**Java程序设计**

**——画图工具设计报告**

**姓名：**

**学号：11003004XX**

**学院：计算机学院**

**班级：1003104**

**摘要**

随着科学技术的不断提高，计算机科学日渐成熟，其强大的功能已为人们深刻认识，它已进入人类社会的各个领域并发挥着越来越重要的作用。 JAVA是一门很优秀的编程语言，具有面向对象、与平台无关、安全、稳定和多线程等特点，是目前软件设计中极为健壮的编程语言。JAVA不仅可以用来开发大型的应用程序。而且特别适合Internet的应用开发。JAVA确实具备“一次写成，处处运行”的特点，JAVA以成为网络时代最重要的编程语言之一。本报告中介绍了用JAVA语言实现画笔系统的全部过程。 本次课程设计是配合JAVA程序设计课程，帮助我们深入学习掌握JAVA语言，熟练运用这个工具来编写的。通过课程设计各个项目的综合训练，培养学生实际分析问题、编程和动手能力、提高学生的综合素质。本课程设计尝试使用一些较生动的示例和设计项目，激发学生学习兴趣，引导学生主动学习，正确理解、接受需要重点掌握的知识点，为参加项目开发及更深入学习做好准备。本次课程设计主要是用JAVA实现画笔系统，其系统主要完成以下功能：设计一个画图程序，能够实现画笔、画直线、画空心圆形、画实心圆形、画空心椭圆、画实心椭圆、画空心矩形、画实心矩形、书写文字、修改字体、修改该画笔的大小以及颜色。该画图程序还需具有橡皮擦功能。

**关键词**：Java，画图工具

目录

摘要**2**

一、题目描述**4**

二、总体设计思想**4**

1、实现的功能**4**

2、功能图**4**

三、详细设计**5**

1、基本图形按钮**5**

2、基本操作按钮**5**

3、菜单及其组成**5**

4、其他功能选择**6**

5、完成的主界面**6**

四、功能详解**6**

1、新建图形文件**6**

2、打开已有的文件**7**

3、保存文件**7**

4、设置画笔粗细**8**

5、选择颜色**9**

6、选择字体与字体风格**9**

五、程序结构说明**10**

六、程序源代码**12**

1. **题目描述**

利用Java语言设计一个简易的画图工具，能够实现基本的画笔、画直线、画矩形、更改颜色、更改画笔的大小、橡皮擦等功能。

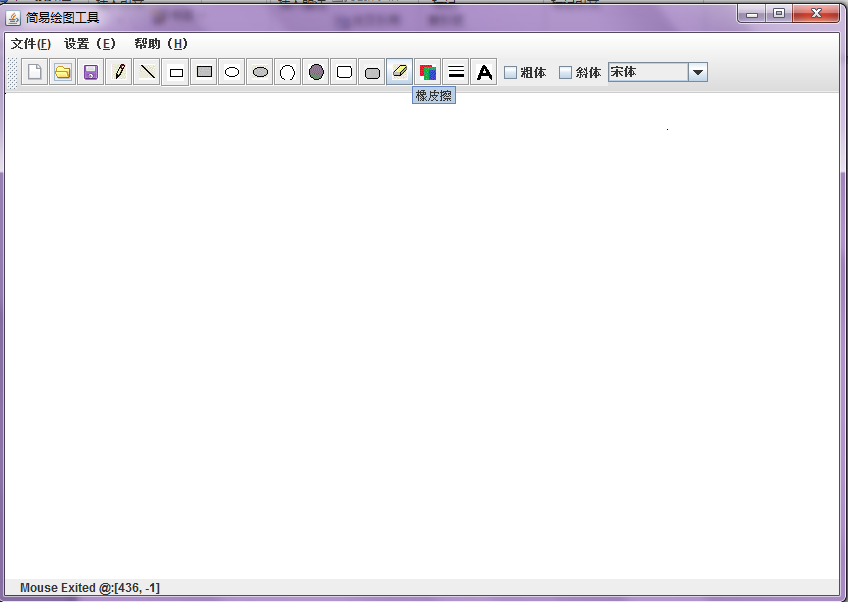
1. **总体设计思想**
2. **实现的功能**
3. 新建空白图形文件，用于绘制新的图形。
4. 打开或保存一个图形文件，便于对图形文件的管理。
5. 绘制一些基本的图形组件：直线，空心矩形，实心矩形，空心圆形，实心圆形，空心椭圆，实心椭圆，空心圆角矩形，实心圆角矩形。
6. 用鼠标在绘图板上单击拖动进行随笔画。
7. 橡皮擦功能。
8. 在绘图区添加文字。
9. 进行文字字体的选取。
10. 选择文字的字体风格：粗体，斜体。
11. 设置画笔的粗细。
12. **功能图**



