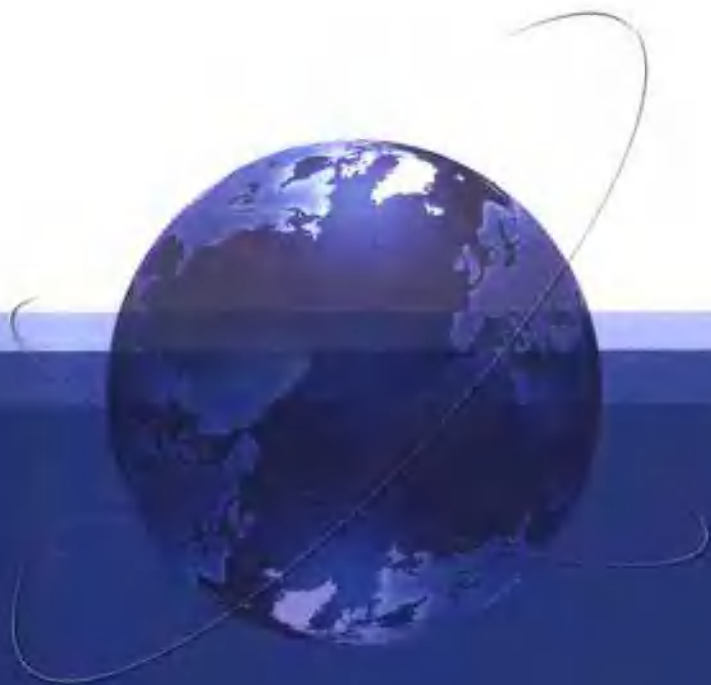




21世纪高职高专规划教材

Java程序设计



赵景晖 主编



机械工业出版社
CHINA MACHINE PRESS

21 世纪高职高专规划教材

Java 程序设计

主 编	辽宁机电职业技术学院	赵景晖
副主编	北京城市学院	杨锡平
参 编	西安理工大学高等技术学院	杨景林
	辽宁机电职业技术学院	孙莉娜
	安徽商贸职业技术学院	秦体德



机械工业出版社

本书是 21 世纪高职高专规划教材之一, 从介绍 Java 语言的基本特点开始, 循序渐进地阐述了 Java 语言的基本概念、面向对象的基本知识和编程方法, 内容包括 Java 语言的基本语法、类和对象的基础知识、类的高级特征和接口、异常处理、数组和字符串、图形界面设计、Java Applet、输入输出与文件管理、多线程程序设计、网络编程和数据库编程等。每章后附有大量复习思考题。

本书可作为高等职业技术学院、高等专科学校、职工大学、业余大学、成人教育学院、电视大学以及其他大专层次的信息类专业的教材, 也可供有关教师以及工程技术人员参考。

图书在版编目 (CIP) 数据

Java 程序设计/赵景晖主编. —北京: 机械工业出版社,
2005

21 世纪高职高专规划教材
ISBN 7-111-17245-0

I. J… II. 赵… III. Java 语言 - 程序设计 - 高等学校:
技术学校 - 教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2003) 第 095294 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 余茂祚

责任编辑: 余茂祚 版式设计: 冉晓华 责任校对: 魏俊云

封面设计: 饶 薇 责任印制: 洪汉军

北京京丰印刷厂印刷 · 新华书店北京发行所发行

2005 年 8 月第 1 版·第 1 次印刷

787mm × 1092mm 1/16 · 15.75 印张 · 388 千字

定价: 24.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

本社购书热线电话 (010) 68326294

封面无防伪标均为盗版

21 世纪高职高专规划教材

编 委 会 名 单

编委会主任 王文斌 郝广发

编委会副主任 (按姓氏笔画为序)

马元兴	王茂元	王明耀	王胜利	王锡铭
田建敏	刘锡奇	杨文兰	杨 颀	李兴旺
李居参	杜建根	余元冠	沈国良	沈祖尧
陈丽能	陈瑞藻	张建华	茆有柏	徐铮颖
符宁平	焦 斌			

编委会委员 (按姓氏笔画为序)

王志伟	付丽华	成运花	曲昭仲	朱 强
齐从谦	许 展	李茂松	李学锋	李连邨
李超群	杨克玉	杨国祥	杨翠明	吴诗德
吴振彪	吴 锐	肖 琬	何志祥	何宝文
陈月波	陈江伟	张 波	武友德	周国良
宗序炎	俞庆生	恽达明	姜 洁	晏初宏
倪依纯	徐炳亭	唐志宏	崔 平	崔景茂

总 策 划 余茂祚

策划助理 于奇慧

前 言

本书是根据教育部教高〔2000〕2号文件精神，由中国机械工业教育协会和机械工业出版社组织全国100多所高等院校合作编写的高职高专规划教材之一。

互联网技术在全球范围内突飞猛进地发展，给计算机的应用领域带来了巨大的变化，计算机应用的网络化已经成为发展的必然趋势。作为网络编程语言，Java已经受到越来越多IT行业科研人员的重视，其应用领域也越加广泛。因此，很多高等院校在相关专业相继开设了Java程序设计的课程。

