**Java语言课程设计**

|  |  |
| --- | --- |
| 题 目 | 简易计算器的设计与实现 |

|  |  |
| --- | --- |
| 学生姓名 | 郭鹏飞 |
| 学 号 | 2018023422 |
| 专 业 | 计算机科学与技术 |
| 教 师 | 刘瑾 |

二〇一九年十一月

# 简易计算器的设计与实现

1. 功能介绍
2. 加法

两个数进行相加，可以为小数。

通过调用getActionCommand获取用户点击的按钮是否为“+”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行相加。

1. 减法

两个数进行相减，可以为小数。

通过调用getActionCommand获取用户点击的按钮是否为“-”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行相减。若除数为0，则文本框显示“除数不能为零”。

1. 除法

两个数相除。

通过调用getActionCommand获取用户点击的按钮是否为“/”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行相除。

1. 乘法

两个数相乘。

通过调用getActionCommand获取用户点击的按钮是否为“-”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行相乘。

1. 幂运算

对于任意一个数进行幂运算。

通过调用getActionCommand获取用户点击的按钮是否为“XˆY”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行幂运算。

1. 开平方

对于非负数进行开平方。

通过调用getActionCommand获取用户点击的按钮是否为“sqrt”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行开平方运算。若是负数，则文本框显示“负数没有平方根”。

1. 平方

对于任意一个数进行平方运算。

通过调用getActionCommand获取用户点击的按钮是否为“X^2”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行平方运算。

1. 倒数

对于任意一个数进行倒数运算。

通过调用getActionCommand获取用户点击的按钮是否为“1/X”，如果是，则调用handleOperator函数，通过getNumberFromText获取用户所点的数字，然后进行倒数运算。若X=0，则文本框显示零没有倒数。

1. Backspace按键

对文本框中的内容进行退一位。

通过调用getActionCommand获取用户点击的按钮是否为“Backspace”，如果是，则调用handleBack函数中substring(0, text.length() – 1)将文本的最后一位字符去掉。

1. C键

将文本框的内容清除。

通过调用getActionCommand获取用户点击的按钮是否为“C”，如果是，则调用handleC函数，将文本框初始化。

1. 数字和·键

通过点击按钮显示的数字，使其在文本框中显示。

通过调用函数"0123456789.".indexOf(e. getActionCommand) >= 0判断是否点击的为”0123456789.”,然后在handleNumber函数中对事件的处理。当第一个数字输入完成后还需要判断第二个输入的是什么，若为”.”则将小数点附在结果文本框的后面,若为数字，则将将数字附在结果文本框的后面。

二、图形界面

标准型计算器的界面


标准型计算器界面 1

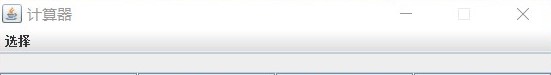
标准型计算器 北部新建一个JPanel panel13中添加的是文本框wbk1(通过JTextField创建)，中部新建一个JPanel panel14中是一个5×4的网格型布局，网格型布局主要依靠GridLayout来设置。网格型布局中通过JButton创建一个又一个的按钮。然后将这些原子组件JButton添加到中层容器JPanel中，最后将中层容器JPanel添加顶层容器JFame中。然后通过继承JFame类，利用setLayout来设置整体的布局，将其设置为BorderLayout；利用setTitle来设置标题，将其设置为计算器；利用setVisible来设置窗体是否可以显示出来，此程序将其设置为true即可；利用setResizable来设置窗体是否可以缩放，此程序将其设置为false，防止影响整体布局；利用setLocation来设置窗体位于屏幕中的位置；利用pack使窗体自动大小；利用setDefaultCloseOperation(EXIT\_ON\_CLOSE)，进程通过点击×来结束。

标题



标题设置成计算器

文本框

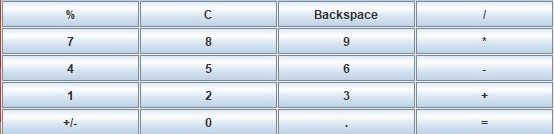


通过wbk1 = **new** JTextField(50) 设置文本框的大小。

通过wbk1.setHorizontalAlignment(JTextField.***RIGHT***) 设置文本框右对齐。

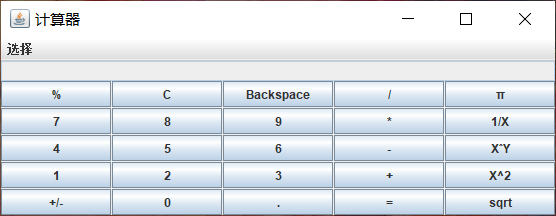
通过wbk1.setEditable(**false**) 设置文本框不可编辑。

