莆田学院新工科产业学院

课 程 设 计

课程名称：Java程序设计课程设计

题 目：简易计算器

学生姓名：傅泳森

学 号：202012900512

专业班级：软工205

指导教师：林少宇

**2022** 年 4 月 14 日

课程设计任务书

|  |  |
| --- | --- |
| **设计题目** | **简易计算器** |
| **课**  **程**  **设**  **计**  **主**  **要**  **内**  **容**  **及**  **要**  **求** | 利用 Java 开发一个可以进行简单的四则运算的图形化计算器（以下为参考界面，同学们自行根据功能设计界面）。  F05G6]RY)F4V6NG8YA@G}2Y   1. 基本功能要求如下：   1、要求计算器至少能够实现基本的加减乘除运算和清空功能。  2、C按钮用于实现清空操作，直接将显示置零。  以上为基本功能，想要更好的成绩，需要有扩展功能，扩展功能鼓励同学发挥自己的想象力，以下仅为参考。   1. 扩展功能：   a.实际上按资料做完的计算器存在一定问题，比如连续的加减乘除并没有实现。例如输入7\*8然后不输入=号，而是输入+号，这个时候不能得到结果。  b.能否有更多的计算功能，比如开方等  c.除了显示结果，还有显示输入。比如：当你输入7+8+9的时候，屏幕有两行，第一显示7+8+9，第2行显示结果24  d.在c基础上能否实现括号功能  其他：更多功能(进制转换)、更漂亮界面、更方便的操作、支持对大数的加减法、支持对运算结果的保存与读取。  等等等。 |
| **日**  **程**  **安**  **排** | 2022.02.28-2022.03.01：完成分组、任务的调研、需求的分析以及组内任务的分工。  2022.03.02-2022.03.20：软件的设计、代码的编写与调试。  2022.03.20-2022.04.10：软件的测试和设计报告的撰写。  本学期第7周期末前，需完成答辩、报告以及代码的提交、资料整理等工作。 |
| **指导老师签名： 202 年 月 日** | |

**课程设计成绩评定表**

|  |  |  |  |
| --- | --- | --- | --- |
| 课程设计题目 |  | | |
| 学生姓名 |  | 总成绩 |  |
| 平时成绩 |  | 设计报告成绩 |  |
| 指导教师评语（建议：从学生的工作态度、工作量、设计任务的完成情况及设计报告书面表达能力等方面给出评价。）  **指导教师（签名）：**  **年 月 日** | | | |

目录

1. 概述

使用Java开发进行一个可以进行简单四则运算的计算器，该计算器能够实现基本的加减乘除运算和清空功能，C按钮用于实现清空操作，直接将显示置零，同时在基础上进行了扩展功能，能够进行连续的加减乘除、开方、括号运算和简单的三角函数运算。

1. 方案设计
2. 新建相关文件并查询相关文件
3. 引入java中相关的包
4. 新建类，并继承类Exception，实现了接口Linkedlist
5. 定义相关的变量，创建相关的组件，并对组件的属性进展设置
6. 对所创建的组件进展布局，完成界面的实现
7. 为各个组件添加监听器

1. 功能描述与实现

**1.窗口的设置**

开局在main仿佛中设置一个长900px，宽600px的可视窗口，该窗口利用setVisible(true)将窗口可视化。

public class Main {

public static void main(String[] args) throws MalformedURLException { //抛出异常

final int width = 600; //宽

final int height = 900; //高

BuildUI calculator = new BuildUI();

calculator.buildLayout();//构建布局

calculator.setSize(width, height);

calculator.setVisible(true);

}

}

**2.绘制计算机面板**

绘按钮、文本框，定义7排4行的按钮并在actionPerformed中完成监听

class Display {

private final String[] keys = { //键

"C", "(", ")", "Delete",

"7", "8", "9", "+",

"4", "5", "6", "-",

"1", "2", "3", "\*",

"0", ".", "^", "/",

"sin", "cos", "=", "<--"};

private final JTextField result = new JTextField();//建立文本框

private final JButton[] button = new JButton[keys.length];//建立按钮

public Display() {

/\* initialize the text box \*/

result.setText("0");//将组件上的文字定义为0

result.setCaretColor(Color.black);//文本框颜色

/\* text color \*/

result.setForeground(Color.white);//键值颜色

/\* background color \*/

result.setBackground(Color.white);//背景颜色

}

public void setResult(String str) { //将组件上的文字内容设置为str

/\* set number in the text box \*/

result.setText(str);

}

public JTextField getResult() { //获取的文本框当前输入内容的方法

/\* get numbers from text box \*/

return result;

}

public JButton[] getButton() {

return button;

