**Java课程设计计算器**

该程序是一个图形界面的简单Java计算器，具有良好的界面，是使用人员能快捷简单的进行操作。即时准确地获得需要的计算结果，充分减低了数字计算的难度和节约了的时间，对人们的生活有了一定的帮助。可以进行简单的四则运算（加、减、乘、除）等其他功能，其他功能后续完善。其中c表示清零，界面简洁使用方面。

**人员分工配合如下：**

设计简单的模型概念以及界面整理**：**潘星园

资源搜集：赵玉妍、李昊泽

Bug漏洞检查：王宏远

转码格式成型：史航

**1.概念设计**

1.1问题的提出

一个基础的计算器都包括什么？满足那些运算？逻辑关系？优先级？

1.2初步概念流程图如下（图1-1）：

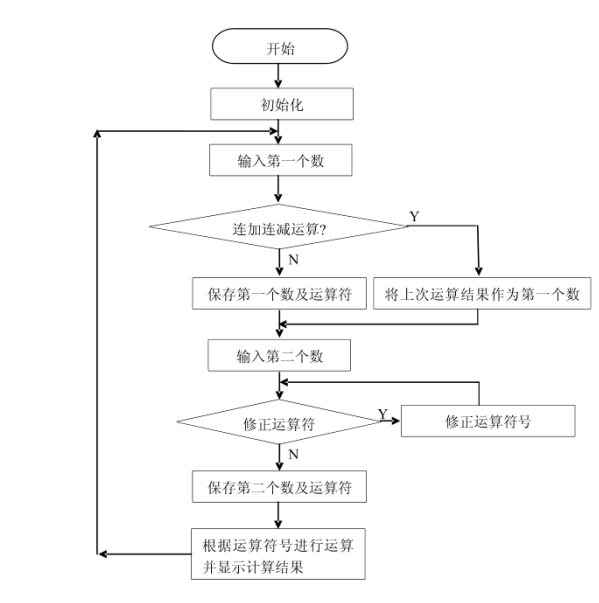


图1-1

1.3准备工作

系统运行环境：window系列

编译工具：MyEclipse 2017cl

转换工具（Java转exe）：exe4j

**2.详细设计**

**2.1界面设计**

显示部分：遵循大多数计算器模型，按照长条格式显示，更为直观明了

按键颜色：采用不同颜色区分，大体可分为三类（基础按键、常用按键、功能按键）

按键布局（以大多数计算器为模型设计）：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 显示屏 | | | | | | | |
| 功能键 | 返回（回退） | | CE | | | C | |
| MC | 7 | 8 | | 9 | / | | 开根号（sqrt） |
| MR | 4 | 5 | | 6 | \* | | % |
| MS | 1 | 2 | | 3 | - | | 1/x |
| M+ | 0 | +/- | | . | + | | = |

**2.2数据分析**

计算器的程序设计，必然有数据的输入，当中就包含运算符和数字，都要通过计算器上的按键选项输入进去。

**2.3类的设计**

**class** Calculator **extends** JFrame **implements** ActionListener {

**2.4功能设计（至于简单的按键我就不介绍了）**

MC：清除存储的数据

MR：读取存储的数据

MS：将所显示的数存入存储器中，存储器中原有的数据被覆盖

M+：将输入的数加上存储器中的数，所得结果存入存储器

M-：将存储器中的数减去输入的数，所得结果存入存储器

CE：在数字输入期间按下此键，将清除输入寄存器中的值并显示"0"，可重新输入

C：清除全部数据结果和运算符

2.4.1扩充概念设计及部分代码示例：

**（1）初始化**

**public** **void** C(){

resultText.setText("0");

firstDigit = **true**;

operator = "=";

}

（2）**实现监听接口**

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

String label = e.getActionCommand();

**if**(label.equals(COMMAND[0])){

Backspace();

}

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

resultText.setText("0");

}

**else** **if**(label.equals(COMMAND[2]))

C();

**else** **if**("0123456789.".indexOf(lab

el)>=0)

Number(label);

**else** { handleOperator(label);

}

}

**（3）处理数字 先判断是不是首数字 再判断是否为小数**

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

**if**(firstDigit){

resultText.setText(key);

}**else** **if** ((key.equals("."))&&(resultText.getText().indexOf(".")< 0 )){

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

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

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

}

firstDigit = **false**;

