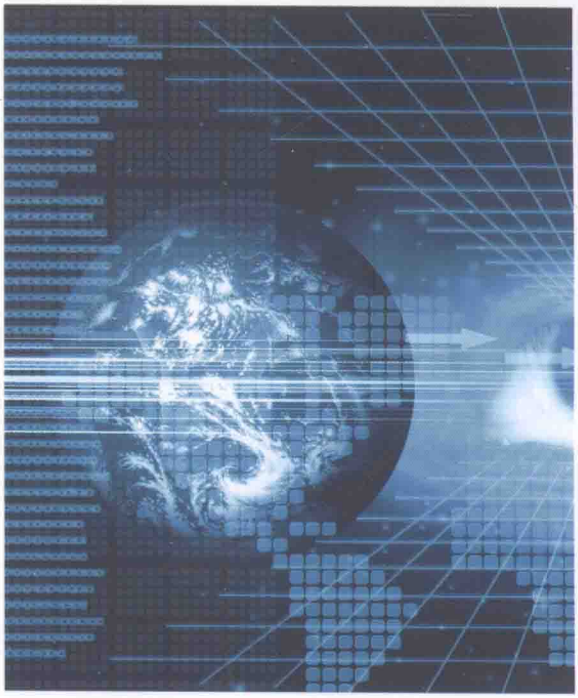
高等学校计算机应用规划教材

Java程序设计教程

(第二版)

- 程序设计基础
- 结构化程序设计→方法
- 面向对象程序设计→类
- 高级编程



林巧民　主编

姜玻　肖艳　何丽萍　副主编

清华大学出版社

高等学校计算机应用规划教材

Java 程序设计教程

(第二版)

林巧民　主编

姜玻　肖艳　何丽萍　副主编

清华大学出版社

北　京

内 容 简 介

本书以 Java 语言为基础，详细介绍计算机语言的结构化编程和面向对象编程。全书共分 12 章，主要内容包括：Java 入门、Java 编程基础、面向对象编程、继承、多态与接口、字符串、多线程与 Applet 技术、图形用户界面、Java I/O、Java 游戏开发基础以及游戏开发实例等。如果说结构化编程的特征是方法，那么面向对象编程的体现就是类的设计和使用，全书对这两种不同的程序设计思想都作了充分介绍。此外，每章的最后都配有思考练习，习题有选择题、填空题、简答题、编程题等多种类型，选择题、填空题和简答题有助于读者对所学知识的理解和掌握，编程题则可以提高读者的动手和实践能力。

本书结构清晰、内容翔实，既可以作为高等院校相关专业的教材，也可作为从事软件开发工作的专业技术人员的参考书。

本书对应的电子教案、实例源代码和习题答案可以到 http://www.tupwk.com.cn 网站下载。

图书在版编目(CIP)数据

Java 程序设计教程 / 林巧民 主编. —2 版. 北京：清华大学出版社，2015
(高等学校计算机应用规划教材)
ISBN 978-7-302-42199-3

Ⅰ. ①J… Ⅱ. ①林… Ⅲ. ①JAVA 语言—程序设计—高等学校—教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 279048 号

责任编辑：胡辰浩　袁建华
装帧设计：孔祥峰
责任校对：曹　阳
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课 件 下 载：http://www.tup.com.cn,010-62794504

印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：22.75　　**字　数**：525 千字
版　　次：2008 年 7 月第 1 版　2015 年 11 月第 2 版　　**印　次**：2015 年 11 月第1 次印刷
印　　数：1～4000
定　　价：45.00 元

产品编号：063207-01

前　言

Java 语言自面世以来，一直受到大学生和广大软件研发人员的青睐。目前，许多高校已改变先讲授 Pascal 语言或 C 语言，再让学生选修 Java 语言的惯例，而开始尝试让学生在大学低年级就学习 Java 语言。还有不少高校甚至对非计算机专业的大一新生就开设了 Java 课程。但目前，市面上大多数的 Java 教程在讲述面向对象技术时几乎都忽视了对 Java 语言基础的介绍，片面追求技术的新、奇、特，无法满足编程初学者的入门需要。

