



普通高等教育“十一五”国家级规划教材

教育部“高等学校教学质量与教学改革工程”立项项目

冯洪海 主编

王世卿 主审

Java面向对象 程序设计基础教程

计算机科学与技术专业实践系列教材

清华大学出版社





普通高等教育“十一五”国家级规划教材

计算机科学与技术专业实践系列教材

Java面向对象 程序设计基础教程

冯洪海 主编

王世卿 主审

清华大学出版社
北京

内 容 简 介

本书主要通过 Java 语言讲解面向对象技术。面向对象一章较其他教材在内容上更详尽,且增加了贴近实际问题的综合应用题,从中可以理解如何应用 Java 语言编写出符合面向对象思想的代码。基本结构程序部分由于其基础性,本书仍给予较详尽的介绍。图形用户界面一章,为了降低事件部分与图形部分的耦合,引入内部类处理事件响应。多线程、异常、Applet、输入输出和文件操作、网络通信、数据库应用等则给予了适当的介绍。

本书不仅讲解语法现象,让读者知其然(what),而且讲解 Java 语法设计的目的,让读者知其所以然(why),更重要的是,针对每个重点语法现象,都给出了如何进行编程的建议(how)。

本书可作为普通高等学校计算机与相近专业 Java 语言程序设计或面向对象技术的课程教材,也可作为软件开发人员的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

Java 面向对象程序设计基础教程/冯洪海主编. —北京:清华大学出版社, 2011.6
(计算机科学与技术专业实践系列教材)
ISBN 978-7-302-25693-9

I. ①J… II. ①冯… 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(2011)第 103135 号

责任编辑:汪汉友

责任校对:焦丽丽

责任印制:李红英

出版发行:清华大学出版社

<http://www.tup.com.cn>

社 总 机:010-62770175

投稿与读者服务:010-62795954, jsjjc@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

地 址:北京清华大学学研大厦 A 座

邮 编:100084

邮 购:010-62786544

印 装 者:清华大学印刷厂

经 销:全国新华书店

开 本:185×260

印 张:17.75

字 数:431 千字

版 次:2011 年 6 月第 1 版

印 次:2011 年 6 月第 1 次印刷

印 数:1~4000

定 价:27.00 元

产品编号:036861-01

普通高等教育“十一五”国家级规划教材
计算机科学与技术专业实践系列教材

编 委 会

主 任：王志英

副 主 任：汤志忠

编 委 委 员：陈向群 樊晓桢 邝 坚

吴 跃 张 莉

前 言

面向对象程序设计方法是目前软件开发的主流方法。Java 语言是目前功能最强、应用最广泛的一种面向对象程序设计语言,具有面向对象、平台无关、多线程以及强大的网络编程功能。因此,多数院校用 Java 语言讲解面向对象技术。

本书有以下几大特色。

(1) 面向对象内容加深加宽。掌握了面向对象思想,即便是应用 C 语言,也能写出面向对象的代码;而没有掌握面向对象思想,即便是应用 Java 语言,写出的也是面向过程的代码。为此,我们采取了以下几个措施。

① 将面向对象放至基本结构程序之前讲。目前国内多数教材都是将基本结构程序放在面向对象之前,因而所有代码不得不放入 main 方法中,有很强的面向过程特征,因而到了“面向对象”一章,学生一时转不过弯来,还习惯将所有代码放入 main 方法中,并在 main 方法中直接赋值。

② 增加内部类的分量。一般教材在这一部分着墨不多,但《Java 编程思想》一书对于内部类叙述得较详细。接口和内部类是 Java 引以为傲的两部分,接口部分地实现了多继承,而内部类完整地实现了多继承。在某些逻辑复杂的地方,用内部类可以较简单和优雅地解决问题。