Java是纯面向对象的程序设计语言，其良好的跨平台特性和安全机制使得它特别适合开发网络应用程序。Java是一种程序设计语言，更是一种构建互联网应用开发的工具和解决方案。作为信息类专业的学生，掌握Java编程技术具有十分重要的意义。

本书从Java的基本特点开始，逐步介绍了Java的基础知识，面向对象编程的概念和特点，面向对象编程的高级特性等。在此基础上，对Java的数组和字符串、异常处理机制、图形界面、输入输出与文件管理、线程、数据库编程和网络通信编程等进行了详细的阐述。为提高读者运用Java解决实际问题的能力，在每章中安排了较多的应用实例。本书在内容阐述方面，尽量做到语言浅显易懂；书中给出的实例保持风格一致，代码书写规范。

根据教育部有关文件精神及高职高专发展趋势，本教材基本上是按照高中毕业二年制高职高专教学计划所需内容编写的，同时兼顾三年制高职高专的教学计划。教学内容以工程应用为目的，以理论适度、讲清概念、强化应用为重点，突出实用性、综合性，注重学生基本技能的训练和综合能力的培养。本书可作为高等职业技术学院、高等专科学校、职工大学、业余大学、成人教育学院、电视大学以及其他大专层次的信息类专业的教材，也可供有关教师以及工程技术人员参考。

本教材共12章，其中第1、6、11、12章由辽宁机电职业技术学院赵景晖编写；第2、3章由辽宁机电职业技术学院孙莉娜编写；第4、5章由安徽商贸职业技术学院秦体德编写；第7、8章由北京城市学院杨锡平编写；第9、10章由西安理工大学高等技术学院杨景林编写。本书由赵景晖任主编，杨锡平任副主编。

本书在编写过程中得到辽宁机电职业技术学院领导的大力支持和热情帮助，在此也表示感谢。

由于编者水平有限，加之时间仓促，书中难免有错误和疏漏之处，恳请读者给予批评指正。

目 录

前言	
第 1 章 Java 语言概述	1
1.1 Java 语言的发展及特点	1
1.2 Java 虚拟机	2
1.3 简单的 Java 程序	3
1.4 Java 程序的编译与运行	5
1.5 常用开发工具介绍	5
复习思考题	14
第 2 章 Java 基本语法	16
2.1 Java 字符	16
2.2 数据类型、常量和变量	18
2.3 运算符、表达式和语句	21
2.4 语句和程序流程控制	27
复习思考题	36
第 3 章 类和对象的基础知识	39
3.1 面向对象程序设计的概念	39
3.2 类的概念	39
3.3 类的定义	40
3.4 对象的创建及类成员的引用	42
3.5 类的成员	45
3.6 类的继承	52
3.7 包 (package)	56
复习思考题	60
第 4 章 类的高级特性和接口	63
4.1 方法的覆盖	63
4.2 多态性	64
4.3 抽象类和最终类	67
4.4 接口	69
4.5 内部类	74
4.6 匿名类	76
复习思考题	78
第 5 章 异常处理	81
5.1 异常的概念	81
5.2 异常的分类及层次	82
5.3 捕获异常	82
5.4 丢弃异常	84
5.5 抛出异常	86
5.6 异常类常用的方法	87
5.7 自定义异常	88
复习思考题	90
第 6 章 数组与字符串	92
6.1 数组	92
6.2 字符串	100
复习思考题	106
第 7 章 图形界面设计	108
7.1 图形用户界面的基础知识	108
7.2 常用组件	109
7.3 布局管理	127
7.4 事件处理	134
7.5 图形图像处理	146
7.6 Swing 组件简介	149
复习思考题	151
第 8 章 Java Applet	155
8.1 Applet 基础	155
8.2 Applet 类	155
8.3 简单 Applet 程序举例	156
8.4 <applet> 标记语法	157
8.5 Applet 和 HTML 的参数传递	158
8.6 在 Applet 中显示图像	160
8.7 在 Applet 中播放声音	162
8.8 Applet 响应鼠标事件	164
8.9 Applet 响应键盘事件	167
复习思考题	169
第 9 章 输入输出与文件管理	174
9.1 流的概念	174
9.2 流的分类及层次	174
9.3 字符流类	175
9.4 字节流类	178

9.5 文件管理与操作	179	11.1 网络基础	208
9.6 文件的随机读写	182	11.2 Java 网络包 (Java.net)	209
9.7 文件流	184	11.3 使用 TCP 协议编程	211
9.8 对象的序列化	187	11.4 使用 UDP 协议编程	218
复习思考题	190	11.5 使用 URL 类编程	223
第 10 章 多线程程序设计	191	复习思考题	228
10.1 线程的概念	191	第 12 章 数据库编程	229
10.2 Java 的多线程机制	192	12.1 JDBC 的概念	229
10.3 线程的运行状态	192	12.2 JDBC 驱动程序的类型	230
10.4 线程的创建方法	193	12.3 JDBC API 简介	230
10.5 线程的优先级	197	12.4 创建与数据库的连接	231
10.6 多线程和线程同步	199	12.5 访问数据库	234
10.7 线程的调度	203	复习思考题	241
复习思考题	206	附录 安装和配置 MySQL 数据库	242
第 11 章 网络编程	208	参考文献	244