本书旨在突破市面上大多数 Java 教材的局限，尝试用一种语言来充分阐述两种编程思想，即结构化程序设计和面向对象程序设计，以满足普通初学者的需要。事实上，结构化程序设计是面向对象程序设计的基础，面向对象程序的基本组成还是结构化程序。面向对象程序设计引入了类的概念，使得程序设计人员可以站在设计类(而不是方法)的高度，对程序进行设计和实现，同时必须重视结构化程序设计基本功的锻炼，因为类的设计恰恰是建立在结构化程序设计的基础之上的。因此，本书以 Java 语言为工具，从结构化程序设计和面向对象程序设计两种不同编程思想的角度，分别对 Java 编程的相关基础知识予以介绍，希望能对广大编程爱好者尤其是初学者有所裨益。

全书共分 12 章，各章的主要内容如下。

第 1 章是 Java 入门，简要介绍 Java 的诞生、Java 语言的特点、Java 开发工具以及具体的开发步骤等。

第 2 章是 Java 编程基础，主要介绍 Java 的基本数据类型、赋值语句、条件表达式、运算符等。

第 3 章是 Java 程序基本结构，详细介绍程序的 3 种基本流程结构：顺序结构、分支结构和循环结构。

第 4 章是方法和数组，主要介绍方法的概念与定义、方法的调用、变量的作用域、数组以及数组与方法的关系等。

第 5 章是类和对象，详细介绍类的概念和定义、对象的创建与使用、访问控制符和包等。

第 6 章是继承、多态与接口，详细介绍了继承与多态技术、抽象类和接口等知识。

第 7 章是字符串，主要介绍 Java 提供的 String 和 StringBuffer 类。

第 8 章是多线程与 Applet 技术，详细介绍了线程的概念、创建、生命周期及状态、线程同步、优先级和调度等；还介绍了 Applet 的概念和工作原理、基本开发技术以及多媒体编程等。

第 9 章是图形用户界面，详细介绍 AWT 组件集中的常用组件，包括容器类组件、布局类组件、普通组件以及事件处理机制等。此外，本章最后还简要介绍了 Swing 组件集。

第 10 章是 Java 输入输出，详细介绍了 Java 输入输出流的概念、字节流类、字符流类、File 类以及 RandomAccessFile 类等。

第 11 章是 Java 游戏开发基础，介绍游戏编程的相关知识，包括图形环境的坐标体系、图形图像的绘制、各种坐标变换、动画的生成和动画闪烁的消除等。

第 12 章是游戏开发实例，以星球大战这款游戏为例介绍 Java 游戏的开发过程，同时还介绍了独立应用程序和小应用程序两种不同形式的游戏开发。

本书在编写过程中力求做到概念清楚、由浅入深、通俗易懂、论述详尽、实例丰富以方便读者自学。全书内容具有较强的实用性。

本书由林巧民、姜玻、肖艳和何丽萍共同编著。由于作者水平所限，书中难免会有不足之处，敬请广大同行和读者给予批评和指正。我们的邮箱是 huchenhao@263.net，电话是 010-62796045。

本书对应的电子教案、实例源代码和习题答案可以到 http://www.tupwk.com.cn 网站下载。

作　者

2015 年 8 月

目　录

第1章　Java入门

本章学习目标：

- 了解 Java 语言的历史和特点
- 理解 Java 与其他编程语言的关系
- 掌握 Java 语言开发环境的配置
- 掌握 Java 程序的基本结构及其编译运行过程
- 了解流行的 Java 语言集成开发环境

1.1　概　　述

Java 是由美国 Sun 公司(现已被 Oracle 公司收购)开发的支持面向对象程序设计的计算机语言。它的最大优势在于借助虚拟机机制实现了跨平台特性，即实现所谓的“一次编译，随处运行”，使代码的移植工作变得不再复杂。也正因为如此，Java 语言迅速流行起来，成为深受广大开发者喜爱的编程语言之一。目前，随着 Java ME、Java SE 和 Java EE 的发展，Java 已经不仅仅是一门计算机开发语言了。以 Java 为核心，已经拓展出了一系列先进的技术。