键盘



通过panel4.setLayout(**new** GridLayout(5,4,1,1)) 将其设置为5×4并且按钮上下左右的间距为1的网格型布局。

通过an.addActionListener(**this**) 为每一个按钮添加事件。



科学型计算器界面

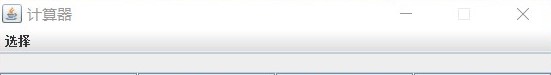
科学型计算器 北部新建一个JPanel panel13中添加的是文本框wbk1(通过JTextField创建)，中部新建一个JPanel panel14中是一个5×5的网格型布局，网格型布局主要依靠GridLayout来设置。网格型布局中通过JButton创建一个又一个的按钮。然后将这些原子组件JButton添加到中层容器JPanel中，最后将中层容器JPanel添加顶层容器JFame中。然后通过继承JFame类，利用setLayout来设置整体的布局，将其设置为BorderLayout；利用setTitle来设置标题，将其设置为计算器；利用setVisible来设置窗体是否可以显示出来，此程序将其设置为true即可；利用setResizable来设置窗体是否可以缩放，此程序将其设置为false，防止影响整体布局；利用setLocation来设置窗体位于屏幕中的位置；利用pack使窗体自动大小；利用setDefaultCloseOperation(EXIT\_ON\_CLOSE)，进程通过点击×来结束。

标题



标题设置成计算器。

文本框

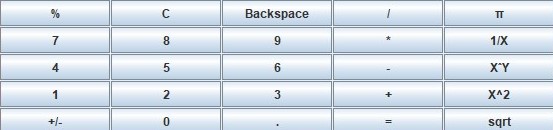


通过wbk1 = **new** JTextField(50) 设置文本框的大小。

通过wbk1.setHorizontalAlignment(JTextField.***RIGHT***) 设置文本框右对齐。

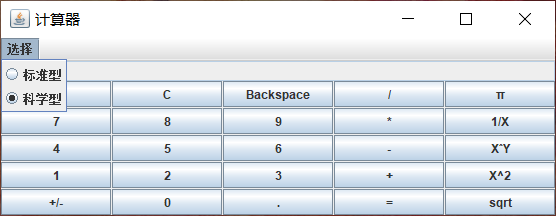
通过wbk1.setEditable(**false**) 设置文本框不可编辑。

键盘



通过panel4.setLayout(**new** GridLayout(5,5,1,1)) 将其设置为5×5并且按钮上下左右的间距为1的网格型布局。

通过an.addActionListener(**this**) 为每一个按钮添加事件。



选择按钮

菜单栏的选择可以在标准型和科学型计算器之间切换。通过建立两个单选按钮rb1，rb2，

rb1 = **new** JRadioButton("标准型") rb2 = **new** JRadioButton("科学型",**true**)，

分别给它们添加成标准型和科学型。运用rb1.ItemListener(this)和rb2.ItemListener(this)为其添加事件，以实现标准型和科学型的选择。

通过重写itemStateChanged函数来完成标准型和科学型之间选择的事件处理，如果是选择的单选按钮则通过创建新的类来完成标准型和科学型计算器之间的切换。

三、程序代码

标志性计算器

**import** java.awt.BorderLayout;

**import** java.awt.GridLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ItemEvent;

**import** java.awt.event.ItemListener;

**import** javax.swing.ButtonGroup;

**import** javax.swing.JButton;

**import** javax.swing.JFrame;

**import** javax.swing.JMenu;

**import** javax.swing.JMenuBar;

**import** javax.swing.JPanel;

**import** javax.swing.JRadioButton;

**import** javax.swing.JTextField;

**public** **class** Cal **extends** JFrame **implements** ActionListener,ItemListener

{

**double** number=0.0; //初始的数字

String operator="="; //当前的运算符

**boolean** bool = **true**; //操作是否合法

**boolean** firstDigit = **true**; //输入的第一个数

String []button= {"%","C","Backspace","/","7","8","9","\*","4","5","6","-","1","2","3","+","+/-","0",".","="};