1. **详细设计**
2. **基本图形按钮**

基本图形按钮包括：随笔画F:\编程\MiniDrawPad\images\Pencil.gif，直线F:\编程\MiniDrawPad\images\Line.gif，空心矩形F:\编程\MiniDrawPad\images\Rect.gif，实心矩形F:\编程\MiniDrawPad\images\fRect.gif，空心圆形F:\编程\MiniDrawPad\images\Cricle.gif，实心圆形F:\编程\MiniDrawPad\images\fCricle.gif，空心椭圆F:\编程\MiniDrawPad\images\Oval.gif，实心椭圆F:\编程\MiniDrawPad\images\fOval.gif，空心圆角矩形F:\编程\MiniDrawPad\images\RoundRect.gif，实心圆角矩形F:\编程\MiniDrawPad\images\fRoundRect.gif，字F:\编程\MiniDrawPad\images\Word.gif，橡皮擦F:\编程\MiniDrawPad\images\Rubber.gif。

1. **基本操作按钮**
2. New F:\编程\MiniDrawPad\images\newFile.gif：新建图形文件。
3. Open F:\编程\MiniDrawPad\images\open.gif：打开图形文件。
4. Save F:\编程\MiniDrawPad\images\save.gif：保存当前图形文件。
5. **菜单及其组成**
6. “文件”菜单项，包括：新建，打开，保存，退出。
7. “设置”菜单项，包括：颜色（线条颜色，字体颜色，图形填充颜色，图形颜色），线条（可设置线条粗细）。
8. “帮助”菜单项：有关程序帮助提示
9. **其他功能选择**
10. 选择当前字体风格的选择框为：粗体，斜体，用来设置当前字体的风格。
11. 选择字体：复选框
12. **完成的主界面**