从应用的角度来看，Microsoft、IBM、DEC、Adobe、SiliconGraphics、HP、Toshiba、Netscape 和 Apple 等大公司均已购买了 Java 许可证，Microsoft 还在其 IE 浏览器中增加了对 Java 的支持。通过 Java 编写的应用程序可以在各种主流平台上运行。另外，对于开发者而言，众多的软件开发商都设计了许多支持 Java 开发的软件产品，如美国 Borland 公司的 JBuilder，蓝色巨人 IBM 的 Eclipse 和 Visual Age for Java，Sun 公司的 NetBeans 和 Sun Java Studio 5，以及 BEA 公司的 WebLogic Workshop 等。在数据库应用方面，数据库厂商 Sybase 也在开发支持 HTML 和 Java 的 CGI(Common Gateway Interface)。Oracle 公司甚至将自己的数据库产品用 Java 语言来进行开发。对于企业内部的 Intranet 而言，无论用户使用何种类型的机器和操作系统，界面都是统一的 Web 浏览器，而数据库、Web 页面(HTML 和用 Java 编写的 JSP、Servlet 等)、中间件(Java Bean 或 Enterprise Java Bean 等)则存在于 WWW 和应用服务器上。开发人员只需要维护一个软件版本，管理人员也省去了为用户单机安装、升级客户端以及培训人员等的繁琐工作。用户则只需要一个操作系统和一个 Internet 浏览器，即采用 B/S(浏览器/服务器)模式。B/S 与 C/S(客户端/服务器)模式最显著的不同就在于其是“瘦客户端”的，即程序运行对客户端的要求很低。Java EE 系列技术正是致力于帮助客户构建此类应用的。Java ME、Java SE 和 Java EE 的侧重点各有不同，现将其列举如下：

- Java ME(Java Micro Edition)是 Java 的微型版，常用于嵌入式设备及消费类电器(如手机等)上的 Java 开发。
- Java SE(Java Standard Edition)是 Java 的标准版，用于针对普通 PC 的标准应用程序开发。
- Java EE(Java Enterprise Edition)是 Java 的企业版，用于针对企业级应用服务进行开发。

Java ME、Java SE 和 Java EE 是 Java 针对不同的应用领域而提供的不同服务，即提供不同类型的类库。初学者一般可以从 Java SE 入手学习 Java 语言。Java SE 是一个优秀的开发环境，开发者可以基于这一环境创建功能丰富的交互式应用程序，并且可以把这些应用部署到不同的平台上。本书将以 Java SE 1.4 版本为例进行讲述，虽然该版本不是 Java 的最新版，但对于初学者的学习已经足够了。Java SE 1.4 版本具有 GUI 控制功能、快速的 Java 图形功能、支持国际化与本地化扩展以及新的配置选项，并对 Windows XP 提供扩展支持。此外，Java SE 还是多种不同风格软件的开发基础，例如，客户端 Java 小程序和应用程序，以及独立的服务器应用程序等。同时，Java SE 也是 Java ME 和 Java EE 的基础。事实上，大部分非企业级软件几乎都是在 Java SE 上开发部署的。首先，多数开发者选择 Java SE 进行应用软件的开发；其次，Java SE 和 Java EE 是兼容的，企业版是在标准版基础上进行扩充的，在 Java SE 上开发的软件，拿到企业版平台上可以无缝运行；再次，通常情况下手机及嵌入式设备上的应用开发和调试也是在 Java SE 环境中完成的。

1.1.1 Java 语言的诞生

早在1990年12月，SUN公司就由Patrick Naughton、Mike Sheridan和James Gosling成立了一个名为Green Team的小组。该小组的主要目标是发展一种分散式系统架构，使其能在消费级电子产品平台上执行，如PDA、手机、家用电器等。1992年9月3日，Green Team发布了一款名为Star Seven(*7)的机器。它有点像早期的PDA(个人数字助理)产品，不过有着比PDA更强大的功能，如无线通信(Wireless Network)、5寸彩色LCD、PCMCIA界面等。

