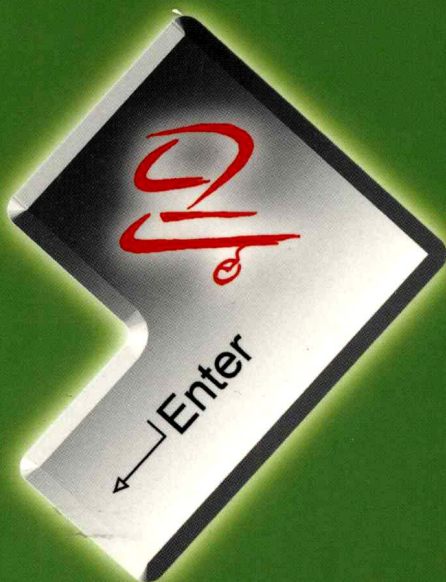


高等院校

计算机技术系列教材



Java语言程序设计

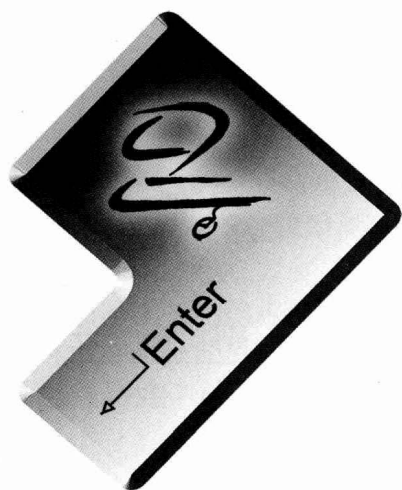
■ 倪永军 汪金友 王淑礼 陈功平 编著



WUHAN UNIVERSITY PRESS

武汉大学出版社

高等院校
计算机技术系列教材



Java语言程序设计

■ 倪永军 汪金友 王淑礼 陈功平 编著



WUHAN UNIVERSITY PRESS

武汉大学出版社

图书在版编目(CIP)数据

Java 语言程序设计/倪永军,汪金友,王淑礼,陈功平编著. —武汉:武汉大学出版社,2008.9

高等院校计算机技术系列教材

ISBN 978-7-307-06524-6

I. J… II. ①倪… ②汪… ③王… ④陈… III. JAVA 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2008)第 143823 号

责任编辑:杨 华 李 文 责任校对:王 建 版式设计:詹锦玲

出版发行:武汉大学出版社 (430072 武昌 珞珈山)

(电子邮件:wdp4@whu.edu.cn 网址:www.wdp.whu.edu.cn)

印刷:湖北金海印务公司

开本:787×1092 1/16 印张:18.5 字数:443 千字 插页:1

版次:2008 年 9 月第 1 版 2008 年 9 月第 1 次印刷

ISBN 978-7-307-06524-6/TP·312 定价:30.00 元

版权所有,不得翻印;凡购买我社的图书,如有缺页、倒页、脱页等质量问题,请与当地图书销售部门联系调换。



总 序

进入 21 世纪以来,人类已步入了知识经济的时代。作为知识经济重要组成部分的信息产业已经成为全球经济的主导产业。计算机科学与技术的信息产业中占据了极其重要的地位,计算机技术的进步直接促进了信息产业的发展。在国内,随着社会主义市场经济的高速发展,国民生活水平的不断提高,尤其 IT 行业在国民经济中的迅猛渗透和延伸,越来越需要大量从事计算机技术方面工作的高级人才加盟充实。

另一方面,随着我国教育的不断深入,高等教育已经完成了从精英教育向大众化教育的转变,在校大学本科和专科计算机专业学生的人数大量增加,接受计算机科学与技术教育的对象发生了变化。我国的高等教育进入了前所未有的大发展时期,时代的进步与发展对高等教育提出了更高、更新的要求。早在 2001 年 8 月,教育部就颁发了《关于加强高等学校本科教学工作,提高教学质量的若干意见》。文件明确指出,本科教育是高等教育的主体和基础,抓好本科教学是提高整个高等教育质量的重点和关键。2007 年 1 月,国家教育部和财政部又联合启动了“高等学校本科教学质量与教学改革工程”(以下简称“质量工程”)。“质量工程”以提高高等学校本科教学质量为目标,以推进改革和实现优质资源共享为手段,按照“分类指导、鼓励特色、重在改革”的原则,加强内涵建设,提升我国高等教育的质量和整体实力。