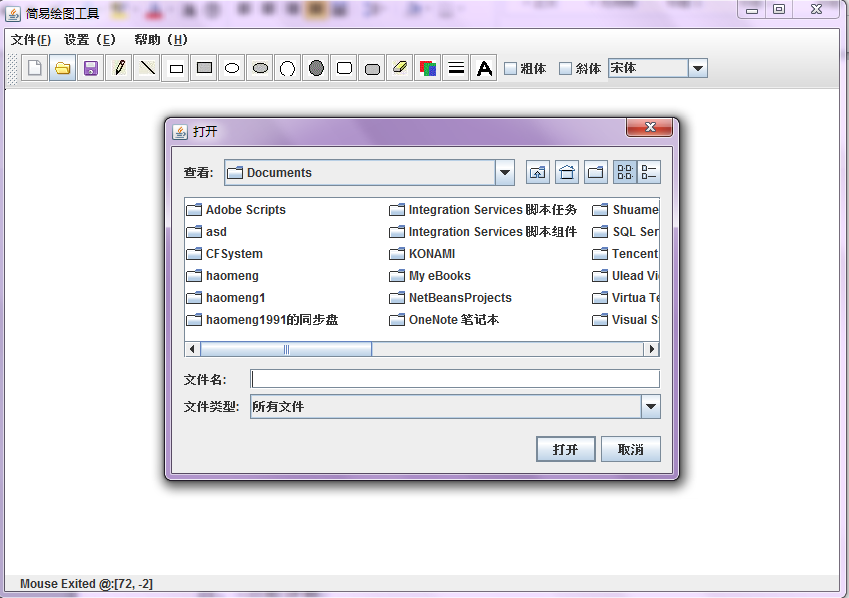


1. **功能详解**
2. **新建图形文件**

单击工具栏上的图标F:\编程\MiniDrawPad\images\newFile.gif，或选择菜单栏 文件->新建，则清除当前所绘制的图像，回到初始画面。

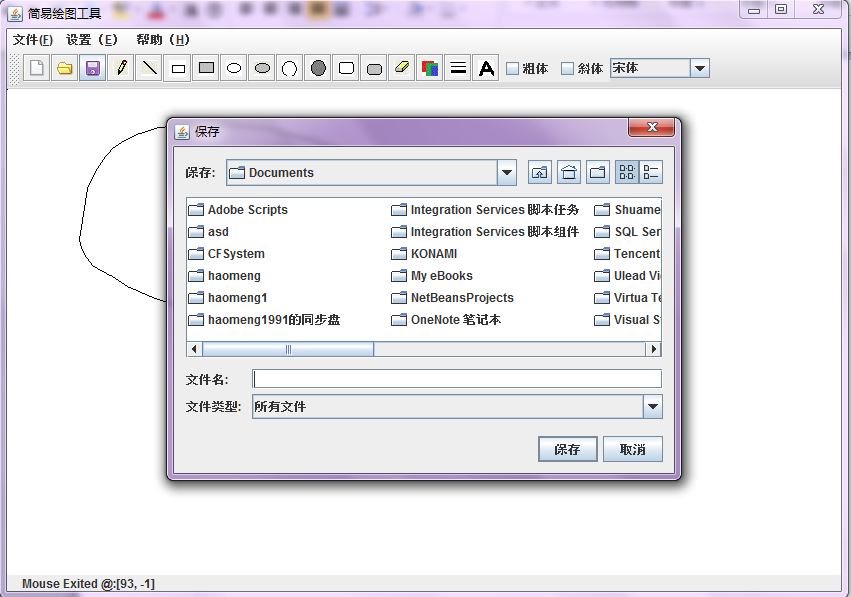
1. **打开已有的文件**

单击工具栏上的图标F:\编程\MiniDrawPad\images\open.gif，或选择菜单 文件->打开，则清除当前所绘制的图像并弹出选择文件对话框要求选择打开的文件，如图



