信息科学与技术丛书

Java编程模式与范例

——基础开发技巧

严桂兰 刘甲耀 刘波 编著

机械工业出版社

信息科学与技术丛书

Java 编程模式与范例
—基础开发技巧

严桂兰　刘甲耀　刘　波　编著

机 械 工 业 出 版 社

《Java 编程模式与范例》分为基础部分和高级部分两册。本书为基础开发技巧部分，共分为 8 种模式，近 200 个范例，取材广泛，由浅入深。内容涉及：Java 基本编程模式与范例；Java 基本编程构件与范例；使用对象工作的模式与范例；控制流结构的模式与范例；数组对象的模式与范例；Java 类与应用程序的模式与范例；Java 小应用程序的模式与范例；字符串处理的模式与范例。书后附录提供了 TextPad 与 JDK 的使用步骤。书中所有的范例，均在 Core Java 2（使用 TexPad 工具）环境中上机通过，实用性强，覆盖面广。许多例题采用多种解决方案，充分体现了 Java 编程的灵活性和趣味性。

本书可作为大专院校计算机及相关专业的配套教材，并可供各行各业从事计算机编程的人员参考。

图书在版编目（CIP）数据

Java 编程模式与范例：基础开发技巧/严桂兰等编著．—北京：机械工业出版社，2002.8
（信息科学与技术丛书）
ISBN 7-111-10781-0

Ⅰ．J…　Ⅱ．严…　Ⅲ．Jave 语言—程序设计　Ⅳ．TP312

中国版本图书馆 CIP 数据核字（2002）第 061211 号

机械工业出版社（北京市百万庄大街 22 号　邮政编码 100037）
策　　划：胡毓坚
责任编辑：孙　业
责任印制：路　琳

北京市樱花印刷厂印刷　·新华书店北京发行所发行

2002 年 8 月第 1 版·第 1 次印刷
1000mm × 1400mm B5·10.625 印张·412 千字

0001-5000册

定价：28.00 元

出 版 说 明

随着信息科学与技术的迅速发展，人类每时每刻都会面对层出不穷的新技术、新概念。毫无疑问，在节奏越来越快的工作和生活中，人们需要通过阅读和学习大量信息丰富、具备实践指导意义的图书，未获取新知识和新技能，从而不断提高自身素质，紧跟信息化时代展的的步代。

众所周知，在计算机硬件方面，高性价比的解决方案和新型技术的应用一直倍受青睐；在软件技术方面，随着计算机软件的规模和复杂性与日俱增，软件技术受到不断挑战，人们一直在为寻求更先进的软件技术而奋斗不止。目前，计算机在社会生活中日益普及，随着因特网延伸到人类世界的层层面面，掌握计算机网络技术和理论已成为大众的文化需求。正是这种在社会各领域的全方位渗透，信息科学与技术正在电工、电子、通信、工业控制、智能建筑、工业产品设计与制造等专业领域中得到充分、广泛的应用。相应地，这些专业领域中的研究人员和工程技术人员将越来越迫切需要汲取自身领域信息化所带来的新理念和新方法。

针对人们对了解和掌握新知识、新技能的热切期待，以及由此促成的人们对语言简洁、内容充实、融合实践经验的图书迫切需要的现代，机械工业出版计适时推出了“信息科学与技术丛书”。这套丛书涉及计算机软件、硬件、网络、工程应用等内容，注重理论与实践相结合，内容实用，层次分明，语言流畅，是信息科学与技术领域专业人员学习和参考不可或缺的图书。

现今，信息科学与技术的发展可谓一日千里，机械工业出版社欢迎从事信息技术方面工作的科研人员、工程技术人员积极参与我们的工作，为推进我国的信息化建设做出贡献。

机械工业出版社

前　言

Java是基于网络的纯面向对象的程序设计语言，适用于编写各式各样的软件，适用于各种平台与操作系统，编译后的代码能在互联网上传送，并确保用户安全运行，因而是当前最富生命力的计算机编程语言之一。

