学号 E21514033 专业 软件工程 姓名 何铭春

实验日期 **6.7**  教师签字 成绩

实验报告

【实验名称】 Java实现计算器

【实验目的】

1. 熟练使用Java布局来实现计算器布局
2. 通过事件函数来实现计算器功能

【实验原理】

1. 网格布局可以用于规范化布局按钮。
2. 事件函数通过改变文本框的内容
3. 通过eval函数来进行运算

【实验内容】

//主体框架类

public class MyFrame extends JFrame {

Box baseBox = Box.createVerticalBox();

public Display display = new Display();

public JButton backSpace = new JButton("BackSpace");

public JButton CE = new JButton("CE");

public JButton C = new JButton("C");

public JButton num1 = new JButton("1");

public JButton num2 = new JButton("2");

public JButton num3 = new JButton("3");

public JButton num4 = new JButton("4");

public JButton num5 = new JButton("5");

public JButton num6 = new JButton("6");

public JButton num7 = new JButton("7");

public JButton num8 = new JButton("8");

public JButton num9 = new JButton("9");

public JButton num0 = new JButton("0");

public JButton add = new JButton("+");

public JButton sub = new JButton("-");

public JButton mul = new JButton("\*");

public JButton div = new JButton("/");

public JButton or = new JButton("+/-");

public JButton point = new JButton(".");

public JButton sqrt = new JButton("sqrt");

public JButton mod = new JButton("%");

public JButton div1 = new JButton("1/x");

public JButton cal = new JButton("=");

// 构造函数

MyFrame(){

setTitle("计算器");

setBounds(200,200,450,200);

setLayout(new FlowLayout());

initFirst();

initSecond();

initRest();

BondAction();

this.add(baseBox);

this.pack();

setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

}

// 初始化第一行

private void initFirst(){

display.setPreferredSize(new Dimension(450,50));

baseBox.add(display);

}

// 初始化第二行

private void initSecond(){

................

}

// 绑定事件函数

public void BondAction(){

................

}

}

// 显示文本框类

public class Display extends JTextField {

Display(){

}

// 数字键或者运算键函数

public void tailAdd(String num){

String current = getText();

current = current + num;

setText(current);

}

// sqrt函数

public void headAdd(String op){

String current = getText();

current = op + "(" + current + ")";

setText(current);

}

// 正负号函数

public void or(){

String curret = getText();

char op = curret.charAt(0);

if (op == '-'){

curret = curret.substring(1);

}

else {

curret = "-" + curret;

}

setText(curret);

}

// 退格函数

public void backSpace(){

String current = getText();

if (current.length() != 0) {

current = current.substring(0, current.length() - 1);

}

setText(current);

}

// 清空函数

public void CAndCE(){

setText("");

}

}

// 数字键事件和加减乘除事件，使用了单件模式

public class TailAddAction implements ActionListener {

private static TailAddAction single = null;

private TailAddAction(){}

private TailAddAction(Display display){

this.display = display;

}

public static TailAddAction getTailAddAction(Display display){

if (single == null){

single = new TailAddAction(display);

}

return single;

}

public Display display;

@Override

public void actionPerformed(ActionEvent e) {

String num = e.getActionCommand();

display.tailAdd(num);

}

}

运算事件函数：

public class CalculateAction implements ActionListener {

private Display display;

CalculateAction(){}

CalculateAction(Display display){

this.display = display;

}

@Override

public void actionPerformed(ActionEvent e) {

ScriptEngine jse = new ScriptEngineManager().getEngineByName("JavaScript");

String strs = display.getText();

strs = strs.replace("√","Math.sqrt");

String answer;

try{

answer = jse.eval(strs).toString();

if (answer == "Infinity"){

answer = "除数不能为0,请重新输入";

}

}

catch (Exception t){

answer = "错误的运算";

}

display.setText(answer);

}

}

Demo类：

public class Demo {

public static void main(String[] args){

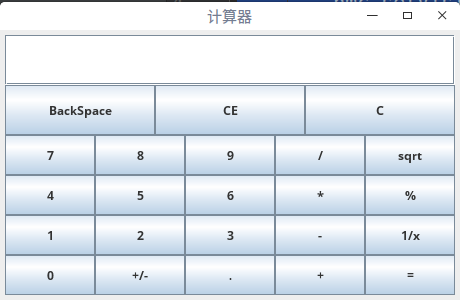
MyFrame hmc = new MyFrame();

hmc.setVisible(true);

}

}

实验截图：





【小结或讨论】

1. 利用网格布局能够使得按钮位置整齐划一。
2. 按照不同事件类型添加事件，并使用单件模式来让类型相同的按钮使用相同的事件。