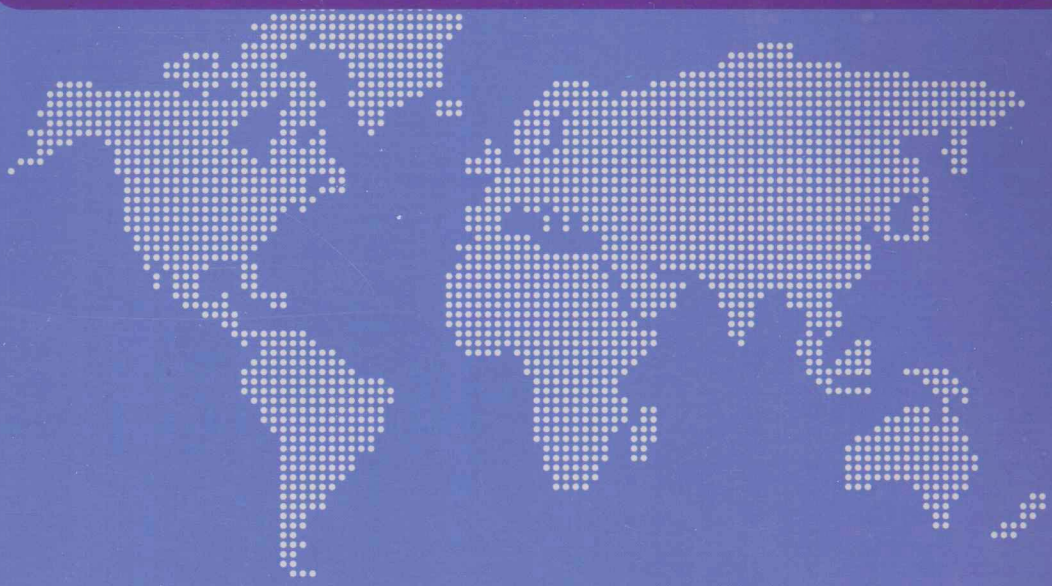
应用型经管类高等院校专业能力拓展系列教材·计算机



Java课程设计案例教程

主　编　李梅生

中国人民大学出版社

应用型经管类高等院校专业能力拓展系列教材·计算机

Java 课程设计案例教程

主　编　李梅生

中国人民大学出版社
·北京·

图书在版编目（CIP）数据

Java 课程设计案例教程/李梅生主编．—北京：中国人民大学出版社，2013.8
应用型经管类高等院校专业能力拓展系列教材．计算机
ISBN 978-7-300-17432-7

Ⅰ.①J… Ⅱ.①李… Ⅲ.①JAVA 语言-程序设计-课程设计-高等学校-教材 Ⅳ.①TP312

中国版本图书馆 CIP 数据核字（2013）第 188111 号

应用型经管类高等院校专业能力拓展系列教材·计算机

Java 课程设计案例教程

主　编　李梅生

Java Kecheng Sheji Anli Jiaocheng

出版发行	中国人民大学出版社		
社　　址	北京中关村大街 31 号	**邮政编码**	100080
电　　话	010－62511242（总编室）		010－62511398（质管部）
	010－82501766（邮购部）		010－62514148（门市部）
	010－62515195（发行公司）		010－62515275（盗版举报）
网　　址	http://www.crup.com.cn		
	http://www.ttrnet.com(人大教研网)		
经　　销	新华书店		
印　　刷	秦皇岛市昌黎文苑印刷有限公司		
规　　格	185 mm×260 mm　16 开本	**版　　次**	2013 年 8 月第 1 版
印　　张	12.25	**印　　次**	2013 年 8 月第 1 次印刷
字　　数	300 000	**定　　价**	27.00 元

总 序

应用型经管类高等院校是我国高等教育的重要组成部分，担负着为社会培养应用型经管类人才的重要任务，当前，我国高等教育正面临着深刻社会变革带来的挑战，科学技术的迅猛发展，应用型人才的社会需要，都对经管类院校教学实践提出了新要求。刚刚颁布的《国家中长期教育改革和发展规划纲要（2010—2020年）》明确指出："加强就业创业教育和就业指导服务，创立高校与科研院所、行业、企业联合培养人才的新机制"，这为各应用型经管类院校人才培养模式的设计与构建指明了方向。

