2017 - 2018 学年第 1 学期

JAVA 大作业报告

作业题目名称 画图

学院（系） 计算机科学与工程学院

小组成员人数 3 人

小组成员名单(学号和姓名)：

组长 班级 计算机 1508 学号 20154536 姓名 李天志 成绩___

组员 班级 计算机 1508 学号 20154537 姓名 邓翼 成绩___

组长 班级 计算机 1508 学号 20154305 姓名 刘卓成 成绩___

目录

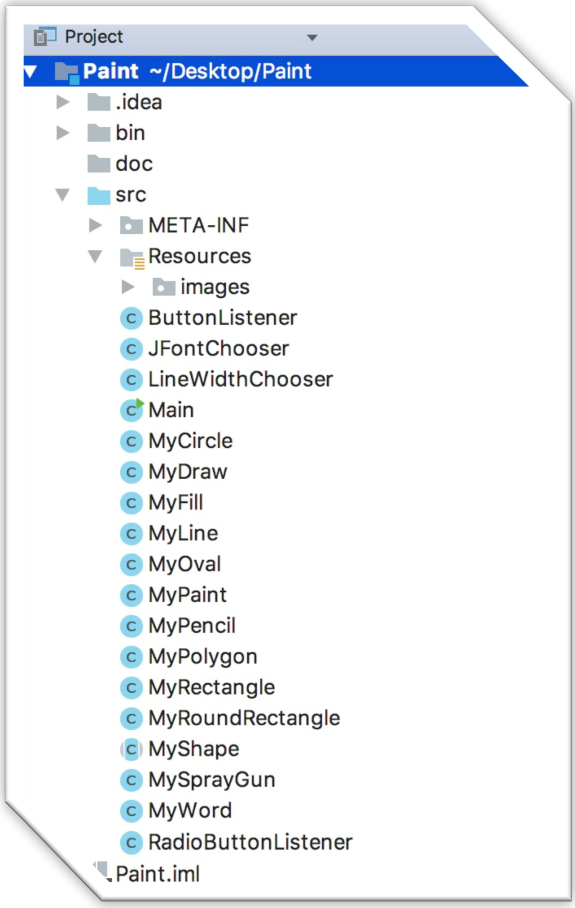
画图程序设计

一、作业内容

设计一个图画软件，包括对图形进行编辑、存取等。

二、文件与执行环境

2.1 电子文档打包文件名及文件列表：



Paint 中包含三个主要文件夹

第一个包含的是一些 idea IDE 的配置文件。

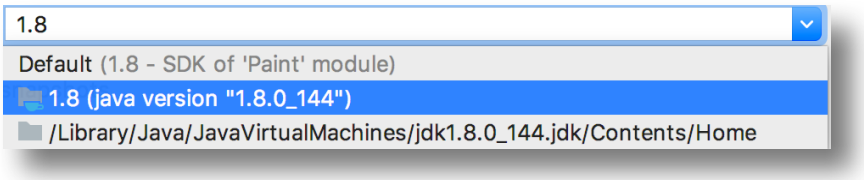
第二个包含的是可执行字节码的文件。

第三个是报告文件夹，内含 word 和 PDF 文件，推荐后者。

第四个则是源代码，内含所有源代码，Resouces/image 存储的是 Button 的 icon。

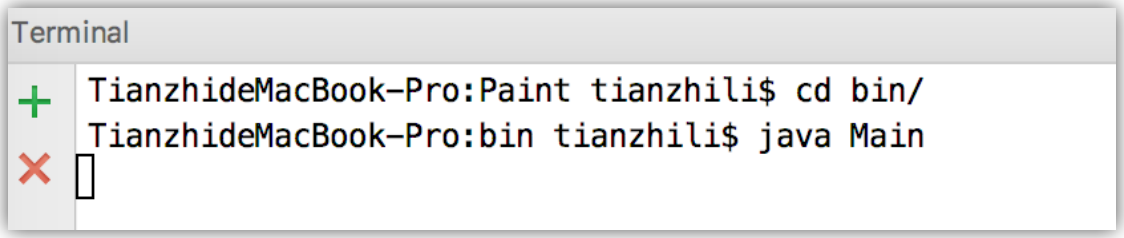
2.2 编译执行环境与命令

Java JDK 版本是 1.8.0_144



Java 版本是 Java1.8.0_151-b12

命令：

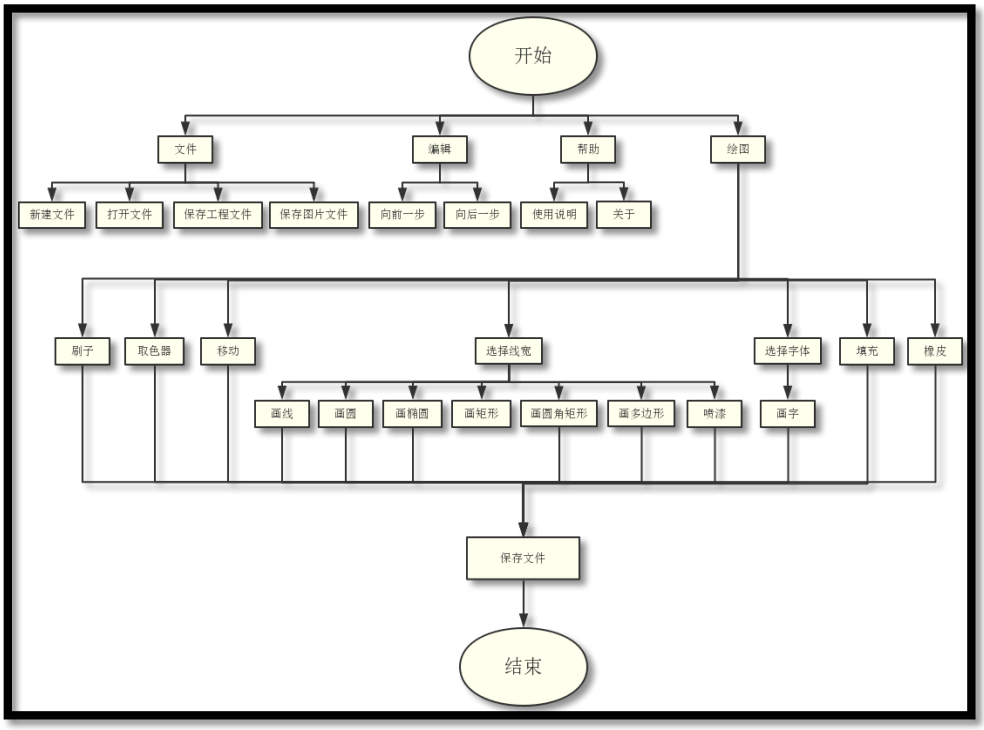


三、主要功能

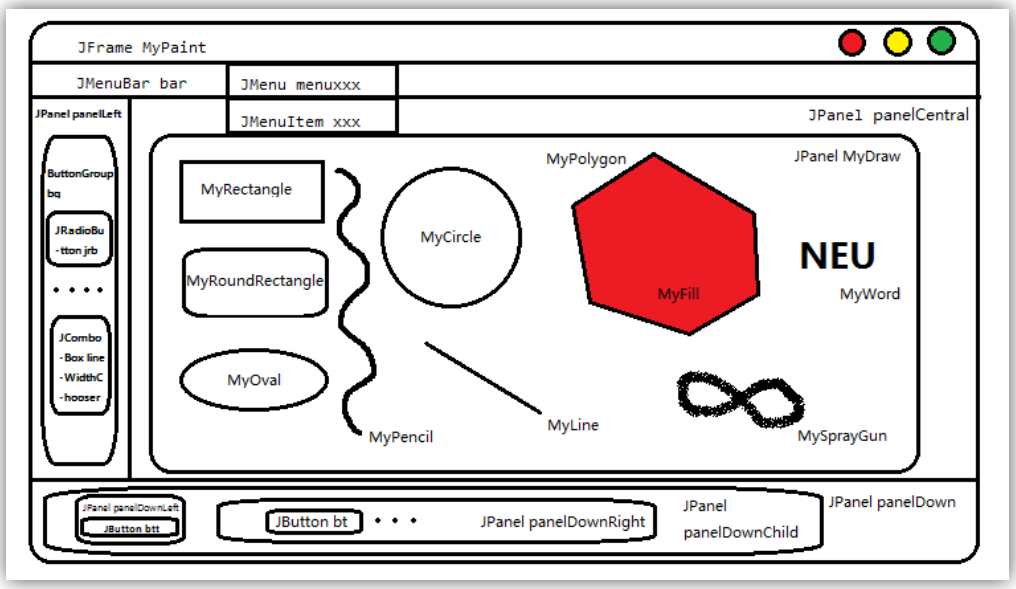
- 新建一个空白图形文件
- 打开或保存一个图形文件
- 绘制基本图形
- 设置线型并画直线、矩形、圆角矩形、圆、椭圆
- 设定画笔类型和大小，其中包括铅笔、刷子、喷枪
- 设定颜色与文字风格并添加文字
- 鼠标拖动画图和添加文字
- 设定画笔、图像修改（橡皮擦）
- 图形填充
- 撤销和反撤销操作
- 菜单
- 移动操作

四、系统的设计

4.1 使用流程图



4.2 功能模块图及说明



在功能模块说明中主要分为两部分，第一部分是功能实现的简介，第二部分是关于所有类的简单介绍和类之间的相互关系。

功能模块简单介绍一下一个图形从创建到显示的简单过程，我首先定义了一个 MyShape 类，其他图形类都是继承该类，根据不同图形添加了不同的私有变量（如 MyRoundRectangle 类中添加了 ArcWidth 和 ArcHeight 两个变量）。鼠标点击不同的图形按钮时，改变 MyDraw 中的 shapeType 的值，之后鼠标 Press 的时候，则会根据 shapeType 的值为 currentShape 创建不同的图形类对象，在鼠标 Drag 时根据鼠标位置不断更改图形 currentShape 位置属性的参数并重绘，做到实时显示图形的功能，最后在鼠标 Released 的时候把 currentShape 添加到 shapeList 中。其中重绘功能通过重写 paintComponent 函数实现，这个函数里主要就是把 shapeList 里的图形都画完之后，再画 currentShape。其他的具体功能在程序关键类中仔细介绍。

最后简单介绍一下所有类和类之间的相互关系：

首先是 MyShape，它是一个图形的抽象父类，之后通过子类去继承它，根据不同形状补充不同属性与方法，同时重写 Draw 函数，它的子类有 10 个 :MyLine、MyRectangle、MyRoundRectangle、MyOval、MyCircle、MyPolygon、MyWord、MyPencil、MySprayGun、MyFill。

第二个要介绍的是 MyDraw（JPanel），它是画板，里面有一个 List<MyShape>，存放绘画的图形，通过 repaint 把 list 里的图形全部绘制。

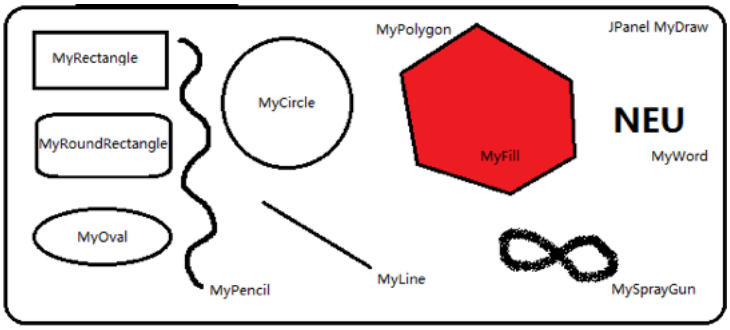
之后介绍一下 MyPaint（JFrame）类，它主要是负责界面的，内部含有 bar（JMenuBar）、panelLeft（JPanel）、panelCentral（JPanel）、panelDown（JPanel），

bar 中含有三个 menu（JMenu），不同 menu 中也含有相应的 item（JMenuItem）；panelLeft 含有一个 bg（ButtonGroup），里面，含有许多 jrb（JRadioButton），panelCentral 里面含有一个 MyDraw（JPanel），panelDown 含有左右两个 JPanel，左边的 panelDownlLeft 有一个 btt（JButton），用来展示当前选择的颜色的，右边的 panelDownRight 含有许多 bt（JButton），用来提供选择颜色。

此外还有四个小类，两个按钮监听器类（ActionListener），一个是 JRadioButtonListener，另一个是 JButtonListener，两个是选择器类，一个是 JFontChooser（JPanel）（抄自网络），另一个是 LineWidthChooser（JComboBox）。

五、程序关键类的实现

1. MyDraw（JPanel）



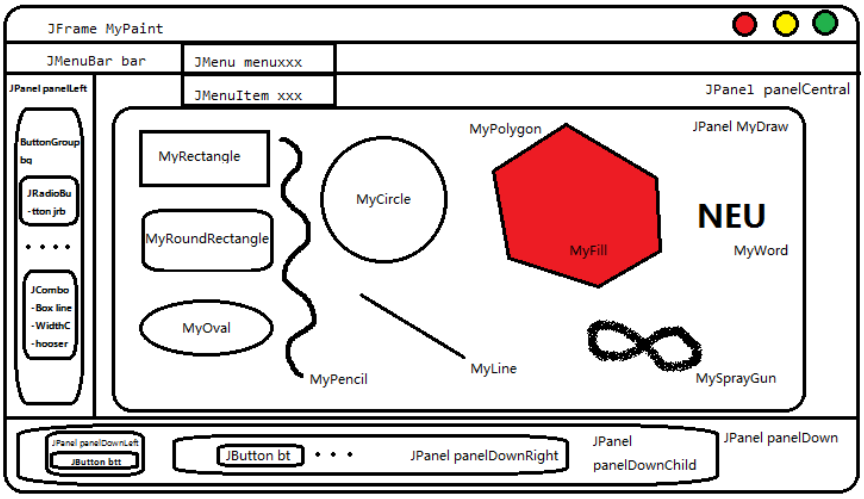
域成员：

```
private ArrayList<MyShape> shapeList;
private ArrayList<MyShape> deletedShapeList;
private int shapeType;
/*
0 for line, 1 for rectangle, 2 for roundrectangle,3 for oval, 4 for
picker,5 for brush, 6 for pencil, 7 for spray gun, 8 for erasor, 9 for
polygon, 10 for fill, 11 for  character, 12 for move, 13 for circle
*/
private MyShape currentShape;              //When I create a shape, i use
this currentShape to new different object
private MouseMonitor ma;                   //Mouse moniter in this panel,
which is very important
private int x1,x2,y1,y2,old_x,old_y;     //x1,y1 for pressed positon;
x2,y2 for dragged position; old_x,old_y for polygon
private boolean flag;                      //To judge whether it is the
first point in polygon
private JButton bt;                       //This is the current Color
button in the lower left corner
private int[][] fillJud;                   //Judge array in BFS
private int width;                        //Stoke's width
private Color color;                      //Shapes's color
private Font font;                        //Word's font
private String sentences = " ";            //Word
private int moveJud = 0;                   //judge whether have moved shape
```

