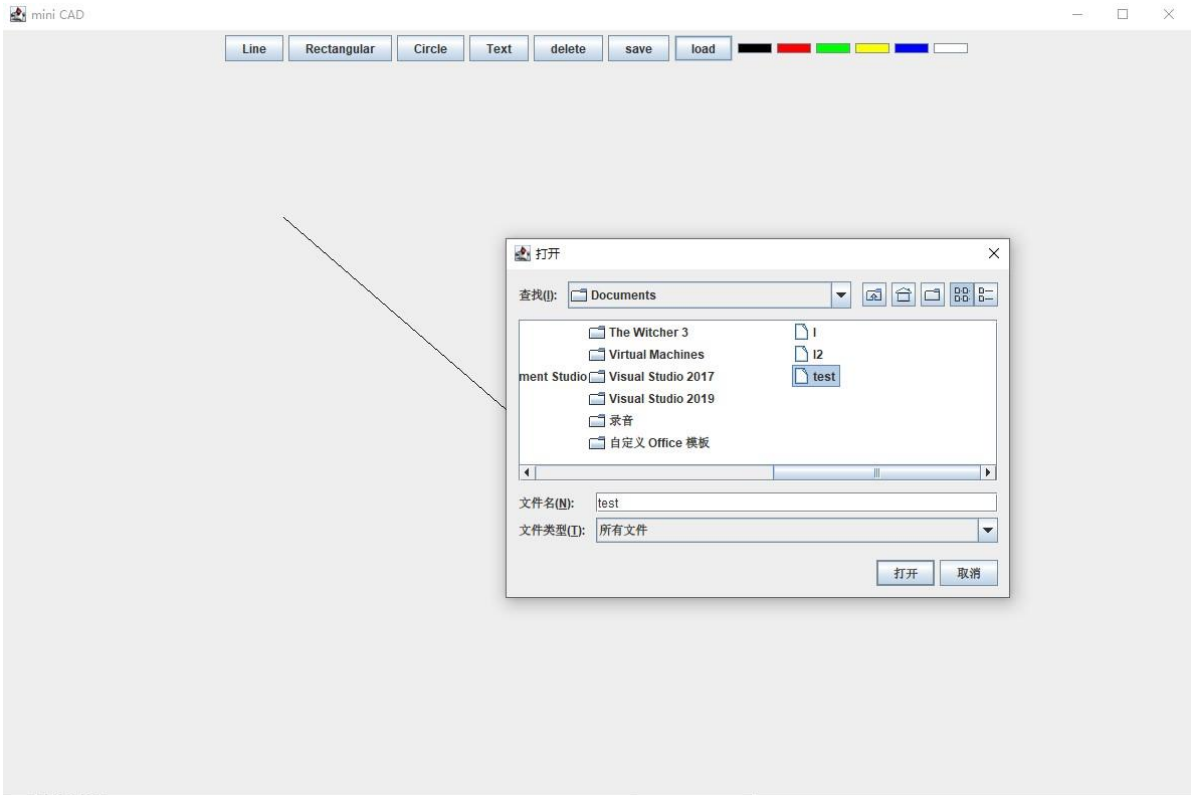
保存图案

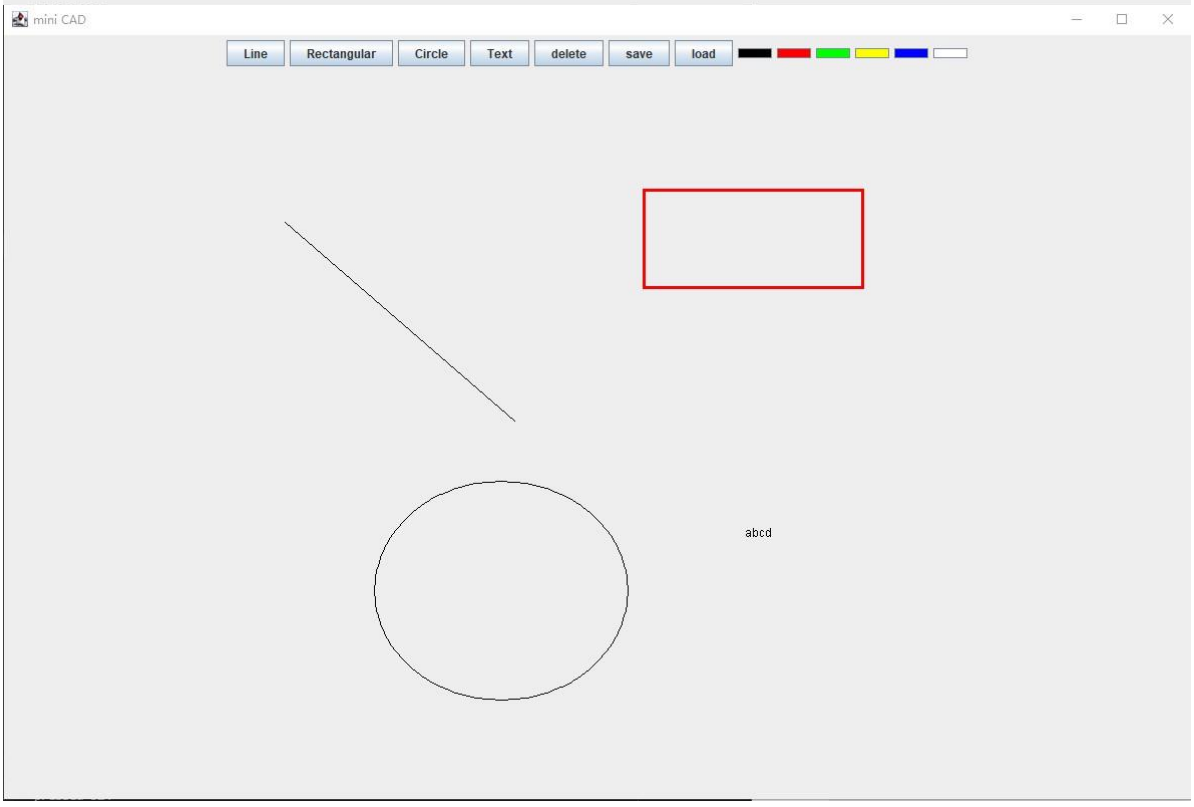


选取图像后按 delete 按钮，可以删除图像，随后可以点击删除。



恢复图像





另有视频演示 project 2 demo.mp4