为适应当前 Internet 的迅猛发展及各行业学习 Java 语言的需要，特别是大专院校为研究生及本科生开设面向对象程序设计课程的需要，我们根据多年对 Java 教学和科研的实践以及 Java 版本的升级，特编写了《Java 编程模式与范例》，分为基础开发技巧和高级应用开发两册。本书阐述了基础编程模式与范例，共分为 8 种模式，近 200 个范例，取材广泛，由浅入深。其内容涉及：Java 基本编程模式与范例；Java 基本编程构件与范例；使用对象工作的模式与范例；控制流结构的模式与范例；数组对象的模式与范例；Java 类与应用程序的模式与范例；Java 小应用程序的模式与范例；字符串处理的模式与范例。所有范例均在 Core Java 2 环境下（使用 TextPad 工具）通过，并在附录中提供 TextPad 与 JDK 的使用步骤。

本书有以下三个特点：

1. 开发工具与语言相结合，本书采用了最新版本 Core Java 2 以及 TextPad 工具。

2. 取材广泛，由浅入深，重点、难点分明，易学，易掌握。

3. 编程方法与实例并举。

在本书编写中，刘涌博士提供了大量资料，江炜昌、马小路及黎颖等为本书做了大量录入工作，谨此表示感谢。

编　者

目　录

第 1 章　Java 基本编程模式与范例

:.1　基本编程模式

.1.1　Application（应用程序）

```
class 用户定义的类名    //定义类
{
  pubic static void main(String args[])      //调用 main( )方法
  {
    方法体
  }
}
```

注意：在关键字 class 前面亦可冠以关键字 public

1.1.2　Applet（小应用程序）

```
import java.awt.Graphics;          //引入 java.awt 软件包中的 Graphics 类
import java.applet.Applet;         //引入 java.applet 软件包中的 Applet 类
class 用户定义的类名 extends Applet      //定义类
{
  pubic void paint(Graphics g)     //调用 paint( )方法
  {
    方法体
  }
}
```

注意：其中

(1) import java.awt.Graphics;
import java.applet.Applet;

可写成

import java.awt.*;
import java.applet.*;

星号(*)用于说明 java.awt 和 java.applet 软件包中所有的类均可提供给编译器进行编译，但整个软件包被引入时，编译器将只装入该程序中用到的类，即“*”表示只装入(引入)该程序中用到的所有类。

(2) import java.awt.Graphics;
import java.applet.Applet;
class 用户定义的类名 extends Applet

可写成

```
import java.awt.Graphics;
class 用户定义的类名 extends java.applet.Applet
{
    ...
}
```

(3) 在关键字 class 前面亦可冠以关键字 public。

● 要使用 HTML 文件嵌入 applet，HTML 文件格式为

```
<HTML>
<HEAD>
<TITLE> 标题 </TITLE>
</HEAD>
<BODY>
<P>  分段说明  </P>
<APPLET
    CODE = 类名.class
    WIDTH = 宽度
    HEIGHT = 高度>
</APPLET>
</BODY>
</HTML>
```

也可简写成：

```
<APPLET
    CODE = 类名.class
    WIDTH = 宽度
    HEIGHT = 高度>
</APPLET>
```

1.2 范例

【例 1-1】 在屏幕上显示“The pattern and paradigm for Java programming”（使用 Java 编写 Application）。

方案一：(在 DOS 状态的命令行上输入源文件名运行程序)

```
//ParadigmApplication1.java
```

```
class ParadigmApplication1   //定义类 paradigmApplication
{
  public static void main(String args[])   //调用 main()方法,程序执行的起点
  {
    System.out.println("The pattern and paradigm for Java programming");   //打印语句
    //Sytem.out 是类变量,通过它调用 println()方法来实现
  }
}
```

运行结果:

```
C:\WINDOWS\JAVA>jviewd:\user\ParadigmApplication1↙
The pattern and paradigm for Java programming
```

注意:在 jview 运行本程序时,看不清楚运行结果,即结束程序。如果希望运行后看到结果,必须使用 Java 的 jview 编译器,并在命令行上输入目录与源文件名,如上所示,一般格式为:

```
C: \ WINDOWS \ JAVA > jview 目录 文件名 ↙
```

方案二:(使用 throws IOException 与 read( )方法)

```
//ParadigmApplication2.java
import java.io.*;   //引入 java.io 软件中所有类。这里是由于要用到 read()方法
class ParadigmApplication2
{
  public static void main(String args[]) throws IOException   //调用 main()方法,同时抛出异常
  {
    System.out.println("The pattern and paradigm for Java programming");
    System.in.read();   //等待输入数据(一般是按 Enter 键),目的是观看结果
    //这里是通过 System.in 类变量调用 read()方法来实现
  }
}
```