③ 多数教材在访问权限、初始化顺序、多态等部分叙述较简单,甚至有概念模糊的地方,本教材则叙述详细,概念解释更清楚。

(2) 将图形用户界面提前。原因是图形用户界面直观,学生学起来容易、有兴趣,而且该部分内容面向对象性较强。

(3) 引入 Java 编程原则,尤其是面向抽象编程和开闭原则。将代码的编写、测试和维护联系起来,通过应用组合、多态等解决继承的逻辑耦合问题。

(4) 针对一些语法难点和重点,讲述 Java 为什么设计该语法,以及如何应用该语法编程,解决学生学完 Java 语法不知道如何应用语法编程的问题。

(5) 在内容较多或层次多的段落设置段首语。段首语提纲挈领,有助于学生阅读,也可作为教师讲解的详细提纲。

本书由冯洪海主编,朱小柯、李建伟副主编,周星、陈丹捷、周兵、刘宇、杜莹、乔保军、王海荣等参加了部分编写。胡山峰参与了一些程序调试工作。全书由王世卿教授审阅。感谢王教授的审阅以及清华大学出版社的支持。书中有一些资料借鉴于网络,在此感谢那些在网上提供资料的作者。

书中有不少是作者一家之言,对于不妥之处,欢迎批评指正。

本书的全部例题、习题、实验题和配套课件可从清华大学出版社网站本书相应页面免费下载。

编者

2011 年 5 月

• III •

目 录

第 1 章 Java 语言概述	1
1.1 Java 语言的起源及发展	1
1.1.1 起源	1
1.1.2 发展	2
1.2 Java 语言的特点	2
1.3 Java 程序是如何在系统中运行的	5
1.4 Java 开发环境	5
1.5 编写、编译与运行 Java 程序	7
1.6 包	10
1.7 应用 Eclipse 开发项目	13
1.7.1 Eclipse 开发环境	13
1.7.2 创建 Java 项目并运行	14
1.7.3 程序调试技术	17
第 2 章 Java 基本语法规则	20
2.1 标识符与保留字	20
2.1.1 标识符	20
2.1.2 保留字	20
2.1.3 分隔符	21
2.1.4 注释	21
2.2 数据类型	22
2.2.1 整数类型	22
2.2.2 浮点数类型	23
2.2.3 布尔类型	23
2.2.4 字符类型	24
2.2.5 数据类型之间的相互转换	24
2.2.6 变量与常量	25
2.3 运算符和表达式	27
2.3.1 运算符	27
2.3.2 表达式	33
2.3.3 运算符优先级	33
2.3.4 编程风格	34
第 3 章 Java 中的面向对象技术	36
3.1 现实世界由对象组成	36

3.2	人类的思维方式	37
3.3	对象模型的特点	37
3.4	过程模型的特点	39
3.5	类及其成员	39
3.5.1	类	40
3.5.2	对象	41
3.5.3	成员变量	44
3.5.4	成员方法	45
3.5.5	构造方法	49
3.5.6	方法的重载	50
3.5.7	this 引用	51
3.5.8	static 成员	53
3.5.9	成员的访问权限	55
3.5.10	类的访问权限	58
3.6	类的继承	59
3.6.1	继承	60
3.6.2	对象的赋值相容性与上转型	62
3.6.3	隐藏、覆盖与动态绑定	63
3.6.4	super 引用	65
3.6.5	对象初始化顺序	66
3.7	方法调用的优先顺序	71
3.8	abstract 修饰的类	73
3.9	final 修饰	74
3.10	接口	76
3.10.1	接口与实现接口的类	76
3.10.2	编程中选择抽象类还是接口	78
3.11	多态	80
3.11.1	多态的 3 种情形	80
3.11.2	多态的具体实现	82
3.11.3	总结	83
3.12	继承与多态的综合用例	84
3.13	内部类	89
3.13.1	实例成员内部类	89
3.13.2	局部内部类	92
3.13.3	匿名内部类	93
3.13.4	静态成员内部类	94
3.13.5	内部类的继承	95
3.13.6	内部类的两种特殊用法	95
3.13.7	内部类总结	97