成员方法：

```
//Standard set and get method.
public ArrayList<MyShape> getShapeList()
public void setType(int type)
public void setColor(Color color)
public void setFont(Font f)
public void setSentences(String s)
public void setWidth(int w)
//Mouse Monitor is very important
private class MouseMonitor extends MouseAdapte
//Renew a paint canvas
public void reset()
//Make an image of whole JPanel
private BufferedImage giveMeImage()
//Step backward
public void revoke()
//Step forward
public void voke()
//Override paintComponent for repaint
public void paintComponent(Graphics g)
//BFS to find the points need to be painted and add them into MyFill
private void BFS
//Get the point(x,y) in image
private Color pickColor
//To judge whether the point(x,y) is in the JPanel or not
private boolean isInsideImage(int x, int y)
```

2. MyPaint（JPanel）



域成员：

```
private JButton btt;                    //Current color  button
private ButtonListener bl;               //Color button listener
private RadioButtonListener rbl;          //Function button listener
private MyDraw md;                      //Canvas
private JMenuBar bar;
private JMenu menuFile;
private JMenu menuOperate;
private JMenu menuHelp;
private JMenuItem newFile;
private JMenuItem openFile;
private JMenuItem saveFile;
private JMenuItem saveimgFile;
private JMenuItem revoke;
private JMenuItem voke;
private JMenuItem help;
private JMenuItem about;
private JPanel panelCentral;
private JPanel panelLeft;
private ButtonGroup bg;
private LineWidthChooser lineWidthChooser;
private JPanel panelDown;
private JPanel panelDownLeft;
private JPanel panelDownRight;
private JPanel panelDownChild;
```

成员方法：

```
//Init Frame
public void initFrame()
//Save file into project file or image file
public void saveFile(int type)
```

3. MyShape （抽象类）

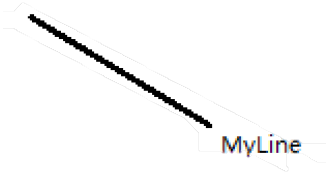
域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
```

成员方法：

```
//Abstract method Draw
public abstract void Draw(Graphics g)
//For moving operation
public void moveWhere(int x,int y)
//To judge whether be selecte or not
public boolean containsPix(int x, int y)
//Add points for MULTIPOINTS shape
public void addPoint(int x1, int y1)
//Upgrade points position
public void upgradePoint(int x1, int y1, int x2,int y2)
```

4. MyLine （MyShape）



域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
```

成员方法：

```
public MyLine(int x1, int y1, int x2, int y2, Color color, int width)
//Make two vertex move (x,y)
public void moveWhere(int x,int y)
//To judge if (x,y) is upon the segment between (x1,y1) and (x2,y2)
public boolean containsPix(int x, int y)
//Override Draw
public void Draw(Graphics g)
```

5. MyRectangle （MyShape）

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
```

成员方法：

```
public MyRectangle(int x1, int y1, int x2, int y2, Color color, int width)
//Override Draw
public void Draw(Graphics g)
```

6. MyRoundRectangle （MyShape）

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
public int arcWidth,arcHeight;            //arcWidth and arcHeight are
```

成员方法：

```
public MyRoundRectangle(int x1, int y1, int x2, int y2, Color color, int
width)
//Override Draw
public void Draw(Graphics g)
```

7. MyOval （MyShape）

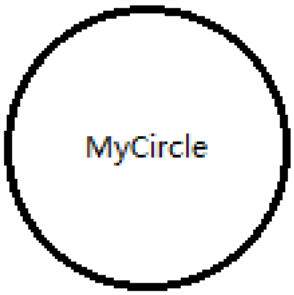
域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
```

成员方法：

```
public MyOval(int x1, int y1, int x2, int y2, Color color, int width)
//Override Draw
public void Draw(Graphics g)
```

8. MyCircle （MyShape）



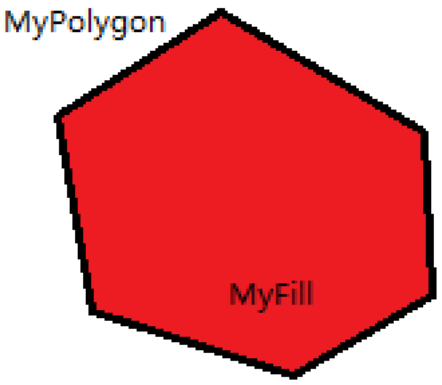
域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
```

成员方法：

```
public MyCircle(int x1, int y1, int x2, int y2, Color color, int width)
//Override Draw
public void Draw(Graphics g)
```

9. MyPolygon （MyShape）



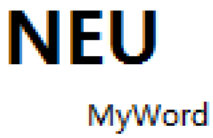
域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
private ArrayList<Point> points;       //The set of vertex points
int minx,miny,maxx,maxy;
```

成员方法：

```
public MyPolygon (int x1, int y1, Color c, int width)
//To judge if (x,y) is in the limited zone
public boolean containsPix(int x, int y)
//add point should be painted into set
public void addPoint(int x2, int y2)
//Make every point in set move  (x,y)
public void moveWhere(int mx,int my)
//Override Draw
public void Draw(Graphics g)
```

10. MyWord （MyShape）



MyWord

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
private Font font;          //Word's font
private String s;           //Word
private int size;           //Word's size
```

成员方法：

```
public MyWord(int x1, int y1,String s, Font f, Color c)
//To judge if (x,y) is in the limited zone
public boolean containsPix(int x, int y)
//Override Draw
public void Draw(Graphics g)
```

11. MyPencil （MyShape）

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
public ArrayList<Point> points;        //The set of points pencil moved
```

成员方法：

```
public MyPencil (int x1, int y1, Color c, int width)
//add point should be painted into set
public void addPoint(int x2, int y2)
//Make every point in set move  (x,y)
public void moveWhere(int mx,int my)
//Override Draw
public void Draw(Graphics g)
```

12. MySprayGun （MyShape）

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
private Random r;                //To create random integer
public ArrayList<Point> points;    //The set of points should be painted
```

成员方法：

```
public MySprayGun(int x1, int y1, Color c, int width)
//add point should be painted into set
public void addPoint(int x2, int y2)
//Make every point in set move  (x,y)
public void moveWhere(int mx,int my)
//Override Draw
public void Draw(Graphics g)
```

13. MyFill （MyShape）

域成员：

```
public int x1,y1,x2,y2;
public Color color;
public int width;
public ArrayList<Point> points;    //The set of points should be painted
```

成员方法：

```
public MyFill (int x1, int y1, Color c, int width)
//add point should be painted into set
public void addPoint(int x2, int y2)
//Make every point in set move  (x,y)
public void moveWhere(int mx,int my)
//Override Draw
public void Draw(Graphics g)
```

14. ButtonListener （ActionListener）

域成员：

```
Color c;
MyDraw md;
JButton cb;    //This is the current Color button in the lower left corner
```

成员方法：

```
public ButtonListener(MyDraw mydraw, JButton colorButton)
public void actionPerformed(ActionEvent e)
```

15. RadioButtonListener（ActionListener）

域成员：

```
int type;
MyDraw md;
```

成员方法：

```
public RadioButtonListener(MyDraw md)
public void actionPerformed(ActionEvent e)
```

16. LineWidthChooser（JComboBox）

域成员：

```
private String[] s = {"细","中","粗"};           //Three lengths of line
private ImageIcon[] icons = new ImageIcon[3];      //Three icons
private JComboBox combo;
private MyDraw md;
//Two useful classes in this line width chooser
class ItemObj
class ACellRenderer extends JLabel implements ListCellRenderer
```

成员方法：

```
public LineWidthChooser(MyDraw md)
public JComboBox getBox()
```

17. JFontChooser（JPanel）

域成员：

```
private String current_fontName = "宋体";//当前的字体名称,默认宋体.
private String Str = "";//展示的文字
private int current_fontStyle = Font.PLAIN;//当前的字样,默认常规.
private int current_fontSize = 9;//当前字体大小,默认 9 号.
private Color current_color = Color.BLACK;//当前字色,默认黑色.
private JDialog dialog;                                    //用于显示模态的窗体
private JLabel lblString;
private JLabel lblFont;                                   //选择字体的 LBL
private JLabel lblStyle;                                  //选择字型的 LBL
private JLabel lblSize;                                   //选择字大小的 LBL
private JLabel lblColor;                                  //选择 Color 的 label
private JLabel otherColor;                                //其它颜色
private JTextField txtFont;                                   //显示选择字体的
TEXT
private JTextField txtStyle;                                  //显示选择字型的
TEXT
private JTextField txtSize;                                   //显示选择字大小的
TEXT
private JTextField showTF;                                    //展示框（输入框）
private JList lstFont;                                   //选择字体的列表.
```

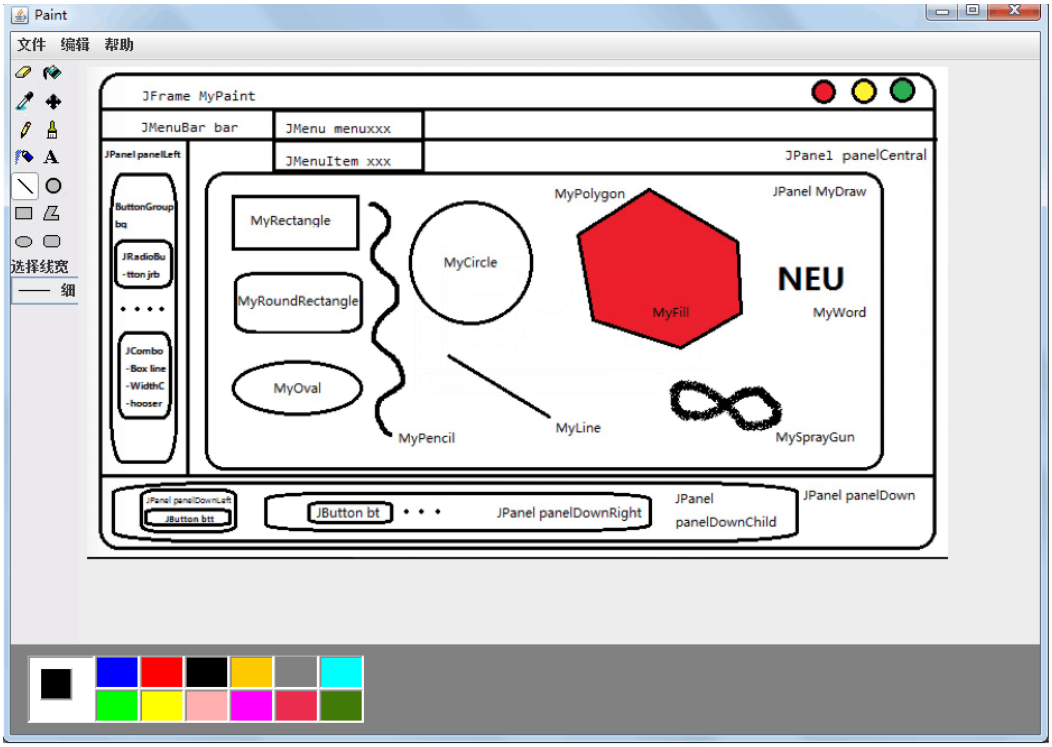
```
private JList lstStyle;                                    //选择字型的列表.
private JList lstSize;                                     //选择字体大小的列表.
private JComboBox cbColor;                                 //选择 Color 的下拉
框.
private JButton ok, cancel;                                //"确定","取消"按钮.
private JScrollPane spFont;
private JScrollPane spSize;
private JPanel showPan;                                    //显示框.
private Map sizeMap;                                       //字号映射表.
private Map colorMap;                                      //字着色映射表.
private Font selectedfont;                                 //用户选择的字体
private Color selectedcolor;                               //用户选择的颜色
```

成员方法：

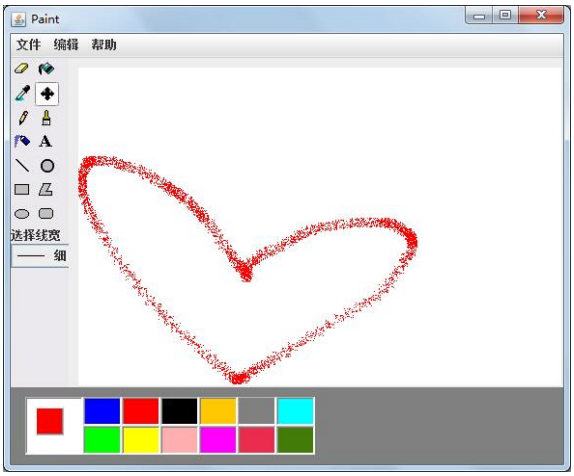
```
//无参初始化
public JFontChooser()
//重载构造，有参的初始化 用于初始化字体界面
public JFontChooser(Font font, Color color)
//可供外部调用的方法
public Font getSelectedfont()
public void setSelectedfont(Font selectedfont)
public Color getSelectedcolor()
public String getStr()
public void setSelectedcolor(Color selectedcolor)
/*初始化界面*/
private void init(Font font, Color color)
/*显示字体选择器对话框(x,y 表示窗体的启动位置)*/
public void showDialog(Frame parent, int x, int y)
```