本科教学质量工程的启动对高等院校的从事计算机科学与技术教学的教师提出了一个新的课题:如何在新形势下培养高素质创新型的计算机专业人才,以适应于社会进步的需要,适应于国民经济的发展,增强高新技术领域在国际上的竞争力。

毋庸置疑,教材建设是“本科教学质量工程”的重要内容之一。新时期计算机专业教材应做到以培养学生会思考问题、发现问题、分析问题和解决问题的实际能力为干线,以理论教学与实际操作相结合,“案例、实训”与应用问题相结合,课程学习与就业相结合为理念,设计学生的知识结构、能力结构、素质结构的人才培养方案。为了适应新形势对人才培养提出的要求,在教材的建设上,应该体现内容的科学性、先进性、思维性、启发性和实用性,突出中国学生学习计算机专业的特点和优势,做到“够用、能用、实用、活用”。这就需要从总体上优化课程结构,构造脉络清晰的课程群;精练教学内容,设计实用能用的知识点;夯实专业基础,增强灵活应用的支撑力;加强实践教学,体现理论实践的连接度,力求形成“基础课程厚实,专业课程宽新,实验课程创新”的教材格局。

提高计算机科学与技术课程的教学质量,关键是要不断地进行教学改革,不断地进行教材更新,在保证教材知识正确性、严谨性、结构性和完整性的条件下,使之能充分反映当代科学技术发展的现状和动态,使之能为学生提供接触最新计算机科学理论和技术的机会;教材内容应提倡学生进行创新性的学习和思维,鼓励学生动手能力的培养和锻炼。在这个问题上,计算机科学与技术这个领域表现得尤为突出。



正是在这种编写思想指导下，在武汉大学出版社的大力支持下，我们组织中南地区的华中科技大学、武汉大学、华中师范大学、武汉理工大学、武汉科技学院、湖北经济学院、武汉生物工程学院、信阳师范学院、咸宁职业技术学院、江门职业技术学院、广东警官干部学院、深圳技师学院等院校长期工作在教学和科研第一线的骨干教师，按照 21 世纪大学本科计算机科学与技术课程体系要求，反复研究写作大纲，广泛猎取相关资料，精心设计教材内容，认真勘正知识谬误。经过大家努力的工作，辛勤的劳动，这套高等院校计算机技术系列教材终于与读者见面了。我相信通过这套教材的编写和出版，能够为我国计算机科学与技术教材的建设有所贡献，能够为我国高等院校计算机专业本科教学质量的提高有所帮助，能够为更多具有高素质的、创新型的计算机专业人才的培养有所作为。

魏长华

2007 年 7 月于武昌





前 言

面向对象程序设计是建立在具体的编程语言之上,利用它可以将复杂的软件系统分解成多个对象部件,通过对象模型可以清晰、简洁地模拟客观现实,并且可以方便地重复使用这些模型。Java 程序设计语言作为面向对象的程序设计语言,正是借助类来实现各种功能和提供行之有效的资源共享。

本书在介绍 Java 程序设计语言的过程中,注重培养读者正确使用面向对象的思维方法、分析问题和解决问题的能力。因此,在内容的取舍上,作了精心的筹划和选择,确保有一定的深度和广度;在内容的编排上体现了由浅入深、循序渐进的学习规律。同时,注重理论与实践相结合,提供了大量实例,注重基本知识的理解与基本技能的培养,将复杂的面向对象理论融合于众多的实例之中,使读者学会面对一个具体问题时,能够使用面向对象的思维方式分析问题和解决问题。因此,本书所提供的实例均经过上机验证,并给出实验结果,以便初学者对照学习和方便理解。