JPanel panel3,panel4;

JTextField wbk1,wbk2;

JMenuBar jmb;

JMenu jm;

JRadioButton rb1,rb2;

**public** Cal() {

**this**.setLayout(**new** BorderLayout()); //整体布局

**this**.setTitle("计算器"); //标题

**this**.setVisible(**true**); //可见性

**this**.setResizable(**false**); //是否可调整

**this**.setLocation(600, 600); //窗口位于屏幕的位置

**this**.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***); //窗口可以关闭

//菜单

jmb = **new** JMenuBar();

jm =**new** JMenu("选择");

rb1 = **new** JRadioButton("标准型",**true**);

rb2 = **new** JRadioButton("科学型");

ButtonGroup bg = **new** ButtonGroup();

**this**.setJMenuBar(jmb);

jmb.add(jm);

jm.add(rb1);

jm.add(rb2);

bg.add(rb1);

bg.add(rb2);

rb1.addItemListener(**this**);

rb2.addItemListener(**this**);

//文本框

panel3 = **new** JPanel();

panel3.setLayout(**new** BorderLayout());

wbk1 = **new** JTextField(50); //文本框的大小

wbk1.setHorizontalAlignment(JTextField.***RIGHT***); //右对齐

wbk1.setEditable(**false**); //文本框禁止编辑

panel3.add(wbk1);

**this**.add(panel3,BorderLayout.***NORTH***);

//按钮

panel4 = **new** JPanel();

panel4.setLayout(**new** GridLayout(5,4,1,1));

**for**(**int** i=0;i<button.length;i++)

{ JButton an=**new** JButton(button[i]); //按钮添加

an.addActionListener(**this**); //为按钮添加事件监听器

panel4.add(an);

**this**.add(panel4,BorderLayout.***CENTER***);}

**this**.pack(); //窗口的大小

}

//处理事件

@Override

**public** **void** actionPerformed(ActionEvent e)

{

String label = e.getActionCommand();

**if** (label.equals(button[2])) {

//用户按了"Backspace"

handleBack();

}**else** **if**(label.equals(button[1])) {

//用户按了"C"

handleC();

}**else** **if** ("0123456789.".indexOf(label) >= 0) {

//用户按了数字键

handleNumber(label);

}**else** {

//用户按了运算符键

handleOperator(label);

}

}

**public** **void** handleBack() {

String text = wbk1.getText();

**if** (text.length() > 0) {

// 退格，将文本最后一个字符去掉

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

**if** (text.length() == 0) {

// 如果文本没有了内容，则初始化计算器的各种值

wbk1.setText("0");

firstDigit=**true**;

operator = "=";

} **else** {

// 显示新的文本

wbk1.setText(text);

}

}

}

**public** **void** handleC() {

//初始化文本框

wbk1.setText("0");

number=0.0;

operator="=";

}

**public** **void** handleNumber(String key) {

**if** (firstDigit)

{

// 输入的第一个数字

wbk1.setText(key);

}

**else** **if** ((key.equals(".")) && (wbk1.getText().indexOf(".") < 0))

{

// 输入的是小数点，并且之前没有小数点，则将小数点附在结果文本框的后面

wbk1.setText(wbk1.getText() + ".");

}

**else** **if** (!key.equals("."))

{

// 如果输入的不是小数点，则将数字附在结果文本框的后面

wbk1.setText(wbk1.getText() + key);

}

// 以后输入的肯定不是第一个数字了

firstDigit = **false**;

}

**public** **void** handleOperator(String key) {

**if** (operator.equals("/"))

{

// 除法运算

// 如果当前结果文本框中的值等于0

**if** (getNumberFromText() == 0.0)

{

// 操作不合法

bool = **false**;

wbk1.setText("除数不能为零");

}

**else**

{

number /= getNumberFromText();

}

} **else** **if** (operator.equals("+"))

{

// 加法运算

number += getNumberFromText();

}

**else** **if** (operator.equals("-"))

{

// 减法运算

number -= getNumberFromText();

}

**else** **if** (operator.equals("x"))

{

// 乘法运算

number \*= getNumberFromText();

}**else** **if** (operator.equals("%")) {

// 百分号运算，除以100

number /=100;

} **else** **if** (operator.equals("+/-")) {

// 正数负数运算

number \*=(-1);

} **else** **if** (operator.equals("=")) {

// 赋值运算

number = getNumberFromText();

}

**if** (bool) {

// 双精度浮点数的运算

**long** t1;

**double** t2;

t1 = (**long**)number;

t2 = number - t1;

**if** (t2 == 0) {

wbk1.setText(String.*valueOf*(t1));

} **else** {

wbk1.setText(String.*valueOf*(number));

}

}

// 运算符等于用户按的按钮

operator = key;

firstDigit=**true**;

bool = **true**;

}

**public** **double** getNumberFromText() {

**double** result = 0;

**try** {

result = Double.*valueOf*(wbk1.getText()).doubleValue(); //将文本框中的内容转换为double型

} **catch** (NumberFormatException e) { //数字格式化错误

}

**return** result;

}

**public** **static** **void** main(String []args) {

**new** Cal();

}

**public** **void** itemStateChanged(ItemEvent e) //选择项发生改变时事件处理

{

**if** (e.getSource() **instanceof** JRadioButton) //表示获得事件的对象源是不是单选按钮

{

**if**(rb2.isSelected()){ //表示你所选的

Cal2 a = **new** Cal2() ;

a.setVisible(**true**) ;

**this**.setVisible(**false**) ;

}

}

}

}

科学性计算器

**import** java.awt.BorderLayout;

**import** java.awt.GridLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ItemEvent;

**import** java.awt.event.ItemListener;

**import** javax.swing.ButtonGroup;

**import** javax.swing.JButton;

**import** javax.swing.JFrame;

**import** javax.swing.JMenu;

**import** javax.swing.JMenuBar;

**import** javax.swing.JPanel;

**import** javax.swing.JRadioButton;

**import** javax.swing.JTextField;

**public** **class** Cal2 **extends** JFrame **implements** ActionListener,ItemListener

{

**double** number=0.0; //初始的数字

String operator="="; //当前的运算符

**boolean** bool = **true**; //操作是否合法

**boolean** firstDigit = **true**; //输入的第一个数

String []button= {"%","C","Backspace","/","π","7","8","9","\*","1/X","4","5","6","-","XˆY","1","2","3","+","X^2","+/-","0",".","=","sqrt"};

JPanel panel3,panel4;

JTextField wbk1;

JMenuBar jmb;

JMenu jm;

JRadioButton rb1,rb2;

**public** Cal2() {

**this**.setLayout(**new** BorderLayout()); //整体布局

**this**.setTitle("计算器"); //标题

**this**.setVisible(**true**); //可见性

**this**.setResizable(**false**); //是否可调整

**this**.setLocation(600, 600); //窗口位于屏幕的位置

**this**.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***); //窗口可以关闭

//菜单

jmb = **new** JMenuBar();

jm =**new** JMenu("选择");

rb1 = **new** JRadioButton("标准型");

rb2 = **new** JRadioButton("科学型",**true**);

ButtonGroup bg = **new** ButtonGroup();

**this**.setJMenuBar(jmb);

jmb.add(jm);

jm.add(rb1);

jm.add(rb2);

bg.add(rb2);

bg.add(rb1);

rb1.addItemListener(**this**);

rb2.addItemListener(**this**);

//文本框

panel3 = **new** JPanel();

panel3.setLayout(**new** BorderLayout());

wbk1 = **new** JTextField(50); //文本框的大小

wbk1.setHorizontalAlignment(JTextField.***RIGHT***); //右对齐

wbk1.setEditable(**false**); //文本框禁止编辑

panel3.add(wbk1);

**this**.add(panel3,BorderLayout.***NORTH***);

//按钮

panel4 = **new** JPanel();

panel4.setLayout(**new** GridLayout(5,6,1,1));

**for**(**int** i=0;i<button.length;i++)

{ JButton an=**new** JButton(button[i]); //按钮添加

an.addActionListener(**this**); //为按钮添加事件事件

panel4.add(an);

**this**.add(panel4,BorderLayout.***CENTER***);}

**this**.pack(); //窗口的大小

}

**public** **static** **void** main(String []args) {

**new** Cal2();

}

//处理事件

@Override

**public** **void** actionPerformed(ActionEvent e)

{

String label = e.getActionCommand();

**if** (label.equals(button[2])) {

//用户按了"Backspace"

handleBack();

}**else** **if**(label.equals(button[1])) {

//用户按了"C"

handleC();

}**else** **if** ("0123456789.".indexOf(label) >= 0) {

//用户按了数字键

handleNumber(label);

}**else** {

//用户按了运算符键

handleOperator(label);

}

}

**public** **void** handleBack() {

String text = wbk1.getText();

**if** (text.length() > 0) {

// 退格，将文本最后一个字符去掉

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

**if** (text.length() == 0) {

// 如果文本没有了内容，则初始化计算器的各种值

wbk1.setText("0");

firstDigit=**true**;

operator = "=";

} **else** {

// 显示新的文本

wbk1.setText(text);

}

}

}

**public** **void** handleC() {

//初始化文本框

wbk1.setText("0");

number=0.0;

operator="=";

}

**public** **void** handleNumber(String key) {

**if** (firstDigit)

{

// 输入的第一个数字

wbk1.setText(key);

}

**else** **if** ((key.equals(".")) && (wbk1.getText().indexOf(".") < 0))

{

// 输入的是小数点，并且之前没有小数点，则将小数点附在结果文本框的后面

wbk1.setText(wbk1.getText() + ".");

}

**else** **if** (!key.equals("."))

{

// 如果输入的不是小数点，则将数字附在结果文本框的后面

wbk1.setText(wbk1.getText() + key);

}

// 以后输入的肯定不是第一个数字了

firstDigit = **false**;

}

**public** **void** handleOperator(String key) {

**if** (operator.equals("/"))

{

// 除法运算

// 如果当前结果文本框中的值等于0

**if** (getNumberFromText() == 0.0)

{

// 操作不合法

bool = **false**;

wbk1.setText("除数不能为零");

}

**else**

{

number /= getNumberFromText();

}

} **else** **if** (operator.equals("+"))

{

// 加法运算

number += getNumberFromText();

} **else** **if** (operator.equals("-"))

{

// 减法运算

number -= getNumberFromText();

} **else** **if**(operator.equals("π")) {

//Π运算

number=Math.***PI***;

}**else** **if**(operator.equals("1/X")) {

//1/X运算

**if** (number == 0.0)

{

// 操作不合法

bool = **false**;

wbk1.setText("零没有倒数");

}

**else**

{

number = 1 / number;

}

}**else** **if**(operator.equals("sqrt")) {

//开根号运算

**if**(number < 0) {

//操作不合法

bool=**false**;

wbk1.setText("负数没有平方根");

}

**else**

{

number=Math.*sqrt*(getNumberFromText());

}

}**else** **if**(operator.equals("X^2")) {

//平方运算

number\*=number;

}**else** **if**(operator.equals("XˆY")) {

//幂运算

number=Math.*pow*(number,getNumberFromText() );

}**else** **if** (operator.equals("x"))

{

// 乘法运算

number \*= getNumberFromText();

}**else** **if** (operator.equals("%")) {

// 百分号运算，除以100

number /=100;

} **else** **if** (operator.equals("+/-")) {

// 正数负数运算

number \*=(-1);

} **else** **if** (operator.equals("=")) {

// 赋值运算

number = getNumberFromText();

}

**if** (bool) {

// 双精度浮点数的运算

**long** t1;

**double** t2;

t1 = (**long**)number;

t2 = number - t1;

**if** (t2 == 0) {

wbk1.setText(String.*valueOf*(t1));

} **else** {

wbk1.setText(String.*valueOf*(number));

}

}

// 运算符等于用户按的按钮

operator = key;

firstDigit=**true**;

bool = **true**;

}

**public** **double** getNumberFromText() {

**double** result = 0;

**try** {

result = Double.*valueOf*(wbk1.getText()).doubleValue(); //将文本框中的内容转换为double型

} **catch** (NumberFormatException e) { //数字格式化错误

}

**return** result;

}

@Override

**public** **void** itemStateChanged(ItemEvent e) //选择项发生改变时事件处理

{

**if** (e.getSource() **instanceof** JRadioButton) //表示获得事件的对象源是不是单选按钮

{

**if**(rb1.isSelected()){ //表示你所选的

Cal a = **new** Cal() ;

a.setVisible(**true**) ;

**this**.setVisible(**false**) ;

}

}

}

}