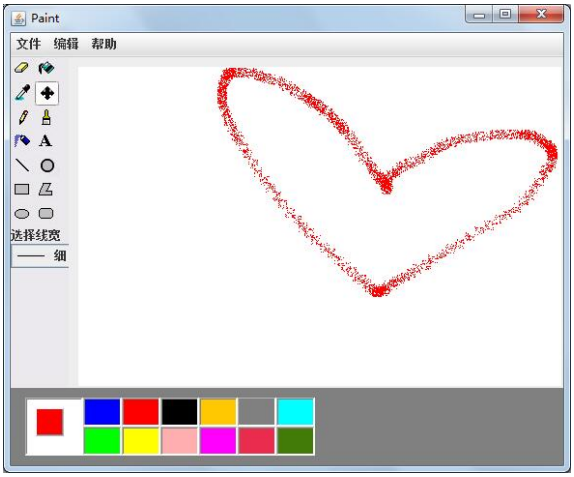
六、软件的主要界面截图

基本的绘图如下所示（基本图形的绘画全部做到，下对特色功能再次截图说明）

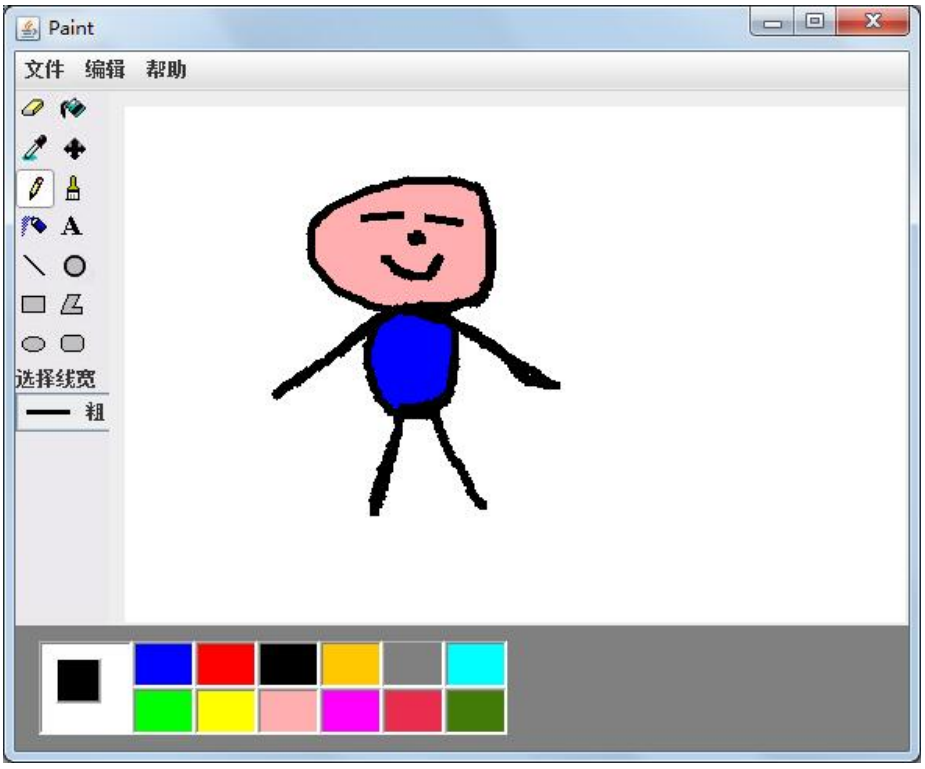


移动操作

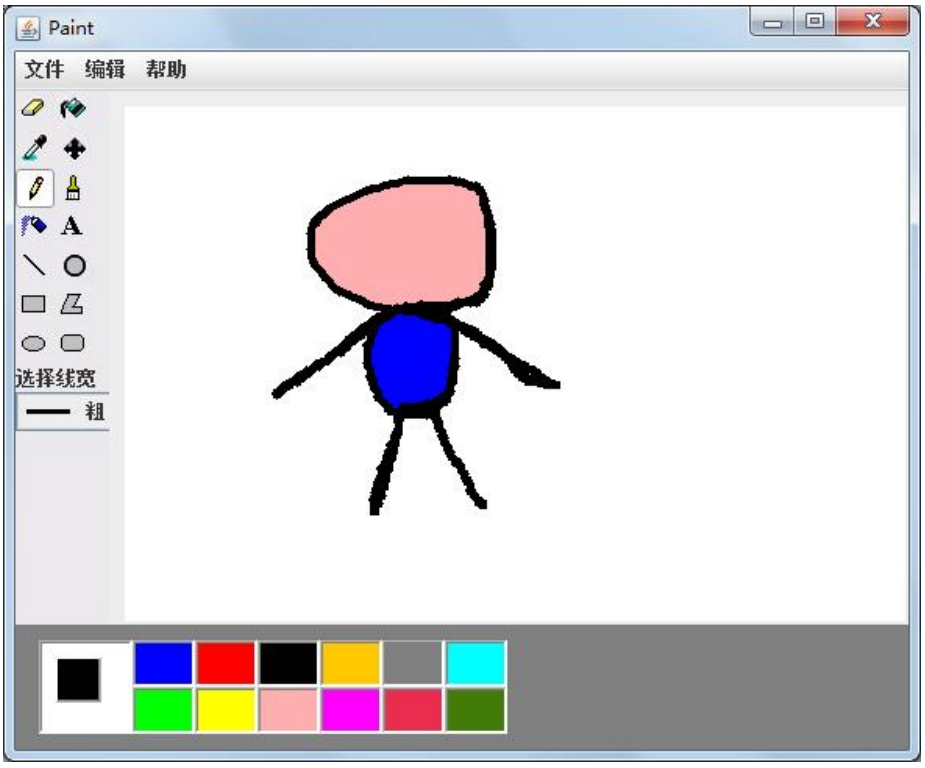




填充操作（加画笔操作）



撤销操作



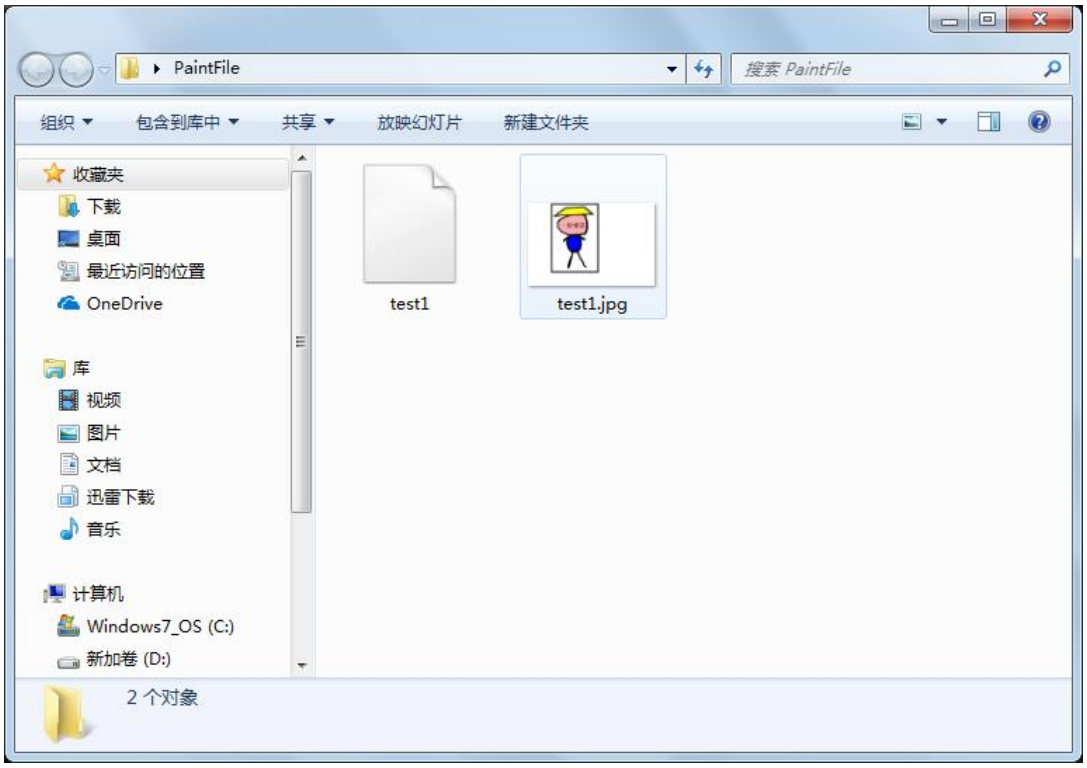
添加文字功能

添加多边形功能（加填充功能）

橡皮擦功能

文件存储功能

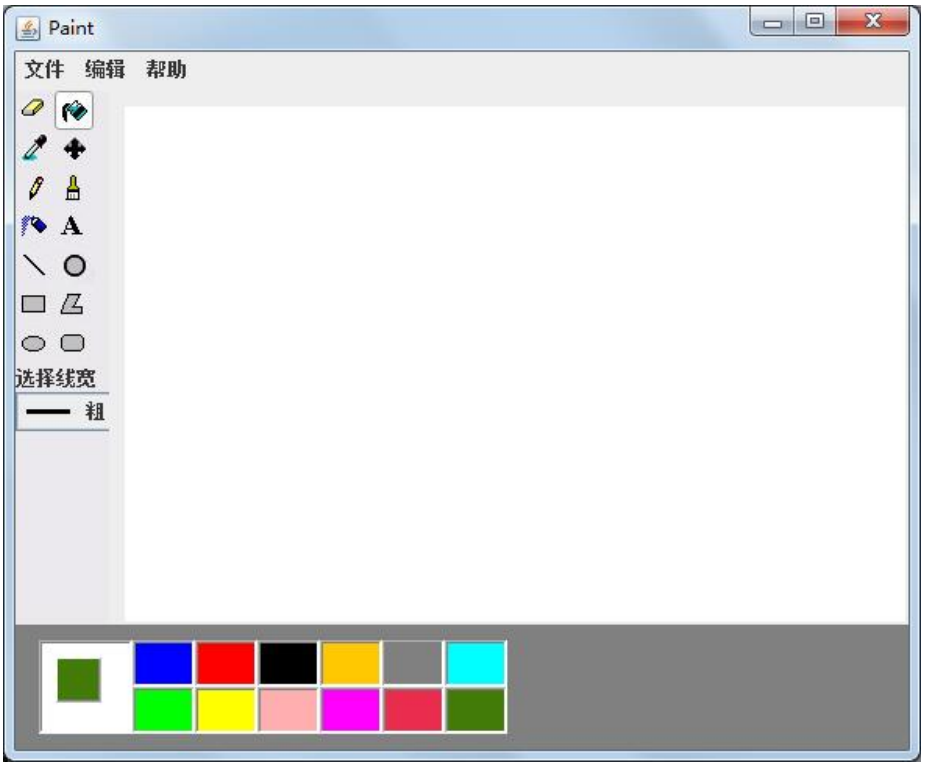

Paint
文件 编辑 帮助
选择线宽
粗
一脸懵逼
消息
保存成功！
确定


PaintFile
搜索 PaintFile
组织
包含到库中
共享
放映幻灯片
新建文件夹
收藏夹
下载
桌面
最近访问的位置
OneDrive
库
视频
图片
文档
迅雷下载
音乐
计算机
Windows7_OS (C:)
新加卷 (D:)
test1
test1.jpg
2 个对象

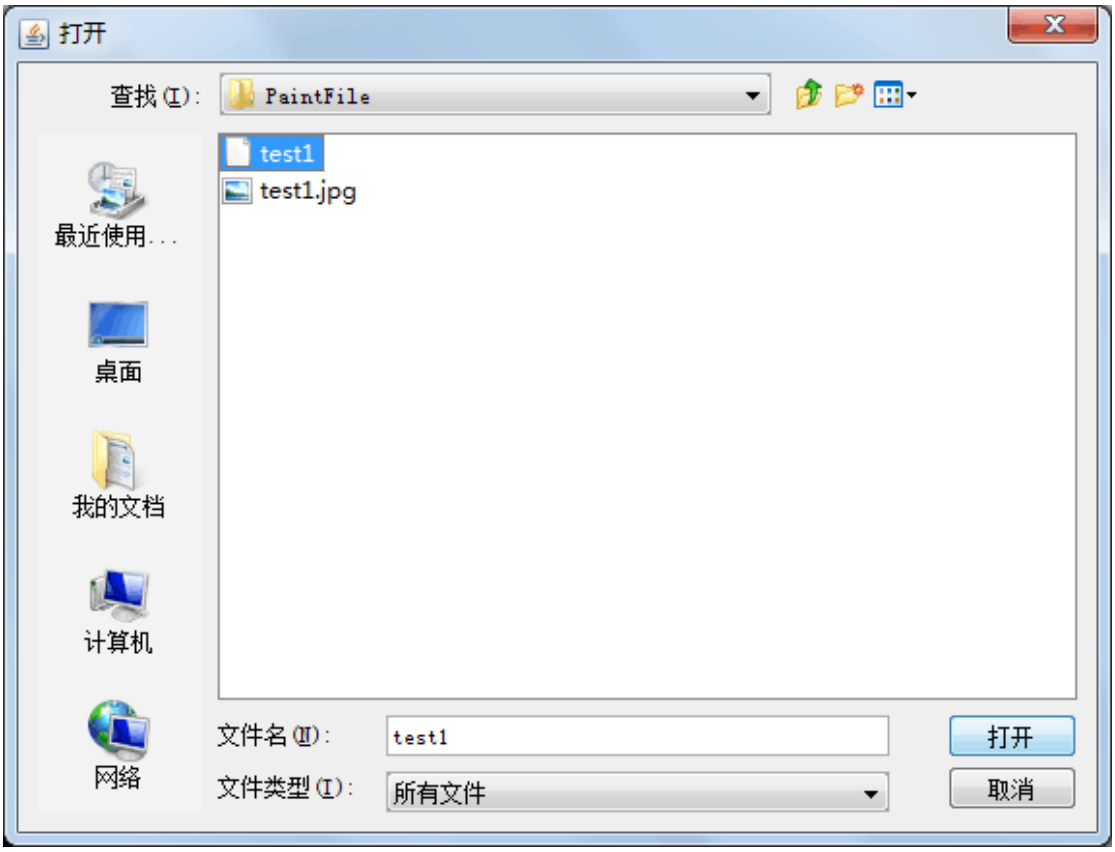
文件新建功能


Paint
文件 编辑 帮助
选择线宽
粗
一脸懵逼
提示信息
是否需要保存当前文件？
是(Y)
否(N)