运行结果:

```
The pattern and paradigm for Java programming
```

注意:本程序也可在 Windows 命令行上输入源文件名来执行,只要键入"jview 目录 文件名"即可。本程序存放在 D: \ user 目录,因此,点击开始→点击运行,在运行框中输入

```
C: \ windows \ jview   d: \ user \ ParadigmApplication2 ↙
```

即得运行结果。

方案三:(使用 try/catch 与 read()方法)

1)

```
//ParadigmApplication3.java
import java.io.*;
class ParadigmApplication3
{
  public static void main(String args[])
  {
    try
    {
      System.out.println("The pattern and paradigm for Java programming");
      System.in.read();  //等待输入数据,通常是按 Enter 键,目的是观看结果
    }
    catch(Exception e)  //Exception 为异常类型,又为对象变量
    {
      System.out.println(e);  //打印出现异常的信息
    }
  }
}
```

运行结果:

```
The pattern and paradigm for Java programming
```

说明:try{…}catch{}为异常处理的结构模式,当一个方法引发一个异常后,可将异常抛出,由该方法的调用者处理异常。

2)

```
//ParadigmApplication4.java
import java.io.*;
class ParadigmApplication4
{
  public static void main(String args[])
  {
    try
    {
      System.out.println("The pattern and paradigm for Java programming");
      System.in.read();
    }
    catch(Exception e)
    {}  //空体,表示不要求输出出现异常的信息

  }
```

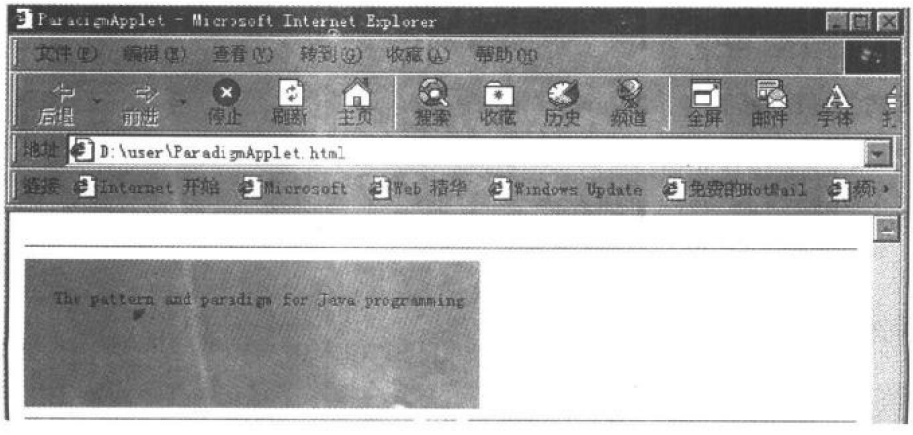
运行结果:

```
The pattern and paradigm for Java programming
```

【例 1-2】 在网页浏览器上显示“The pattern and paradigm for Java programming”(用 Java 编写 Applet)。

```
//ParadigmApplet.java
import java.awt.Graphics;   //引入 java.awt 包中的 Graphics 类
import java.applet.Applet;   //引入 java.applet 包中的 Applet 类
class ParadigmApplet extends Applet//定义 ParadigmApplet 类,它是 Applet 类的扩展
{
  public void paint(Graphics g)//调用 paint()方法,其参数 Graphics 类的对象 g
  {
    g.drawString("The pattern and paradigm for Java programming",20,30);   //打印语句(通过 g 对象调用
    //drawString()方法来实现)。其指定在 t 为 20,g 为 30(以像素为单位)位量开始显示结果
  }
}
//HTML 文件,这里是用于设置窗口宽度与高度
<html>
<head>
<title> ParadigmApplet </title>
</head>
<body>
<hr>
<applet
code = ParadigmApplet
width = 300
height = 100 >
</applet>
</hr>
</body>
</html>
```

运行结果:



【例 1-3】 输入一个整数,除以 2,并输出其结果(使用 Application)。

方案一:(DivideByTwo.java)

```
// DivideByTwo.java
//Input and output for data
import java.io.*;
public class DivideByTwo
{
  public static void main(String[] args)
  {
    DataInputStream is = new DataInputStream(System.in);  //创造 DataInputStream 类的对象 is
    //System.in 流用读取用户的键盘输入
    int iX;  //说明 iX 为整型变类
    String oneLine;  //说明 oneLine 为 String 型变量
    System.out.println("Enter an integer: ");
    try
    {
      oneLine = is.readLine();  //读入一行字符串
      iX = Integer.parseInt(oneLine);  //将字符串转换成整型数
      System.out.println("Half of iX is " + (iX/2));
      System.in.read();
    }
    catch(Exception e)
    {
     System.out.println(e);
    }
  }
}
```