全书共分 8 章,第 1 章是绪论部分,主要讲述 Java 语言的特点和配置方法以及使用的编辑工具;第 2 章是 Java 语言基础,主要讲述 Java 语言的语法规则及编程格式和编程方法。第 3 章面向对象程序设计,讲述类、对象、方法、接口、包等;第 4 章对数组的建立、初始化等作了详细的论述;第 5 章介绍 Java 的基本可重用类,包括数学类、IO 类、File 类及字符流和字节流;第 6 章介绍 Java 的多线程机制和异常处理,对线程的创建、线程的同步和让步以及线程的优先级等作了深入详细的论述;第 7 章介绍 Java 的图形用户界面,GUI 程序设计、事件驱动程序设计、创建用户图形界面等;第 8 章介绍 Java 的小应用程序 Applet,对于 Applet 的基本知识、图形图像的处理,声音动画的制作作了深入的论述。

倪永军承担本书的内容策划和主编任务。本书第 3、4、6 章由倪永军副教授编写,第 5、8 章由汪金友副教授编写,第 1、2 章由王淑礼副教授编写,第 7 章由陈功平讲师编写,最后由倪永军审定、修改和统稿。在本书编写过程中,作者参考了许多书籍和网站,也得到了许多同事和同仁的关心、支持和帮助,在此一一表示衷心的感谢。

本书编写的目的是针对 Java 语言程序设计提供一些有价值的程序设计的方法和理念。尽管书稿经过多次修改,但由于作者水平和时间所限,书中难免存在缺点和不足之处,热忱欢迎各位同行和广大读者批评指正。

编者

2008 年 5 月





目 录

第 1 章 绪论	1
1.1 计算机语言	1
1.2 Java 语言及其特点	2
1.2.1 Java 语言的历史	2
1.2.2 Java 语言的特点	4
1.3 Java 语言的开发工具	7
1.3.1 JDK	7
1.3.2 JBuilder	11
1.4 第一个 Java 程序	11
本章小结	12
习题 1	13
第 2 章 Java 语言基础	14
2.1 Java 语言的基本结构	14
2.1.1 类首部	15
2.1.2 类体	15
2.1.3 注释语句	16
2.1.4 标识符	17
2.1.5 分隔符	18
2.2 变量和常量	18
2.2.1 变量	18
2.2.2 常量	19
2.3 数据类型和运算符	19
2.3.1 基本数据类型	19
2.3.2 数据类型转换	22
2.3.3 算术运算符	24
2.3.4 比较和条件运算符	24
2.3.5 逻辑运算符	26
2.3.6 位运算符	26
2.3.7 移位运算符	28
2.3.8 增减运算符	29
2.3.9 赋值运算符	30





2.4	表达式	31
2.4.1	运算符的优先级	31
2.4.2	表达式的运算顺序	32
2.5	输出格式控制	33
2.6	流程控制	35
2.6.1	简单的 if 语句	36
2.6.2	if_else 语句	37
2.6.3	if 语句的嵌套	38
2.6.4	switch 语句	41
2.6.5	while 循环	45
2.6.6	do_while 循环	47
2.6.7	for 循环	48
2.6.8	循环嵌套	52
2.6.9	break 和 continue 语句	55
	本章小结	59
	习题 2	59
第 3 章	Java 的面向对象程序设计	63
3.1	面向对象程序设计的基本概念	63
3.2	类	63
3.2.1	类的声明	64
3.2.2	类体定义	66
3.3	对象	70
3.3.1	对象的创建	71
3.3.2	对象的调用与销毁	71
3.4	方法	75
3.4.1	方法的调用	75
3.4.2	方法过载	83
3.5	继承和多态	85
3.5.1	类继承的实现	86
3.5.2	继承中的访问控制符	88
3.5.3	覆盖	90
3.5.4	this、super 和 null	91
3.5.5	多态性	96
3.6	接口	97
3.6.1	接口声明	98
3.6.2	接口的实现	99
3.6.3	抽象类	101