选择“否”后

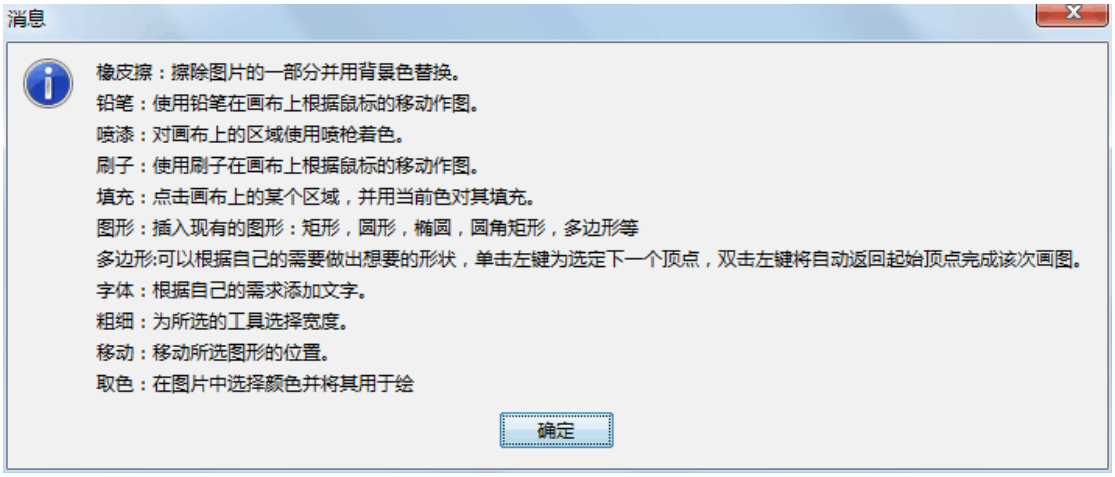

Paint
文件 编辑 帮助
选择线宽
粗

文件打开功能


打开
查找(I):
PaintFile
test1
test1.jpg
最近使用...
桌面
我的文档
计算机
网络
文件名(N):
test1
打开
文件类型(T):
所有文件
取消


Paint
文件
编辑
帮助
选择线宽
粗
一脸懵逼

使用帮助


消息
橡皮擦：擦除图片的一部分并用背景色替换。
铅笔：使用铅笔在画布上根据鼠标的移动作图。
喷漆：对画布上的区域使用喷枪着色。
刷子：使用刷子在画布上根据鼠标的移动作图。
填充：点击画布上的某个区域，并用当前色对其填充。
图形：插入现有的图形：矩形，圆形，椭圆，圆角矩形，多边形等
多边形:可以根据自己的需要做出想要的形状，单击左键为选定下一个顶点，双击左键将自动返回起始顶点完成该次画图。
字体：根据自己的需求添加文字。
粗细：为所选的工具选择宽度。
移动：移动所选图形的位置。
取色：在图片中选择颜色并将其用于绘
确定

七、任务分工与联系方式：

姓名	完成内容	联系方式(手机号)
李天志	1. 系统功能的设计，任务的分工。 2. 界面 MyPaint 类的改写，包括目录、各种。 3. 画板 MyDraw 类的编写，包括鼠标监听器、BFS 算法、撤销反撤销操作、存储 image 图片等。 4. 图形父类 MyShape 类的编写，图形子类 MyPencil 类、MySprayGun 类、MyFIll 类的编写。 5. 监听器类 ButtonListener 类和 RadioButtonListener 类的编写。 6. 选择器类 LineWidthChooser 类的编写 7. 部分图标设计。 8. 实验报告中主要负责的板块有：文件与执行环境、系统的设计、程序关键类的实现、任务分工与联系方式。	15940255453
邓翼	1. 图形子类 MyWord 类、MyOval 类和 MyCircle 类的编写。 2. 使用说明的书写。 3. 实验报告中主要负责的板块有：作业内容、使用流程图、软件截图、程序框图。	15804050891
刘卓成	1. 图形子类 MyRectangle 类、MyRoundRectangle 类和 MyLine 类的编写 2. 实验报告里主要负责的板块有 :作业内容、基本要求。	15940258972

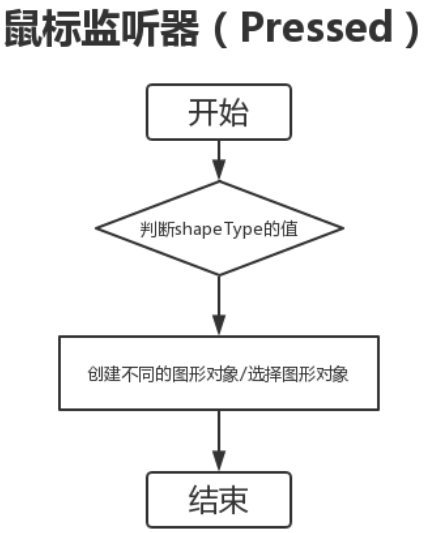
7.1 李天志代码:

7.1.1 MyDraw 类

```
/*
This is most Important part of my project! I have commented some codes which
I have try my best to accomplish some goals.
Though I know I need to delete useless codes when I hand this homework, I
want to keep it. It's not only code, but my attempts.
 */


public class MyDraw extends JPanel {
```

```
    private ArrayList<MyShape> shapeList;
    private ArrayList<MyShape> deletedShapeList;
    private int shapeType;
    /*
    0 for line, 1 for rectangle, 2 for roundrectangle,3 for oval, 4 for
picker,
    5 for brush, 6 for pencil, 7 for spray gun, 8 for erasor, 9 for polygon,
    10 for fill, 11 for  character, 12 for move, 13 for circle
    */
    private MyShape currentShape;             //When I create a shape, i use
this currentShape to new different object
    private MouseMonitor ma;                 //Mouse moniter in this panel,
which is very important
    private int x1,x2,y1,y2,old_x,old_y;       //x1,y1 for pressed positon;
x2,y2 for dragged position; old_x,old_y for polygon
    private boolean flag;                    //To judge whether it is the
first point in polygon
    private JButton bt;                      //This is the current Color
button in the lower left corner
    private int[][] fillJud;                 //Judge array in BFS
    private int width;                       //Stoke's width
    private Color color;                     //Shapes's color
    private Font font;                       //Word's font
    private String sentences = " ";           //Word
    private int moveJud = 0;                 //judge whether have moved shape
or not

    public MyDraw(JButton bt){
        shapeType=0;
        flag = true;
        this.bt = bt;
        width = 1;
        color = Color.black;
        ma=new MouseMonitor();
        this.addMouseListener(ma);
        this.addMouseMotionListener(ma);
        shapeList=new ArrayList<MyShape>();
        deletedShapeList=new ArrayList<MyShape>();
    }

    //Standard set and get method.
    public ArrayList<MyShape> getShapeList() {
        return shapeList;
    }
```

```
    public void setType(int type){
        this.shapeType = type;
    }
    public void setColor(Color color){
        this.color = color;
    }
    public void setFont(Font f){
        this.font = f;
    }
    public void setSentences(String s){
        this.sentences = s;
    }
    public void setWidth(int w) {
        this.width = w;
    }

    //Mouse Monitor is very important
    private class MouseMonitor extends MouseAdapter {
```

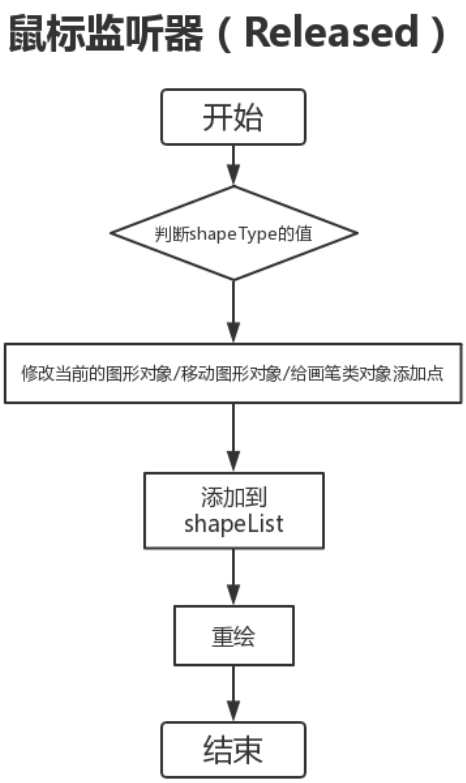



```
        public void mousePressed(MouseEvent e) {
            //When the mouse pressed, we should consider the user will create a new shape. So we should save the current condition and delete the shapes in deletedList
            deletedShapeList.clear();
            //get the mouse position when mouse pressed
            x1=e.getX();
            y1=e.getY();
```

```
        //Judge what operation user choose (create rectangle? pick color? fill? etc..)
        /*
           If the current operation is creating a new shape, then we just new an object fot currentShape, and when we draged to change the properties of currentShape.
           If the current operation is move shapes, than we just need to find which has been selected.
         */
        switch (shapeType){
            case 0:
                currentShape = new MyLine(x1,y1,0,0,color,width);
                break;
            case 1:
                currentShape = new MyRectangle(x1,y1,0,0,color,width);
                break;
            case 2:
                currentShape = new MyRoundRectangle(x1,y1,0,0,40,40,color,width);
                break;
            case 3:
                currentShape = new MyOval(x1,y1,0,0,color,width);
                break;
            case 5:
                currentShape = new MyPencil(x1,y1,color,10);
                break;
            case 6:
                currentShape = new MyPencil(x1,y1,color,width);
                break;
            case 7:
                currentShape = new MySprayGun(x1,y1,color,width);
                break;
            case 8:
                currentShape = new MyPencil(x1,y1,Color.white,16);
                break;
            case 11:
                String s = sentences;
                currentShape = new MyWord(x1,y1,s,font,color);
                shapeList.add(currentShape);
                repaint();
                sentences="";
                break;
            /*
                If the current operation is to move some shapes, then we
```

```
should know which shape will be selected.
                So I find from the last in shapeList, if the shape contains the point mouse clicked, than add this
                shape into moveShapeList. However, it is not easy as I considered. Since some shapes use two points
                to draw(like rectangle, oval, etc.), and some shapes is a set of point(like pencil, fill, etc.) As a
                result, I need to use two different ways to judge whether the point is in the shape or not. What's more,
                if a shape is upon another shape, what should I do when the user clicked shared zone. My solution is
                move only one TWO-POINTS shape and as many as possible MULTIPOINTS shape.
                 */
                case 12:
                    moveJud=0;
                    for(int i=shapeList.size()-1 ;i >=0; i--){
                        if(shapeList.get(i) instanceof  MyFill){
                            MyFill tempFill = (MyFill) shapeList.get(i);
                            if(tempFill.points.contains(new Point(x1,y1))){
                                currentShape = tempFill;
                                deletedShapeList.add(tempFill);
                                shapeList.remove(i);
                                moveJud=1;
                                break;
                            }
                        } else if(shapeList.get(i) instanceof  MySprayGun){
                            MySprayGun tempSG = (MySprayGun) shapeList.get(i);
                            if(tempSG.points.contains(new Point(x1,y1))){
                                currentShape = tempSG;
                                deletedShapeList.add(tempSG);
                                shapeList.remove(i);
                                moveJud=1;
                                break;
                            }
                        }  else if(shapeList.get(i) instanceof  MyPencil){
                            MyPencil tempPC = (MyPencil) shapeList.get(i);
                            if(tempPC.points.contains(new Point(x1,y1))){
                                currentShape = tempPC;
                                deletedShapeList.add(tempPC);
                                shapeList.remove(i);
                                moveJud=1;
                                break;
                            }
```

```
                    } else{
                        currentShape = (MyShape) shapeList.get(i);
                        if(currentShape.containsPix(x1,y1)){
                            deletedShapeList.add(currentShape);
                            shapeList.remove(i);
                            moveJud=1;
                            break;
                        }
                    }
                }
                break;
            case 13:
                currentShape = new MyCircle(x1,y1,0,0,color,width);
                break;
        }
    }

    public void mouseMoved(MouseEvent e){
        x1=e.getX();
        y1=e.getY();
        /*
            Drawing word is somewhat different from others. For others,
user can know real-time look of the canvas,
            however, if the operation is drawing a word, users cannot know
what looks like after adding the word.
            So I add this method in mouse moved.
         */
        if(shapeType==11){
            String s = sentences;
            currentShape = new MyWord(x1,y1,s,font,color);
            repaint();
        }
        //Same for erasor
        else if(shapeType==8){
            currentShape = new MyRectangle(x1-4,y1-4,8,8,Color.white,8);
            repaint();
        }
    }
```