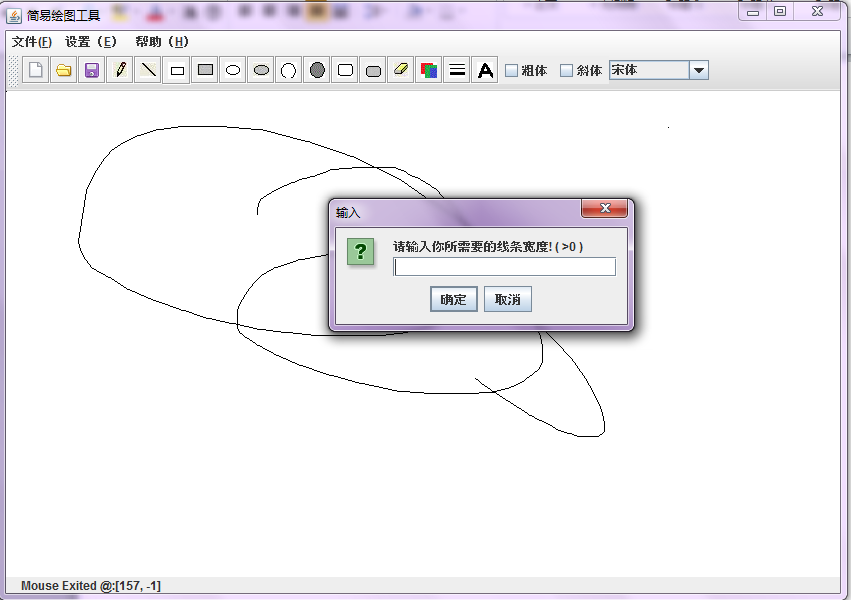
1. **保存文件**

单击工具栏上的图标F:\编程\MiniDrawPad\images\save.gif，或选择菜单 文件->保存，则弹出保存文件对话框，如图



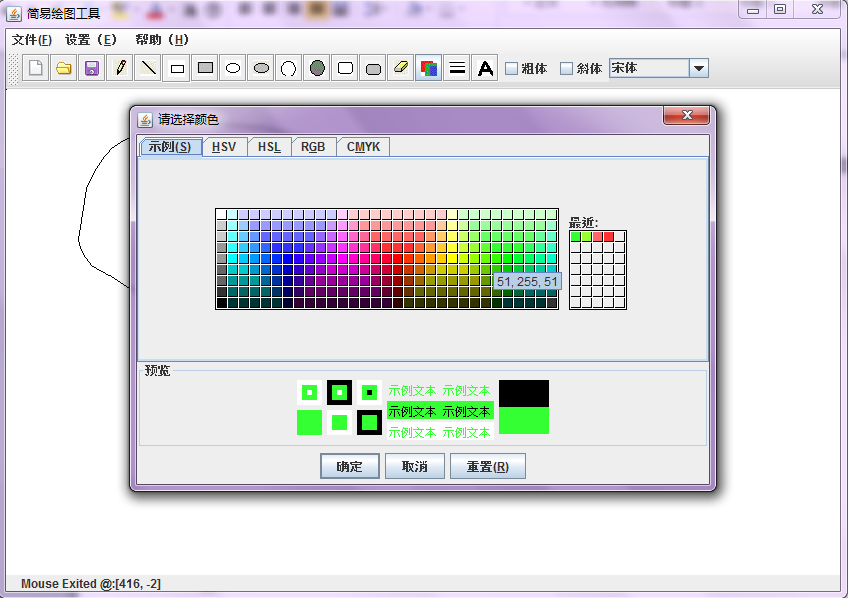
1. **设置画笔粗细**

单击工具栏上的图标F:\编程\MiniDrawPad\images\Stroke.gif，或选择菜单栏 设置->线条，则弹出设置画笔粗细对话框，如图



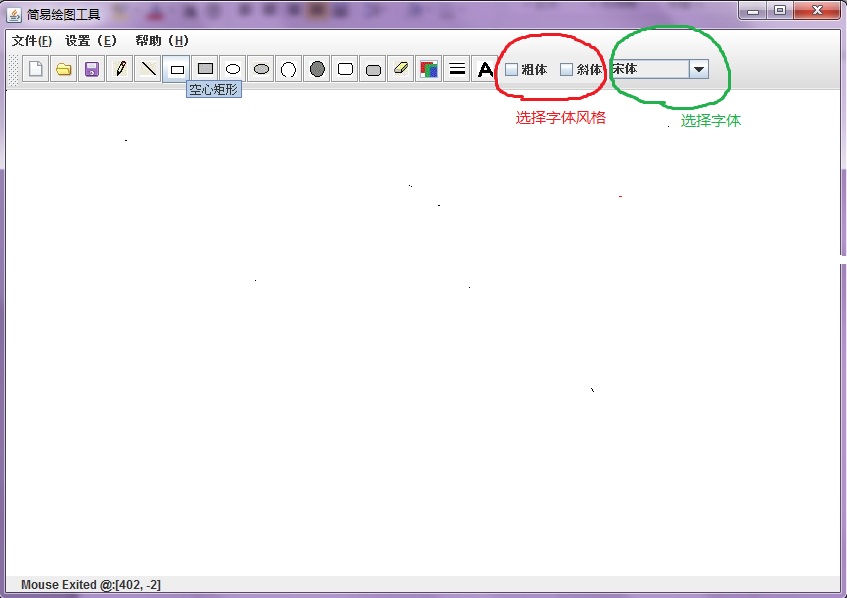
1. **选择颜色**

单击工具栏上的F:\编程\MiniDrawPad\images\Color.gif，或选择菜单 设置->颜色，弹出颜色选择对话框，如图



1. **选择字体与字体风格**

操作区域如图



1. **程序结构说明**

程序中的类

public class MiniDrawPad extends JFrame \\主类

public class ButtonHandler implements ActionListener

//按钮侦听器ButtonHanler类，内部类，用来侦听基本按钮的操作

public class ButtonHandler1 implements ActionListener

//按钮侦听器ButtonHanler1类，用来侦听颜色选择、画笔粗细设置、文字输入按钮的操作

class mouseA extends MouseAdapter

//鼠标事件mouseA类，继承了MouseAdapter，用来完成鼠标相应事件操作

class mouseB extends MouseMotionAdapter

//鼠标事件mouseB类继承了MouseMotionAdapter，用来完成鼠标拖动和鼠标移动时的相应操作

private class checkBoxHandler implements ItemListener

//选择字体风格时候用到的事件侦听器类，加入到字体风格的选择框中

class DrawPanel extends JPanel

//画图面板类，用来画图

class drawings implements Serializable//父类，基本图形单元，用到串行化接口，保存时所用

class Line extends drawings //直线类

class Rect extends drawings//矩形类

class fillRect extends drawings//实心矩形类

class Oval extends drawings//椭圆类

class fillOval extends drawings//实心椭圆

class Circle extends drawings//圆类

class fillCircle extends drawings//实心圆

class RoundRect extends drawings//圆角矩形类

class fillRoundRect extends drawings//实心圆角矩形类

class Pencil extends drawings//随笔画类

class Rubber extends drawings//橡皮擦类

class Word extends drawings//输入文字类

1. **程序源代码**

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package minidrawpad;

import java.awt.\*;

import java.io.\*;

import java.awt.event.\*;

import javax.swing.\*;

/\*\*

\*

\* @author haomeng

\*/

public class MiniDrawPad extends JFrame {

private ObjectInputStream input;

private ObjectOutputStream output;

private JLabel statusBar;

private DrawPanel drawingArea;

private int width = 850, heigh = 600;

drawings[] itemList = new drawings[5000];

private JButton choices[];

JToolBar buttonPanel;

private int currentChoice = 3;

int index = 0;

private Color color = Color.black;

int R, G, B;

int f1, f2;

String style1;

private float stroke = 1.0f;

JCheckBox bold, italic;

JComboBox styles;

private String names[] = {

"newFile",

"open",

"save",

"Pencil",

"Line",

"Rect",

"fRect",

"Oval",

"fOval",

"Cricle",

"fCricle",

"RoundRect",

"fRoundRect",

"Rubber",

"Color",

"Stroke",

"Word"