Java 语言的前身 Oak 就是在那个时候诞生的，其主要目的是用来撰写在 Star 7 上运行的应用程序。为什么取名 Oak 呢？因为在 James Gosling 办公室的窗外，正好有一棵橡胶树(Oak)，于是顺手就取了这么个名字。Java 所提供的一些特性，在 Oak 中就已经具备了，例如安全性、网络通信、面向对象、垃圾收集(Garbage Collection)、多线程等。Oak 已经是一个相当优秀的程序设计语言，为什么 Oak 会改名为 Java 呢？因为当 Oak 要去注册商标时，发现已经有另外一家公司先使用了 Oak 这个名字。既然 Oak 不能用，那要取什么新名字呢？工程师们边喝咖啡边讨论着，看看手上的咖啡，灵机一动，就叫 Java 好了。Java 是印度尼西亚爪哇岛的英文名称，因盛产咖啡而闻名。就这样，它就变成了业界所熟知的 Java 语言了。

1995 年 5 月 23 日，JDK(Java Development Kits) 1.0 版本正式对外发布，它标志着 Java 语言的正式诞生。2014 年 3 月 18 日，Oracle 正式发布了 Java SE 8，这是 Java 的最新发行版本，其中包含许多新特性和 bug 修复。

1.1.2　Java 语言的特点

Java 语言之所以流行并得到广泛应用，这和它的优秀特性是分不开的。

1. 平台独立性

平台独立性意味着 Java 程序可以在任何支持 Java 语言的平台上运行。Java 虚拟机独立于所有其他软硬件而运行。因此，不管操作系统是 Windows、Linux、Unix 还是 Mac OS，也不管计算机是大型机、小型机还是普通 PC 机，甚至是 PDA 或手机、智能家电，Java 程序都能运行。当然，在这些平台上都要装有相应版本的 JVM(Java 虚拟机)，即运行平台必须支持 Java 语言。

现在，绝大部分手机都支持 Java，很多手机应用也采用 Java 开发。这样，任何支持 Java 的手机都能运行这些应用，这就是平台独立性所带来的好处，如图 1-1 所示。

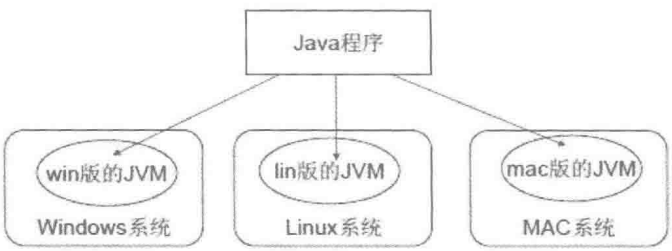


图 1-1　Java 应用程序可以跨平台运行

平台独立性保证了软件的可移植性，而软件的可移植性是软件投资在未来的保证。用 Java 开发的软件保证了程序在将来移植时的便捷性。其他语言，如 C 和 C++也宣称其是可移植的，但其移植工作仍然需要花费大量的时间和精力来适配不同的平台和硬件。

2. 安全性

目前用 Java 语言开发的应用很多都属于网络应用程序，因此，对软件的安全性有很高的要求。如果没有安全保证，用户运行从网络上下载的 Java 应用程序将会是十分危险的行为。Java 语言通过一系列安全措施，在很大程度上避免了病毒程序的产生和网络程序对本地系统的破坏，具体体现在如下几点：

(1) 去除指针这种数据类型，简化了编程，避免了程序对内存可能的非法访问。

(2) Java是一种强类型的程序设计语言，要求显式的声明。这样，可以保证编译器提前发现程序错误，提高程序的可靠性。

(3) 垃圾自动回收机制。让程序员从繁琐的内存管理工作中解脱出来，专注于程序逻辑本身的开发。更重要的是，通过这种内存自动回收机制，可以很好地保证内存管理的正确性，避免出现“内存泄漏”等问题。

(4) Java 语言提供了异常处理机制。

(5) Java 程序运行时，解释器会对其进行数组和字符串等的越界检查，确保程序的安全执行。

3. 多线程

在古老的 DOS 时代，人们一次只能运行一个程序，执行完一个程序后才能运行另一个。后来出现了视窗系统，如 Windows 之后，人们可以同时运行几个程序，并且可以在各个运行的程序之间进行切换，如一边听音乐一边编辑 Word 文档。这时的操作系统引入了进程的概念，每个运行中的程序都是一个进程。再后来，为了提高程序的并发性，又引入了线程的概念。