3.7 包	103
3.7.1 包的创建	103
3.7.2 包的引用	105
3.7.3 包中成员的访问	106
本章小结	107
习题 3	108
第 4 章 数组	116
4.1 一维数组	116
4.1.1 一维数组的定义	116
4.1.2 创建数组	117
4.1.3 数组初始化	117
4.2 多维数组	123
4.2.1 二维数组的定义与创建	123
4.2.2 二维数组的初始化	125
4.3 对象数组	126
4.4 给方法传递数组参数	128
4.5 数组查找	130
4.6 数组的复制	133
4.7 数组的排序	137
本章小结	140
习题 4	140
第 5 章 Java 的基本可重用类	143
5.1 java.lang 包	143
5.1.1 数学类 Math	143
5.1.2 字符串类 String	146
5.1.3 字符串缓冲区类 StringBuffer	151
5.2 java.io 包	153
5.2.1 文件类 File	154
5.2.2 RandomAccessFile 类	158
5.2.3 流 (Stream)	161
5.2.4 标准的输入/输出流	161
5.2.5 字符流	163
5.2.6 字节流	169
5.3 java.util 包	175
本章小结	178
习题 5	178





第 6 章 Java 的多线程机制和异常处理	181
6.1 线程	181
6.1.1 线程的建立	182
6.1.2 Thread 子类	185
6.1.3 线程的状态	187
6.1.4 线程让步和合并	190
6.1.5 线程的优先级与调度	192
6.2 多线程的实现机制	193
6.2.1 线程的同步与锁定	194
6.2.2 生产者和消费者	199
6.3 异常处理	202
6.3.1 异常类	203
6.3.2 异常处理	204
6.3.3 抛出异常	207
6.3.4 异常的使用	212
本章小结	213
习题 6	213
 第 7 章 Java 的图形用户界面	217
7.1 引言	217
7.2 组件分类	218
7.2.1 容器 (Container)	218
7.2.2 框架 (Frame)	218
7.2.3 面板 (Panel)	219
7.2.4 对话框 (Dialog)	221
7.2.5 文件对话框 (FileDialog)	222
7.3 组件 (Component)	223
7.3.1 按钮 (Button)	223
7.3.2 标签 (Label)	224
7.3.3 文本域和文本区	225
7.3.4 复选框和组合框	227
7.3.5 列表框 (List)	229
7.3.6 滚动条 (Scrollbar)	230
7.3.7 画布 (Canvas)	232
7.4 布局管理器	233
7.4.1 FlowLayout	234
7.4.2 GridLayout	235
7.4.3 BorderLayout	236





7.5 菜单 (Menu) 组件	237
7.5.1 菜单栏 (MenuBar) 和菜单	237
7.5.2 菜单项 (MenuItem)	239
7.5.3 弹出式菜单	240
7.6 颜色类	241
7.7 字体类	241
7.8 绘制简单图形	242
7.9 显示图像	245
7.10 事件处理	247
7.10.1 事件处理机制	247
7.10.2 按钮点击事件	249
7.10.3 鼠标和窗口事件	250
7.10.4 事件适配器	254
7.10.5 内部类和匿名内部类	255
本章小结	257
习题 7	257
 第 8 章 Java 的小应用程序 Applet	259
8.1 Applet 的特点	259
8.2 Applet 类	259
8.3 Applet 的开发过程	260
8.4 HTML 标记及属性	262
8.5 Applet 的生命周期	264
8.6 Applet 的安全	267
8.7 Applet 的应用	267
8.7.1 Applet 与 Application	267
8.7.2 利用 Applet 绘图	269
8.7.3 利用 Applet 显示图像	273
8.7.4 利用 Applet 播放声音	274
8.7.5 利用 Applet 制作动画	276
本章小结	281
习题 8	282
 参考文献	283



1.1 计算机语言

计算机语言是计算机能够理解的、用于实现人机通信的语言。随着计算机的迅猛发展,计算机的应用语言也有上千种之多,程序员采用这些语言编写的程序,有的可以在计算机上直接运行,有些则需要经过翻译或编译之后才能执行。但是,不论哪种语言,均可以归纳为机器语言、汇编语言和高级语言三种类型。

机器语言也可以称为低级语言,是计算机能够直接识别的语言,是由“0”和“1”组成的二进制序列,是由计算机硬件设计者所定义的。它依赖于特定类型的机器,也就是说不同类型的计算机具有不同的机器语言。机器语言枯燥乏味且难以记忆,因此,程序员们开始引入一种助记符号来代替晦涩难记的二进制指令,这种语言称为汇编语言。