人才培养模式是实现人才培养目标的保证。作为地方性应用型高等院校，突出行业性和应用性办学特色，注重教育与经济社会发展的实际相结合，努力培养具有一定职业素养和创新精神，满足企业可持续发展需要的应用型人才已经成为我们进一步努力的办学目标。经管类课程与就业导向如何紧密结合，如何缩短大学生知识结构和能力结构与行业用人需求的距离，比以往任何时候都更多地引起广大教育者的关心和重视。广东金融学院在吸取传统课程培养模式优点的基础上，秉承面向职业世界、面向生活世界的人才培养理念，探索与实行了由基础核心课程、专业课程和就业导向课程组成的课程培养模式，得到各级教育行政部门领导和教育专家的肯定和支持，并逐步得到推广和应用。该课程培养模式以行业需求为导向，注重课程结构的调整和优化，在学科通识课程基础上嫁接反映职业群集的就业导向课程。探索了大学普通教育与大学职业教育的融合，凸显"厚基础、精专业、强能力"的人才培养特色。就业导向课程以职业群集课程为主体，以拓展提高课程和创新创业课程为两翼，主要根据各专业学生毕业后将从事的岗位要求或考取从业资格证需要，设置相应的课堂教学和校内综合实践课程。为了适应各专业以就业为导向的专业能力拓展的教学需要，我们在经过几年的探索和实践基础上，组织编写了针对应用型经管类高等院校的专业能力拓展系列教材。本系列教材突出以下特色：

1. 各部教材的内容针对性强，就业导向明确。教材内容与行业需求相适应，与职业岗位群和职业认证紧密结合，注重经管类本科生职业能力和社会适应能力培养，强调行业性，突出实用性。

2. 各部教材的内容与现代典型案例相结合，体现应用型人才培养特色。各部教材在编撰过程中注重以培养应用型人才为原则，依据各专业就业导向教学大纲，注重理论联系实际，体现现代经管类企业的最新运作流程，具有较强的实践性和启发性，有利于培养学生的实践能力和创新能力。

3. 各部教材的内容新颖，由易到难，便于自主学习。该系列教材汇集了应用型经管类高等院校教学实践体系构建、教学内容与方法改革、人才培养模式创新等方面的最新研究成果，全方位地展示了广大一线教师在教学实践中的所思所想。其中很多研究成果已经在实际过程中得到了应用和推广，取得了良好的成效。同时，各部教材由易到难，增加了课外学习内容，课内教学与课外教学结合紧密；增加了学生自主学习的时间和空间，满足学生多样化

的学习需求，促进了学生的个性发展；符合最后一学年学生对教学的现实需求，体现了现代大学制度条件下的弹性学习课程设置特性。

4. 该系列教材由既有经管类企业工作阅历和经验，又有高校教学经验的双师型教师编写，编写程序严格。编写者不但熟悉教学过程，而且熟悉现代企业运作，能够将用人单位的需求真实反映到教学活动中和教材编写中，让学生提前感受到未来的工作环境，从而提高学生将来在实际工作中解决问题的能力，有效提升毕业生就业的核心竞争力。该系列教材在出版过程中，严格遵循如下程序：首先，由专家及各级领导对申请立项的教材进行立项审查；其次，由相关同行专家对教材初稿进行审核鉴定；最后，送交出版社评审出版。

本系列教材是在各部教材编者和广大教师共同努力下完成的，凝结着诸位作者在应用型经管类高等院校教学实践过程中的收获和体会，也内含着对经管类本科应用性人才培养模式的理念和认识。教材的编写得到了广大业界专家的充分肯定，得到了多所经管类高校专家的悉心指导，尤其是得到了广东金融学院各级领导及有关部门的鼎力支持和直接帮助，谨致以衷心谢意！当然，由于经管类人才培养与就业导向紧密结合的教学实践还处在探索过程中，作者在编写过程中也难免有考虑不周之处，因而个别观点还需要进一步商榷，热切地期盼各位读者不吝赐教。

应用型经管类高等院校
专业能力拓展系列教材编委会

前　言

Java是一种先进的、面向对象的语言，也是目前最流行的软件开发语言之一。Java具有简单高效、面向对象、与平台无关的特点，支持多线程、分布和并发机制。所以，Java可以广泛适用于高层商业应用和底层系统的开发，还可应用于Internet系统管理、Web页面设计和Internet可视化软件开发等方面。