第 1 章 Java 语言概述

本章主要介绍 Java 语言的发展概况、技术特点和 Java 虚拟机的概念,用几个简单的例子说明 Java 程序的类型、结构以及编译和运行 Java 程序的方法。为使初学者能够以较高的效率开发 Java 应用程序,还介绍了 Java 辅助开发工具 Jcl 的使用方法。

1.1 Java 语言的发展及特点

1.1.1 Java 语言的发展

Java 语言诞生于 20 世纪 90 年代初,它是由美国 Sun 公司的一个开发小组在开发 Green 项目时完成的,该项目旨在为消费类电子产品开发一种交互式的软件系统。最初该小组尝试用 C++ 作为编程语言,但在项目进行过程中,发现 C++ 语言存在编程复杂、可移植性差等缺点。于是选择基于 C++ 开发一种新的计算机编程语言,该项目小组将之命名为 Oak。Oak 继承了 C++ 面向对象等优点,摒弃了 C++ 编程复杂、可移植性差等缺点。在 Sun 公司正式为该语言注册时,发现 Oak 已经被其他产品注册了,于是将该语言正式命名为 Java。

Java 诞生之初,并未引起计算机界专业人士的重视,直到 20 世纪 90 年代中期,随着 Internet 的蓬勃发展,受 Mosaic 和 Netscape 的启发,该项目小组用 Java 开发了浏览器软件 HotJava。该浏览器可以运行用 Java 编写的小程序,实现了在 Web 页中进行交互操作的功能。HotJava 上佳的表现,使计算机界专业人士认识到 Java 的巨大应用潜力和发展前景。如今,许多世界著名的计算机软硬件公司如 IBM、微软等相继购买了 Java 语言的使用许可,越来越多的软件和硬件产品将与 Java 技术结合,Java 已经成为网络、多媒体等编程的强大工具。

1.1.2 Java 语言的特点

同其他程序设计语言相比,Java 技术的发展十分迅速,其应用也越来越广泛,这与 Java 所具有的突出特点密切相关。归纳起来,Java 的主要特点如下:

1. 简单性 Java 是面向对象的编程语言,在语法规则上和 C++ 类似,C++ 程序员会比较容易地掌握 Java 编程技术。Java 摒弃了 C++ 容易引起错误的内容,如指针操作和内存管理。Java 本身提供了丰富的类库,程序员可以直接使用类的对象实现较为复杂的程序设计任务,使程序设计变得简单、高效。

2. 面向对象 面向对象是 Java 的重要特性。面向对象编程技术具有很多优点,比如通过对象的封装,减少了对数据非法操作的风险,使数据更加安全;通过类的继承,实现了代码的重用,提高编程效率等。Java 语言的编程主要集中在类、接口的描述和对象引用方面,面向对象编程技术适合开发大型、复杂的应用程序,且程序代码易于理解和维护。

3. 分布式 Java 是面向网络的编程语言,它提供了基于 TCP/IP 协议通信的类库,使用这些类,Java 应用程序可以很容易地访问网络上的资源。编译后生成的目标代码可以分布在网络的任何主机上,客户机能够方便地实现和这些目标程序的通信。

4. 健壮性 Java 提供了较完备的错误处理机制,在编译和运行程序时,系统对代码进行逐级检查,指出可能产生错误的地方,要求必须对可能存在错误的代码进行必要的处理,以

消除因产生错误而造成系统崩溃的情况。

5. 安全性 作为网络编程语言,安全是至关重要的。一方面,由于 Java 不支持指针,消除了因指针操作带来的安全隐患;另一方面,Java 具有完备的安全结构和策略,代码在编译和运行过程中,被逐级检查,可以防止恶意程序和病毒的攻击。另外,对下载到本地的 Applet,严格地限制其访问本地资源的权限,降低了利用 Applet 非法访问本地资源的风险。

6. 平台独立与可移植性 互联网是由各种各样的计算机系统组成的,如果应用程序在网络中任何计算机上都能够正常运行,必须使程序具有平台独立性,即软件本身不受计算机硬件和操作系统的限制。Java 是一种具有平台独立性的编程语言,Java 源程序编译后生成与具体机器指令无关的字节码,通过 Java 虚拟机(JVM)在不同的平台上运行。Java 的基本数据类型在设计上不依赖于具体硬件,为程序的移植提供了方便。