3.14 设计原则	99
第4章 Java 基本结构程序	112
4.1 流程控制语句	112
4.1.1 顺序结构	112
4.1.2 分支语句	112
4.1.3 循环语句	117
4.1.4 跳转语句	120
4.2 递归	123
第5章 Java 基本类	125
5.1 java.lang 语言包	125
5.1.1 Object 类	125
5.1.2 字符串类	126
5.1.3 数组	129
5.1.4 基本数据类型的包装类	134
5.1.5 Math 数学类	135
5.2 java.util 实用包	136
5.2.1 日期类	136
5.2.2 集合 API	138
5.2.3 随机数类 Random	141
第6章 图形用户界面	143
6.1 概述	143
6.2 容器和布局	145
6.2.1 容器组件	145
6.2.2 框架 JFrame	147
6.2.3 布局管理器	148
6.3 常用组件	152
6.3.1 标签组件	152
6.3.2 文本编辑组件	153
6.3.3 按钮组件	154
6.3.4 列表框和组合框	156
6.3.5 对话框	156
6.3.6 菜单组件	158
6.4 事件处理机制	161
6.4.1 基本概念	161
6.4.2 委托模型	162
6.4.3 Java 中常用的事件、监听器及事件处理方法	163
6.4.4 事件处理机制的编程方法	164
6.4.5 几个事件处理的例子	167

第 7 章 异常处理	180
7.1 异常处理概述	180
7.1.1 程序中常见的错误	180
7.1.2 Java 中的异常类	181
7.1.3 异常处理过程	182
7.2 Java 的异常处理	183
7.2.1 try...catch...finally 异常处理语句	183
7.2.2 用 throw 抛出自定义异常对象	186
7.2.3 用 throws 将异常抛给上级调用方法处理	186
7.2.4 自定义异常类	187
第 8 章 多线程	189
8.1 创建线程	189
8.1.1 继承 Thread 类创建线程	189
8.1.2 实现 Runnable 接口创建线程	192
8.1.3 线程状态和线程控制	194
8.2 线程间的同步机制	198
8.2.1 同步机制解决资源竞争问题	198
8.2.2 同步机制与线程通信	202
第 9 章 Java Applet 程序	207
9.1 初识 Applet	207
9.2 Applet 的生命周期	208
9.3 Applet 中的方法	209
9.4 Applet 和 Web 页面	210
9.5 Applet 播放声音	211
9.6 Applet 实现动画	212
9.7 Application 与 Applet 的关系	214
第 10 章 输入输出流与文件处理	215
10.1 输入输出流概述	215
10.2 字节流类	216
10.2.1 字节输入输出流	217
10.2.2 文件字节输入输出流类	220
10.2.3 过滤输入输出流	222
10.2.4 对象输入输出流类	225
10.3 字符流类	228
10.3.1 字符输入输出流	229
10.3.2 文件字符输入输出流类	230
10.3.3 字符缓冲流类	231
10.4 文件处理	233

10.4.1	文件的基本概念	233
10.4.2	File 类	234
10.4.3	文件过滤器接口	236
10.4.4	随机存取文件类	239
第 11 章	Java 网络编程	242
11.1	网络编程基本概念	242
11.1.1	网络基础知识	242
11.1.2	HTTP 简介	243
11.1.3	两类传输协议 TCP 和 UDP	243
11.2	使用 URL 访问网络资源	244
11.2.1	统一资源定位符 URL	244
11.2.2	URL 类	244
11.2.3	URLConnection 类	247
11.3	Socket 通信	249
11.3.1	套接字	249
11.3.2	Socket 通信	249
11.4	UDP 数据报	254
11.4.1	数据报	254
11.4.2	UDP 通信	255
第 12 章	数据库应用	260
12.1	数据库技术	260
12.2	初识 JDBC 程序	261
12.2.1	建立 student 表	261
12.2.2	新建 first_v1.0 项目	261
12.3	创建数据库应用程序	263
12.3.1	数据库驱动	263
12.3.2	URL	264
12.3.3	建立连接	266
12.3.4	建立会话	266
12.3.5	操作数据库	267
12.4	JDBC 综合应用举例	268

第 1 章 Java 语言概述

1.1 Java 语言的起源及发展

1.1.1 起源

在计算机编程语言的历史上,C 语言是一个里程碑,第一次让程序员相对轻松地编写适度复杂的程序,但工程规模达到一定程度后,结构化编程方法无法对复杂性进行有效的管理。当代码超过 2.5 万至 10 万行后,便很难从整体上把握其复杂性了。C++ 引入面向对象的思想,解决了复杂度管理的问题。因此,可以说,面向对象是应用继承、封装、多态帮助管理程序复杂性的编程技术。于是人们称 C++ 是带类的 C。