线程也称为轻量级的进程。进程是系统分配资源的基本单位，而线程则是 CPU 调度执行的基本单位。一个进程可以只有一个线程，也可以有多个线程。在很多情况下，开发多线程程序是很有必要的。例如，在早期单线程进程时代，安装程序在开始执行安装操作后，就只能一路安装下去了，而现在的软件安装程序一般都提供了“取消”操作，允许安装者在安装过程中随时取消安装，这也是软件“多线程”的一个表现。

多线程的目的就是降低总程序的执行粒度，让子程序们“同时”并发地执行。这里的“同时”只是为了强调CPU执行各个子程序的速度很快，从宏观上看，像是同时在执行。如果要实现真正的同时，就要借助多核甚至多处理器，如现在已经流行起来的双核、四核CPU。另外，随着程序规模的扩大以及对效率的重视，在线程之后又出现了纤程技术。纤程对线程又做了进一步细分，成为CPU调度的基本单位，使得人们在设计并发程序时更加灵活。

Java 是支持多线程程序开发的，它提供了 Thread 类，负责线程的启动运行、终止线程等，并可测试线程状态。后面章节会有关于多线程的专门介绍。

4. 网络化

在网络环境中，程序可以在本地或远程机器上执行。Java 程序可以通过网络来打开和访问对象，就像访问本地系统一样简单。Java 语言提供了丰富的类库，保证其可以在 HTTP、FTP 和 TCP/IP 协议中良好运行。Java Applet 程序需要客户端浏览器的支持，并且通过标签对<applet></applet>将 Java Applet 程序嵌入到 HTML 文件中。当用户浏览该 Web 页面时，Java Applet 程序才从服务器端下载到客户端，并解释执行。因此，也称 Java Applet 是可移动代码，这种移动性为分布式程序开发提供了一种新的途径。关于 Java Applet，后面章节会有详细的介绍。

5. 面向对象

随着软件业的蓬勃发展，面向对象程序设计方法已经流行起来，出现了很多面向对象的程序设计语言，如 C++、SmallTalk 等。Java 也属于面向对象程序设计语言。简单地说，面向对象主要是通过引入类的概念，使得原本面向过程的程序设计方法有了质的飞跃。类中不仅包含数据部分，而且还包含操作方法。这个囊括了数据和方法的类成为面向对象程序设计中最关键的要素。可以说，所有功能的实现都是围绕类展开的。同样，面向对象技术的特征也是由类体现出来的。面向对象主要有 3 大特征。

(1) 封闭性

类定义的一般形式如下：

```
class 类名
{
    类的细节
};
```

其中，“类的细节”以类的形式封装起来了，该细节就是类的成员。它可以是数据，也可以是操作这些数据的方法(在面向过程程序设计中称为函数)。当这些数据和方法的访问权限被设置为私有后，它们就不能被其他对象从外部访问，像是被隐藏了起来。而对外部只暴露那些访问权限被设置为公有的成员。

(2) 继承性

类是可继承的，就像遗产一样。一个已经设计好的类，可以被其他类继承，并扩充数据和方法部分，成为功能更为强大的新类。这样可以大大提高程序的可复用性，提高程序的开发效率，同时也能降低系统复杂性，提升代码的可读性。

(3) 多态性

在传统的面向过程程序设计中，函数一般是不能重名的。即便功能类似而只是处理的数据类型不同的函数也需要不同的名字，这就大大增加了程序的复杂性和维护的难度。而在面向对象程序设计中允许同名方法的存在。多态性正是说明系统如何在多个同名方法中找到正确的一个执行的机制。多态性也是面向对象技术的三大特征之一。