本书共分6章，精选了6个案例，这些案例各具特色。

案例1：简易计算器。主要是使用Swing技术进行图形界面的设计，熟悉Java的基本知识。

案例2：股票K线图绘制。采用Swing技术进行图形界面的设计，牵涉Java中的文件和流的概念和使用，涉及算法的实现。

案例3：21点纸牌游戏。采用面向对象的思想，尤其是封装、继承和多态，其中牵涉图形界面的显示和更新。在这个游戏的开发中还应用了一些设计模式。

案例4：聊天室。融合多线程技术、网络编程技术和图形界面技术，实现多人同时进行聊天。

案例5：通信录管理系统。一个比较完整的MIS系统，涉及多方面的知识，包括数据库技术、JDBC技术、图形界面技术等。

案例6：客户关系管理系统。进一步加深读者对Java开发数据库系统的认识。采用MVC模块化程序设计的方法，既便于系统功能的组合和修改，也便于系统的补充和维护。

本书采用Eclipse软件作为开发工具，数据库主要采用MySQL，所用到的表、各种结构每个例子中都有介绍。

本书由李梅生主编，期间得到陈国君教授、邹琳达老师和潘章明老师的热心帮助与指导，并得到了广东金融学院教务处的大力支持，在此表示衷心的感谢。由于编写时间仓促，加之编者水平有限，书中难免存在一些不足，欢迎读者指教。

编　者

目　录

第 1 章　简易计算器程序

计算器是 Windows 操作系统附带的一个程序，也是一个非常实用的程序。设计一个与之类似的计算器，也是检验一个人对 Java 程序语言、界面设计和基本程序逻辑知识掌握熟练程度的有效途径。

1.1　程序概述

Windows 操作系统附带的计算器，提供两种类型，一种是标准型计算器，一种是科学型计算器。其中，标准型可以进行基本的算术运算和简易存储等功能。本程序就是实现与标准计算器具有相似功能的简易计算器，程序运行时的界面如图 1.1 所示。

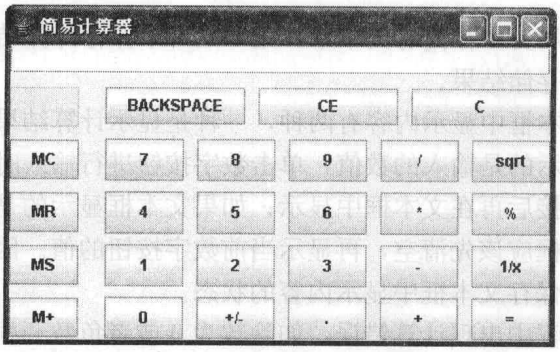


图 1.1　简易计算器运行效果图

由于程序只模拟标准型计算器的功能，因此程序没有提供菜单栏，界面主要由按钮和文本框组成。

1.2　程序的概要设计

根据设计要求，首先要设计其 GUI 界面，主界面由显示区、存储按钮区、清除按钮区和数字与计算按钮区等四部分组成，如图 1.2 所示。在进行界面设计时，总体界面用 BorderLayout 布局，其中文本框放置在最 NORTH，存储器区的面板放置在 WEST，清除区和数字与计算按钮区放在一个面板中，该面板放置在 CENTER。而存放清除区和数字与计算按钮区的面板也采用 BorderLayout 布局，其中清除区的面板放置在 NORTH，而数字与计算

按钮区的面板放置在 CENTER。

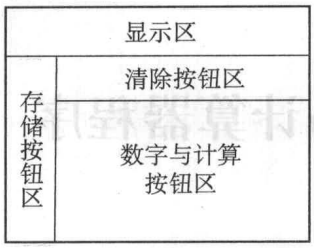


图 1.2　计算器界面布局示意图

接着设计计算流程。当点击数字按钮时，将该数字显示在文本框中，再次点击数字按钮时，将之前保存的字符串与新的数值拼接起来，再显示在文本框中，直到点击运算符按钮时，将文本框中存在的字符串保存在一个字符串变量中，然后重置文本框内容，将运算符显示到文本框；接下来输入第二个技术数据，用同样的办法保存数据，最后通过控制“=”运算符先将字符串数据转化成双精度类型，然后计算出结果显示到文本框中。