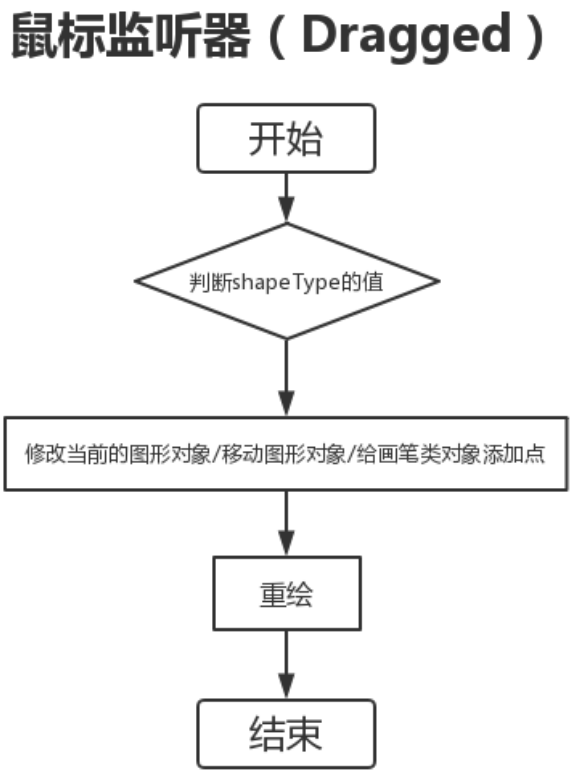



```
    public void mouseReleased(MouseEvent e) {
        int x=e.getXOnScreen();
        int y=e.getYOnScreen();
        x2=e.getX();
        y2=e.getY();
        switch (shapeType){
            case 0:
                currentShape.upgradePoint(x1,y1,x2,y2);
                repaint();
                shapeList.add(currentShape);
                break;
            case 1:
                currentShape.upgradePoint(Math.min(x2, x1),Math.min(y2, y1),
Math.abs(x2-x1),Math.abs(y1-y2));
                repaint();
                shapeList.add(currentShape);
                //currentShape=null;
                break;
            case 2:
                currentShape.upgradePoint(Math.min(x2, x1),Math.min(y2, y1),
Math.abs(x2-x1),Math.abs(y1-y2));
```

```
                repaint();
                shapeList.add(currentShape);
                break;
            case 3:
                currentShape.upgradePoint(Math.min(x2, x1),Math.min(y2, y1),
Math.abs(x2-x1),Math.abs(y1-y2));
                repaint();
                shapeList.add(currentShape);
                break;
            case 4:
                BufferedImage image1 = giveMeImage();
                Color bicolor=pickColor(x2,y2,image1);
                color = bicolor;
                bt.setBackground(bicolor);
                break;
            case 5:
                currentShape.addPoint(x2,y2);
                repaint();
                shapeList.add(currentShape);
                break;
            case 6:
                currentShape.addPoint(x2,y2);
                repaint();
                shapeList.add(currentShape);
                break;
            case 7:
                currentShape.addPoint(x2,y2);
                repaint();
                shapeList.add(currentShape);
                break;
            case 8:
                currentShape.addPoint(x2,y2);
                repaint();
                shapeList.add(currentShape);
                break;
            //For polygon, if this is the first point, we should record the
old_x and old_y, else just add the point to MyPolygon class
            case 9:
                if(flag){
                    old_x = x2;
                    old_y = y2;
                    currentShape = new MyPolygon(x2,y2,color,width);
                    flag=false;
                }
```

```
                else{
                    currentShape.addPoint(x2,y2);
                }
                repaint();
                break;
            //For the function of filling, I use BFS algorithm to get which
point should be paint
            /*
                This is somewhat difficult for me and very interesting to
accomplish this feature. Since before attempting this function,
                I just use two points or several points(Polygon) to record a
shape. However filling a paint should add plenty of points,
                which seems not very to be very consistent to other class.

                And the beginning, I made some mistakes. For example, in
order to get the color of a position, every time I want to
                "pick" color, I use robot to make a 1*1(pixel) screenshot
and get the (0,0) position. And without a doubt, it's very slow.
                And then, I change the way that make a full JPanel(MyDraw)
screenshot( private BufferdImage giveMeImage() ), and than use
                this image to pick color. And then use BFS to record points
to be painted and add to MyFill.

             */
            case 10:
                BufferedImage image = giveMeImage();
                currentShape = new MyFill(x2, y2, color, 1);
                fillJud = new int[507][337];
                Color currentColor = pickColor(x2, y2, image);
                BFS(x2, y2, currentColor, image);
                repaint();
                shapeList.add(currentShape);
                break;
            case 12:
                if(moveJud == 1){
                    shapeList.add(currentShape);
                }
                break;
            case 13:
                currentShape.upgradePoint(Math.min(x2, x1),Math.min(y2, y1),
Math.max(Math.abs(x2-x1),Math.abs(y1-y2)), Math.max(Math.abs(x2-
x1),Math.abs(y1-y2)));
                repaint();
                shapeList.add(currentShape);
```

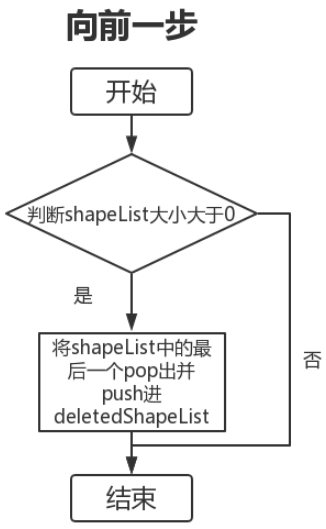
```
                break;
            }
            if(shapeType!=9){
                currentShape = null;
            }
        }
```

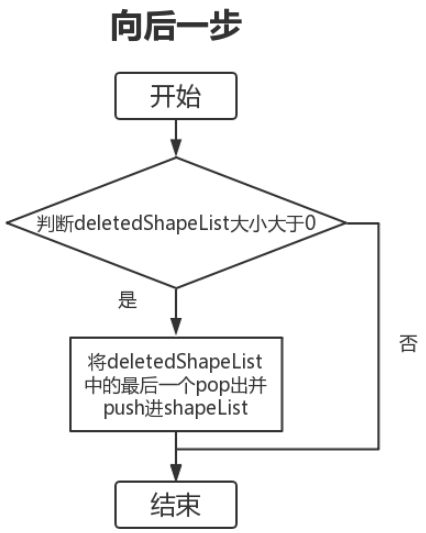



```
        public void mouseDragged(MouseEvent e) {
            /*
                When the mouse dragged, what I need to do is update the current
position and repaint the shape.
             */
            x2=e.getX();
            y2=e.getY();
            if(shapeType == 0){
                currentShape.upgradePoint(x1,y1,x2,y2);
            } else if(shapeType>0&&shapeType<4){
                currentShape.upgradePoint(Math.min(x2, x1), Math.min(y2, y1),
Math.abs(x2 - x1), Math.abs(y1 - y2));
                /*
```

```
                I want to have a keyboard listener to listen whether users
pressed SHIFT, but it doesn't work.
             */

//             addKeyListener(new KeyAdapter() {
//                @Override
//                public void keyPressed(KeyEvent e) {
//                   if(e.getKeyCode() == KeyEvent.VK_SHIFT){
//                      currentShape.upgradePoint(Math.min(x2,
x1),Math.min(y2, y1), Math.min(Math.abs(x2-x1),Math.abs(y1-y2)),
Math.min(Math.abs(x2-x1),Math.abs(y1-y2)));
//                   } else {
//                      currentShape.upgradePoint(Math.min(x2, x1),
Math.min(y2, y1), Math.abs(x2 - x1), Math.abs(y1 - y2));
//                   }
//                }
//             });

         } else if(shapeType>4 && shapeType <9){
            currentShape.addPoint(x2,y2);
         } else if(shapeType==12){
            if(moveJud == 1){
               currentShape.moveWhere(x2-x1,y2-y1);
               x1=x2;y1=y2;          //Without, the distance will be
accumulated.
            }
         } else if(shapeType==13){
            currentShape.upgradePoint(Math.min(x2, x1),Math.min(y2, y1),
Math.max(Math.abs(x2-x1),Math.abs(y1-y2)), Math.max(Math.abs(x2-
x1),Math.abs(y1-y2)));
         }
         repaint();
      }

      public void mouseClicked(MouseEvent e) {
         /*
            If the number of mouse clicking is 2, than add a new line
between the last point and the "oldest" point.
          */
         int count =e.getClickCount();
         if(count==2 && shapeType == 9){
            currentShape.addPoint(old_x,old_y);
            flag=true;
            shapeList.add(currentShape);
```

```
            repaint();
            currentShape = null;
        }
    }

}

//Renew a paint canvas
public void reset(){
    shapeList.clear();
    deletedShapeList.clear();
    repaint();
}

//Make an image of whole JPanel
private BufferedImage giveMeImage(){
    Dimension imageSize = this.getSize();
    BufferedImage image = new BufferedImage(imageSize.width,
            imageSize.height, BufferedImage.TYPE_INT_RGB);
    Graphics2D g = image.createGraphics();
    this.paint(g);
    g.dispose();
    return image;
}

//Save the canvas as a jpg file
public void savePic(File f){
    Dimension imageSize = this.getSize();
    BufferedImage image = new BufferedImage(imageSize.width,
            imageSize.height, BufferedImage.TYPE_INT_RGB);
    Graphics2D g = image.createGraphics();
    this.paint(g);
    g.dispose();
    try {
        ImageIO.write(image, "jpg", f);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```






```
//Step backward
public void revoke(){
    if(shapeList.size()>0) {
        //pop a shape from deletedShapeList and push it into shapeList
        deletedShapeList.add(shapeList.get(shapeList.size() - 1));
        shapeList.remove(shapeList.size() - 1);
        repaint();
    }
}

//Step forward
public void voke(){
    if(deletedShapeList.size()>0) {
        //pop a shape from shapeList and push it into deletedShapeList
        shapeList.add(deletedShapeList.get(deletedShapeList.size() - 1));
        deletedShapeList.remove(deletedShapeList.size() - 1);
        repaint();
    }
}

//Override paintComponent for repaint
public void paintComponent(Graphics g){
    super.paintComponent(g);
    //Draw every shapes in shapeList
    for(int i=0; i<shapeList.size(); i++){
        shapeList.get(i).Draw(g);
    }
```

```
        //Then draw the current shape
        if(currentShape!=null){
            currentShape.Draw(g);
        }
    }

    //BFS to find the points need to be painted and add them into MyFill
    private void BFS(int x1, int y1, Color c, BufferedImage image){
        Point temppoint;
        ArrayList<Point> sp = new ArrayList<Point>();  //Use this ArrayList
as a Queue
        sp.add(new Point(x1,y1));
        fillJud[x1][y1] = 1;
        while(sp.size()!=0){          //While the queue is not empty, then get
the front point and find jacent points.
            temppoint = sp.get(0);
            sp.remove(0);
            int tempx=temppoint.x, tempy = temppoint.y;
            if(isInsideImage(tempx+1,tempy) && fillJud[tempx+1][tempy] == 0 &&
pickColor(tempx+1,tempy,image).equals(c)){
                sp.add(new Point(tempx+1,tempy));
                fillJud[tempx+1][tempy] = 1;
            }
            if(isInsideImage(tempx-1,tempy) && fillJud[tempx-1][tempy] == 0 &&
pickColor(tempx-1,tempy,image).equals(c)){
                sp.add(new Point(tempx-1,tempy));
                fillJud[tempx-1][tempy] = 1;
            }
            if(isInsideImage(tempx,tempy+1) && fillJud[tempx][tempy+1] == 0 &&
pickColor(tempx,tempy+1,image).equals(c)){
                sp.add(new Point(tempx,tempy+1));
                fillJud[tempx][tempy+1] = 1;
            }
            if(isInsideImage(tempx,tempy-1) && fillJud[tempx][tempy-1] == 0 &&
pickColor(tempx,tempy-1,image).equals(c)){
                sp.add(new Point(tempx,tempy-1));
                fillJud[tempx][tempy-1] = 1;
            }
            currentShape.addPoint(tempx, tempy);
        }
    }

    //Get the point(x,y) in image
    private Color pickColor(int x, int y, BufferedImage bi){
```

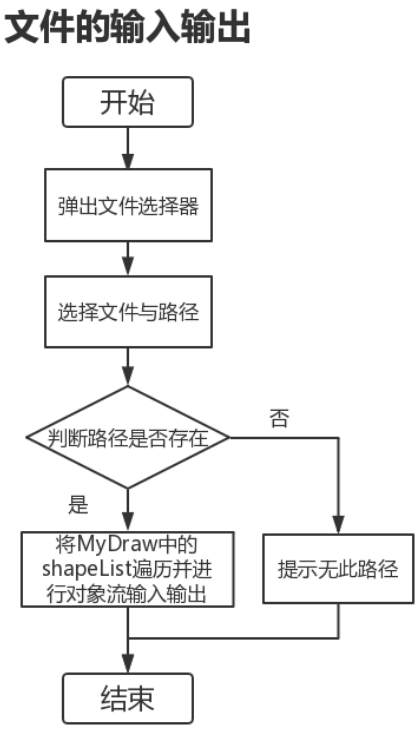
```
        int c=bi.getRGB(x, y);
        Color bicolor=new Color(c);
        return bicolor;
    }

    //To judge whether the point(x,y) is in the JPanel or not
    private boolean isInsideImage(int x, int y){
        if(x>=0 && x<500 && y>=0 && y<330){
            return true;
        } else{
            return false;
        }
    }
}
```

7.1.2 MyPaint 类

```
/*
    This is the main frame. I admit I copied plenty of codes from Internet in
MyPaint.java. However I added more function and some components to make it
stronger
    and make it fit with my whole project. To be honest, I just use some UI
components and change the frame of the project from Internet.

    from http://blog.csdn.net/bluesky_usc/article/details/53128460
    I must thanks "BlueSky_USC", whose blog teach me a lot.
 */

public class MyPaint extends JFrame{
    private JButton btt;                    //Current color  button
    private ButtonListener bl;               //Color button listener
    private RadioButtonListener rbl;          //Function button listener
    private MyDraw md;                      //Canvas
    private JMenuBar bar;
    private JMenu menuFile;
    private JMenu menuOperate;
    private JMenu menuHelp;
    private JMenuItem newFile;
    private JMenuItem openFile;
    private JMenuItem saveFile;
    private JMenuItem saveimgFile;
    private JMenuItem revoke;
    private JMenuItem voke;
    private JMenuItem help;
```

```
private JMenuItem about;
private JPanel panelCentral;
private JPanel panelLeft;
private ButtonGroup bg;
private LineWidthChooser lineWidthChooser;
private JPanel panelDown;
private JPanel panelDownLeft;
private JPanel panelDownRight;
private JPanel panelDownChild;

//Init Frame
public void initFrame(){
    JPanel panel = new JPanel();
    btt = new JButton();
    md = new MyDraw(btt);
    bl = new ButtonListener(this.md, btt);
    rbl = new RadioButtonListener(this.md);
    bar= new JMenuBar();
    menuFile = new JMenu("文件");
    menuOperate = new JMenu("编辑");
    menuHelp = new JMenu("帮助");
    panelCentral = new JPanel();
    panelLeft = new JPanel();
    bg = new ButtonGroup();
    lineWidthChooser = new LineWidthChooser(md);
    panelDown = new JPanel();
    panelDownChild = new JPanel();
    panelDownLeft = new JPanel();
    panelDownRight = new JPanel();

    //Set properties
    this.setSize(600,500);
    this.setDefaultCloseOperation(3);
    this.setTitle("Paint");
    this.setLocationRelativeTo(null);
    panel.setLayout(new BorderLayout());
    this.add(panel);

    //Add menu bar
    panel.add(bar,BorderLayout.NORTH);

    //Add three menu into menu bar
    bar.add(menuFile);
```

```
        bar.add(menuOperate);
        bar.add(menuHelp);
```




```
        //Add four menu item into the first menu(File)
        newFile= new JMenuItem("新建文件");

newFile.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_N,Event.CTRL_MASK)
);        //Add shortcutkeys
        newFile.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int value=JOptionPane.showConfirmDialog(null, "是否需要保存当前文
件？", "提示信息", 0);
                if(value==0){
                    saveFile(0);
                }
                if(value==1){
                    md.reset();
                }
            }
        });
```