**3.程序运行结果（图1-2）：**

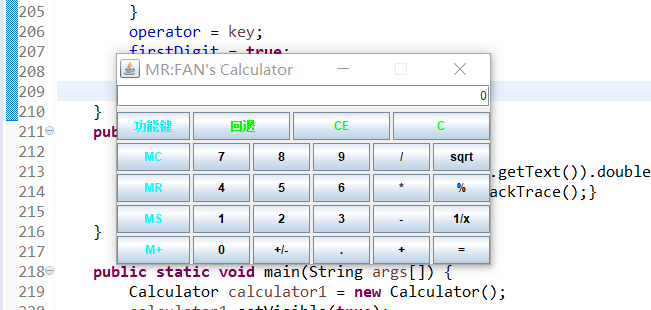


图1-2

界面生成功。进行按键尝试：

随机按一个按键，显示如下（图1-3）：

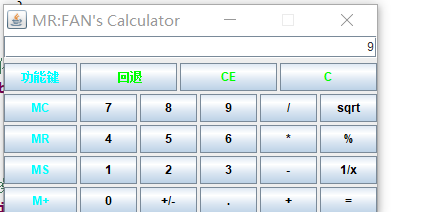


图1-3

对1+4=进行运算，运算结果如下（图1-4）：

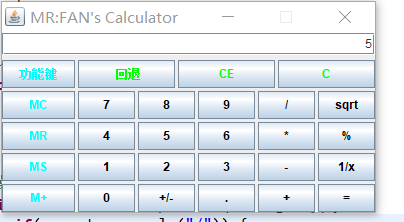


图1-4

显然运行无大碍，都能顺利执行。

**4.运行调试与分析讨论**

4.1程序运行中所遇到的不足以及后期改正

初步设计已经完成，并无大碍。相对应的功能也能顺利执行，但部分细节还有待提高。

经检验未考虑其处理操作运算符 ”/“ 和”1/x“未考虑 0为除数问题

修补，并添加代码如下：

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

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

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

operatorValidFlag = **false**;

resultText.setText("除数不为0");

} **else** {

resultNum /= getNumberFromText();

}

}

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

**if**(resultNum == 0.0) {

operatorValidFlag = **false**;

resultText.setText(" 0 没有倒数");

}

**else**{

resultNum = 1 / resultNum;

}

}

**else** **if** (operator.equals("+")){ resultNum += getNumberFromText(); }

**else** **if** (operator.equals("-")){ resultNum -= getNumberFromText(); }

**else** **if** (operator.equals("\*")){ resultNum \*= getNumberFromText(); }

**else** **if** (operator.equals("sqrt")){resultNum = Math.*sqrt*(resultNum);}

**else** **if** (operator.equals("%")){ resultNum /=100;}

**else** **if** (operator.equals("+/-")){resultNum = resultNum\*(-1);}

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

**if**(operatorValidFlag){

**long** t1;

**double** t2;

t1 = (**long**)resultNum;

t2 = resultNum - t1;

**if**(t2 == 0)

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

**else**

resultText.setText(String.*valueOf*(resultNum));

}

operator = key;

firstDigit = **true**;

operatorValidFlag = **true**;

}

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

**double** result = 0;

**try**{ result = Double.*valueOf*(resultText.getText()).doubleValue();}

**catch**(NumberFormatException E){E.printStackTrace();}

**return** result;

}

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

Calculator calculator1 = **new** Calculator();

calculator1.setVisible(**true**);

calculator1.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

}

**5.文件转换**

为了能进一步使用，我们打算将jav格式转化为Windows通用的exe格式。

在对MyEclipse 2017cl一系列保存，生成如下文件（图1-5）：



图1-5

转换格式如下（1-6）。



图1-6

运行没问题(图1-7)。



图1-7

**附件：整体代码如下：**

**package** calculator;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** java.security.PrivateKey;

**import** java.util.Vector;

**import** java.math.BigDecimal;

/\*

\* @ClassName:Calculator

\* @Description: 计算器的设计尝试

\* @author: xiaopan

\* @date: 2020/04/25

MC：清除存储的数据

MR：读取存储的数据

MS：将所显示的数存入存储器中，存储器中原有的数据被覆盖

M+：将输入的数加上存储器中的数，所得结果存入存储器

M-：将存储器中的数减去输入的数，所得结果存入存储器

CE：在数字输入期间按下此键，将清除输入寄存器中的值并显示"0"，可重新输入

C：清除全部数据结果和运算符

\*/

**public** **class** Calculator **extends** JFrame **implements** ActionListener {

**private** **final** String[] KEYS = {"7", "8", "9", "/", "sqrt",//创建数组，用于存储操作按键

"4", "5", "6", "\*", "%",

"1", "2", "3", "-", "1/x",

"0", "+/-", ".", "+", "="};

**private** **final** String[] COMMAND = {" 回退", "CE", "C"};//创建数组用于存储功能按键

// CE：在数字输入期间按下此键，将清除输入寄存器中的值并显示"0"，可重新输入 C：清除全部数据结果和运算符

**private** **final** String[] M = {"功能键", "MC", "MR", "MS", "M+"};//创建数组 存储M类型的操作按键

/\*MC：清除存储的数据

MR：读取存储的数据

MS：将所显示的数存入存储器中，存储器中原有的数据被覆盖

M+：将输入的数加上存储器中的数，所得结果存入存储器

M-：将存储器中的数减去输入的数，所得结果存入存储器\*/

//创建三个JButton数组用于存储上述String数组的各个功能

**private** JButton keys[] = **new** JButton[KEYS.length];

**private** JButton commands[] = **new** JButton[COMMAND.length];

**private** JButton m[] = **new** JButton[M.length];

**private** JTextField resultText = **new** JTextField("0");

**private** **boolean** firstDigit = **true**;

**private** **double** resultNum = 0.0;

**private** String operator = "=";

**private** **boolean** operatorValidFlag = **true**;

**public** Calculator() {

**super**();

init();