基本运算设计完成以后则开始考虑其他个别功能的实现，如倒数、清零、退格等功能的实现。

1.3　程序的详细设计及相关代码

1. 全局变量的定义

首先是定义一些全局变量存储有关结果，用 result 来保存计算结果，用 memory 来保存单击存储按钮时保存的存储结果。

由于显示数值的文本框中显示内容有两种，一种是显示计算结果，另一种是显示输入的数值。如果文本框中显示的是输入的数值，单击数字按钮进行输入时，则应该将文本框的内容与当前按钮的数字连接后再在文本框中显示；如果文本框显示的是计算结果，单击数字按钮进行输入时，则文本框应该先清空，再显示当前数字按钮的值。因此，在设计时，需要定义一个变量（start）来保存文本框中显示内容的状态。

如果在进行计算过程中出现计算错误，如除数为 0 或者负数开根号，则在文本框中显示错误提示信息，并且不允许点击除 C 和 CE 按钮之外的其他按钮。在设计时，需要定义一个变量（valid）来保存计算错误的状态。

在进行双目运算时，当单击完运算符，如“+”、“−”、“×”、“/”等之后，必须将运算符保存起来，待第二个数据也输入后才能进行计算。因此，在设计时，需要定义一个变量（lastCommand）来保存待运算的运算符。

具体变量定义如下：

```
//声明保存计算结果的全局变量
private double result = 0;
//声明单击存储按钮时保存存储结果的全局变量
private double memory = 0;
//保存待运算的运算符
private String lastCommand = " = ";
```

```
//保存文本框中显示内容的状态
private boolean start = true;
//计算错误的状态,true 表示其他按钮可单击,false 表示其他按钮不可单击
private boolean valid = true;
```

2. 界面设计

接下来，进行主界面设计，根据设计思想，界面用 BorderLayout 布局，其中文本框放置在最 NORTH，存储器区的面板放置在 WEST，清除区和数字与计算按钮区放在一个面板中，该面板放置在 CENTER。创建主界面面板及面板中各个区块面板的程序如下：

```
private JPanel getJContentPane( ) {
        if(jContentPane == null) {
          //定义布局管理器对象及设置相关属性
          BorderLayout borderLayout = new BorderLayout( );
          borderLayout. setHgap(20);
          borderLayout. setVgap(10);
          jContentPane = new JPanel( );
          //设置面板的布局为 BorderLayout
          jContentPane. setLayout(borderLayout);
          //在面板的 NORTH 放文本框
          jContentPane. add(getTxtResult( ),BorderLayout. NORTH);
          //在面板的 WEST 放置存储器区面板
          jContentPane. add(getMemoryPanel( ),BorderLayout. WEST);
          //在面板的 CENTER 放置清除区和数字与计算按钮区的面板
          jContentPane. add(getMainPanel( ),BorderLayout. CENTER);
        }
        return jContentPane;
    }
```

在存储面板（memoryPanel）中共包含一个显示框和四个按钮，五个组件排列成一列，因此在该面板中布局可采用 5×1 的网格式布局，于是创建存储区面板及相关组件的程序代码：

```
private JPanel getmemoryPanel( ) {
        if(memoryPanel == null) {
          //定义布局管理器对象及设置相关属性
          GridLayout gridLayout = new GridLayout( );
          gridLayout. setRows(5);//为 5 行 1 列
          gridLayout. setVgap(10);
          gridLayout. setHgap(10);
          memoryPanel = new JPanel( );
          //设置面板的布局为 GridLayout
          memoryPanel. setLayout(gridLayout);
          //对面板添加相关组件
          memoryPanel. add(getTxtInfo( ),null);
          memoryPanel. add(getBtnMC( ),null);
          memoryPanel. add(getBtnMR( ),null);
```

```
            memoryPanel.add(getBtnMS( ),null);
            memoryPanel.add(getBtnM( ),null);
        }
        return memoryPanel;
    }
```

主面板（mainPanel）包含了清除区和数字与计算按钮区，在该面板，我们采用比较简单的 BorderLayout 进行布局，其中清除区面板放置在 NORTH 位置，数字和计算按钮区放置在 CENTER，创建该面板及添加组件的代码如下：

```
private JPanel getmainPanel( ) {
        if(mainPanel == null) {
            //定义布局管理器对象及设置相关属性
            BorderLayout borderLayout1 = new BorderLayout( );
            borderLayout1.setHgap(0);
            borderLayout1.setVgap(11);
            mainPanel = new JPanel( );
            //设置面板的布局为 BorderLayout
            mainPanel.setLayout(borderLayout1);
            //将清除区面板放置在该面板的 NORTH
            mainPanel.add(getClearPanel( ),BorderLayout.NORTH);
            //将数字与计算区面板放置在该面板的 CENTER
            mainPanel.add(getComputePanel( ),BorderLayout.CENTER);
        }
        return mainPanel;
    }
```