7. 解释执行 Java 是解释执行的编程语言,Java 源程序经过编译后生成被称作字节码(byte code)的二进制文件,Java 解释器直接执行字节码文件。解释器在执行字节码文件时,能够对代码进行安全检查,以保证没有被修改的代码才能执行,提高了系统的安全性。

8. 多线程 多线程机制使程序代码能够并行或重叠执行,而且线程的同步机制保证了对共享数据的正确操作。通过使用多线程,程序设计者可以用不同的线程完成不同的行为,而不需要采用全局的事件循环机制,这样就实现了网络上的实时交互行为。

9. 动态性 Java 在设计上力求适合不断发展的环境。在类库中可以自由地加入新的方法和实例而不会影响用户程序的执行。Java 通过接口来支持多重继承,使之比严格的类继承具有更灵活的方式且易于扩展。

1.2 Java 虚拟机

Java 虚拟机(JVM)是在实际计算机上用软件实现的计算机模型,它有自己的虚拟的处理单元、堆栈、寄存器等,还具有相应的指令系统。Java 语言的一个非常重要的特点就是平台独立性,使用 Java 虚拟机是实现这一特点的关键所在。

对一般的高级语言而言,如果要在不同的平台上运行,要用适用于不同平台的编译器对源程序进行重新编译,生成不同的目标代码,比如 Windows 系统下的 C++ 编译器生成的是 Intelx86 处理器上的二进制代码。Java 编译器生成的目标代码不是而向具体处理器的代码,而是而向虚拟机的中间代码,称为字节码(byte code),字节码只能在 Java 虚拟机上运行。

目前很多计算机系统都支持 Java 虚拟机。Java 虚拟机概括起来有三部分组成:类下载器、字节码校验器和执行单元。图 1-1 是 Java 编译器、虚拟机和计算机系统之间的关系。

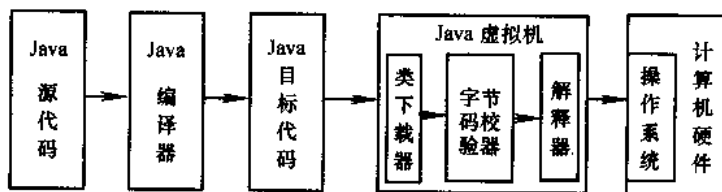


图 1-1 Java 编译器、虚拟机和计算机系统之间的关系

虚拟机用类下载器从计算机或网络上获取字节码,字节码经过字节码校验器验证之后,

送到执行单元执行。执行单元是一个解释器，它能够字节码翻译成具体计算机系统下的本地代码。一般情况下，Java 程序是解释执行的，解释执行的程序在运行速度上要比编译执行的程序慢一些。但对于支持即时编译（JIT 即 Just-In-Time compilation）的 Java 运行环境来说，Java 程序的运行效率会提高很多，因为 JIT 在字节码被装载时，会将其编译成能够在客户计算机上运行的本地代码，在这种运行环境中，只是在程序代码第一次被下载时，会使用较长的时间，在运行时，和本地代码具有同样的运行速度。

目前很多系统都提供 JIT 运行环境，比如常用的浏览器 Microsoft Internet Explorer、Netscape Navigator 等。除了桌面办公系统外，在嵌入系统和智能卡的芯片上已经实现了 Java 虚拟机，Java 将在个人家电、智能电话和网络计算机等设备上得到广泛应用。

1.3 简单的 Java 程序

使用 Java 语言可以开发基于某种平台的应用程序（Application）以及嵌入在网页中的小应用程序（Applet），应用程序又可分为字符界面应用程序和图形界面应用程序。

1. 字符界面的应用程序

例 1.1 编写一个在屏幕上显示“Hello! I like Java”的程序。

【程序代码】

```
public class HelloApp//定义公共类 HelloApp
{
    public static void main(String[] args)//应用程序入口即 main()方法
    {
        System.out.println("Hello! I like Java.");//输出 Hello! I like Java!
    }
}
```

例 1.1 是最简单的 Java 应用程序，它给出了 Java 应用程序的基本框架。类是构成 Java 程序的主体，class 是类的说明符号，类中包含了实现具体操作的方法，每个应用程序中必须包含一个主方法 main()，主方法是程序的入口。该程序运行后，在屏幕上显示“Hello! I like Java.”。

2. 图形界面的应用程序

例 1.2 编写一个基于 Windows 的窗口程序，窗口标题为“A simple GUI application”，在窗口内部显示“This is a simple GUI application.”。

【程序代码】

```
import java.awt.*; //用 import 导入 Java 类库中的类
import java.awt.event.*;
//定义 GUI 应用程序的公共类
public class GuiApp extends WindowAdapter
{
    public GuiApp() //定义方法 GuiApp()
    {
        //定义并创建框架 frm
```

```

Frame frm=new Frame("A simple GUI application");
Label lb=new Label();//定义并创建一个标签 lb
//设置标签文字属性
lb.setText("This is a simple GUI application.");
frm.setSize(300,100);//设置框架的尺寸
frm.add(lb);//将标签放到框架中
frm.setVisible(true);//设置框架的显示属性为“真”
frm.addWindowListener(this);//为框架注册监听器
}

public void windowClosing(WindowEvent e)//关闭框架的方法
{
    System.exit(0);
}

public static void main(String[] args)
{
    new GuiApp();//创建 GuiApp 类对象
}
}