```
        openFile = new JMenuItem("打开工程文件");

openFile.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_O,Event.CTRL_MASK
));        //Add shortcutkeys
        openFile.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                int value=JOptionPane.showConfirmDialog(null, "是否需要保存当前文
件？", "提示信息", 0);
                if(value==0){
                    saveFile(0);
                }
                if(value==1){
                    try {
                        //Show file chooser to let user select file
                        JFileChooser chooser = new JFileChooser();
                        chooser.showOpenDialog(null);
                        File file =chooser.getSelectedFile();
                        //if file doesn't exist, show dialog "NO FILE"
                        if(file==null){
                            JOptionPane.showMessageDialog(null, "没有选择文件");
                            md.repaint();
                        }
                        else {
                            //delete all shapes in Shapelist
                            md.getShapeList().clear();
                            md.repaint();
                            FileInputStream fis = new FileInputStream(file);
                            ObjectInputStream ois = new ObjectInputStream(fis);
                            //Get objects from file
                            ArrayList<MyShape> list
=(ArrayList<MyShape>)ois.readObject();
                            //Add objects into list
                            for (int i = 0; i <list.size(); i++) {
                                MyShape shape=(MyShape)list.get(i);
                                md.getShapeList().add(shape);
                                //Repaint to show the painting
                                md.repaint();
                            }
                            ois.close();
                        }
                    } catch (Exception e1) {
                        e1.printStackTrace();
                    }
```

```
            }
        }
    });

    saveFile = new JMenuItem("保存工程文件");

saveFile.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_S,Event.CTRL_MASK
));       //Add shortcutkeys
    saveFile.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            saveFile(0);
        }         //0 means save project (Objects)
    });

    saveimgFile = new JMenuItem("保存图片文件");

saveimgFile.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_S,Event.SHIFT_
MASK+Event.CTRL_MASK));        //Add shortcutkeys
    saveimgFile.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            saveFile(1);
        }         //1 means save a picture
    });

    menuFile.add(newFile);
    menuFile.add(openFile);
    menuFile.add(saveFile);
    menuFile.add(saveimgFile);

    //Add two menu item into the second menu(Edit)
    revoke = new JMenuItem("向前一步");

revoke.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_Z,Event.CTRL_MASK))
;       //Add shortcutkeys
    revoke.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            md.revoke();
        }      //Call method revoke in MyDraw
    });

    voke = new JMenuItem("向后一步");
```

```
voke.setAccelerator(KeyStroke .getKeyStroke(KeyEvent.VK_Z,Event.SHIFT_MASK+E
vent.CTRL_MASK));        //Add shortcutkeys
        voke.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                md.voke();
            }      //Call method voke in MyDraw
        });

        menuOperate.add(revoke);
        menuOperate.add(voke);

        //Add two menu item into the third menu(Help)
        help = new JMenuItem("使用说明");

help.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_H,Event.CTRL_MASK));
//Add shortcutkeys
        help.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                //Show help dialog
                JOptionPane.showMessageDialog(null,
                        "橡皮擦：擦除图片的一部分并用背景色替换。\n" +
                        "铅笔：使用铅笔在画布上根据鼠标的移动作图。\n" +
                        "喷漆：对画布上的区域使用喷枪着色。\n" +
                        "刷子：使用刷子在画布上根据鼠标的移动作图。\n" +
                        "填充：点击画布上的某个区域，并用当前色对其填充。\n" +
                        "图形：插入现有的图形：矩形，圆形，椭圆，圆角矩形，多边形等\n" +
                        "多边形:可以根据自己的需要做出想要的形状，单击左键为选定下一个顶
点，双击左键将自动返回起始顶点完成该次画图。\n" +
                        "字体：根据自己的需求添加文字。\n" +
                        "粗细：为所选的工具选择宽度。\n" +
                        "移动：移动所选图形的位置。\n" +
                        "取色：在图片中选择颜色并将其用于绘");
            }
        });

        about = new JMenuItem("关于");

about.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK_A,Event.CTRL_MASK));
//Add shortcutkeys
        about.addActionListener(new ActionListener() {
            @Override
```

```
            public void actionPerformed(ActionEvent e) {
                //Show about dialog
                JOptionPane.showMessageDialog(null, "计算机1508班 李天志 邓翼 刘卓成");
            }
        });

        menuHelp.add(help);
        menuHelp.add(about);


        //Add central JPanel
        panelCentral.setLayout(new FlowLayout(FlowLayout.LEFT,10,10));
        panel.setBackground(Color.gray);
        panel.add(panelCentral);
        //Set MyDraw's properties
        md.setBackground(Color.WHITE);
        md.setPreferredSize(new Dimension(500,330));
        //Add MyDraw into central JPanel
        panelCentral.add(md);

        //Add left JPanel (Function RadioButton JPanel)
        panelLeft.setLayout(new FlowLayout(FlowLayout.LEFT,0,0));
        panelLeft.setBackground(new Color(235,233,238));
        panelLeft.setPreferredSize(new Dimension(60, 0));
        //Add RatioButtons in the central left JPanel
        for(int i=2;i<16;i++){
            //Add four different images to each RadioBuuton
            ImageIcon img1 = new ImageIcon("images/draw"+i+".jpg");
//Default image
            ImageIcon img2 = new ImageIcon("images/draw"+i+"-1"+".jpg");
//Image when mouse moves upon button
            ImageIcon img3 = new ImageIcon("images/draw"+i+"-2"+".jpg");
//Image when mouse pressed
            ImageIcon img4 = new ImageIcon("images/draw"+i+"-3"+".jpg");
//Image when selected
            JRadioButton jrb = new JRadioButton();
            jrb.setActionCommand("pic"+i);
            //Set default function is draw Line
            if(i==10){
                jrb.setSelected(true);
            }
            bg.add(jrb);          //add to buttonGroup
            jrb.addActionListener(rbl);
```

```
            jrb.setIcon(img1);
            jrb.setRolloverIcon(img2);
            jrb.setPressedIcon(img3);
            jrb.setSelectedIcon(img4);
            jrb.setBorder(null);          //Set no Border, since we added the
border in image. :)
            panelLeft.add(jrb);
        };

        //Add line width chooser into left JPanel
        panelLeft.add(new JLabel("选择线宽"));
        panelLeft.add(lineWidthChooser.getBox());

        //Add down JPanel
        panelDown.setBackground(Color.gray);
        panelDown.setLayout(null);
        panelDown.setPreferredSize(new Dimension(0, 80));

        //Add down child JPanel
        panelDownChild.setLayout(null);
        panelDownChild.setBounds(15, 10, 300, 60);
        panelDownChild.setBackground(Color.green);

        //Add down left JPanel (for current color)
        panelDownLeft.setLayout(null);
        panelDownLeft.setBackground(Color.white);
        panelDownLeft.setBounds(0, 0, 60, 60);

        //Add down right JPanel (for choosing color)
        panelDownRight.setBackground(Color.magenta);
        panelDownRight.setLayout(new FlowLayout(FlowLayout.LEFT,0,0));
        panelDownRight.setBounds(60, 0, 240, 60);

        //Add child JPanel into parent JPanel and set position
        panel.add(panelLeft,BorderLayout.WEST);
        panel.add(panelDown, BorderLayout.SOUTH);
        panelDown.add(panelDownChild);
        panelDownChild.add(panelDownLeft);
        panelDownChild.add(panelDownRight);

        //Add Button to show current color
        btt.setBackground(Color.black);
        btt.setBounds(10, 10, 30, 30);
        panelDownLeft.add(btt);
```

```
        //Add onSet special effect
        BevelBorder  bb = new BevelBorder(0,Color.gray,Color.white);
//Set border for button
        BevelBorder  bb1 = new BevelBorder(1,Color.gray,Color.white);
        panelDownLeft.setBorder(bb);
        btt.setBorder(bb1);

        //Add right side color
        Color colors[]
={Color.BLUE,Color.red,Color.black,Color.ORANGE,Color.gray,

Color.CYAN,Color.GREEN,Color.YELLOW,Color.PINK,Color.magenta,new
Color(234,45,78),new Color(67,123,9)};
        for(int i=0;i<12;i++){
            JButton bt = new JButton();
            bt.setBackground(colors[i]);
            bt.setBorder(bb);
            bt.setPreferredSize(new Dimension(40,30));
            //Add button listener for each button
            bt.addActionListener(bl);
            panelDownRight.add(bt);
        }
        this.setVisible(true);
    }

    //Save file into project file or image file
    public void saveFile(int type){
        //Get file (including name and path)
        JFileChooser chooser = new JFileChooser();
        chooser.showSaveDialog(null);
        File file =chooser.getSelectedFile();

        if(file==null){
            JOptionPane.showMessageDialog(null, "没有选择文件");
        }else {

            try {
                if(type==1){
                    //Save as an jpg file
                    File file1 = new File(file.getAbsoluteFile()+".jpg");
//Add ".jpg"
                    md.savePic(file1);
                } else{
```

```
                FileOutputStream fis = new FileOutputStream(file);
                ObjectOutputStream oos = new ObjectOutputStream(fis);
                //Add all shapes in list into output stream
                oos.writeObject(md.getShapeList());
                oos.close();
            };
            JOptionPane.showMessageDialog(null, "保存成功！");
        } catch (Exception e1) {
            e1.printStackTrace();
        }
    }
}
}
```

7.1.3 MyShape 类

```
//Abstract parent class
public abstract class MyShape extends JPanel{
    public int x1,y1,x2,y2;
    public Color color;
    public int width;

    //Abstract method Draw
    public abstract void Draw(Graphics g);

    //For moving operation
    public void moveWhere(int x,int y){
        this.x1 += x;
        this.y1 += y;
    }

    //To judge whether be selecte or not
    public boolean containsPix(int x, int y){
        if(x>=x1 && x<=x1+x2 && y>=y1 && y<=y1+y2){
            return true;
        } else{
            return false;
        }
    }

    //Add points for MULTIPOINTS shape
    public void addPoint(int x1, int y1) {

    }
```

```
    //Upgrade points position
    public void upgradePoint(int x1, int y1, int x2,int y2){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
    }
}
```

7.1.4 MyPenccil 类

```
public class MyPencil extends MyShape {
    public ArrayList<Point> points;        //The set of points pencil moved

    public MyPencil (int x1, int y1, Color c, int width){
        this.x1=x1;
        this.y1=y1;
        this.color=c;
        this.width=width;
        this.points = new ArrayList<Point>();
        Point temppoint = new Point(x1,y1);
        this.points.add(temppoint);
    }

    //add point should be painted into set
    public void addPoint(int x2, int y2){
        Point temppoint = new Point(x2,y2);
        points.add(temppoint);
    }

    //Make every point in set move  (x,y)
    public void moveWhere(int mx,int my){
        for(int i=0; i<points.size(); i++){
            points.get(i).move(points.get(i).x+mx,points.get(i).y+my);
        }
    }

    //Override Draw
    public void Draw(Graphics g){
        Graphics2D g1 = (Graphics2D)g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        int tempx1,tempx2,tempy1,tempy2;
```

```
        tempx1 = points.get(0).x;
        tempy1 = points.get(0).y;
        //draw a line between two jacent points
        for(int i=1;i < points.size(); i++){
            tempx2 = points.get(i).x;
            tempy2 = points.get(i).y;
            g1.drawLine(tempx1, tempy1, tempx2, tempy2);
            tempx1 = tempx2;
            tempy1 = tempy2;
        }
    }
}
```

7.1.5 MySprayGun 类

```
public class MySprayGun extends MyShape {
    private Random r;                //To create random integer
    public ArrayList<Point> points;    //The set of points should be painted

    public MySprayGun(int x1, int y1, Color c, int width) {
        this.color = c;
        if(width <3){
            this.width=5;
        } else if(width <5){
            this.width=10;
        } else{
            this.width=15;
        }
        points = new ArrayList<Point>();
        Point temppoint = new Point(x1,y1);
        points.add(temppoint);
        r = new Random();
    }

    //Make every point in set move  (x,y)
    public void moveWhere(int mx,int my){
        for(int i=0; i<points.size(); i++){
            points.get(i).move(points.get(i).x+mx,points.get(i).y+my);
        }
    }

    //add point should be painted into set
    public void addPoint(int x2, int y2){
        for (int i = 0; i < 30; i++) {
```

```
            int xp=r.nextInt(2*width)-width;
            int yp=r.nextInt(2*width)-width;
            //create random point around the mouse position
            Point temppoint = new Point(x2+xp,y2+yp);
            points.add(temppoint);
        }
    }

    //Override Draw
    public void Draw(Graphics g){
        Graphics2D g1 = (Graphics2D)g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(1));
        int tempx, tempy;
        for(int i=0; i<points.size();i++){
            tempx = points.get(i).x;
            tempy = points.get(i).y;
            g1.drawLine(tempx, tempy, tempx, tempy);
        }
    }
)
```

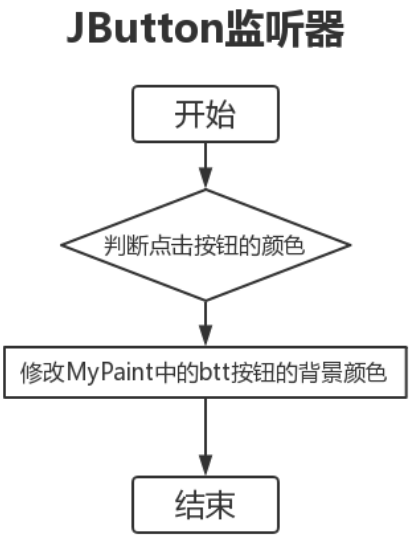
7.1.6 MyFill 类

```
public class MyFill extends MyShape {
    public ArrayList<Point> points;              //The set of points should
be painted

    public MyFill (int x1, int y1, Color c, int width){
        this.x1=x1;
        this.y1=y1;
        this.color=c;
        this.width=width;
        this.points = new ArrayList<Point>();
        Point temppoint = new Point(x1,y1);
        this.points.add(temppoint);
    }

    //Make every point in set move  (x,y)
    public void moveWhere(int mx,int my){
        for(int i=0; i<points.size(); i++){
            points.get(i).move(points.get(i).x+mx,points.get(i).y+my);
        }
    }
```

```
    //add point should be painted into set
    public void addPoint(int x2, int y2){
        Point temppoint = new Point(x2,y2);
        points.add(temppoint);
    }