汇编语言和机器语言相比,使用汇编语言编写程序的突出优点就是可以使用符号,具体说就是可以用助记符来表示指令的操作码和操作数,可以用标号和符号来代替地址、常量和变量。助记符一般都是表示一个操作的英文字母的缩写,便于识别和记忆。然而用汇编语言编写的程序不能由机器直接执行,必须翻译成由机器代码组成的目标程序,这个翻译过程称为汇编。在微型计算机中,汇编过程通常是由软件自动完成的。用来把汇编语言编写的程序自动翻译成目标程序的软件称为汇编程序。汇编过程如图 1-1 所示。

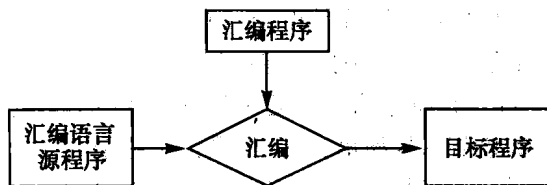


图 1-1 汇编过程

用汇编语言编写的程序称为源程序,而计算机指令系统中的每一条指令都是构成源程序的基本语句。汇编语言的指令和机器语言的指令之间有一一对应的关系。所以,汇编语言是和机器密切相关的,是一种面向机器的语言。采用汇编语言进行程序设计时,要求程序员必须掌握一定的计算机硬件知识,利用计算机的硬件功能和结构特点,编写出的程序既可以有效加快程序的执行速度,又能减小目标程序所占用的存储空间。因此,与高级语言相比,汇编语言为我们提供了直接控制目标代码的手段,而且可以对输入/输出端口进



行控制, 具有实时性能好、执行速度快和节省存储空间等优点。

高级语言完全脱离机器指令, 用人们更加易于理解的方式编写程序, 更接近于科学计算的公式及问题, 所以有时候说, 机器语言和汇编语言是一种面向机器的语言, 而高级语言则是面向科学计算和实际问题的语言。对于程序员来讲, 即使没有掌握任何计算机硬件知识, 只要掌握了高级语言的语法、规则等, 照样可以编写出计算机能够运行的程序来。当然, 高级语言编写的程序不能直接由计算机来执行, 必须由编译程序把它编译成为机器语言的程序, 计算机才能够执行。高级语言的翻译程序有两种形式: 一种是将高级语言的源程序一边进行解释, 一边执行, 这种翻译程序称为解释程序, 如 Basic 语言。另一种是把高级语言编写的源程序翻译成机器语言程序, 然后再在机器上执行, 这种程序称为编译程序。多数高级语言采用这种形式, 如 Pascal、Fortran、C 语言等。

1.2 Java 语言及其特点

Java 语言属于高级语言, 是一种简单的面向对象 (Object-Oriented) 的跨平台语言。其目标是为了满足在一个充满各式各样不同种机器、不同操作系统平台的网络环境中开发软件。利用 Java 程序语言, 可以在网页中加入各式各样的动态效果, 可以放上一段动画, 加入声音, 也可以建立交互式网页等。

1.2.1 Java 语言的历史

Java 是由 Sun Microsystems Inc. 公司开发出来的, 它的诞生主要得益于家用电器芯片的开发。1991 年, 该项目被称为绿色计划 (Green Project)。开始时, 它的开发者想用 C、C++ 语言来开发电器的芯片。但是, 由于芯片的种类各不相同, 程序要进行多次编译。尤其是 C++ 中的指针操作, 一旦操作不慎, 就会引起问题。这些语言的功能与效果无法达到程序员们设计程序所追求的理想目标。同时, 由于当时计算机互联网的蓬勃发展, 于是 Sun 就与一群有志之士开始着手设计一套完全面向对象, 而且不受平台限制的语言。开发者将 C++ 语言进行简化: 去掉了指针操作, 去掉了运算符重载, 去掉了 C++ 中的多重继承。于是, 得到了 Java 语言, 它是一种解释执行的语言。至于它的命名, 刚开始 Java 语言被称之为 Oak 语言 (橡树语言), 但是由于这个名字已经被其他人使用, 最后设计小组突发奇想, 将他们开会时的咖啡厅为这一套计算机语言命名。所以 Java 语言的图标是一杯热滚滚的咖啡。