//设置计算器背景颜色

**this**.setBackground(Color.***LIGHT\_GRAY***);

//设置计算器标题

**this**.setTitle("MR:FAN's Calculator");

//设置计算器的左上角坐标

**this**.setLocation(500, 300);

//禁止修改计算器窗口尺寸

**this**.setResizable(**false**);

//使计算器自动调整组件大小

**this**.pack();

}

**private** **void** init() {

//将JtextField的内容设置为右对齐

resultText.setHorizontalAlignment(JTextField.***RIGHT***);

//将文本框的背景设置为白色

resultText.setBackground(Color.***white***);

/\*\*\*\*\*\*\*\*\*\*\*\*将String数组的内容添加到JButton数组中\*\*\*\*\*\*\*\*\*\*\*\*\*/

//按钮panel

JPanel calckeyPanel1 = **new** JPanel();

calckeyPanel1.setLayout(**new** GridLayout(4, 5, 3, 3));

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

keys[i] = **new** JButton(KEYS[i]);

calckeyPanel1.add(keys[i]);

keys[i].setForeground(Color.***black***);

}

//命令panel

JPanel commandsPanel = **new** JPanel();

commandsPanel.setLayout(**new** GridLayout(1, 3, 3, 3));

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

commands[i] = **new** JButton(COMMAND[i]);

commandsPanel.add(commands[i]);

commands[i].setForeground(Color.***green***);

}

//计算panel

JPanel calmsPanel = **new** JPanel();

calmsPanel.setLayout(**new** GridLayout(5, 1, 3, 3));

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

m[i] = **new** JButton(M[i]);

calmsPanel.add(m[i]);

m[i].setForeground(Color.***cyan***);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//创建一个新的panel将按钮和命令panel按照格式置放

JPanel panel1 = **new** JPanel();

panel1.setLayout(**new** BorderLayout(3, 3));

panel1.add("North", commandsPanel);

panel1.add("Center", calckeyPanel1);

//创建一个新的panel将点击的数字输入到该panel中

JPanel top = **new** JPanel();

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

top.add("Center", resultText);

//将三个JPanel 添加到当前的面板

getContentPane().setLayout(**new** BorderLayout(3, 5));

getContentPane().add("North", top);

getContentPane().add("Center", panel1);

getContentPane().add("West", calmsPanel);

//所有JButton添加监听

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

keys[i].addActionListener(**this**);

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

commands[i].addActionListener(**this**);

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

m[i].addActionListener(**this**);

}

//实现监听接口

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

String label = e.getActionCommand();

**if**(label.equals(COMMAND[0])){

Backspace();

}

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

resultText.setText("0");

}

**else** **if**(label.equals(COMMAND[2]))

C();

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

Number(label);

**else** { handleOperator(label);

}

}

//处理回退功能 将字符串利用subString截取

**public** **void** Backspace(){

String text = resultText.getText();

**int** i = text.length();

**if**(i > 0){

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

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

resultText.setText("0");

firstDigit = **true**;

operator = "=";

}

**else**{

resultText.setText(text);

}

}

}

// 处理数字 先判断是不是首数字 再判断是否为小数

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

**if**(firstDigit){

resultText.setText(key);

}**else** **if** ((key.equals("."))&&(resultText.getText().indexOf(".")< 0 )){

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

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

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

}

firstDigit = **false**;

}

//初始化

**public** **void** C(){

resultText.setText("0");

firstDigit = **true**;

operator = "=";

}

//处理操作运算符 ”/“ 和”1/x“需考虑 0为除数

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

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

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

operatorValidFlag = **false**;

resultText.setText("除数不为0");

} **else** {

resultNum /= getNumberFromText();

}

}

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

**if**(resultNum == 0.0) {

operatorValidFlag = **false**;

resultText.setText(" 0 没有倒数");

}

**else**{

resultNum = 1 / resultNum;

}

}

**else** **if** (operator.equals("+")){ resultNum += getNumberFromText(); }

**else** **if** (operator.equals("-")){ resultNum -= getNumberFromText(); }

**else** **if** (operator.equals("\*")){ resultNum \*= getNumberFromText(); }

**else** **if** (operator.equals("sqrt")){resultNum = Math.*sqrt*(resultNum);}

**else** **if** (operator.equals("%")){ resultNum /=100;}

**else** **if** (operator.equals("+/-")){resultNum = resultNum\*(-1);}

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

**if**(operatorValidFlag){

**long** t1;

**double** t2;

t1 = (**long**)resultNum;

t2 = resultNum - t1;

**if**(t2 == 0)

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

**else**

resultText.setText(String.*valueOf*(resultNum));

}

operator = key;

firstDigit = **true**;

operatorValidFlag = **true**;

}

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

**double** result = 0;

**try**{ result = Double.*valueOf*(resultText.getText()).doubleValue();}

**catch**(NumberFormatException E){E.printStackTrace();}

**return** result;

}

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

Calculator calculator1 = **new** Calculator();

calculator1.setVisible(**true**);

calculator1.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

}