1.1.3　Java 与其他编程语言间的关系

程序开发语言可以分为 4 代：机器语言、汇编语言、高级语言和面向对象程序设计语言。机器语言是机器最终执行时所能识别的二进制序列，任何其他语言编写的程序最终都要转化为相应的机器语言才能运行。在电子计算机刚刚诞生的一小段时间内，人们只能用 0、1 序列进行编程，难度可想而知。后来为了提高编程效率，引入了英文助记符，这样就出现了汇编语言。汇编语言的出现，大大提升了代码的编写速度，同时也使代码可读性和可维护性大大提高。直到今天，仍然有人在用汇编语言进行编程。当然，这主要是因为效率的关系。系统底层应用，如一些硬件驱动类的代码，用汇编语言的执行效率更高。但是，汇编语言仍然不容易理解，对程序员自身有很高的要求。这就限制了其他领域的科技工作者们利用计算机进行辅助工作的可能性。为了普及计算机，使之成为社会各行业的一种工具，就需要开发语法简单，编写容易的编程语言。Bill Gates 的第一桶金据说就是从这个需求中赚来的，他在大学时代设计并开发了 Basic 语言，并将其出售给 IBM 公司。除了 Basic 语言，还有很多其他高级语言，如 Pascal、Fortran、C 等。随着软件行业的不断发展，软件的规模变得越来越大，迫切需要更高效的编程语言，Java、C++、Visual Basic 和 Delphi 等应运而生。除此之外，世界上还有很多其他编程语言，只是它们不是很流行，并不为人们所熟知。事实上，每一种流行的开发语言都有其优缺点：C 语言适合开发系统程序，很多操作系统及驱动程序都是用 C 语言编写的；Fortran 适合用来进行数值计算；Pascal 语言结构严谨，适合作为教学语言；Visual Basic 和 Delphi 适合用来开发中小型可视化应用程序；C++适合开发大型系统级应用程序；Java 适合开发跨平台的应用程序。

总之，每种语言都有其特色，至于选用什么语言作为开发工具，要看具体的开发需求。没有最好的，只有合适的。很多开发任务可能需要同时使用几种语言共同来完成。本教程主要面向没有任何编程基础的初学者学习Java语言而编写，下面就开始简单的Java之旅吧。

1.2　Java 开发环境配置

如果只是想运行 Java 编写好的程序，则只需下载 Java 运行时库 JRE(Java Runtime Environment)即可。但如果要用 Java 语言进行开发、编译和运行 Java 程序，则需要下载并安装 JDK(Java Development Kits)。

1.2.1　软件安装

从 Oracle 的官方网站(http://www.oracle.com/)上可以下载所需要的软件。下载时，读者需要关注自己所使用的操作系统版本(Windows XP/7/8或其他)以及是32位还是64位的系统。单击 Oracle 网站首页的“Downloads(下载)”链接，并选择“Java for Developers”后，可以下载最新版本的 Java SE。如果想下载老版本的 Java，可以在打开的页面中选择“Java Archive”。下面以 Windows XP 32位系统为例，从“Java Archive”里下载安装包 j2sdk-1_4_0_03-windows-i586.exe。该文件是一个自解压文件，双击即可解压缩，并进行安装工作。安装程序首先收集一些信息，用于安装的选择，然后才开始复制文件、设置 Windows 注册表等。安装过程中，只需按照提示一步一步操作即可。假设安装目录设置为 C:\jdk1.4(注意：该路径后面配置环境变量时要用到)。安装完毕后，切换到 C:\jdk1.4目录，可以发现有如下一些子目录。

(1) bin 文件夹：该文件夹包含了 Java 编译器(javac.exe)、解释器(java.exe)、Applet 查看器(appletviewer.exe)等 Java 命令，如图 1-2 所示。

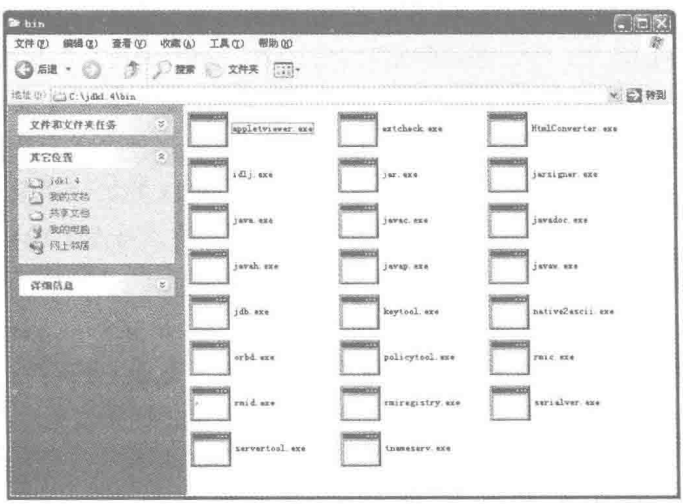


图 1-2　Java 安装目录中的 bin 文件夹

(2) demo 文件夹：包含一些示例程序的源代码。

(3) lib 文件夹：存放了一系列 Java 类库。

(4) jre 文件夹：存放 Java 运行时可能需要的一些可执行文件和类库。

(5) include 文件夹：存放头文件。

以上目录中，bin 文件夹是需要特别注意的，因为这个目录中的编译器(javac.exe)、解释器(java.exe)是后面编译程序时需要用到的。另外，最好将这个目录的绝对路径(如 C:\jdk1.4\bin)设置到操作系统的环境变量 path 中，这样，在进入命令行窗口后就可以直接编译和执行 Java 程序了。