1995 年, Sun 公司在一个重要的会议上宣布了 Java, 但是当时它并没有引起人们的足够重视。随着 WWW 的出现和随之而来的商机, 广大商家开始对 Internet 投入了极大的兴趣。由于 Java 所采用的含有动态内容的网络主页, Java 语言也从此引起了人们的关注。Java 不同于 Pascal 这样的个人开发的语言, 也不同于由一个小组为了自己的使用目的而开发的 C 或 C++, Java 纯粹是为了商业目的而开发的。

Java 语言现在逐渐成熟起来, 它的类已经有近千个, 它的类无所不包, 而且还可以通过第三方购买类, 为 Java 语言的发展提供了良好的发展前景。同时它也是跨平台的语





言，因此许多软件开发商、硬件开发商也争先恐后地想乘上 Java 语言的快车，都声称支持 Java 语言，它对微软发起了有力的挑战，而且 Sun 公司正努力开发 Java 芯片。

目前，Java 的应用已经推广到各种平台，尤其是在 Internet 盛行的情况下，Java Applet 和 Application 几乎都可以通过网络来下载运行，其他的应用更是数不胜数。虽然 Sun 早期只维护一套 Java 的运行环境。但是在近年 Java 的应用日趋广泛的时候，就需要针对不同类型的用户，提供不同的操作平台。因此，Sun 推出了 4 种不同运行环境的 Java 语言版本，它们分别是应用于个人计算机的 J2SE (Java 2 Standard Edition)，即标准版；应用于小型设备上的 J2ME (Java 2 Micro Edition)，微型版；应用于企业服务器上的 J2EE (Java 2 Enterprise Edition)，企业版；应用在 Smart Card 上的 Java Card。本书中所出现的 Java 程序的运行环境均在标准版中。

在 Java 的系统结构中，Java 主要由虚拟机 (Virtual Machine) 和应用程序接口 (API) 两部分构成。虚拟机可以看成是一套虚拟的计算机，有一个标准的规格，可以用软件或硬件来实现。Sun 开放其虚拟机的源代码，也有一些公司如 IBM 自己编写虚拟机。位于 Java 虚拟机下层的移植接口层和 Adapter 是为了使 Java 虚拟机能够方便地移植到不同的操作系统之上而开发的。而位于 Java 虚拟机之上的 Java API 部分则包含了基本型的 API 类与标准延伸的 API 类。

1998 年底，Sun 公司正式发布了 Java 1.2 版本。之后第 3 天，Sun 的市场部门就把名字改为 Java 2 标准版软件开发工具箱 1.2 版。该版本用老练的、可伸缩的版本取代了早期的 GUI 和图形工具箱，从而比其先前版本“一次编程，到处运行”的能力更强。随后出现的 1.3 版、1.4 版都是在最初的 Java 2 版本的基础上的增量改进，增加了一些新特性，提升了性能，也修改了其中的一些 BUG。在 Java 标准版中，应用程序的接口、类、方法和字段数目均随版本的提高而增多。表 1-1 给出了标准版本从 1.0 至 1.5 中的 API 发展情况。

表 1-1

Java 标准版 API 的发展

版本	类和接口的数目	方法和字段数目
1.0	212	2125
1.1	504	5478
1.2	1781	20935
1.3	2130	23901
1.4	3020 (135 个包)	32138
1.5 (5.0)	4000 以上 (166 个包)	

Java 也号称是能跨平台使用的语言，这主要是因为 Java 本身被编译之后，并不是直接产生可执行的代码，而是产生一种中间码叫做 ByteCode (字节码)。这种码必须通过 Java 的直译器来解读它，才能够真正被执行，所以只要平台上装有这种 Java 的直译器，就能解读 ByteCode，也就能执行 Java 编译过的程序，故与 Java 程序是在哪种平台上被编译的，就完全没有关系了，如图 1-2 所示。Java 写出来的程序可分为两类，分别是 Java





Applet 与一般的 Application，而 Application 这一类就与一般的程序如 C++ 的作用比较类似，是一个独立可执行的应用程序。如 HotJava 浏览器，就是使用 Java 程序所发展出来的。最常见的 Java 程序包括应用程序和 Applets。应用程序是单独的程序，诸如 HotJava 浏览器软件就是用 Java 语言编写的。Applets 类似于应用程序，但是它们不能单独运行，Applets 可以在支持 Java 的浏览器中运行。Applet 主要是内置于 HTML 网页中，在浏览时发挥作用。