};

private String styleNames[] = {

"宋体", "隶书", "华文彩云", "仿宋\_GB2312", "华文行楷", "方正舒体",

"Times New Roman", "Serif", "Monospaced", "SonsSerif", "Garamond"

};

private Icon items[];

private String tipText[] = {

"新建",

"打开",

"保存",

"铅笔画",

"直线",

"空心矩形",

"填充矩形",

"空心椭圆",

"填充椭圆",

"空心圆形",

"填充圆形",

"空心圆角矩形",

"填充圆角矩形",

"橡皮擦",

"颜色",

"线条",

"输入文本"

};

public MiniDrawPad() {

super("简易绘图工具");

this.setLocation(200, 100);

JMenuBar bar = new JMenuBar();

JMenu fileMenu = new JMenu("文件(F)");

fileMenu.setMnemonic('F');

JMenuItem newItem = new JMenuItem("新建(N)");

newItem.setMnemonic('N');

newItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_N, InputEvent.CTRL\_MASK));

newItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

newFile();//新建文件

}

});

fileMenu.add(newItem);

JMenuItem saveItem = new JMenuItem("保存(S)");

saveItem.setMnemonic('S');

saveItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_S, InputEvent.CTRL\_MASK));

saveItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

saveFile();//保存文件

}

});

fileMenu.add(saveItem);

JMenuItem openItem = new JMenuItem("打开(L)");

openItem.setMnemonic('L');

openItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_L, InputEvent.CTRL\_MASK));

openItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

openFile();//保存文件

}

});

fileMenu.add(openItem);

fileMenu.addSeparator();

JMenuItem exitItem = new JMenuItem("退出(X)");

exitItem.setMnemonic('X');

exitItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_X, InputEvent.CTRL\_MASK));

exitItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

fileMenu.add(exitItem);

bar.add(fileMenu);

JMenu setMenu = new JMenu("设置（E）");

setMenu.setMnemonic('E');

JMenuItem colorItem = new JMenuItem("颜色(O)");

colorItem.setMnemonic('O');

colorItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_O, InputEvent.CTRL\_MASK));

colorItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

chooseColor();//颜色选择

}

});

setMenu.add(colorItem);

JMenuItem strokeItem = new JMenuItem("线条(K)");

strokeItem.setMnemonic('K');

strokeItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_K, InputEvent.CTRL\_MASK));

strokeItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

setStroke();

}

});

setMenu.add(strokeItem);

bar.add(setMenu);

JMenu helpMenu = new JMenu("帮助（H）");

helpMenu.setMnemonic('H');

JMenuItem aboutItem = new JMenuItem("Readme(A)");

aboutItem.setMnemonic('A');

aboutItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_A, InputEvent.CTRL\_MASK));

aboutItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

readMe();

}

});

helpMenu.add(aboutItem);

JMenuItem writerItem = new JMenuItem("作者(W)");

writerItem.setMnemonic('W');

writerItem.setAccelerator(

KeyStroke.getKeyStroke(

KeyEvent.VK\_W, InputEvent.CTRL\_MASK));

writerItem.addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(null, "哈尔滨工业大学\n10级计算机学院\n郝 萌", "作者信息", JOptionPane.INFORMATION\_MESSAGE);

}

});

helpMenu.add(writerItem);

bar.add(helpMenu);

items = new ImageIcon[names.length];

drawingArea = new DrawPanel();

choices = new JButton[names.length];

buttonPanel = new JToolBar(JToolBar.HORIZONTAL);

ButtonHandler handler = new ButtonHandler();

ButtonHandler1 handler1 = new ButtonHandler1();

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

items[i] = new ImageIcon("images/" + names[i] + ".gif");

choices[i] = new JButton("", items[i]);

choices[i].setToolTipText(tipText[i]);

ToolTipManager.sharedInstance().setDismissDelay(5000);// 设置为5秒

choices[i].setPreferredSize(new Dimension(33, 33));

buttonPanel.add(choices[i]);

}

for (int i = 3; i < choices.length - 3; i++) {

choices[i].addActionListener(handler);

}

choices[0].addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

newFile();

}

});

choices[1].addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

openFile();

}

});

choices[2].addActionListener(

new ActionListener() {

public void actionPerformed(ActionEvent e) {

saveFile();