清除区面板（clearPanel）包含退格按钮，CE 按钮和 C 按钮，在该面板采用 GridLayout 进行布局，创建清除区面板和相关组件的代码如下：

```
private JPanel getclearPanel( ) {
        if(clearPanel == null) {
            //定义布局管理器对象及设置相关属性
            GridLayout gridLayout1 = new GridLayout( );
            gridLayout1.setRows(1);//设置为 1 行
            gridLayout1.setVgap(5);
            gridLayout1.setHgap(10);
            clearPanel = new JPanel( );
            //设置面板的布局为 BorderLayout
            clearPanel.setLayout(gridLayout1);
            //依次添加退格键、CE 和 C 键按钮
            clearPanel.add(getBtnBS( ),null);
            clearPanel.add(getBtnCE( ),null);
            clearPanel.add(getBtnC( ),null);
        }
        return clearPanel;
    }
```

数字与计算区面板（computePanel）共包含了20个数字和计算按钮，并按照4×5均匀排布，因此该面板应该采用GridLayout比较简易，于是创建该面板以及相关按钮的代码如下：

```
private JPanel getcomputePanel( ) {
        if(computePanel == null) {
          //定义布局管理器对象及设置相关属性
          GridLayout gridLayout2 = new GridLayout( );
          gridLayout2. setRows(4);//设置为4行
          gridLayout2. setHgap(10);
          gridLayout2. setVgap(8);
          gridLayout2. setColumns(5);//设置为5列
          computePanel = new JPanel( );
          //设置面板的布局为BorderLayout
          computePanel. setLayout(gridLayout2);
          //按照按钮排列次序依次添加按钮组件
          computePanel. add(getBtn7( ),null);
          computePanel. add(getBtn8( ),null);
          computePanel. add(getBtn9( ),null);
          computePanel. add(getBtnDiv( ),null);
          computePanel. add(getBtnSqrt( ),null);
          computePanel. add(getBtn4( ),null);
          computePanel. add(getBtn5( ),null);
          computePanel. add(getBtn6( ),null);
          computePanel. add(getBtnMul( ),null);
          computePanel. add(getBtnMod( ),null);
          computePanel. add(getBtn1( ),null);
          computePanel. add(getBtn2( ),null);
          computePanel. add(getBtn3( ),null);
          computePanel. add(getBtnSub( ),null);
          computePanel. add(getBtnCvt( ),null);
          computePanel. add(getBtn0( ),null);
          computePanel. add(getBtnSign( ),null);
          computePanel. add(getBtnDot( ),null);
          computePanel. add(getBtnAdd( ),null);
          computePanel. add(getBtnEqual( ),null);
        }
        return computePanel;
  }
```

3. 事件处理

由于界面中的按钮很多，所有按钮都有相应的事件处理程序，为了便于代码的维护和管理，将事件处理程序根据按钮功能成四个方法，分别是clearHandle( ) 方法用于执行清除计算结果及文本框显示内容程序，numberHandle( ) 方法用执行单击数字按钮后再文本框显示输入的程序，computeHandle( ) 方法用于执行运算的程序，memoryHandle( ) 方法用于执行存储操作的程序。所有按钮的事件处理程序根据按钮的性质调用相应的方法即可。

clearHandle( ) 是用于执行清除计算结果及文本框显示内容的方法。当单击的是 C 或 CE 按钮时，如果因为之前的运算出现运算错误从而使 vaild 的值为 false，则可以将 valid 恢复为正常状态，即为 true。computeHandle( ) 方法的代码如下：

```
private void clearHandle(ActionEvent e) {
        String input = e. getActionCommand( );
        //如果之前运算错误,点击按钮 C 或 CE 后将变量 valid 恢复为 true
        if(input. equals("C")|| input. equals("CE")){
          valid = true;
        }
        //如果 valid 为 false,表示不允按钮执行单击操作
        if(! valid)return;
        //如果点击的是 BACKSPACE 按钮,在清除输入框中末尾的字符
        if(input. equals("BACKSPACE")){
          String txt = txtResult. getText( );
          if(! start && txt. length( )>0)
            txtResult. setText(txt. substring(0,txt. length( ) - 1));
        }
        else if(input. equals("CE")){//点击的是 CE 按钮,清空输入框
          txtResult. setText("");
          start = true;
        }
        else if(input. equals("C")){//点击的是 C 按钮,清空所有计算内容
          txtResult. setText("");
          result = 0;
          lastCommand = " = ";
          start = true;
        }
  }
```