} //接受鼠标点击的值

public void setButton(JButton button, int i) {

this.button[i] = button;

}

public String getKey(int i) {

return keys[i];

} //获取keys【i】的值

}

class BuildUI extends JFrame implements ActionListener {

Display dis = new Display();

String expression;

final Font resultFont = new Font("Helvetica", Font.PLAIN, 60);

final Font buttonFont = new Font("Helvetica", Font.PLAIN, 30);

ResultBuffer rb = new ResultBuffer();

public void buildLayout() {

JPanel calFrame = new JPanel();

calFrame.setBackground(Color.black);

/\* seven rows, four cols \*/

calFrame.setLayout(new GridLayout(7, 1));//布局

calFrame.add(dis.getResult());//接口调用

dis.getResult().setFont(resultFont);

Container[] con = new Container[6];

/\* set each key \*/

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

con[i] = new Container();

con[i].setLayout(new GridLayout(1, 4));

calFrame.add(con[i]);

}

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

JButton newButton = setBtnForm(i);

// newButton.setForeground(Color.DARK\_GRAY);

dis.setButton(newButton, i);

con[i / 4].add(newButton);

}

getContentPane().add(calFrame);

expression = "";

}

public JButton setBtnForm(int i) {

/\* set font, color, shape of the buttom \*/

JButton newButton = new JButton(dis.getKey(i));

newButton.setBorderPainted(true);

if (0 == i || 1 == i || 2 == i) {

newButton.setForeground(Color.black);

newButton.setBackground(Color.lightGray);

// Image icon = getDefaultToolkit().getImage("/Users/fengfeitong/IdeaProjects/Calculator/IMG\_4018.jpg");

// newButton.setIconImage(icon);

} else if (0 == (i + 1) % 4) {

newButton.setForeground(Color.black);

newButton.setBackground(Color.white);

} else {

newButton.setForeground(Color.black);

newButton.setBackground(Color.green);

}

newButton.addActionListener(this);

newButton.setFont(buttonFont);

return newButton;

}

@Override

public void actionPerformed(ActionEvent e) { //ActionListener这个接口里面的方法

String action = e.getActionCommand();

if (dis.getKey(0).equals(action)) {

/\* press C \*/

pressC();

} else if (dis.getKey(3).equals(action)) {

/\* press Delete \*/

pressDelete();

} else if (dis.getKey(22).equals(action)) {

/\* press = \*/

pressCalculate();

} else if (dis.getKey(23).equals(action)) {

/\* press <-- \*/

pressResultBuffer();

} else {

/\* press 1, 2, 3, +, - ...\*/

pressExpression(action);

}

}

public void pressC() {

/\* clear the text box \*/

expression = "";

dis.setResult("0");

rb.resetPointer();

}

public void pressDelete() {

/\* delete the last character in the text box \*/

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

dis.setResult(expression);

rb.resetPointer();

}

public void pressCalculate() {

/\* compute \*/

Calculate cal = new Calculate();

try {

String re = cal.evaluationExpression(expression + '@');

dis.setResult(re);

rb.addElement(expression + " = " + re);

Log l = new Log();

l.writeFile(expression, re);

} catch (Exception e) {

dis.setResult("INPUT ERROR!!!");

}

expression = "";

rb.resetPointer();

}

public void pressExpression(String action) {

/\* input... \*/

expression += action;

dis.setResult(expression);

rb.resetPointer();

}

public void pressResultBuffer() {

dis.setResult(rb.getElement());

rb.increasePointer();

}

}

**3.设置缓冲区**

在ResultBuffer中建立缓冲区，实现删除返回

public class ResultBuffer { //结果缓冲建立

final int maxBufferLength = 10; //最大缓冲区长度

int currentBufferLength = 0; //当前缓冲区长度

int pointer = -1;

LinkedList<String> buffer = new LinkedList<>();

public void addElement(String ele) { //附加元素

buffer.addFirst(ele);

if (currentBufferLength < maxBufferLength) {

currentBufferLength++;

} else {

buffer.removeLast(); //删除选项

}

}

public String getElement() { //获得要素

return buffer.get(pointer); //返回获取point值

}

public void increasePointer() {

pointer = (pointer + 1) % currentBufferLength;

}

public void resetPointer() {

pointer = 0;

}

}

**4.写文件**

writeFile向磁盘上写文件，将字符一个一个的写入磁盘中

public class Log {

public void writeFile(String expression, String result) { //设置文件

try {

final String currentLogPath = System.getProperty("user.dir");

final String fileName = currentLogPath + "/log.txt";

FileWriter writer = new FileWriter(fileName, true);

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

String time = sdf.format(new java.util.Date());

writer.write(time + ": " + expression + " = " + result + "\n");

writer.close();