1.2.2　环境变量配置

由于程序开发过程中需要经常用到各种头文件、库文件和编译器文件等，一种比较方便的做法是把这些文件对应的路径添加到系统环境变量中，以方便开发环境能够找到所需要的文件。

不同的操作系统，其设置系统环境变量的方法也各不相同，这里我们以 Windows XP 操作系统为例，设置环境变量的具体操作如下：

(1) 在桌面上右击“我的电脑”图标，从弹出的快捷菜单中选择“属性”命令，打开“系统属性”对话框。然后选择“高级”选项卡，如图 1-3 所示。

(2) 从打开的“系统属性”对话框的“高级”选项卡里，单击“环境变量”按钮，打开“环境变量”对话框，如图 1-4 所示。

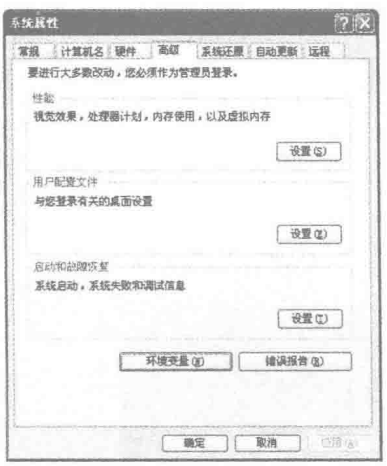


图 1-3　“系统属性”对话框

图 1-4　“环境变量”对话框

(3) 在“环境变量”对话框中，“XX 的用户变量”选项组中的内容是用户个人的环境变量，而“系统变量”选项组中的内容是系统变量。它们的区别是：用户变量只对本用户有效，且设置后无须重新启动计算机；而系统变量则对任何用户均有效，设置后需重新启动计算机才能生效。一般情况下，配置为用户变量即可。这里共需要配置两个用户变量：path 和 classpath。

(4) 如果原本没有 path 用户变量，就新建一个，并将变量值设置为 C:\jdk1.4\bin，如图 1-5 所示。

如果原本就有 path 用户变量，则只需重新编辑它，并在变量值的最后添加“;C:\jdk1. 4\bin”，然后单击一系列“确定”按钮后即可生效。生效后在命令行窗口中输入 javac 命令，将有如图 1-6 所示的信息。