1991 年,Sun 公司的 Jame Gosling 所领导的开发小组为了对家用消费类电子产品进行交互式控制,欲开发一个分布式系统,使用这个系统可以把 E-mail 发给电冰箱、电视机等家用电器。但是,消费电子产品种类繁多,即使是同一类消费电子产品所采用的处理芯片和操作系统也不相同,因此要想成功开发此类系统,必须首先解决跨平台的问题。最初,开发小组想用 C++ 语言,但用 C++ 写的程序必须针对特定的计算机芯片进行编译,一旦编译好后就很难适应其他芯片,并且由于 C++ 语言中指针等问题导致其安全性差,并不适合项目的要求。因此,开发小组基于 C++ 语言开发了一种能更适合于消费类电子设备软件开发的新的程序设计语言,称为 Oak(橡树)。Oak 程序可以在需要时下载,可以在设备出厂前预先编程,而且当设备新增功能时,客户也可以立即用上,而不用再将设备送回工厂。

由于当时对软件的迫切需求是速度和对复杂度的管理,对于可移植性的要求并不迫切。Oak 语言初步设计完成后,在市场上并未显示其优势,因而没有得到推广和广泛承认。直到 1994 年,面对 Internet 的迅猛发展以及环球信息网 WWW 的快速增长,Sun 公司发现因特网是一个完全异构的世界,Oak 语言的可下载技术和平台无关性可以用到 Web 上,并发挥其威力。这就需要将原先针对消费设备的语言改造成适于网络的语言。于是将 Oak 改名为 Java 并着手研制基于 Java 的浏览器。Java(爪哇)是印尼的一个岛屿,那里盛产咖啡,人们把那里产的咖啡称为 Java。同时,为了推动 Java 编程语言的发展,Sun 公司决定对 Java 采取免费提供的方式。

1995 年 5 月 HotJava 浏览器推出。HotJava 是第一个能够自动装载和运行 Java 程序的浏览器,它将程序从服务器端下载到客户端,解决了服务器的性能瓶颈,并使得 Web 上的图形和文本由静态的变为动态的、交互的,极大地促进了 Web 的服务功能。HotJava 浏览器不仅仅使 Java 语言更加成熟,更重要的是向世人展示了 Java 的强大功能。从此,Java 语言便逐渐成为 Internet 上受欢迎的开发与编程语言。一些著名的计算机公司如 IBM、Netscape、Apple、Oracle 等纷纷表示对 Java 语言的支持。当今,Oracle、DB2 这些数据库大部分代码都是用 Java 写的,甚至 Eclipse 这样普及的编程环境,即便其可以用于 C++ 编译,其大部分代码也是用 Java 编成的。

1.1.2 发展

Java 不仅是一门编程语言,还是一个开发平台。2005 年,Sun 公司公开 Java SE6。Java 的各种版本更名,取消 Java 1.2 版本后 J2ME,J2SE,J2EE 中的 2,更名为 Java ME,Java SE,Java EE。

(1) Java SE。Java SE(Java Platform,Standard Edition)含有基本的 Java SDK、工具和 API,开发者可以用来编写、部署和运行 Java 应用程序和 Applet。主要用于工作站、PC 的标准平台。

(2) Java EE。Java EE(Java Platform,Enterprise Edition)建立在 Java SE 的基础上,它是 JSP(Java Server Page)、Servlet、EJB(Enterprise JavaBean)、JTS(Java Transaction Service)、Java mail 以及 JMS(Java Message Service)等多项技术的混合体,并且还含有兼容性测试套件,主要用于开发分布式的、服务器端的多层结构的应用系统,例如电子商务网站。