    //Override Draw
    public void Draw(Graphics g){
        Graphics2D g1 = (Graphics2D)g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        //draw a line between two jacent points
        int tempx=points.get(0).x,tempy=points.get(0).y,curx,cury;
        for(int i=0;i < points.size(); i++){
            curx = points.get(i).x;
            cury = points.get(i).y;
            g1.drawLine(tempx,tempy,curx,cury);
            tempx=curx;
            tempy=cury;
        }
    }
}
```

7.1.7 ButtonListener 类

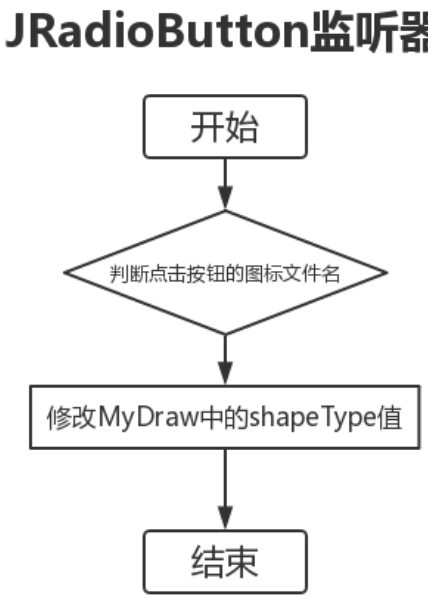
```
public class ButtonListener implements ActionListener {
    Color c;
    MyDraw md;
    JButton cb;                 //This is the current Color button in the
lower left corner

    public ButtonListener(MyDraw mydraw, JButton colorButton) {
        this.md = mydraw;
        this.cb = colorButton;
    }
```




```
    public void actionPerformed(ActionEvent e) {
        JButton bt = (JButton) e.getSource();
        c = bt.getBackground();     //to get the clicked button's color
        md.setColor(c);            //set color in MyDraw
        cb.setBackground(c);       //set the current Color button's color
    }
}
```

7.1.8 RadioButtonListener 类

```
public class RadioButtonListener implements ActionListener {
    int type;
    MyDraw md;

    public RadioButtonListener(MyDraw md) {
        this.md = md;
    }
```



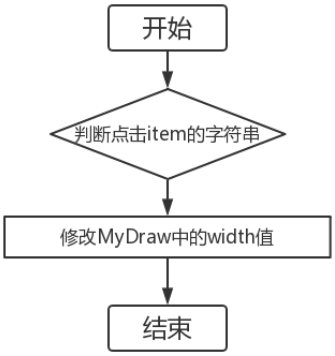

```
//Different radio buttons change the shapeType in different modes
public void actionPerformed(ActionEvent e) {
    JRadioButton rbt = (JRadioButton) e.getSource();
    String command = rbt.getActionCommand();
    if("pic10".equals(command)){
        type=0;
        md.setType(0);
    } else if("pic12".equals(command)){
        type=1;
        md.setType(1);
    } else if("pic15".equals(command)){
        type=2;
        md.setType(2);
    } else if("pic14".equals(command)){
        type=3;
        md.setType(3);
    } else if("pic4".equals(command)){
        type=4;
        md.setType(4);
    } else if("pic7".equals(command)){
        type=5;
        md.setType(5);
    } else if("pic6".equals(command)){
        type=6;
```

```
            md.setType(6);
        } else if("pic8".equals(command)){
            type=7;
            md.setType(7);
        } else if("pic2".equals(command)){
            type=8;
            md.setType(8);
        } else if("pic13".equals(command)){
            type = 9;
            md.setType(9);
        } else if("pic3".equals(command)){
            type = 10;
            md.setType(10);
        } else if("pic9".equals(command)){
            type = 11;
            md.setType(11);
            JFontChooser one = new JFontChooser(new Font("宋体", Font.PLAIN,
24), new Color(0,0,0));
            one.showDialog(null,500,200);
            //获取选择的字体
            Font font=one.getSelectedfont();
            //获取选择的颜色
            Color color=one.getSelectedcolor();
            if(font!=null&&color!=null){
                md.setFont(font);
                md.setColor(color);
                md.setSentences(one.getStr());
            }
        } else if("pic5".equals(command)){
            type = 12;
            md.setType(12);
        } else if("pic11".equals(command)){
            type = 13;
            md.setType(13);
        }
    }
}
```

7.1.9 LineWidthChooser 类

```
/* This is worked with from my classmate */
```

JComboBox监听器




```
public class LineWidthChooser extends JComboBox{
   private String[] s = {"细","中","粗"};           //Three lengths of line
   private ImageIcon[] icons = new ImageIcon[3];      //Three icons
   private JComboBox combo;
   private MyDraw md;
   //Two useful classes in this line width chooser
   class ItemObj
   {
      //it contains a name and an icon
      String name;
      ImageIcon icon;
      public ItemObj(String name, ImageIcon icon){
         this.name = name;
         this.icon = icon;
      }
   }

   class ACellRenderer extends JLabel implements ListCellRenderer
   {
      ACellRenderer()
      {
         setOpaque(true);
      }
      public Component getListCellRendererComponent(JList list,
                                                    Object value,
                                                    int index,
                                                    boolean isSelected,
                                                    boolean cellHasFocus)
      {
         if (value != null)
```

```
            {
                setText(((ItemObj)value).name);
                setIcon(((ItemObj)value).icon);
            }

            if (isSelected) {
                setBackground(list.getSelectionBackground());
                setForeground(list.getSelectionForeground());
            }
            else {
                setBackground(list.getBackground());
                setForeground(list.getForeground());
            }

            return this;
        }
    }

    public LineWidthChooser(MyDraw md) {
        this.md = md;
        ItemObj[] obj = new ItemObj[3];
        //Get three icons
        for(int i=0; i < 3; i++)
        {
            icons[i] = new ImageIcon("images/lineSize_"+(i+1)+".gif");
            obj[i] = new ItemObj(s[i],icons[i]);
        }
        combo = new JComboBox(obj);    //use the array of ItemObj to fill the
new JComboBox
        combo.addItemListener(new ItemListener() {
            @Override
            public void itemStateChanged(ItemEvent e) {
                ItemObj selectedObj = (ItemObj) e.getItem();
                String width = selectedObj.name;          //use selected Item's
name to change the width in MyDraw
                if(width.equals("细")){
                    md.setWidth(1);
                } else if(width.equals("中")){
                    md.setWidth(3);
                } else if(width.equals("粗")){
                    md.setWidth(5);
                }
            }
        });
```

```
        combo.setRenderer(new ACellRenderer());
        combo.setSize(50, 10);
        combo.setMaximumRowCount(3);
        combo.setVisible(true);
    }

    //get the JComboBox
    public JComboBox getBox(){
        return combo;
    }

}
```

7.1.10 Main 类

```
public class Main {
    public static void main(String[] args) {
        MyPaint paint = new MyPaint();
        paint.initFrame();
    }
}
```

7.2 邓翼代码：

7.2.1 MyWord 类

```
public class MyWord extends MyShape {
    private Font font;          //Word's font
    private String s;           //Word
    private int size;           //Word's size

    public MyWord(int x1, int y1,String s, Font f, Color c) {
        this.x1 = x1;
        this.y1 = y1;
        this.s = s;
        this.font = f;
        this.color = c;
        this.size = f.getSize();
    }

    //To judge if (x,y) is in the limited zone
```

```
    public boolean containsPix(int x, int y){
        if(x>=x1 && x<=x1+s.length()*size && y>=y1 && y<=y1-size){
            return true;
        } else{
            return false;
        }
    }

    //Override Draw
    public void Draw(Graphics g){
        Graphics2D g1 = (Graphics2D) g;
        g1.setPaint(color);
        g1.setFont(font);
        g1.drawString(s,x1,y1);
    }
}
```

7.2.2 MyOval 类

```
public class MyOval extends MyShape{

    public MyOval(int x1, int y1, int x2, int y2, Color color , int width){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
        this.color=color;
        this.width=width;
    }

    //Override Draw
    public void Draw(Graphics g) {
        Graphics2D g1 = (Graphics2D) g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        g1.drawOval(x1, y1, x2, y2);
    }
}
```

7.2.3 MyCircle 类

```
public class MyCircle extends MyShape {
    public MyCircle(int x1, int y1, int x2,int y2, Color color , int width){
        this.x1=x1;
        this.y1=y1;
```

```
        this.x2=x2;
        this.y2=y2;
        this.color=color;
        this.width=width;
    }

    //Upgrade points position
    public void upgradePoint(int x1, int y1, int x2,int y2){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
    }

    //Override Draw
    public void Draw(Graphics g) {
        Graphics2D g1 = (Graphics2D) g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        g1.drawOval(x1, y1, x2, y2);
    }
}
```

7.3 刘卓成代码：

7.3.1 MyLine 类

```
public class MyLine extends MyShape{

    public MyLine(int x1, int y1, int x2, int y2, Color color, int width){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
        this.color=color;
        this.width=width;
    }

    //Make two vertex move (x,y)
    public void moveWhere(int x,int y){
        this.x1 += x;
        this.y1 += y;
        this.x2 += x;
```

```
        this.y2 += y;
    }

    //To judge if (x,y) is upon the segment between (x1,y1) and (x2,y2)
    public boolean containsPix(int x, int y){
        if(x>=x1 && x<=x2 && y>=y1 && y<=y2 && (x-x1)*(y2-y)== (y-y1)*(x2-x)){
            return true;
        } else{
            return false;
        }
    }

    //Override Draw
    public void Draw(Graphics g) {
        Graphics2D g1 = (Graphics2D)g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        g1.drawLine(x1, y1, x2, y2);
    }
}
```

7.3.2 MyRectangle 类

```
public class MyRectangle extends MyShape{

    public MyRectangle(int x1,int y1,int x2,int y2,Color color,int width){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
        this.color=color;
        this.width=width;
    }

    //Override Draw
    public void Draw(Graphics g) {
        Graphics2D g1 = (Graphics2D)g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        g1.drawRect(x1, y1, x2, y2);
    }

}
```

7.3.3 MyRoundRectangle 类

```
public class MyRoundRectangle extends MyShape{
    public int arcWidth,arcHeight;            //arcWidth and arcHeight are
properties for arc

    public MyRoundRectangle(int x1, int y1, int x2, int y2, int arcWidth, int
arcHeight, Color color , int width){
        this.x1=x1;
        this.y1=y1;
        this.x2=x2;
        this.y2=y2;
        this.arcWidth=arcWidth;
        this.arcHeight=arcHeight;
        this.color=color;
        this.width=width;
    }

    //Override Draw
    public void Draw(Graphics g) {
        Graphics2D g1 = (Graphics2D) g;
        g1.setColor(this.color);
        g1.setStroke(new BasicStroke(width));
        g1.drawRoundRect(x1, y1, x2, y2, this.arcWidth, this.arcHeight);
    }

}
```

7.4 网络代码

JFontChooser 类

```
/*
   Copied from http://blog.csdn.net/tangcaijun/article/details/8372943
   However, I make some change in order to fit my app.
 */

public class JFontChooser extends JPanel {

    // 设置界面风格
    {
        try {
```

```
javax.swing.UIManager.setLookAndFeel(javax.swing.UIManager.getSystemLookAndF
eelClassName());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    //[start] 定义变量
    private String current_fontName = "宋体";//当前的字体名称,默认宋体.
    private String Str = "";//展示的文字
    private int current_fontStyle = Font.PLAIN;//当前的字样,默认常规.
    private int current_fontSize = 9;//当前字体大小,默认 9 号.
    private Color current_color = Color.BLACK;//当前字色,默认黑色.
    private JDialog dialog;                                    //用于显示模态的窗体
    private JLabel lblString;
    private JLabel lblFont;                                   //选择字体的 LBL
    private JLabel lblStyle;                                  //选择字型的 LBL
    private JLabel lblSize;                                   //选择字大小的 LBL
    private JLabel lblColor;                                  //选择 Color 的 label
    private JLabel otherColor;                                //其它颜色
    private JTextField txtFont;                                   //显示选择字体的
TEXT
    private JTextField txtStyle;                                  //显示选择字型的
TEXT
    private JTextField txtSize;                                   //显示选择字大小的
TEXT
    private JTextField showTF;                                    //展示框（输入框）
    private JList lstFont;                                   //选择字体的列表.
    private JList lstStyle;                                  //选择字型的列表.
    private JList lstSize;                                   //选择字体大小的列表.
    private JComboBox cbColor;                                    //选择 Color 的下拉
框.
    private JButton ok, cancel;                                //"确定","取消"按钮.
    private JScrollPane spFont;
    private JScrollPane spSize;
    private JPanel showPan;                                   //显示框.
    private Map sizeMap;                                     //字号映射表.
    private Map colorMap;                                    //字着色映射表.
    private Font selectedfont;                                //用户选择的字体
    private Color selectedcolor;                               //用户选择的颜色
    //[end]

    //无参初始化
    public JFontChooser() {
```

```
        this.selectedfont = null;
        this.selectedcolor = null;
            /* 初始化界面 */
        init(null, null);
    }

    //重载构造，有参的初始化 用于初始化字体界面
    public JFontChooser(Font font, Color color) {
        if (font != null) {
            this.selectedfont = font;
            this.selectedcolor = color;
            this.current_fontName = font.getName();
            this.current_fontSize = font.getSize();
            this.current_fontStyle = font.getStyle();
            this.current_color = color;
            /* 初始化界面 */
            init(font, color);
        } else {
            JOptionPane.showMessageDialog(this, "没有被选择的控件", "错误",
JOptionPane.ERROR_MESSAGE);
        }
    }

    //可供外部调用的方法
    public Font getSelectedfont() {
        return selectedfont;
    }

    public void setSelectedfont(Font selectedfont) {
        this.selectedfont = selectedfont;
    }

    public Color getSelectedcolor() {
        return selectedcolor;
    }

    public String getStr() {
        return Str;
    }

    public void setSelectedcolor(Color selectedcolor) {
        this.selectedcolor = selectedcolor;
    }
```

```
    /*初始化界面*/
    private void init(Font font, Color color) {
        //实例化变量
        lblString = new JLabel("输入文字");
        lblFont = new JLabel("字体:");
        lblStyle = new JLabel("字型:");
        lblSize = new JLabel("大小:");
        lblColor = new JLabel("颜色:");
        otherColor = new JLabel("<html><U>其它颜色</U></html>");
        txtFont = new JTextField("宋体");
        txtStyle = new JTextField("常规");
        txtSize = new JTextField("9");

        //取得当前环境可用字体.
        GraphicsEnvironment ge =
GraphicsEnvironment.getLocalGraphicsEnvironment();
        String[] fontNames = ge.getAvailableFontFamilyNames();

        lstFont = new JList(fontNames);

        //字形.
        lstStyle = new JList(new String[]{"常规", "粗体", "斜体", "粗斜体"});