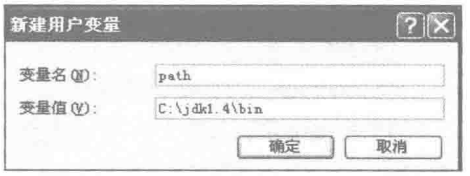


图 1-5　新建 path 用户变量

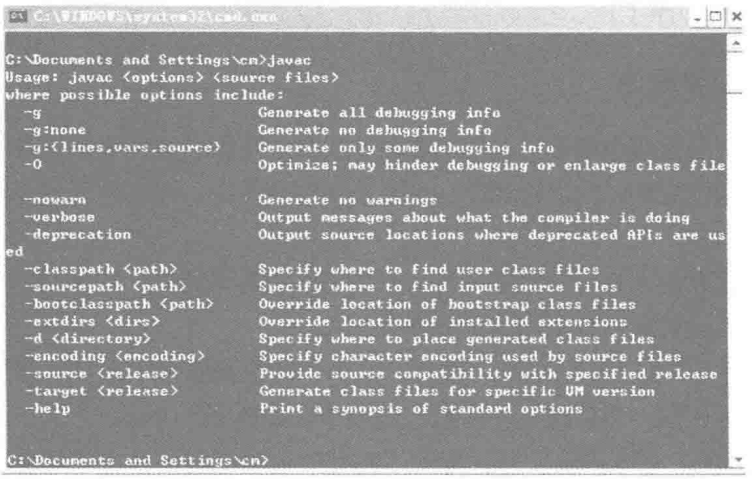


图 1-6　path 设置生效后可直接调用 Java 命令

(5) 如果原来没有 classpath 用户变量，则新建一个，并将变量值设置为“C:\jdk1.4\lib”。此外，当运行所编写的 Java 程序时，一般还需要将相应的工作目录(即存放 Java 程序及编译过的字节码文件的目录)添加到 classpath 变量值中，以便程序运行时能找到用户所编写的 Java 类。这一点一定要格外注意，很多人在初学 Java 时会忘记，导致程序运行失败。

设置完上述环境变量后，可以通过选择“开始”|“运行”命令，输入 cmd，在打开的命令行窗口中输入 set 命令，验证刚才的设置是否成功，如图 1-7 所示。

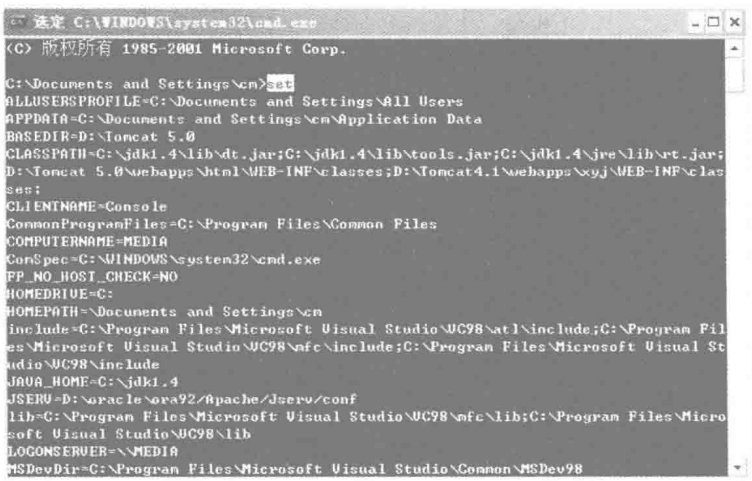


图 1-7　通过 set 命令查看环境变量设置情况

1.3　第一个 Java 程序

1.3.1　Java 程序的结构

用 Java 编写的程序有两种类型：Java 应用程序(Java Application)和 Java 小应用程序(Java Applet)。虽然二者的语法是完全一样的，但后者需要客户端浏览器的支持才能运行，并且运行前须将其嵌入到 HTML 文件的<applet>和</applet>标签对中。当用户浏览该 HTML 页面时，首先从服务器端下载 Java Applet 程序，然后被客户端已安装的 Java 虚拟机解释并执行。由于 Applet 与 HTML 联系紧密，且编程相对复杂，因而将其放至后面章节中详细讲述，这里只对 Java Application 程序的结构进行介绍。下面看一个最简单的 Java 程序。

```
public class Hello {
    public static void main(String args[])
    {
        System.out.println("Hello, welcome to Java programming.");
    }
}
```

Java 源程序是以文本文件的形式存放的，文件扩展名必须为.java。上面的程序代码，可将其保存为 Hello.java 文件。这里有一个非常细小但一定要注意的问题：文件名必须与(主)类名一致，包括字母大小写也要一致，因为 Java 语言是大小写敏感的。通常在定义类时，类名的第一个字母都大写。在正确编辑以上代码后，保存时应确保文件名正确，否则后面将不能通过编译，更无法运行。

所有的Java语句都必须以英文的分号“;”结束。编辑程序时千万注意不要输入中文的“；”，因为中文“；”不能被编译器识别。此外，Java语言是大小写敏感的，编辑程序时应注意区分关键字及标识符中的大小写字母。

下面通过图 1-8 对该程序的结构做简要介绍。

图 1-8　第一个 Java 程序

上述程序中，首先用关键字 class 声明一个新类，类名为 Hello，它是一个公共类(public)。整个类的定义由一对大括号{}括起来。在该类中定义了一个 main()方法。其中，public 表示访问权限，指明所有类都可以调用(使用)这一方法；static 指明该方法是一个静态类方法，