(3) Java ME。Java ME(Java Platform,MicroEdition)主要用于开发电子产品,例如移动电话、数字机顶盒、汽车导航系统等。

1.2 Java 语言的特点

1. 简单性

Java 去掉了 C++ 中许多复杂的、冗余的、有二义性的概念,例如操作符重载、多继承、数据类型自动转换等。为了将程序员从复杂的内存管理的负担中解脱出来,同时也是为了减少错误,Java 使用了自动内存垃圾收集机制,程序员只要在需要的时候申请即可,不需要释放,而由 Java 自己来收集、释放内存中的无用的块。Java 语言采用了 C 语言中的大部分语法,熟悉 C 语言的程序员会发现 Java 语言在语法上与 C 语言极其相似,可以十分容易地学会 Java 语言。

2. 面向对象

C++、Objec、Pascal 等都是一种混合型的语言,即在过程式的语言中加上面向对象的扩展,因此在编程时会导致面向对象思想与面向过程思想的转换,甚至两种思想打架。Java 的面向对象特性几乎可以与 Smalltalk 媲美,但是其适用于分布式计算环境的特性却远远超过了 Smalltalk。Java 具有抽象、封装、继承与多态四大特点。这些特点使得 Java 能够做到信息隐藏,易于模块化,尤其是适合分布式开发、多人开发、大项目开发、委托(外包)开发。使得开发速度提高,开发方式灵活,开发成本降低。面向对象技术使得大项目和复杂项目的开发变得容易,维护也容易。Java 是迄今为止稳定性、可靠性、安全性最好的语言。2002 年成功登陆火星的“勇气”号探测器所使用的程序就是由 Java 编写的。正如 Java 的创始人 Gosling 所说的,如果不用 Java,很难想象这么复杂的一个大项目能够完成。而探测器在火星上陷入一个坑中不能自拔时,正是由于 Java 优秀的易维护性,让软件工程师在地球上利用一周的时间修复了它。

3. 面向网络

Java 是一门在网络上运行的语言。有人将 Java 和 XML 比喻成网络的两个翅膀或两

个轮子。XML 负责网络数据的存储,而 Java 负责网络数据的处理。二者完美的结合,使得基于网络的应用如日中天,其发展和对各个行业的渗透速度出乎人们的意料,正以一种前所未有的速度改变着人类的生活方式和工作方式。当然,Java 应用也很广泛,比如说人工智能游戏、汽车、铁路机车上的即时控制系统、甚至军事方面也得到广泛应用。

4. 鲁棒性

Java 不支持指针,杜绝了内存的非法访问,从而保证了程序的可靠性。在 C++ 程序中,指针的错误使用通常是程序差错的元凶;在 C 和 C++ 中,不加封装的全局变量会由于使用不当而造成运算错误甚至系统的崩溃。Java 使全局变量(即类变量)封装在类中,保证了安全性;C++ 语言中用 goto 语句实现无条件跳转,灵活的同时导致逻辑混乱。Java 取消了 goto 语句;C++ 语言中,不同的平台上,编译器为简单的数据类型如 int、float 等分别分配不同的内存存储位数。在 Java 中,对数据类型的位数分配总是固定的,而无视计算机平台特性。因此就保证了 Java 数据的平台无关性和可移植性;C++ 语言中,可以通过指针进行任意的类型转换,存在不安全因素。而在 Java 语言中系统要对对象的处理进行严格的相容性检查,在编译阶段防止不安全的转换;Java 自动收集无用的内存单元,进而防止了由于内存泄露导致的动态内存分配问题;Java 完善的异常处理机制,既简化了错误处理任务和恢复,也增加了程序的可读性。

5. 安全性

安全性是 Internet 开发面临的最重要的问题。因为 Java 小程序需要下载到客户端解释执行,所以,如果没有安全控制,就会给一些网络黑客以可乘之机。Java 的安全机制可以有效地防止病毒程序的产生、下载程序对本地文件系统的破坏,以及网络黑客窃取密码和入侵。

第一,在 C++ 中,通过指针对内存地址进行显示类型转换后,可以访问到类的私有成员,破坏了安全性。Java 不支持指针,一切对内存的访问都必须通过对象的实例变量来实现,Java 程序员不能强制引用内存指针。这样就防止程序员使用欺骗手段访问对象的私有成员,同时也避免了指针操作中容易产生的错误。