        //字号.
        String[] sizeStr = new String[]{
                "8", "9", "10", "11", "12", "14", "16", "18", "20", "22", "24",
"26", "28", "36", "48", "72", "初号", "小初",
                "一号", "小一", "二号", "小二", "三号", "小三", "四号", "小四", "五号
", "小五", "六号", "小六", "七号", "八号"
        };
        int sizeVal[] = {8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 36,
48, 72, 42, 36, 26, 24, 22, 18, 16, 15, 14, 12, 11, 9, 8, 7, 6, 5};
        sizeMap = new HashMap();
        for (int i = 0; i < sizeStr.length; ++i) {
            sizeMap.put(sizeStr[i], sizeVal[i]);
        }
        lstSize = new JList(sizeStr);
        spFont = new JScrollPane(lstFont);
        spSize = new JScrollPane(lstSize);

        //颜色
        String[] colorStr = new String[]{
                "黑色", "蓝色", "青色", "深灰", "灰色", "绿色", "浅灰", "洋红", "桔黄
", "粉红", "红色", "白色", "黄色"
```

```
        };
        Color[] colorVal = new Color[]{
                Color.BLACK, Color.BLUE, Color.CYAN, Color.DARK_GRAY,
Color.GRAY, Color.GREEN, Color.LIGHT_GRAY, Color.MAGENTA, Color.ORANGE,
Color.PINK, Color.RED, Color.WHITE, Color.YELLOW
        };
        colorMap = new HashMap();
        for (int i = 0; i < colorStr.length; i++) {
            colorMap.put(colorStr[i], colorVal[i]);
        }
        cbColor = new JComboBox(colorStr);
        showPan = new JPanel();
        ok = new JButton("确定");
        cancel = new JButton("取消");


        //布局控件
        //字体框
        this.setLayout(null);   //不用布局管理器
        add(lblFont);
        lblFont.setBounds(12, 10, 30, 20);
        txtFont.setEditable(false);
        add(txtFont);
        txtFont.setBounds(10, 30, 155, 20);
        txtFont.setText("宋体");
        lstFont.setSelectedValue("宋体", true);
        if (font != null) {
            txtFont.setText(font.getName());
            lstFont.setSelectedValue(font.getName(), true);
        }

        add(spFont);
        spFont.setBounds(10, 50, 155, 100);

        //样式
        add(lblStyle);
        lblStyle.setBounds(175, 10, 30, 20);
        txtStyle.setEditable(false);
        add(txtStyle);
        txtStyle.setBounds(175, 30, 130, 20);

lstStyle.setBorder(javax.swing.BorderFactory.createLineBorder(Color.gray));
        add(lstStyle);
        lstStyle.setBounds(175, 50, 130, 100);
```

```
txtStyle.setText("常规"); //初始化为默认的样式
lstStyle.setSelectedValue("常规", true);   //初始化为默认的样式
if (font != null) {
    lstStyle.setSelectedIndex(font.getStyle()); //初始化样式 list
    if (font.getStyle() == 0) {
        txtStyle.setText("常规");
    } else if (font.getStyle() == 1) {
        txtStyle.setText("粗体");
    } else if (font.getStyle() == 2) {
        txtStyle.setText("斜体");
    } else if (font.getStyle() == 3) {
        txtStyle.setText("粗斜体");
    }
}


//大小
add(lblSize);
lblSize.setBounds(320, 10, 30, 20);
txtSize.setEditable(false);
add(txtSize);
txtSize.setBounds(320, 30, 60, 20);
add(spSize);
spSize.setBounds(320, 50, 60, 100);
lstSize.setSelectedValue("9", false);
txtSize.setText("9");
if (font != null) {
    lstSize.setSelectedValue(Integer.toString(font.getSize()), false);
    txtSize.setText(Integer.toString(font.getSize()));
}

//颜色
add(lblColor);
lblColor.setBounds(18, 220, 30, 20);
cbColor.setBounds(18, 245, 100, 22);
cbColor.setMaximumRowCount(5);
add(cbColor);
otherColor.setForeground(Color.blue);
otherColor.setBounds(130, 245, 60, 22);
otherColor.setCursor(new Cursor(Cursor.HAND_CURSOR));
add(otherColor);

//展示框
showTF = new JTextField();
```

```
        showTF.setEnabled(true);
        showTF.setFont(new Font(current_fontName, current_fontStyle,
current_fontSize));
        showTF.setBounds(10, 10, 300, 50);
        showTF.setHorizontalAlignment(JTextField.CENTER);
        showTF.setText("");
        showTF.setBackground(Color.white);
        showTF.setEditable(true);
        showPan.setBorder(javax.swing.BorderFactory.createTitledBorder("输入文
字"));
        add(showPan);
        showPan.setBounds(13, 150, 370, 80);
        showPan.setLayout(new BorderLayout());
        showPan.add(showTF);
        if (font != null) {
            showTF.setFont(font); // 设置示例中的文字格式
        }
        if (font != null) {
            showTF.setForeground(color);
        }

        //确定和取消按钮
        add(ok);
        ok.setBounds(230, 245, 60, 20);
        add(cancel);
        cancel.setBounds(300, 245, 60, 20);
        //布局控件_结束

        //listener.....
        /*用户选择字体*/
        lstFont.addListSelectionListener(new ListSelectionListener() {
            public void valueChanged(ListSelectionEvent e) {
                current_fontName = (String) lstFont.getSelectedValue();
                txtFont.setText(current_fontName);
                showTF.setFont(new Font(current_fontName, current_fontStyle,
current_fontSize));
            }
        });

        /*用户选择字型*/
        lstStyle.addListSelectionListener(new ListSelectionListener() {
            public void valueChanged(ListSelectionEvent e) {
                String value = (String) ((JList)
e.getSource()).getSelectedValue();
```

```
            if (value.equals("常规")) {
                current_fontStyle = Font.PLAIN;
            }
            if (value.equals("斜体")) {
                current_fontStyle = Font.ITALIC;
            }
            if (value.equals("粗体")) {
                current_fontStyle = Font.BOLD;
            }
            if (value.equals("粗斜体")) {
                current_fontStyle = Font.BOLD | Font.ITALIC;
            }
            txtStyle.setText(value);
            showTF.setFont(new Font(current_fontName, current_fontStyle,
current_fontSize));
        }
    });

    /*用户选择字体大小*/
    lstSize.addListSelectionListener(new ListSelectionListener() {
        public void valueChanged(ListSelectionEvent e) {
            current_fontSize = (Integer)
sizeMap.get(lstSize.getSelectedValue());
            txtSize.setText(String.valueOf(current_fontSize));
            showTF.setFont(new Font(current_fontName, current_fontStyle,
current_fontSize));
        }
    });

    /*用户选择字体颜色*/
    cbColor.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            current_color = (Color)
colorMap.get(cbColor.getSelectedItem());
            showTF.setForeground(current_color);
        }
    });
    /*其它颜色*/
    otherColor.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
            Color col_temp = new JColorChooser().showDialog(null, null,
Color.pink);
            if (col_temp != null) {
```

```
                current_color = col_temp;
                showTF.setForeground(current_color);
                super.mouseClicked(e);
            }
        }
    });
    /*用户确定*/
    ok.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            /*用户用户选择的字体设置*/
            setSelectedfont(new Font(current_fontName, current_fontStyle,
current_fontSize));
             /*用户用户选择的颜色设置*/
            setSelectedcolor(current_color);
            Str = showTF.getText();
            dialog.dispose();
            dialog = null;
        }
    });

    /*用户取消*/
    cancel.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            dialog.dispose();
            dialog = null;
        }
    });
}

/*显示字体选择器对话框(x,y 表示窗体的启动位置)*/
public void showDialog(Frame parent, int x, int y) {
    String title = "字体";
    dialog = new JDialog(parent, title, true);
    dialog.add(this);
    dialog.setResizable(false);
    dialog.setSize(400, 310);
    //设置接界面的启动位置
    dialog.setLocation(x, y);
    dialog.addWindowListener(new WindowAdapter() {

        /*窗体关闭时调用*/
        public void windowClosing(WindowEvent e) {
            dialog.removeAll();
            dialog.dispose();
```

```
                dialog = null;
            }
        });
        dialog.setVisible(true);
    }

}
```

八、JAVA 课程学习心得与改进意见

李天志：

这次 Java 大作业总的来说还是收获很多的，无论是从界面还是绘图，从重绘到监听，都得到了很好的训练。

首先，实践一个项目可以更好更快的去提升自己的代码能力。有目标，有问题的去查找资料，翻阅书籍是非常高效的学习方法，比如在这次大作业中，期初我并不知道如何能在鼠标拖拽的时候让图形不断变化，并且能够不在画板上留痕迹，为此我看了很多网页，首先是查代码，查不到，然后一步步深入，最后一直看到了 swing 的画图机制，里面讲了许多东西，其中也简单提到了重绘机制，之后我又搜索 swing 重绘，然后知道了改写 paintComponent 函数再调用 repaint 的时候就可以根据自己来定义重绘。所以，我觉得目标学习法十分高效，以后在学习生活中也要注意学与用相结合。

第二，这次大作业让我对监听机制有了进一步的训练，我自己比较喜欢 app 开发，写过一些小 app，但是说实话我从来没有自己去 new 一个监听器，因为无论是安卓还是苹果添加 Button 的时候都会自动添加好监听器，你只要填写不同的函数名字进去就可以了。但是，在这次项目中，无论是 JButton 还是 JComboBox，亦或是 JMenu，所有的东西都需要自己添加监听器。虽然心态都要炸了，但还是好好学习了几个监听器的用法，并用在项目中，达到了预期的功能。

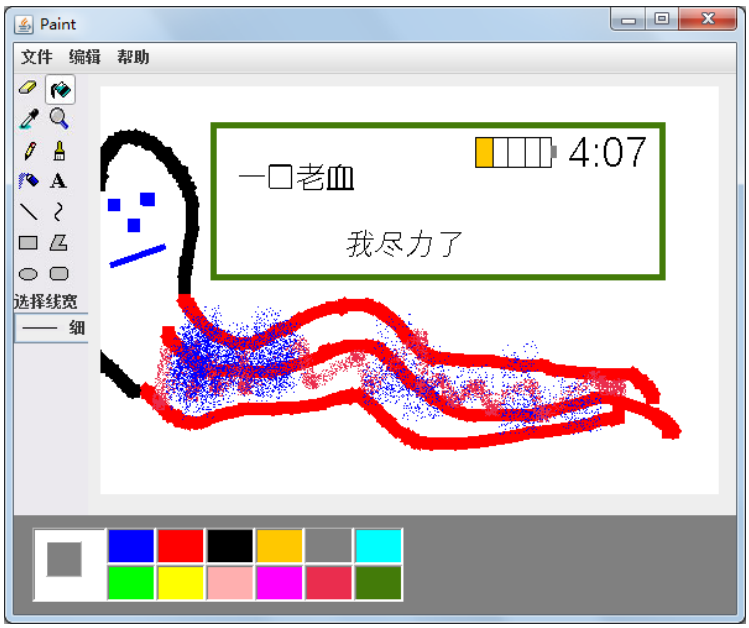
第三，我也对界面开发有了更大的锻炼，因为在手机开发的时候，都会有拖拽控件设置界面的 feature，但是这次 swing 开发中，并没有这样的一个

xml 或者 storyboard 来拖拽，所以一切的界面都考脑中想象，笔下画出来设计，然后手撸代码，对我来说简直是噩耗。虽然 MyPaint 主界面是来源于网络后我自己又加以修改的，但是在期初的字体选择上我还是练习了好多代码写界面的，虽然最后因为在网上看到了一个又美又强大的 JFontChooser 后把自己的 FontEditor 删除了，但是在写自己的字体选择器的时候得到了很多的锻炼。

第四，我要感谢网络的资源吧，在这样一个信息爆炸的年代，我们可以取其精华，学习到很多东西，从一开始看视频学习，看原理博文去思考逻辑，到自己去设计系统，然后代码一点点实现，这一路走来，网络上提供了许多知识给我。

最后，我想感谢老师和我的组员，感谢老师为我们上的一节节课，让我们学习到了很多关于 Java 的知识，也感谢我的组员给了我足够的信任与耐心，在遇到困难时，一起解决，一起讨论。

最后的最后，用一张图来结束我的心得体会。



至于改进的话，希望早日把前几个班考试的班级也解救出来，让他们也做项目，觉得 Java 这种课程，考试考得再高不自己写点东西也没用，还不如考得差点但写写代码、锻炼锻炼的效果好。

邓翼：

在做这个 java 大作业的过程中，我收获了很多，也学到了很多。它让我将所学的知识应用于实践，是理论与实际一次非常好地结合。这是我第一次接触这样的作业。刚开始时，我对于如何弄出一个程序，如何让程序画图都是非常陌生的。由于所学知识有限，为了使程序能够实现自己预先的构想，我通过上网和去图书馆找程序，比较好的流程图及功能模块，向组内同学学习请教，不断阅读修改代码使程序达到预期所要实现的目标。

在实际操作的时候同样遇到了很多问题。说起来编程的经验应该是有一点了，但是在调试的时候还是遇到了相当多的问题。很多的错误都很难体会，有的时候是输入的错误，这种错误还是比较容易找出来的。另外一些问题则是由于 java 自身带有相当多的方法定义，这些方法虽然不用我们自己再去编写了，但是需要花相当多的时间去记忆然后去运用，还有方法名等等。所以对我来说这真的是比 C 里简单几个文件的导入还要难上很多，还是觉得自己编的方法（函数）自己用的习惯一些。

通过这次大作业，我对这学期所学的 JAVA 有了一个全新的理解。对于面向对象和面向过程两种编程方式的不同有了更深刻的体会。面向对象的便捷与简练已经深入我心，以后编程首先想到的不是无脑用 c 语言，而是根据所遇到的问题来决定我所用的方式。学会一种全新的想法是我这次大作业最大的收获。

刘卓成：

经过一个学期的 JAVA 课程学习，现在虽说算不上什么专业的 JAVA 程序员，但我还是很有收获。因为有学习 C++的基础，对 JAVA 的学习过程并不算艰难，并且我觉得 JAVA 这门语言虽然很高级但并不复杂，很多功能都是可以直接调用的，不需要自己亲手去编。通过这学期 JAVA 的学习我对 OOP 思想也有了更深刻的认识，最后的小项目也让我体会到了团队合作的重要性。

自己需要改进的地方：没有多动手写一些小程序，对 JAVA 的语法还有些生涩，对 JAVA 的不同库的掌握需要增进，一旦对 JAVA 各种库的引用融会贯通，对较大程序的编写就能事半功倍。