numberHandle( ) 是执行单击数字按钮后再文本框显示输入的方法，如果全局变量 start 为 false，则单击的数字字符之间连接在显示框字符的末尾；如果变量 start 为 true，则需要先将显示框中清空，然后将单击的数字字符显示在显示框中。单击的“.”号时，如果显示框中有“.”则不能在显示框中出现。numberHandle( ) 事件处理程序代码如下：

```
private void numberHandle(ActionEvent e){
    //如果 valid 为 false,表示不允按钮执行单击操作
    if(! valid)return;
    String input = e. getActionCommand( );
    if(input. equals(". ")){//如果输入的为". "号
      //只有文本框中没有". "时才能输入". "号
      if(txtResult. getText( ). indexOf(". ")<0)
        txtResult. setText(txtResult. getText( ) + input);
    }
    else{//如果输入的是其他数字字符
      if(start){  //如果 start 的状态为 true
```

```
            //先清空显示框
            txtResult.setText("");
            //start 状态设置为 false
            start = false;
        }
        //将显示框中的内容与点击的字符连接起来在显示框中显示
        txtResult.setText(txtResult.getText( ) + input);
    }
}
```

computeHandle( ) 是用于执行运算的方法，在运算中涉及单目运算和双目运算。如果当前输入的为双目运算，该方法是将变量 result 的值与显示框的值执行变量 lastCommand 中运算，而将当前输入的运算符保存到变量 lastCommand 供下次运算。如果当前输入的为单目运算，则将显示框的值执行当前输入单目符的运算，而将变量 lastCommand 中运算延缓至下一次再运算。

```
private void computeHandle(ActionEvent e){
    //如果 valid 为 false,表示不允按钮执行单击操作
    if(!valid)return;
    String input = e.getActionCommand( );
    /* * 如果为单目运算,则先进性单目运算,再执行 lastCommand 中的运算 */
    if(input.equals("sqrt")|| input.equals("1/x")){ //单目运算
      if(txtResult.getText( ).length( )>0){
        //临时保存 lastCommand 中的运算
        String tempCommand = lastCommand;
        //临时保存计算结果 result
        double tempResult = result;
        //将当前单目运算赋予 lastCommand
        lastCommand = input;
        //执行单目运算
        valid = computing(Double.parseDouble(txtResult.getText( )));
        //如果计算合法
        if(valid){
          //恢复之前 lastCommand 的值
          lastCommand = tempCommand;
          //恢复之前 result 的值
          result = tempResult;
          start = true;
        }
      }
    }
    else if(input.equals("/")|| input.equals("*")|| input.equals("-")|| input.equals("+")
        || input.equals("%")|| input.equals("+/-")|| input.equals("=")){
      //执行上次运算
      valid = computing(Double.parseDouble(txtResult.getText( )));
      //计算结果合法
```

```
        if(valid){
          //将当前输入的计算符保存在 lastCommand
          lastCommand = input;
          //start 标识为 true
          start = true;
        }
      }
  }
```