第二,Java 的编译器不处理内存布局,运行时期的实际内存布局无法从类的结构推知。

第三,字节码验证器(Byte Code Verifier)可以防止 Java 运行环境装入有害的类文件。这样,通过炮制字节码来入侵的办法就行不通了,任何字节码都要通过字节码验证器来验明正身。

第四,浏览器限制小程序访问客户机的资源。小程序能够访问的客户机的资源比应用程序要少得多。

第五,Java 类装入机制保证不会装入伪装类。有时,入侵者使用一种称为“伪装”的攻击方法。伪装就是建立与标准类同名的类,希望能够冒名顶替。而被伪装的类通常都是比较敏感的控制资源的类。而 Java 的类装入机制能够防止假冒类的装入。当需要装入标准类时,Java 首先搜索 Java 运行系统自己的类定义集,如果没有,会按照 ClassPath 指定的路径搜索本地机器中文件系统的其他位置,如果还没有,才去搜索远程服务器。

6. 结构中立与可移植性

在网络上的各个结点可以是各种不同类型的机器和操作系统,为使 Java 程序能在网络的任何地方运行,Java 编译器编译生成了与平台无关的字节码文件格式。任何种类的计算机,只要在其处理器和操作系统上有 Java 虚拟机,字节码文件就可以在该计算机上运行。即使是在单一系统的计算机上,结构中立也有非常大的作用。随着处理器结构的发展变化,非跨平台语言的程序员不得不编写各种版本的程序以在不同的处理器上运行,使得开发出能够在所有平台上工作的软件集合是不可能的。而使用 Java 开发的应用程序可以直接运行在所有的平台上而无须开发新平台版本。

7. 解释执行

Java 解释器直接对 Java 字节码进行解释执行。字节码本身携带了许多编译信息,使得连接过程更加简单。

8. 高性能

Java 的开发速度是其他语言无法匹敌的,但其执行速度却不是最快的。Java 牺牲运行速度换来了安全性、跨平台性等其他优点。Java 运行时系统必须内嵌一个内存管理模块,增加了运行时系统的复杂性;同样,Java 程序的解释执行的效率也要低于直接执行编译后的源码的效率。但是 Java 采用了一些很好的措施来弥补这些性能上的差距:(1)生成高效的字节码。Java 字节码的设计充分考虑了性能的因素,字节码的格式简单,解释器可以生成高效的机器码;(2)提供了即时编译和嵌入 C 代码的可选措施。即时编译是指在运行时把字节码编译成机器码;(3)支持多线程。Java 提供了对多线程的语言级的接口,而且 Java 环境本身就是多线程的。

9. 多线程

虽然 Java 的执行速度较 C 和 C++ 慢,但由于其主要用于网络,而网络连接、数据库连接的速度经常慢于 Java 运行的速度,因此,Java 的速度满足网络的需求。当网络连接和数据库连接处于等待时,CPU 就处于闲置状态。Java 提供了多线程机制来解决这一问题。即将多个网络连接或数据库连接设计成多个线程,当一个连接处于停滞时,其他线程可以替补上,可以让 CPU 满负荷运转。

10. 动态性

动态特性是面向对象特性的一个延伸,它使得程序能够适应不断变化的执行环境。Java 的动态性主要表现在以下几个方面。

第一,Java 的类有运行时的表示。这样,即使在运行时刻,程序也能辨别类之间的关系和类型信息,可以动态的从本地或网上把一个类链接到运行系统中去。

第二,后期联编。Java 的类在运行过程中动态的装载,因此,Java 可以在分布式的环境中动态的维护应用程序和 Java 类库之间的一致性。当类库升级后,应用程序无须重新编译,也一样可以利用新类库中新增的功能。