```

例 1.3 使用了系统已经定义的类、对象和方法，如 WindowAdapter 类、Frame 类、windowClosing() 方法等，这些类的使用方法将在后续章节中详细介绍。

3. 嵌入在 Web 页面中的小应用程序 (Applet) Applet 应用程序是嵌入在 HTML (Hypertext Markup Language——超文本标记语言) 文件中的 Java 程序。它可以连同 Web 页面一起被下载到客户的浏览器中，并由实现了 JVM 的浏览器运行。编写 Applet 程序时，还需要编写出相应的 HTML 文件，并在文件中加上调用 Applet 程序的标记。

例 1.4 编写一个在浏览器中显示 “This is a simple Applet.” 的 Java 程序，并编写包含 Applet 的 HTML 文件。

【程序代码】

```

import java.applet.Applet;//打开 Applet 类所在的包
import java.awt.Graphics;//打开 Graphics 类所在的包
//创建继承 Applet 的类 SimpleApplet
public class SimpleApplet extends Applet
{
    public void paint(Graphics g)//调用 Applet 类的 paint() 方法
    {
        //绘制“This is a simple Applet.”
        g.drawString("This is a simple Applet.",50,50);
    }
}

```

用于编写 HTML 文件的工具很多，复杂的 HTML 文件可以用目前较为流行的可视化制作

工具如 FrontPage、Dreamweaver 等来创建，简单的文件可以用文本编辑工具直接编写。本例使用的 HTML 文件的代码如下：

```
<HTML>
<HEAD>
  <TITLE>The Simple Applet</TITLE>
</HEAD>
<BODY>
  <APPLET CODE="SimpleApplet.class"WIDTH=200 HEIGHT=100>
</APPLET>
</BODY>
</HTML>
```

在上面的代码中，“<APPLET CODE="SimpleApplet.class"WIDTH=200 HEIGHT=100>”是调用 SimpleApplet.class 的标记。可以用 Web 浏览器或用 JDK 提供的 appletviewer 运行 Applet 应用程序。

1.4 Java 程序的编译与运行

Java 源程序是文本文件，可以采用任何文本编辑器编写 Java 程序，如用 Windows 系统的 edit、记事本等。源程序的扩展名为 java，如 HelloApp.java。源程序须经过编译，生成字节码后才能运行。编译采用 JDK 提供的编译程序（javac.exe）。在 Windows 的命令行方式下，输入如下命令：

```
javac HelloApp.java
```

如果源程序没有错误，编译后将生成扩展名为 class 的类文件，该文件即为字节码文件，是在 JVM 下运行的目标代码。如果编译过程中出现错误，还需根据编译提供的错误信息重新编辑源程序，直到编译时没有错误为止。

完成编译后，可用 JDK 提供的解释程序（java.exe）执行生成的字节码文件，在 Windows 系统的命令行方式下，键入命令：

```
java HelloApp
```

屏幕上将显示“Hello! I like Java.”的信息。

以上过程是在 javac.exe、java.exe 和源程序都在同一路径的前提下完成的。在安装 JDK 时，假设其根目录为 j2sdk1.4.2（一般称为 JAVA_HOME），则 javac.exe、java.exe 存放在 j2sdk1.4.2\bin 目录下。如果源程序存放在用户自己建立的目录下，执行编译和运行程序时，应当指出编译程序和解释程序所在的路径，这可以通过设置系统环境变量的方式实现，具体方法见 1.5.2 节的内容。

1.5 常用开发工具介绍

1.5.1 常用开发工具简介

早期的 Java 开发人员采用 Sun 公司提供的 JDK（Java Development Kit）作为开发工具，随着 Java 影响力的逐步扩大，许多世界知名公司推出了功能更加完善的集成开发工具，如 Borland 公司的 Jbuilder，IBM 公司的 Visual Age for Java，微软的 Visual J++ 等。另外，还有许多

用于 Java 开发的辅助工具, 如 JCreator、UltraEdit、Gel、eclipse 等。

集成开发工具在很大程度上扩展了 JDK 的功能, 它可以生成各种应用程序的框架, 提供可视化程序设计环境, 使程序设计变得简单、高效。但多数集成开发工具需要它自己提供的类库的支持, 这是 Java 初学者难以掌握的内容。

一般的辅助开发工具, 能够把程序的编辑、编译和运行集成在一起, 尤其在程序的编辑过程中, 可以自动显示 JDK 提供的包 (package) 中的类名称、类的方法的名称等, 提高了编程效率。有些辅助工具是免费的, 如 eclipse、Gel 等。初学者可以在熟悉 JDK 工具后, 选择一个辅助开发工具, 以提高学习效率。