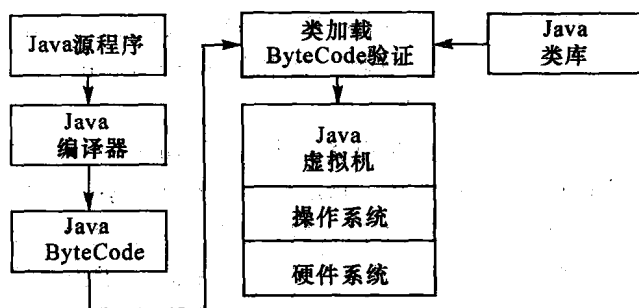


图 1-2 Java 执行过程示意图

Java 语言通过编译器 (Compiler)，将 Java 语言源程序文件 (扩展名为 .java) 编译成被称为字节码 (ByteCode) 的中间层代码文件 (扩展名为 .class)，而字节码与计算机的系统无关，同一个字节码文件可以被任何计算机使用。要在一个计算机系统中运行一个在其他计算机系统中编写 Java 程序，可以下载要执行的程序字节码文件，并把它交给本地的 Java 虚拟机，Java 虚拟机可以把这个字节码文件翻译成本系统的机器码，最后在本机上执行。

1.2.2 Java 语言的特点

Java 作为一种广泛使用的适用于网络环境中的程序设计语言，其迅速的发展和被广泛接受应该归功于它的设计和程序特征。Java 是一种简单的 (Simple)、面向对象的 (Object-Oriented)、分布式的 (Distributed)、解释性的 (Interpreted)、健壮的 (Robust)、安全的 (Secure)、结构中立的 (Architecture-Neutral)、可移植的 (Portable)、高效能的 (High-Performance)、多线程的 (Multithreaded)、动态的 (Dynamic) 语言。

1. Java 语言是简单的

Java 语言为了克服流行的面向对象的程序设计语言 C、C++ 设计程序中的 BUG 问题，将其功能进行了极大的改进和简化，抛弃了 C 和 C++ 语言中容易引起程序错误的部分功能，如指针和内存管理，取消了手工内存的分配和回收，引用了真正的数组而取消了指针算法；消除了指针操作时偏移错误而重写内存；取消了多重继承，引入了接口；同时也消除了条件语句中容易弄混的赋值和相等测试的可能性等。另外，Java 语言的编程风格类似于 C 和 C++，因此，熟悉 C 和 C++ 语言的程序员很容易掌握 Java 编程技术。因此，



Java 语言被称为“C++++-”，即 Java 加进了自动内存回收之类的新任务（++），减去了 C++ 中的费解而又不常用的内容，如运算符重载（--）。

它的简单性还体现在 Java 语言没有任何真正的新东西，它所有的特性集均是借用现成的东西。

2. Java 是面向对象的

面向对象是 Java 语言的最重要的特征，它是一种完全的面向对象程序设计（Object-Oriented Programming, OOP）语言，它不支持类似于 C 语言那种面向过程的程序设计技术。Java 支持静态和动态风格的代码继承和重用，是以对象为模型来描述现实世界的，世界上的任何事物都能抽象为对象。用 Java 编写的程序主要进行对象创建、对象处理，并使对象协调工作。

因此，在进行面向对象程序设计时，重点放在数据（等于对象）和对象接口上，用问题空间中的元素与对象描述问题，而不是用计算机中执行的一系列步骤来描述。这样，好的设计就可以得到可复用、可扩展和可维护的组件。这些组件相当灵活，可以处理环境的改变，因为这些对象的主要工作就是相互之间来回发送信息。实际上，Java APIs 正是这些预建组件的集合，也称为类库。所以，Java 中不需要一切从头开始，可以从标准库开始。这些标准库随着时间的推移而不断增加、改变，每个新版本就可以推出更多的预建组件。