第三,支持动态数据类型和动态协议。通过编写协议句柄,Java 可以支持新的、自定义的传输协议,编写内容句柄,可以支持新的数据类型。

值得一提的是,Java 并不是用来替代 C 和 C++ 的,C 和 C++ 有自己适用的领域。比如,

C 由于其运行速度快适合底层开发,如主板、声卡、显卡等的驱动程序。而 C++ 程序则适合于图像处理程序开发。

1.3 Java 程序是如何在系统中运行的

Java 程序是怎样被运行的呢?首先,编译器将源码转化成字节码(.class),然后由虚拟机将字节码解释执行。虚拟机是平台相关的,但它非常小,容易编写。由于 Java 的强大,任何一个平台都愿意支持它,都配置了虚拟机。这样,Java 就将原先整体的依赖平台的代码分成两部分,其中转化字节码的工作量大,而第二部分即虚拟机解释执行的工作量很小。虽然虚拟机依赖各个不同的平台,但因工作量小,各种平台都很容易实现。于是 Java 就利用这种方式,将大块的耗时耗力的依赖平台的工作,分成了大部分不依赖平台(转化为.class 字节码),小部分依赖平台而每个平台又容易实现(虚拟机)的方法实现了跨平台。

Java 虚拟机是在机器和编译程序之间加入的一层抽象的虚拟的机器。这台虚拟的机器在任何平台上都提供给编译程序一个共同的接口。编译程序只需要面向虚拟机,生成虚拟机能够理解的代码,然后由解释器来将虚拟机代码转换为特定系统的机器码执行。它又包括类加载器、校验器、解释器。字节码的执行需要经过 3 个步骤,首先由类装载器(class loader)负责把类文件(class 文件)加载到 Java 虚拟机中,期间需要检验该类文件是否符合类文件规范;然后由字节码校验器(byte code verifier)检查该类文件的代码中是否存在着某些非法操作,如 Applet 程序中写本机文件系统的读写操作;最后如果字节码校验器检验通过,由 Java 解释器负责把该类文件解释成为机器码进行执行。

1.4 Java 开发环境

1. 安装 JDK

JDK(Java Development Kit)是 Sun 公司推出的 Java 开发工具包,包括 Java 类库、Java 编译器、Java 解释器、Java 运行时环境和 Java 命令行工具。JDK 提供 Java 程序的编译和运行命令,但是没有提供程序编辑环境。其他一些 Java 开发工具提供集成开发环境(Integrated Development Environment, IDE),如 Eclipse、MyEclipse、JBuilder 等,它们都是建立在 JDK 运行环境之上的。

可从 <http://www.oracle.com/technetwork/java/javase/downloads/index.html> 下载以下两个文件:

- jdk-6u23-windows-i586.exe(32 位的 Windows 的 JDK 6 安装程序)。
- jdk-6-doc.zip(JDK 6 文档)。

运行安装文件,在安装过程中可以设置安装路径及选择组件,系统默认安装路径为 C:\Program Files\Java\jdk1.6.0_23,默认组件选择是全部安装。