1.5.2 JDK 开发工具

JDK 是 Sun 公司免费提供的开发工具, 常见的版本有 JDK1.2.2、JDK1.3.0、JDK1.4 等。可以从 Sun 的网站 (<http://java.sun.com>) 上下载适合各种平台的 JDK 工具包。在 Windows 平台下, 下载名为 j2sdk-1_4_2_04-windows-i586-p.exe 的文件。该文件为自解压文件, 将文件安装在适当的目录下, 如“c:\jdk1.4.2”, 安装后的目录结构见图 1-2。



图 1-2 安装 jdk 后生成的目录结构

各目录存放的文件及用途如下:

- bin 存放编译、运行、调试等系统工具;
- include 存放与 C 相关的头文件;
- jre 存放与 Java 运行环境相关的文件;
- lib 存放 JDK 标准类库;
- demo 存放系统提供的应用实例。

在编译、运行和调试 Java 程序时, 所使用的工具存放在 bin 目录下, 这些工具的名称和作用如下:

javac.exe: Java 编译程序, 将 Java 源程序编译成字节码文件。

java.exe: Java 解释程序, 运行字节码文件。

appletviewer.exe: applet 浏览器, 执行包含 Applet 的 HTML 文件。

javadoc.exe: 根据 Java 源代码即说明语句生成 HTML 文档。

jdb.exe: Java 调试器, 可逐行执行程序, 设置断点和检查成员变量。

javap.exe: Java 反汇编工具, 显示字节码文件中的可访问方法和数据。

在安装好 JDK 以后, 为了编译、运行和调试程序的方便, 应当设置 Windows 系统的环境变量。一方面, 可以在任何目录下执行编译、运行和调试程序时, 不用输入这些程序的路径, 另一方面, 在执行类文件或字节码文件时, 也可以不用指出文件的路径。如果使用 Windows95/98 系统, 可以利用文本编辑工具, 在 AUTOEXEC.BAT 文件中加入“PATH=%PATH%;C:\J2SDK1.4.2\BIN”的内容。如果执行的类文件不在当前目录下, 还应当设置 CLASSPATH 参数, 比如类文件分别在“d:\javaok”和“f:\java”目录中, 可以在 AUTOEXEC.BAT 中加入“SET CLASSPATH=d:\javaok;f:\java”, 以便 Java 虚拟机能够找到相应的类文件。

对于 Windows 2000 系统, 需要按以下方式设置系统的环境变量:

1) 启动控制面板→选择“系统”→选择“高级”, 出现如图 1-3 的界面。

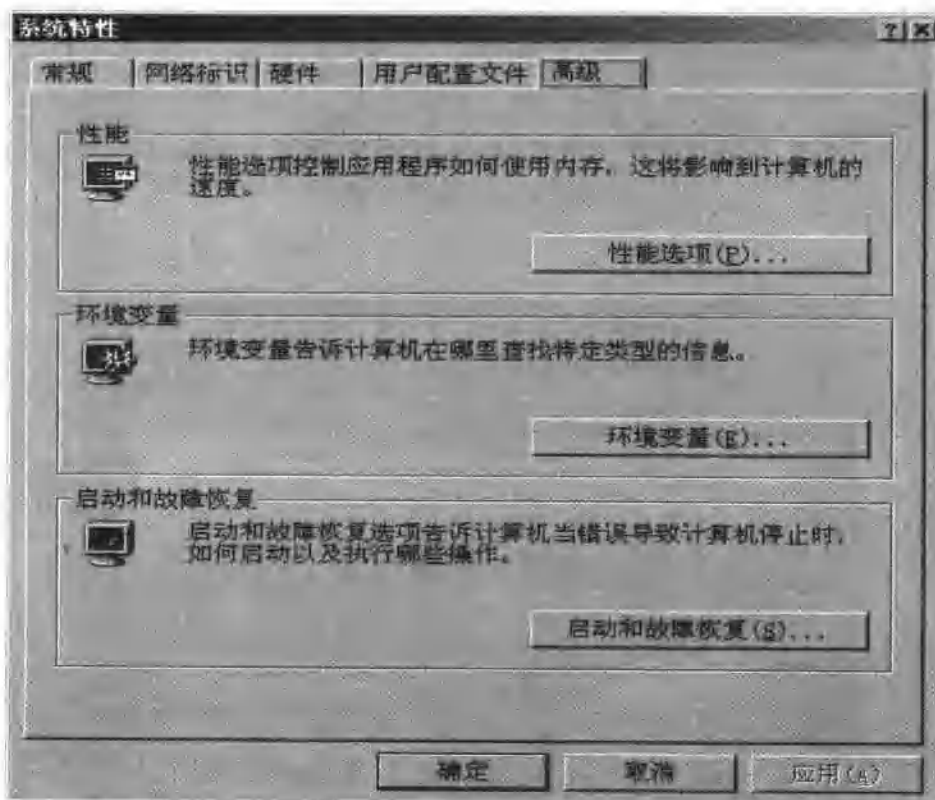


图 1-3 环境变量设置之一

2) 在图 1-3 中选择环境变量, 出现图 1-4 的界面。

3) 在图 1-4 中的“系统变量 (S)”窗口中, 利用窗口右侧的滚动条找到 Path 项目, 选中 Path 项, 出现如图 1-5 的界面。

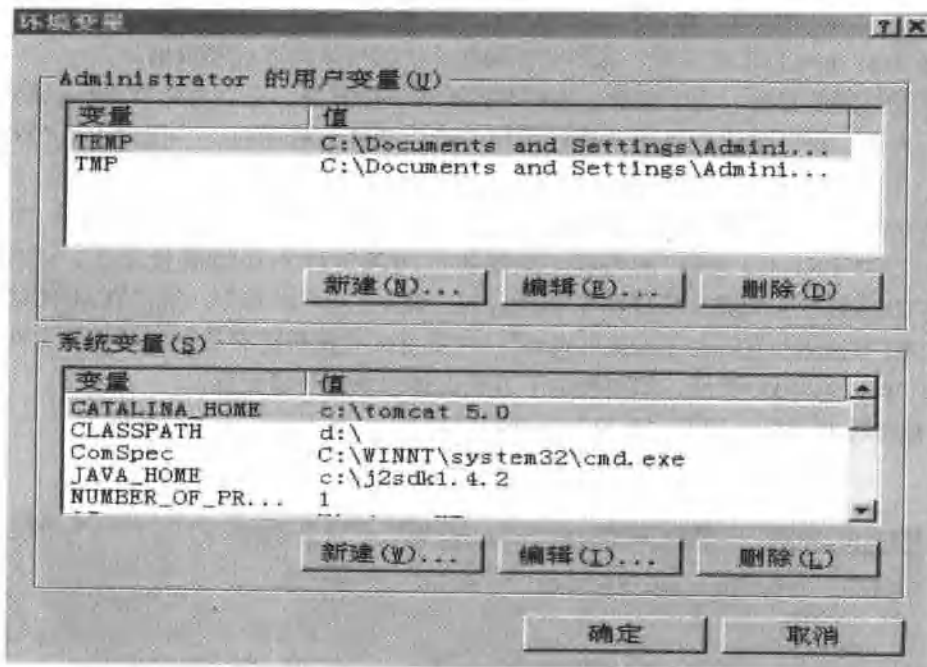


图 1-4 环境变量设置之二

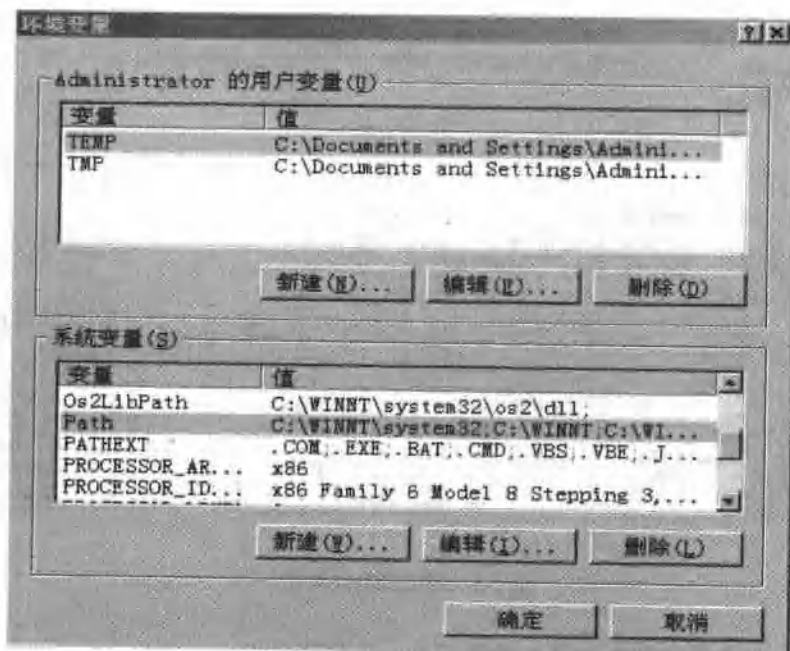


图 1-5 环境变量设置之三

4) 在图 1-5 中选择编辑按钮, 出现如图 1-6 的界面, 在图 1-6 中“变量值 (V)”编辑框中, 添加“c:\j2sdk1.4.2\bin”, 内容要用“;”号与原有内容分开。输入结束后, 用鼠标选择“确定”按钮, Path 参数设置结束。