}

});

choices[choices.length - 3].addActionListener(handler1);

choices[choices.length - 2].addActionListener(handler1);

choices[choices.length - 1].addActionListener(handler1);

styles = new JComboBox(styleNames);

styles.setMaximumRowCount(8);

styles.addItemListener(

new ItemListener() {

public void itemStateChanged(ItemEvent e) {

style1 = styleNames[styles.getSelectedIndex()];

}

});

bold = new JCheckBox("粗体");

italic = new JCheckBox("斜体");

checkBoxHandler cHandler = new checkBoxHandler();

bold.addItemListener(cHandler);

italic.addItemListener(cHandler);

JPanel workPanel = new JPanel();

buttonPanel.add(bold);

buttonPanel.add(italic);

buttonPanel.add(styles);

styles.setMaximumSize(new Dimension(100, 20));

styles.setMinimumSize(new Dimension(50, 20));

Container c = getContentPane();

c.add(buttonPanel, BorderLayout.NORTH);

c.add(drawingArea, BorderLayout.CENTER);

statusBar = new JLabel();

c.add(statusBar, BorderLayout.SOUTH);

statusBar.setText("欢迎进入我的绘图板界面！！！");

setJMenuBar(bar);

createNewItem();

setSize(width, heigh);

setVisible(true);

}

public class ButtonHandler implements ActionListener {

public void actionPerformed(ActionEvent e) {

for (int j = 3; j < choices.length - 3; j++) {

if (e.getSource() == choices[j]) {

currentChoice = j;

createNewItem();

repaint();

}

}

}

}

public class ButtonHandler1 implements ActionListener {

public void actionPerformed(ActionEvent e) {

if (e.getSource() == choices[choices.length - 3]) {

chooseColor();

}

if (e.getSource() == choices[choices.length - 2]) {

setStroke();

}

if (e.getSource() == choices[choices.length - 1]) {

JOptionPane.showMessageDialog(null, "请点击绘图板选择文本输入位置",

"提示", JOptionPane.INFORMATION\_MESSAGE);

currentChoice = 14;

createNewItem();

repaint();

}

}

}

class mouseA extends MouseAdapter {

public void mousePressed(MouseEvent e) {

statusBar.setText(" Mouse Pressed @:[" + e.getX()

+ ", " + e.getY() + "]");//设置状态提示

itemList[index].x1 = itemList[index].x2 = e.getX();

itemList[index].y1 = itemList[index].y2 = e.getY();

//如果当前选择的图形是随笔画或者橡皮擦，则进行下面的操作

if (currentChoice == 3 || currentChoice == 13) {

itemList[index].x1 = itemList[index].x2 = e.getX();

itemList[index].y1 = itemList[index].y2 = e.getY();

index++;

createNewItem();

}

//如果当前选择的图形式文字输入，则进行下面操作

if (currentChoice == 14) {

itemList[index].x1 = e.getX();

itemList[index].y1 = e.getY();

String input;

input = JOptionPane.showInputDialog(

"请输入你想输入的文本!");

itemList[index].s1 = input;

itemList[index].x2 = f1;

itemList[index].y2 = f2;

itemList[index].s2 = style1;

index++;

currentChoice = 14;

createNewItem();

drawingArea.repaint();

}

}

public void mouseReleased(MouseEvent e) {

statusBar.setText(" Mouse Released @:[" + e.getX()

+ ", " + e.getY() + "]");

if (currentChoice == 3 || currentChoice == 13) {

itemList[index].x1 = e.getX();

itemList[index].y1 = e.getY();

}

itemList[index].x2 = e.getX();

itemList[index].y2 = e.getY();

repaint();

index++;

createNewItem();

}

public void mouseEntered(MouseEvent e) {

statusBar.setText(" Mouse Entered @:[" + e.getX()

+ ", " + e.getY() + "]");

}

public void mouseExited(MouseEvent e) {

statusBar.setText(" Mouse Exited @:[" + e.getX()

+ ", " + e.getY() + "]");

}

}

class mouseB extends MouseMotionAdapter {

public void mouseDragged(MouseEvent e) {

statusBar.setText(" Mouse Dragged @:[" + e.getX()

+ ", " + e.getY() + "]");

if (currentChoice == 3 || currentChoice == 13) {

itemList[index - 1].x1 = itemList[index].x2 = itemList[index].x1 = e.getX();

itemList[index - 1].y1 = itemList[index].y2 = itemList[index].y1 = e.getY();

index++;

createNewItem();

} else {

itemList[index].x2 = e.getX();

itemList[index].y2 = e.getY();

}

repaint();

}

public void mouseMoved(MouseEvent e) {

statusBar.setText(" Mouse Moved @:[" + e.getX()

+ ", " + e.getY() + "]");