3. Java 是分布式的

Sun 公司从 1982 年成立起，就一直坚持“网络就是计算机”这一思想。因此从提出 Java 语言开始，就要求其具有支持网络和像本地对象一样方便地访问分布式对象的功能。最初，Java 只能通过 HTTP 之类的标准 TCP/IP 协议访问分布式对象。随着 Java 功能的不断改进和完善，现在利用最新版的 Java 平台，可以通过 CORBA（Common Object Request Broker Architecture）与 RMI（Remote Method Invocation）等协议和 Web Services 功能，像在文件中存取数据一样方便。

4. Java 是解释性的

Java 是解释性的，它不是将源程序编译成自然执行的代码，而是转换成平台中立的字节码（ByteCode）。这些字节码可以传输到任何具有 Java 运行环境（Java Runtime Environment, JRE）的平台，其中包括 Java 虚拟机（Java Virtual Machine, JVM），从而在执行时，不需要重新编译或重新链接。

5. Java 是健壮的

健壮性反映了程序的可靠性，Java 的几个内置特性使程序的可靠性得到保证。Java 的编译器和类装入器保证所有方法调用的正确性，防止了隐式类型失配和版本的不兼容性；Java 使用引用而不使用指针，从而避免了因重写内存而造成数据破坏的可能性；Java 鼓励使用接口而不使用类，接口是定义一组行为，类则是实现这些行为，因此，传递接口就可以隐藏实现的细节。





如果程序在运行过程中出现了异常情况，Java 语言的异常处理程序能够迅速地捕获并响应意外情况，从而确保程序能够继续正常执行错误处理代码，保证程序的执行。

6. Java 程序是安全的

Java 加入了垃圾回收机制，让开发人员无需担心对象资源的回收问题，异常处理机制也可以让开发人员掌握程序中各种突发的异常状况。同时，Java 作为适用于网络/分布式环境的语言，它可以执行多层安全机制来保护系统不受恶意程序的破坏。

7. Java 的结构是中立的

体系结构中立是指 Java 的平台中立字节码。Java 程序不是编译成平台特定的二进制码，而是可以在任何地方执行，不需要重新编译与链接。如果开发出新的硬件，不需要放弃原来的软件投资，只要在新平台中装入 Java 运行环境即可。如果开发出全新的硬件和操作系统，也不必从没有软件的产品上从零开始，只需要放上 Java 运行环境，新设计的平台就可以运行所有现有的 Java 程序。也就是说，Java 不仅仅局限于编写 Web 程序，利用 Java 虚拟机可以编写任何平台上都能运行的程序，或利用 Java 解释器，能够在不同的操作系统上直接运行独立的 Java 应用程序。Java 语言的平台无关性如图 1-3 所示。

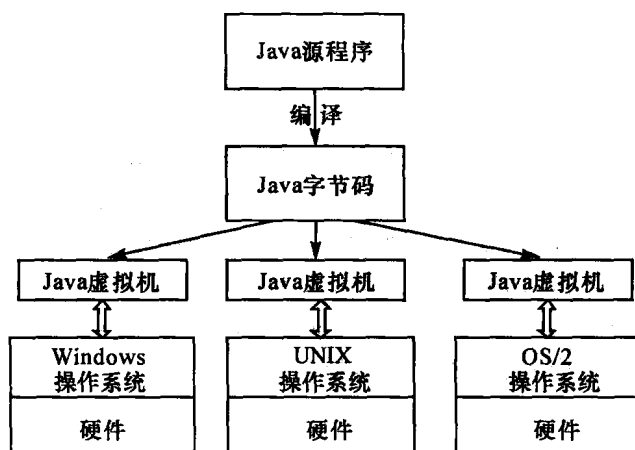


图 1-3 Java 语言的平台无关性

8. Java 的可移植性

Java 的源程序可以不经修改而在不同的硬件或软件平台上运行的特性称为可移植性。它提倡“一次编写，到处运行”（Write Once, Run Anywhere, WORA）。它的目标是在任何特性结构中运行同一个程序。这种移植性不仅利用了平台中立的字节码，而且在 Java 语言规范中定义了所有数字表示的长度、字节顺序和操作，而不是实现细节。

9. Java 是高效的

Java 是解释性的语言，它需要把源程序翻译成字节码然后通过解释器执行，因此它的速度受到一定的影响。为了提高运行的速度，Java 提供了两种运行转换工具，自动完