图 1-6 环境变量设置之四

5) 可在此基础上设置 CLASSPATH 的参数。该参数指出执行类文件的查找路径, 在执行类文件时, Java 虚拟机要在 CLASSPATH 所指定的若干路径中查找相应的类文件。该参数中的内容是以“;”号分割的若干路径。在完成第四步用鼠标单击“确定”按钮以后, 界面重新返回到图 1-5。在图 1-5 中, 首先在“系统变量”的选择框中查找是否有 CLASSPATH 变量, 如果存在该变量, 点击“编辑”按钮, 如果没有 CLASSPATH 变量, 点击“新建”按钮, 出现图 1-7 所示的界面。



图 1-7 环境变量设置之五

在图 1-7 中, 在“变量名 (N)”编辑框中, 输入 CLASSPATH, 在变量值中输入需要查找的类路径, 选择确定按钮, 环境变量设置结束。

正确的设置环境变量, 如设置可执行文件的路径和类路径, 能够给编译、执行和调试程序带来方便, 提高编程效率。

1.5.3 Gel 辅助开发工具

Gel 是由 GExperts 公司免费提供的共享软件, 该软件可以从 <http://www.gexperts.com> 网站下载, 在 Windows 系统下, 下载名为 Gel RC33b.exe 的文件, 该文件是一个自解压的文件, 可以直接安装到用户的系统中。Gel 的开发者们正在不断地更新该软件, 目前会有更新的版本。使用 Gel 可以完成 Java 程序的编辑、编译和运行。下面, 介绍如何使用 Gel 来编辑、编译和执行 Java 程序。图 1-8 是 Gel 运行后的工作界面。

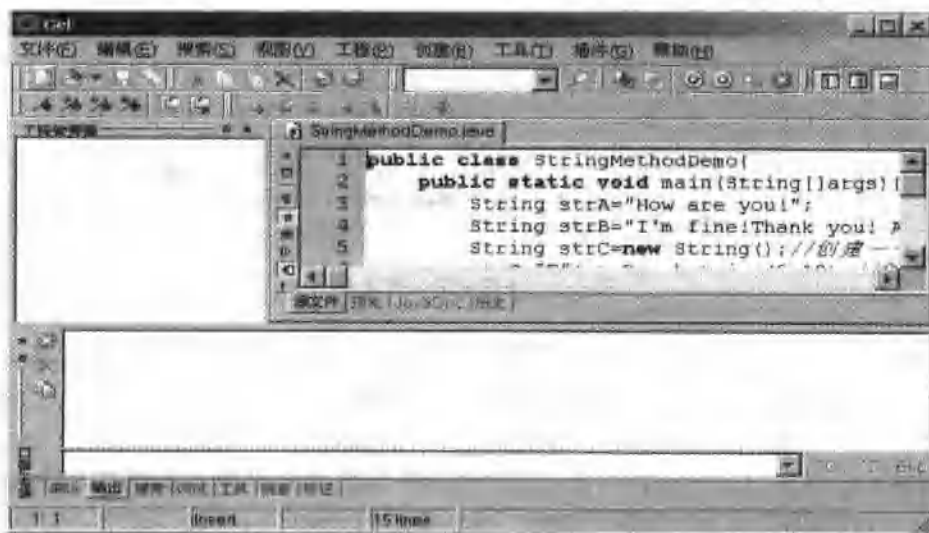


图 1-8 Gel 应用程序界面

1. Gel 运行环境的设置 Gel 是一个辅助开发工具, 该软件需要调用 JDK 来完成程序的编译和运行, 因此在使用 Gel 之前, 应当设置 Gel 的运行环境, 以确定编程风格、JDK 中应用程序的位置 (路径)、类库的位置等。

在 Gel 的主菜单上, “工具 (T)” 菜单中的 “选项 (O)” 项目, 用于设置 Gel 的环境参数。用鼠标点击该项目后, Gel 的界面如图 1-9 所示。界面分两部分: 左面是一个树型项目列表, 右面是以页面形式给出的参数设置列表。Gel 除了能够用于 Java 程序的编辑和调试外, 也可以用于其他语言如 C++ 等的编辑与调试。在选项列表中, 选择与 Java 相关的项目, 设置 Java 程序的编辑环境, 例如在输入左 “|” 时, 让编辑器自动加上右 “|”, 让系统保留字、变量等以不同的颜色显示出来等等。因为选项较多, 可以根据程序的编辑风格, 选中需要的项目, 合理的设置编辑风格会给程序的阅读和编辑带来很大的方便。

在这里必须配置的是 JDK 选项, 因为 Gel 需要执行 JDK 提供的编译、调试、运行等程序, 编译时还需要调用 JDK 提供的类库。该项目如果配置不正确, Gel 便不能正确完成编译、调试、运行程序等功能。图 1-10 是配置 JDK 编译程序的界面, 通过配置该项目, Gel 可以到指定的路径中找到相应的编译程序。具体方法是: 在图 1-10 右侧的页框中, 选择编译标签, 再在下面编译对应的文件栏目中, 输入 “javac.exe” 所在的路径, 如图 1-10 中输入的 “c:\j2sdk1.4\bin\javac.exe”, 则在执行编译时, Gel 自动到相应的路径中执行编译程序。用