在上面的方法中调用了 computing 方法执行运算，该方法使用的运算符保存在变量 lastCommand 中，运算的结果保存在变量 result 中。方法包含了单目运算和双目运算。对于单目运算，则直接将方法传入的参数进行运算。对于双目运算，则第一个运算数为变量 result 的值，第二个运算数为传入的参数。computing 方法的代码如下：

```
  private boolean computing(double x){
      if(lastCommand.equals("/")){//如果是/,要判断除数是否为 0
        if(Math.abs(x)<Math.exp(-10)){
          txtResult.setText("除数不能为 0!");
          return false;
        }
        result /= x;
      }
      else if(lastCommand.equals("*"))//执行 * 运算
        result *= x;
      else if(lastCommand.equals("-"))//执行 - 运算
        result -= x;
      else if(lastCommand.equals("+"))//执行 + 运算
        result += x;
      else if(lastCommand.equals("sqrt")){//执行 sqrt 运算
        if(Math.abs(x)<Math.exp(-10)){//判断是否为负数
          txtResult.setText("函数输入无效!");
          return false;
        }
        result = Math.sqrt(x);
      }
      else if(lastCommand.equals("%"))//执行 % 运算
        result %= x;
      else if(lastCommand.equals("1/x")){//执行 1/x 运算
        if(Math.abs(x)<Math.exp(-10)){//判断除数是否为 0
          txtResult.setText("除数不能为 0!");
          return false;
        }
        result = 1/x;
      }
      else if(lastCommand.equals("+/-"))//执行求相反数
        result *= -1;
```

```
        else if(lastCommand.equals("=")))//执行=运算
            result=x;
        //在显示框中显示计算结果
        txtResult.setText(""+result);
        return true;
    }
```

memoryHandle( ) 是用于执行存储操作的方法。

```
    private void memoryHandle(ActionEvent e){
        String input=e.getActionCommand( );
        if(input.equals("MC")){//如果是MC按钮,清除存储内容
            memory=0;
            txtInfo.setText("");
        }
        else if(input.equals("MR")){//如果是MR按钮,将存储内容显示
            txtResult.setText(""+memory);
            txtInfo.setText("M");
        }
        else if(input.equals("MC")){//如果是MC按钮,存储显示框内容
            memory=Double.parseDouble(txtResult.getText( ));
            txtInfo.setText("M");
        }
        else if(input.equals("MC")){
            memory+=Double.parseDouble(txtResult.getText( ));
            txtInfo.setText("M");
        }
    }
```

本章小结

本章通过介绍简易计算器的开发，介绍了一个简单小程序的开发过程。虽然程序的功能不是很强大，但其中所展示的设计方法，以及编程中的一些技巧还是值得借鉴的。通过本章的学习，应该对一个程序开发过程中包含的界面设计和逻辑控制等有一个简单、清晰的认识。

第 2 章　绘制股票 K 线图程序

股票 K 线图是一种很好的技术分析工具，它被广泛运用于股市的投资分析过程中。绘制股票 K 线图时，需要涉及大量的数据，并且还要对这些数据进行相应的处理，因此通过绘制股票 K 线图，能够帮助更深入地掌握关于数据获得、数据处理和界面绘图等相关技术，同时还可以帮我们了解股票 K 线图分析的原理和方法。

2.1　程序概述

K 线是股市行情分析中的一种参数指标，用股票每日的开盘价、最高价、最低价、收盘价及成交量等数据进行作图，配合五日、十日均线便可反映出一个阶段内该只股票的涨跌走势。本程序是根据每支股票交易的相关数据，绘制出相关图形，通过图形直观地反映股票的行情，程序运行时的界面如图 2.1 所示。

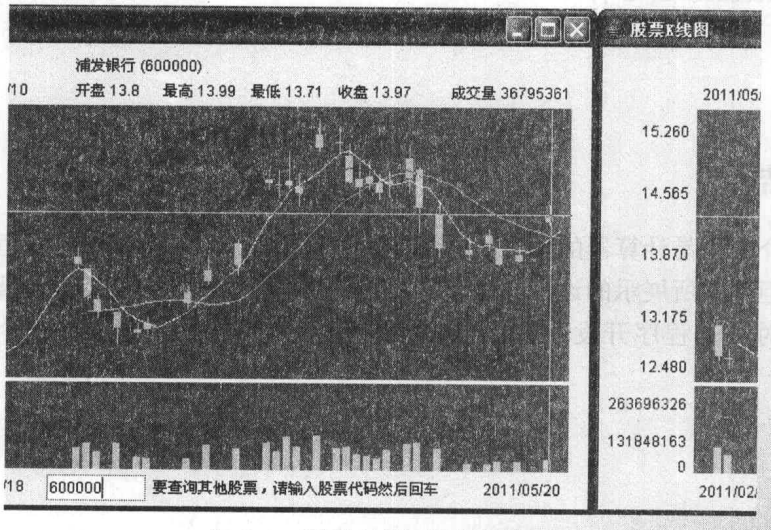


图 2.1　股票 K 线图运行效果图

2.2　程序的概要设计

在编写绘制 K 线时，有几处关键问题要解决，即：（1）数据输入流的应用；（2）界面