安装成功的 JDK 目录结构如图 1-1 所示。

JDK 的目录结构说明如表 1-1 所示,bin 中的主要工具如表 1-2 所示。

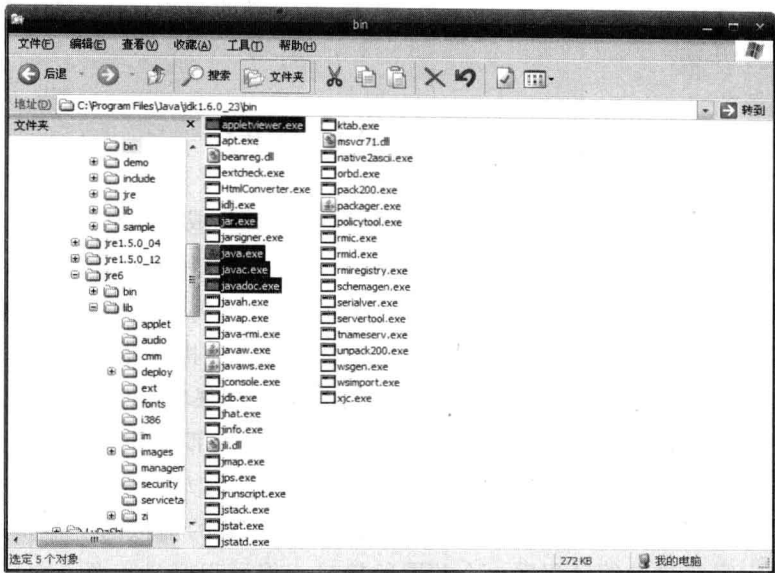


图 1-1 JDK 的目录结构

表 1-1 JDK 的目录结构

文件/目录名	功能说明	文件/目录名	功能说明
src.zip	核心 API 所有类的源文件	include	编写 JNDI 等程序需要的 C 语言头文件
bin	包含编译器、解释器和可执行文件	jre	Java 运行时环境
demo	包含源代码的程序示例	lib	Java 类库

表 1-2 bin 中的主要工具

文件名	功能说明
javac.exe	Java 编译器,将 Java 源程序编译成字节码文件
java.exe	Java 解释器,执行字节码文件对应的 Java 类
appletviewer.exe	Applet 应用程序浏览器
javadoc.exe	根据 Java 源码及说明语句生成 HTML 文档
jbd.exe	Java 调试器,可以逐行执行程序,设置断点和检查变量
jar.exe	压缩文件,扩展名为 JAR(Java Archive,Java 归档),与 Zip 压缩文件格式相同

2. 设置 JDK 环境变量

由于 Java 是平台无关的,安装 JDK 时 Java 不会自动设置路径,也不会修改注册表,需要用用户自己设置环境变量,但是不需要修改注册表。

在 Windows 中需要设置 Path 和 Classpath 两个环境变量。Path 变量指出 bin 包中可执行文件路径,Classpath 变量指出 Java 类包的路径。以下介绍两种方法,选择其中之一即可。

Windows 2000 和 Windows XP 下 Path 和 Classpath 的设置相同,步骤如下。

(1) 右击“我的电脑”图标,在弹出的快捷菜单中执行“属性”命令,在“系统属性”对话框的“高级”选项卡页面上单击“环境变量”按钮;在“环境变量”对话框的“系统变量”列表框中,选中 Path 变量,单击“编辑”按钮;在“编辑系统变量”对话框中,将“.; C:\Program Files\Java\jdk1.6.0_23\bin”输入到“变量值”文本框中,如图 1-2 所示。再单击“确定”按钮,就将 Java 编译器 javac.exe 和解释器 java.exe 等可执行文件的路径添加到 Path 变量中。

(2) 通常,Windows 中没有 Classpath 环境变量,需要新建。在“环境变量”对话框中单击“新建”按钮;在“编辑系统变量”对话框中,输入变量名为“Classpath”,输入变量值为“.; C:\Program Files\Java\jdk1.6.0_23\lib”,这是 Java API 类库的路径,如图 1-3 所示。

Java 虚拟机根据 Classpath 设置的路径,在对应的文件夹及子文件夹中寻找指定类或接口的.class 文件。

注意:这两个环境变量都必须加在“系统变量”中,不能加在“用户变量”中。



图 1-2 环境变量设置



图 1-3 新建 Classpath 变量

重新启动计算机后,设置有效。

设置环境变量需要注意以下 3 点。

① C:\Program Files\Java\jdk1.6.0_23 是 JDK 的默认安装路径,环境变量的值需要根据实际的安装路径更改。方法是逐级打开 bin 的文件夹路径,将安装路径复制到环境变量中。

② Path 路径中,原有路径后要添加;以便和 JDK 的路径区分开。

③ Classpath 路径中,“.”表示当前目录,通常写在最前面,意思是系统优先从当前目录中寻找。

1.5 编写、编译与运行 Java 程序

1. 运行 Java 程序的流程

在一个编辑器中编写好的 Java 应用程序,是扩展名为.java 的文件;通过 java.exe 编译生成扩展名为 class 的字节码文件;最后由 java.exe 执行该字节码文件。

对于 Applet 程序,不是由 java.exe 执行字节码文件,而是将字节码文件放到超文本文件(.html)中。当用浏览器打开该超文本文件时,浏览器中嵌入的字节码解释器运行该字节码文件。

2. 运行 Application 程序

【例 1.1】 显示字符串的 Application 程序。

```
public class FirstApplication {
```