}

}

private class checkBoxHandler implements ItemListener {

public void itemStateChanged(ItemEvent e) {

if (e.getSource() == bold) {

if (e.getStateChange() == ItemEvent.SELECTED) {

f1 = Font.BOLD;

} else {

f1 = Font.PLAIN;

}

}

if (e.getSource() == italic) {

if (e.getStateChange() == ItemEvent.SELECTED) {

f2 = Font.ITALIC;

} else {

f2 = Font.PLAIN;

}

}

}

}

class DrawPanel extends JPanel {

public DrawPanel() {

setCursor(Cursor.getPredefinedCursor(Cursor.CROSSHAIR\_CURSOR));

setBackground(Color.white);

addMouseListener(new mouseA());

addMouseMotionListener(new mouseB());

}

@Override

public void paintComponent(Graphics g) {

super.paintComponent(g);

Graphics2D g2d = (Graphics2D) g; //定义画笔

int j = 0;

while (j <= index) {

draw(g2d, itemList[j]);

j++;

}

}

void draw(Graphics2D g2d, drawings i) {

i.draw(g2d);//将画笔传入到各个子类中，用来完成各自的绘图

}

}

//新建一个画图基本单元对象的程序段

void createNewItem() {

if (currentChoice == 14)//进行相应的游标设置

{

drawingArea.setCursor(Cursor.getPredefinedCursor(Cursor.TEXT\_CURSOR));

} else {

drawingArea.setCursor(Cursor.getPredefinedCursor(Cursor.CROSSHAIR\_CURSOR));

}

switch (currentChoice) {

case 3:

itemList[index] = new Pencil();

break;

case 4:

itemList[index] = new Line();

break;

case 5:

itemList[index] = new Rect();

break;

case 6:

itemList[index] = new fillRect();

break;

case 7:

itemList[index] = new Oval();

break;

case 8:

itemList[index] = new fillOval();

break;

case 9:

itemList[index] = new Circle();

break;

case 10:

itemList[index] = new fillCircle();

break;

case 11:

itemList[index] = new RoundRect();

break;

case 12:

itemList[index] = new fillRoundRect();

break;

case 13:

itemList[index] = new Rubber();

break;

case 14:

itemList[index] = new Word();

break;

}

itemList[index].type = currentChoice;

itemList[index].R = R;

itemList[index].G = G;

itemList[index].B = B;

itemList[index].stroke = stroke;

}

//选择当前颜色程序段

public void chooseColor() {

color = JColorChooser.showDialog(MiniDrawPad.this,

"请选择颜色", color);

R = color.getRed();

G = color.getGreen();

B = color.getBlue();

itemList[index].R = R;

itemList[index].G = G;

itemList[index].B = B;

}

//选择当前线条粗细程序段

public void setStroke() {

String input;

input = JOptionPane.showInputDialog(

"请输入你所需要的线条宽度! ( >0 )");

stroke = Float.parseFloat(input);

itemList[index].stroke = stroke;

}

//保存图形文件程序段

public void saveFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setFileSelectionMode(JFileChooser.FILES\_ONLY);

int result = fileChooser.showSaveDialog(this);

if (result == JFileChooser.CANCEL\_OPTION) {

return;

}

File fileName = fileChooser.getSelectedFile();

fileName.canWrite();

if (fileName == null || fileName.getName().equals("")) {

JOptionPane.showMessageDialog(fileChooser, "无效的文件名",

"无效的文件名", JOptionPane.ERROR\_MESSAGE);

} else {

try {

fileName.delete();

FileOutputStream fos = new FileOutputStream(fileName);

output = new ObjectOutputStream(fos);

drawings record;

output.writeInt(index);

for (int i = 0; i < index; i++) {

drawings p = itemList[i];

output.writeObject(p);

output.flush(); //将所有图形信息强制转换成父类线性化存储到文件中

}

output.close();

fos.close();

} catch (IOException ioe) {

ioe.printStackTrace();

}

}

}

//打开一个图形文件程序段

public void openFile() {

JFileChooser fileChooser = new JFileChooser();

fileChooser.setFileSelectionMode(JFileChooser.FILES\_ONLY);

int result = fileChooser.showOpenDialog(this);

if (result == JFileChooser.CANCEL\_OPTION) {

return;

}

File fileName = fileChooser.getSelectedFile();

fileName.canRead();

if (fileName == null || fileName.getName().equals("")) {

JOptionPane.showMessageDialog(fileChooser, "无效的文件名",

"无效的文件名", JOptionPane.ERROR\_MESSAGE);

} else {

try {

FileInputStream fis = new FileInputStream(fileName);

input = new ObjectInputStream(fis);

drawings inputRecord;

int countNumber = 0;

countNumber = input.readInt();

for (index = 0; index < countNumber; index++) {

inputRecord = (drawings) input.readObject();

itemList[index] = inputRecord;