运行结果:

```
Enter an integer: 1025
Half of iX is 512
```

方案二:(DivideByTwo1.java)

```
//DivideByteTwo1.java
//Use of input and output for data
import java.io.*;
public class DivideByTwo1
{
  public static void main(String[] args)throws IOException//调用 main()方法,同时抛出异常
  {
    DataInputStream is = new DataInputStream(System.in);  //创建数据输入流类对象 is
```

```
        int iX;  //说明整型变量 iX
        String sLine;  //说明字符串变量 SLine
        System.out.print("Enter an integer: ");  //提示输入一个整数
        sLine = is.readLine();  //读入一行字符串
        iX = Integer.parseInt(sLine);  //将读入的字符串转换成整数
        System.out.println("Half of iX is " + (iX/2));  //打印结果
        System.in.read();
    }
}
```

运行结果:

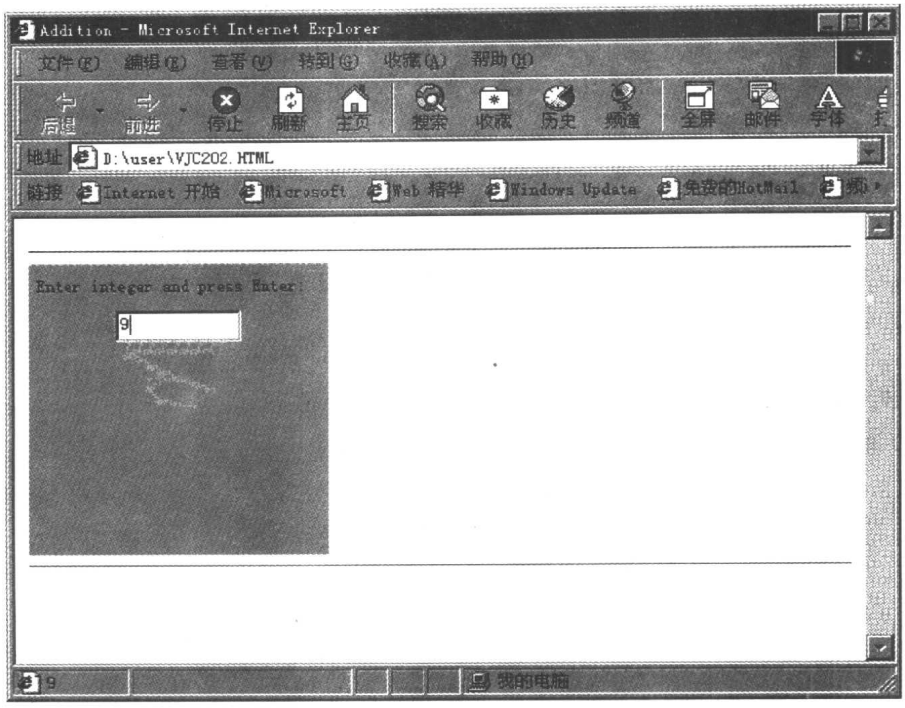
```
Enter an integer: 1025
Half of iX is 512
```