} catch (IOException e) { //IO异常

System.out.println("Problem writing file!!!");

}

}

}

**5.计算**

使用switch case进行简单两个值的加减乘除计算，当设计连续加法或加乘时，赋予符号数值，然后进行比大小，数值较大的先进性运算

class InputException extends Exception {

InputException() {

super();

}

}

public class Calculate { //计算

public String evaluationExpression(String expression) throws InputException {

/\* operator in the stack \*/

LinkedList<Character> stackOptr = new LinkedList<>();

/\* operand in the stack \*/

LinkedList<Double> stackOpnd = new LinkedList<>();

/\* add '@' to the bottom of the stack\*/

stackOptr.addFirst('@');

int i = 0;

while (i < expression.length()) {

if ('0' <= expression.charAt(i) && '9' >= expression.charAt(i)) {

double number;

double integerPart = 0, decimalPart = 0;

boolean beforeDecimalPoint = true;

int decimalRank = 10;

while (('0' <= expression.charAt(i) && '9' >= expression.charAt(i)) || '.' == expression.charAt(i)) {

if ('.' == expression.charAt(i)) {

if (beforeDecimalPoint) {

beforeDecimalPoint = false;

i++;

} else {

throw new InputException();

}

}

if (beforeDecimalPoint) {

integerPart = integerPart \* 10 + (expression.charAt(i) - '0');

} else {

decimalPart = decimalPart + (double) (expression.charAt(i) - '0') / decimalRank;

decimalRank \*= 10;

}

i++;

if ('c' == expression.charAt(i) || 's' == expression.charAt(i) || '(' == expression.charAt(i)) {

throw new InputException();

}

}

number = integerPart + decimalPart;

stackOpnd.addFirst(number);

} else {

switch (precede(stackOptr.getFirst(), expression.charAt(i))) {

case '<':

stackOptr.addFirst(expression.charAt(i));

break;

case '=':

stackOptr.removeFirst();

break;

case '>':

if ('c' == stackOptr.getFirst()) {

double a = stackOpnd.removeFirst();

stackOpnd.addFirst(Math.cos(a));

} else if ('s' == stackOptr.getFirst()) {

double a = stackOpnd.removeFirst();

stackOpnd.addFirst(Math.sin(a));

} else {

double a = stackOpnd.removeFirst();

double b = stackOpnd.removeFirst();

stackOpnd.addFirst(operate(a, stackOptr.getFirst(), b));

}

i--;

stackOptr.removeFirst();

break;

}

if ('s' == expression.charAt(i) || 'c' == expression.charAt(i)) {

i += 3;

} else {

i++;

}

}

}

return String.valueOf(stackOpnd.getFirst());

}

public char precede(char ch1, char ch2) {

int a = 0, b = 0;

switch (ch1) {

case '+':

a = 5;

break;

case '-':

a = 5;

break;

case '\*':

a = 15;

break;

case '/':

a = 15;

break;

case '(':

a = 2;

break;

case ')':

a = 25;

break;

case '^':

a = 17;

break;

case '#':

a = 0;

break;

case 's':

a = 20;

break;

case 'c':

a = 20;

break;

case '@':

a = 0;

break;

}

switch (ch2) {

case '+':

b = 3;

break;

case '-':

b = 3;

break;

case '\*':

b = 10;

break;

case '/':

b = 10;

break;

case '(':

b = 22;

break;

case ')':

b = 2;

break;

case '^':

b = 16;

break;

case 's':

b = 19;

break;

case 'c':

b = 19;

break;

case '#':

b = 0;

break;

case '@':

b = 0;

break;

}

if (a > b)

return '>';

else if (a == b)

return '=';

else

return '<';

}

double operate(double a, char sign, double b) {

BigDecimal aa = BigDecimal.valueOf(a);

BigDecimal bb = BigDecimal.valueOf(b);

BigDecimal answer = null;

switch (sign) {

case '+':

// answer = b + a;

answer = bb.add(aa);

break;

case '-':

// answer = b - a;

answer = bb.subtract(aa);

break;

case '\*':

// answer = b \* a;

answer = bb.multiply(aa);

break;

case '/':

// answer = b / a;

answer = bb.divide(aa);

break;

case '^':

// answer = Math.pow(b, a);

answer = bb.pow(aa.intValue());

break;

default:

break;

}

System.out.println(answer);

return answer.doubleValue();

}

}

1. 结论分析

刚做实验之前，只能勉强做些代码，但在进展了计算器程序设计的时候，才发现很多语句还不能联系，大局部还是通过csdn和bilibili才勉强完成，看来还是得多进行实践操作。

1. 参考文献

Bilibili java码农先生 java计算器程序