}

createNewItem();

input.close();

repaint();

} catch (EOFException endofFileException) {

JOptionPane.showMessageDialog(this, "no more record in file",

"class not found", JOptionPane.ERROR\_MESSAGE);

} catch (ClassNotFoundException classNotFoundException) {

JOptionPane.showMessageDialog(this, "Unable to Create Object",

"end of file", JOptionPane.ERROR\_MESSAGE);

} catch (IOException ioException) {

JOptionPane.showMessageDialog(this, "error during read from file",

"read Error", JOptionPane.ERROR\_MESSAGE);

}

}

}

//新建一个文件程序段

public void newFile() {

index = 0;

currentChoice = 3;

color = Color.black;

stroke = 1.0f;

createNewItem();

repaint();//将有关值设置为初始状态，并且重画

}

public void readMe(){

JFrame frame = new JFrame("readMe");

frame.getContentPane().setLayout(new BorderLayout());

frame.setVisible(true);

frame.setBounds(400, 400, 378, 311);

final JScrollPane scrollPane = new JScrollPane();

scrollPane.setBounds(0, 0, 370, 277);

frame.getContentPane().add(scrollPane);

JTextArea textArea = new JTextArea();

textArea.setEditable(false);

scrollPane.setViewportView(textArea);

byte context[];

try{

InputStreamReader isr = new InputStreamReader(new FileInputStream("readme.txt"), "gb2312");

BufferedReader br = new BufferedReader(isr);

String aline = new String();

while((aline = br.readLine())!=null){

textArea.append(aline+'\n');

}

br.close();

}catch(Exception e){

e.printStackTrace();

}

}

/\*\*

\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

MiniDrawPad newpad = new MiniDrawPad();

newpad.addWindowListener(

new WindowAdapter() {

public void windowCloing(WindowEvent e) {

System.exit(0);

}

});

}

}

class drawings implements Serializable//父类，基本图形单元，用到串行化接口，保存时所用

{

int x1, y1, x2, y2; //定义坐标属性

int R, G, B; //定义色彩属性

float stroke; //定义线条粗细属性

int type; //定义字体属性

String s1;

String s2; //定义字体风格属性

void draw(Graphics2D g2d) {

}

;//定义绘图函数

}

class Line extends drawings //直线类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke,

BasicStroke.CAP\_ROUND, BasicStroke.JOIN\_BEVEL));

g2d.drawLine(x1, y1, x2, y2);

}

}

class Rect extends drawings//矩形类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.drawRect(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2));

}

}

class fillRect extends drawings//实心矩形类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.fillRect(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2));

}

}

class Oval extends drawings//椭圆类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.drawOval(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2));

}

}

class fillOval extends drawings//实心椭圆

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.fillOval(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2));

}

}

class Circle extends drawings//圆类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.drawOval(Math.min(x1, x2), Math.min(y1, y2),

Math.max(Math.abs(x1 - x2), Math.abs(y1 - y2)),

Math.max(Math.abs(x1 - x2), Math.abs(y1 - y2)));

}

}

class fillCircle extends drawings//实心圆

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.fillOval(Math.min(x1, x2), Math.min(y1, y2),

Math.max(Math.abs(x1 - x2), Math.abs(y1 - y2)),

Math.max(Math.abs(x1 - x2), Math.abs(y1 - y2)));

}

}

class RoundRect extends drawings//圆角矩形类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.drawRoundRect(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2),

50, 35);

}

}

class fillRoundRect extends drawings//实心圆角矩形类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke));

g2d.fillRoundRect(Math.min(x1, x2), Math.min(y1, y2),

Math.abs(x1 - x2), Math.abs(y1 - y2),

50, 35);

}

}

class Pencil extends drawings//随笔画类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setStroke(new BasicStroke(stroke,

BasicStroke.CAP\_ROUND, BasicStroke.JOIN\_BEVEL));

g2d.drawLine(x1, y1, x2, y2);

}

}

class Rubber extends drawings//橡皮擦类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(255, 255, 255));

g2d.setStroke(new BasicStroke(stroke + 4,

BasicStroke.CAP\_ROUND, BasicStroke.JOIN\_BEVEL));

g2d.drawLine(x1, y1, x2, y2);

}

}

class Word extends drawings//输入文字类

{

void draw(Graphics2D g2d) {

g2d.setPaint(new Color(R, G, B));

g2d.setFont(new Font(s2, x2 + y2, ((int) stroke) \* 18));

if (s1 != null) {

g2d.drawString(s1, x1, y1);

}

}

}