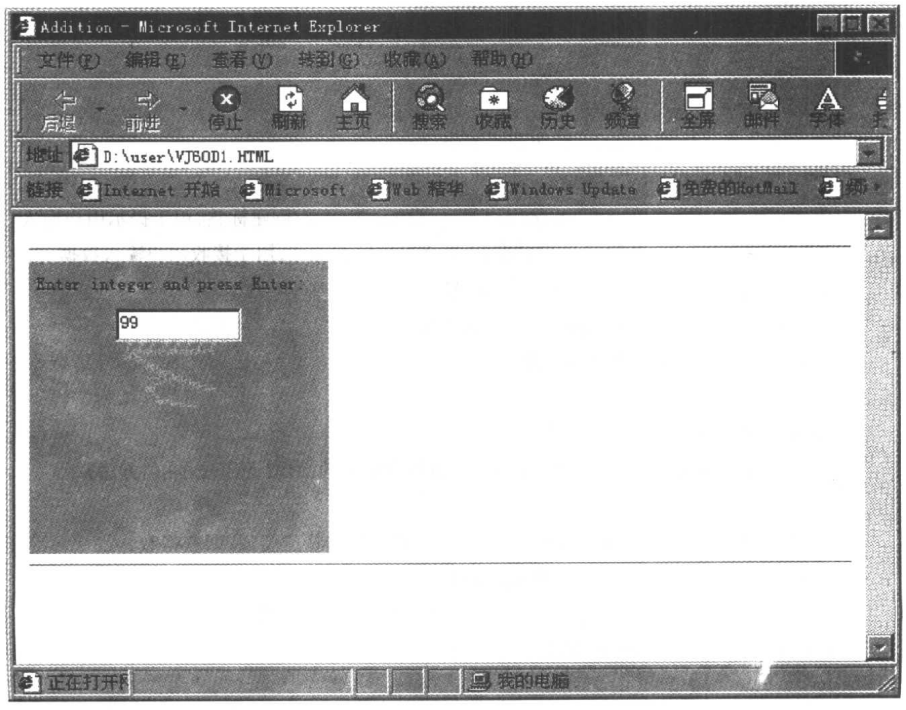
【例 1-4】 输入一个整数,逐个累加求和(使用 Applet)。

方案一:(Addition.java)

```
// Addition.java
import java.awt.*;
import java.applet.Applet;
class Addition extends Applet
{
  Label prompt;       // 定义标签类对象 prompt
  TextField input;    // 定义文本域对象 input
  int number;         // 说明整型变量 number,用于保存输入值
  int sum;            // 说明整型变量 sum,用于保存整数和
// 创建图形用户界面构件,并初始化
public void init()//完成 applet 的初始化工作
{
  prompt = new Label("Enter integer and press Enter:");  //创建标签,用于提示用户输入
  input = new TextField(10);  //创建 10 个字符宽的文本域,用于接收用户输入数据
  add(prompt);      // 在 applet 上添加标签
  add(input);       // 在 applet 上添加文本域
  sum = 0;          // 设置 sum 为 0
}
// 在输入文本域处理用户的动作
public boolean action(Event e, Object o)//完成用户所需的动作(调用 action()方法)
{
  number = Integer.parseInt(o.toString());  //将字串转换成整数,即获得数
  sum = sum + number;          // 将获得的数加入 sum
  showStatus(Integer.toString(sum));  // 在状态行上显示结果
  return true;  // 表示已处理完用户的动作
  }
```

运行结果：


Addition - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 转到(G) 收藏(A) 帮助(H)
后退 前进 停止 刷新 主页 搜索 收藏 历史 频道 全屏 邮件 字体
地址 D:\user\VJC202.HTML
链接 Internet 开始 Microsoft Web 精华 Windows Update 免费的HotMail
Enter integer and press Enter:
9
9
我的电脑


Addition - Microsoft Internet Explorer
文件(F) 编辑(E) 查看(V) 转到(G) 收藏(A) 帮助(H)
后退 前进 停止 刷新 主页 搜索 收藏 历史 频道 全屏 邮件 字体
地址 D:\user\VJ6OD1.HTML
链接 Internet 开始 Microsoft Web 精华 Windows Update 免费的HotMail
Enter integer and press Enter:
99
正在打开F
我的电脑

方案二:(Addition1.java)

```
// Addition1.java
import java.awt.*;
import java.applet.Applet;
public class Addition1 extends Applet
{
  Label prompt;
  TextField input;
  int number;
  int sum;
  public void init()//调用 init()方法,以完成 applet 的初始化工作
  {
  prompt = new Label("Enter integer and press Enter:");
  input = new TextField(10);
  add(prompt);
  add(input);
  sum = 0;
  }
public void paint(Graphics g)//调用 paint()方法,以完成计算,并在指定位置显示结果
{
  String Line = input.getText();
  number = new Integer(Line).intValue();
  sum = sum + number;
  g.drawString(Integer.toString(sum),90,80);  //在指定位置(90,80)显示结果
  g.drawString("Accumulative total is " + sum,20.100);  //在指定位置(20,100)显示结果
}

  public boolean action(Event e, Object o)//调用 action()方法
  {
    repaint();  //调用 repaint()方法,以实现 applet 重画
    return true;  //完成动作后返回真
  }
}
```

运行结果: