

面向对象程序设计系列教材

Java

语言程序设计

主编 殷兆麟
副主编 沈琦 姜秀柱 杨东平

高等教育出版社

面向对象程序设计系列教材

Java 语言程序设计

主 编 殷兆麟

副主编 沈 琦 姜秀柱 杨东平

参 编 赵艳红 朱 敏 秦国华 徐月美

高等教育出版社

内 容 提 要

本书的取材、编写体现了 Java 面向对象编程和网络编程两个特点。立足于满足广大 Java 初学者入门和提高了的需要,满足掌握 Java 达到实用的需要,同时考虑满足 Java 编程 IT 国际认证考试的需要。教材由浅入深,避免开始引入过多的技术术语,以减少编程初学者学习的困难。全书共分为十二章,第一~八章介绍了计算机语言一般知识、Java 开发环境基础知识、Java 语言基础、Java 面向对象的程序设计、异常处理、Java 的基本可重用类、小应用程序类(Applet)和图形用户界面。这些内容可以满足应用 Java 开发交互式网页的基本要求。第九~十一章介绍了体现 Java 特色的其他可重用类的使用:多线程编程、Java 与数据库的连接——JDBC 技术和网络编程基础。最后,第十二章利用典型的综合应用示例进一步介绍了可视化集成开发环境 JBuilder 6 的使用方法。

本书可作为高等院校计算机专业本科、高职、专科及相关专业的语言程序设计或网络编程基础等课程的教材。也可以作为应用开发人员的自学参考书。

图书在版编目(CIP)数据

Java语言程序设计/殷兆麟主编. —北京:高等教育出版社, 2002.12 (2003重印)

ISBN 7-04-011551-4

I. J... II. 殷... III. JAVA语言—程序设计
IV. TP312

中国版本图书馆CIP数据核字(2002)第087049号

Java 语言程序设计

主编 殷兆麟

副主编 沈琦 姜秀穗 杨东平

出版发行 高等教育出版社

社 址 北京市西城区德外大街4号

邮政编码 100011

总 机 010-82028899

购书热线 010-64054588

免费咨询 800-810-0598

网 址 <http://www.hep.edu.cn>

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

经 销 新华书店北京发行所

排 版 高等教育出版社照排中心

印 刷 北京市鑫霸印务有限公司

开 本 787×1092 1/16

印 张 22.75

字 数 550 000

版 次 2002年12月第1版

印 次 2003年7月第2次印刷

定 价 26.20元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

前 言

Java 语言是网络时代广泛使用的面向对象的编程语言,它具有可移植性、稳定性、安全性、多线程机制等众多优点,具有非常高的技术性能。因此,Java 语言得到越来越广泛的应用。为了满足广大计算机编程初学者入门和提高的需要,为了掌握 Java 进而达到实用,同时考虑满足 Java 编程 IT 国际认证考试的需要,我们编写了本教材。教材内容由浅入深,避免开始引入过多的技术术语和介绍过多的 Java 特点,避免介绍 Java 与 C++ 的比较,以减少编程初学者理解的困难;同时,增加编程示例,增加基本算法的介绍,以培养初学者的编程思想,逐步提高其编程能力。

Java 语言程序设计的教学分为两个阶段进行。第一阶段目的是使学生掌握 Java 语言的基本语法,培养学生的编程思想,具有初步的面向对象编程能力。第二阶段目的是进一步提高学生面向对象编程能力,主要讲授进一步体现 Java 特点的扩充的可重用类的运用,使学生理解、掌握一种 Java 可视化集成开发环境。

本书第一~八章为第一阶段讲授的内容。第一章介绍与计算机语言有关的基本知识;第二章介绍 Java 的开发环境;第三章介绍 Java 语言基础;第四章介绍 Java 面向对象的程序设计;第五章介绍异常处理;第六章介绍 Java 的基本可重用类;第七章介绍 Java 小应用程序 Applet;第八章介绍 Java 的图形用户界面。为了达到讲授目的,为后续计算机课程服务,第一阶段还介绍基本算法和单向链、双向链、队列、堆栈的 Java 编程,这是一般 Java 语言教程中所忽视的。这阶段教学约需要 48 学时(不包括上机)。

本书第九~十二章为第二阶段讲授的内容。第九章介绍 Java 的多线程机制;第十章介绍 Java 网络编程基础;第十一章介绍 Java 与数据库的连接——JDBC 技术;第十二章通过综合示例进一步介绍 JBuilder 6 的使用。这阶段按教学对象不同需要 36~48 学时(不包括上机)。

第二阶段的内容要求学生要具备基本的数据库和网络知识,教师可作适当补充。如对计算机专业高年级学生讲授第二阶段内容是不困难的。

与本教材配套的 CAI 课件包括三方面内容:教师讲课用的课件、学生上网浏览用的课件和 Java 上机实验指导(包括教材中的示例源程序)。有兴趣的读者可以 E-mail:zhlyin@cumt.edu.cn 进行查询。

本教材第十二章的 12.1、12.2 和 12.3 节引用了 JBuilder 6.0 Enterprise Help 中的部分内容。这里对原文的作者表示谢意。

上述工作得到中国矿业大学教务处、计算机科学与技术系的关心和支持。该教材列为我校“新世纪教材建设工程项目”之一,教材 CAI 课件得到江苏省教委多媒体课件项目的资助,获得江苏省教委 2002 年多媒体课件三等奖,在此表示衷心感谢。

由于时间仓促,作者水平有限,书中难免有不妥和错误之处,恳请广大读者指正。

主编 殷兆麟

2002.5.18 于中国徐州

目 录

第一章 计算机语言概述

1.1 机器语言与汇编语言	(1)	1.5 Java 程序的运行	(6)
1.2 高级程序设计语言	(1)	1.6 Java 程序的开发、执行流程	(6)
1.3 语言翻译程序	(3)	小结	(7)
1.4 虚拟机、Java 虚拟机与 Java 运行环境	(4)	习题	(7)

第二章 Java 开发环境基础知识

2.1 Java 开发环境概述	(8)	2.3.1 主窗口	(15)
2.2 JDK 的安装、环境配置及使用	(8)	2.3.2 主菜单栏	(15)
2.2.1 JDK 的安装与环境配置	(8)	2.3.3 工具栏	(34)
2.2.2 JDK 的环境工具 及其使用	(9)	2.3.4 组件选项板	(35)
2.2.3 JDK 下的 Java 编程	(13)	2.3.5 JBuilder 下的 Java 编程	(35)
2.3 JBuilder 集成开发环境简介	(14)	小结	(39)
		习题	(39)

第三章 Java 语言基础

3.1 一个简单的 Java 程序实例	(40)	3.5.2 运算符	(47)
3.1.1 类首部	(40)	3.5.3 算术运算符	(47)
3.1.2 类体	(41)	3.5.4 关系运算符	(48)
3.2 标识符、注释和分隔符	(41)	3.5.5 逻辑运算符	(48)
3.2.1 标识符	(41)	3.5.6 位运算符	(49)
3.2.2 注释	(42)	3.5.7 赋值运算符	(49)
3.2.3 分隔符	(42)	3.5.8 条件运算符	(49)
3.3 变量和常量	(42)	3.6 运算符的优先级	(50)
3.3.1 变量	(42)	3.7 数据类型转换	(51)
3.3.2 常量	(43)	3.7.1 自动类型转换	(51)
3.4 基本类型	(43)	3.7.2 强制类型转换	(51)
3.4.1 整型	(44)	3.8 简单语句和复合语句	(52)
3.4.2 浮点型	(45)	3.9 控制语句	(54)
3.4.3 布尔型	(45)	3.9.1 选择语句	(54)
3.4.4 字符型	(46)	3.9.2 switch 语句	(57)
3.5 运算符及表达式	(47)	3.9.3 循环语句	(59)
3.5.1 表达式	(47)	3.9.4 for 语句	(61)

3.9.5 循环的嵌套	(64)	小结	(68)
3.9.6 break 和 continue 语句	(65)	习题	(68)
3.10 综合应用举例	(67)		

第四章 Java 面向对象的程序设计

4.1 面向对象的理论基础	(71)	4.6 null、this、super 和多态性	(85)
4.2 类与对象的基本概念	(71)	4.6.1 null、this、super	(85)
4.2.1 对象	(71)	4.6.2 多态性	(86)
4.2.2 类	(71)	4.7 接口	(86)
4.3 Java 的类	(71)	4.7.1 接口的概念	(87)
4.3.1 类首说明	(72)	4.7.2 接口的说明	(87)
4.3.2 类体	(72)	4.7.3 接口的使用	(88)
4.3.3 方法说明	(75)	4.8 包	(89)
4.4 继承	(80)	4.8.1 包说明	(89)
4.4.1 类继承的语法形式	(81)	4.8.2 包的层次结构	(89)
4.4.2 类的层次结构	(82)	4.8.3 包的使用	(90)
4.4.3 抽象类和方法	(83)	4.8.4 Java 包	(92)
4.4.4 最终类(final 类)	(84)	4.9 综合应用示例	(92)
4.5 对象的创建与使用	(84)	4.9.1 单向链表	(92)
4.5.1 对象说明	(84)	4.9.2 队	(97)
4.5.2 对象的实例化 和初始化	(84)	4.9.3 栈	(98)
4.5.3 对象的成员变量 及方法的访问	(85)	小结	(99)
		习题	(99)

第五章 异常处理

5.1 异常类的层次	(104)	5.5 finally 语句	(109)
5.2 throws 抛出异常	(105)	5.6 正确地使用异常	(113)
5.3 throw 抛出异常	(106)	小结	(113)
5.4 try - catch 异常处理	(107)	习题	(114)

第六章 Java 的基本可重用类

6.1 Java 可重用类的结构	(115)	6.3.2 日期时间类	(123)
6.2 Java 语言包(java.lang)	(116)	6.3.3 向量类及其使用	(125)
6.2.1 字符串类	(116)	6.3.4 哈希表类及其应用	(128)
6.2.2 数组	(118)	6.3.5 栈类	(130)
6.2.3 Math 类	(122)	6.4 输入/输出包(java.io)	(130)
6.3 java.util 包	(123)	6.4.1 流(stream)	(130)
6.3.1 java.util 包的构成	(123)	6.4.2 基本输入/输出流	(132)

6.4.3 File 类····· (134)	6.4.7 文件拷贝实例····· (141)
6.4.4 字符流文件····· (136)	小结····· (149)
6.4.5 字节流文件····· (137)	习题····· (149)
6.4.6 过滤流····· (138)	

第七章 Java 小应用程序 Applet

7.1 Applet 程序开发步骤····· (153)	7.5 Applet 的应用····· (156)
7.1.1 编辑 Applet 的 Java 源文件····· (153)	7.5.1 利用 Applet 接收从 HTML 中 传递过来的参数····· (156)
7.1.2 编译 Applet····· (154)	7.5.2 利用 Applet 显示图像·· (161)
7.1.3 创建 HTML 文件····· (154)	7.5.3 利用 Applet 播放声音·· (163)
7.1.4 使用 applet viewer 运行 JavaWorld.HTML····· (154)	7.5.4 在 Applet 之间进行 通信····· (166)
7.1.5 使用浏览器观察 JavaWorld.HTML 运行·· (154)	7.6 Applet 的事件及其处理····· (172)
7.2 Applet 的安全····· (154)	7.6.1 Applet 中的鼠标操作·· (172)
7.3 Applet 类····· (155)	7.6.2 Applet 中的键盘操作·· (175)
7.4 Applet 的 4 种常用方法····· (156)	7.7 利用浏览器浏览 Applet····· (179)
7.4.1 init()方法····· (156)	7.7.1 远程浏览 Applet····· (179)
7.4.2 start()方法····· (156)	7.7.2 查看 Java 错误····· (180)
7.4.3 stop()方法····· (156)	小结····· (181)
7.4.4 destroy()方法····· (156)	习题····· (181)

第八章 图形用户界面(GUI)

8.1 抽象窗口工具箱 ——AWT 组件····· (183)	8.4 菜单····· (196)
8.2 基本组件····· (184)	8.4.1 菜单和菜单条····· (196)
8.2.1 按钮····· (184)	8.4.2 菜单项····· (196)
8.2.2 标签····· (185)	8.4.3 画布····· (197)
8.2.3 文本框和多行文本框·· (186)	8.4.4 使用 AWT 组件绘图···· (197)
8.2.4 复选框····· (188)	8.5 AWT 中的其他类····· (201)
8.2.5 单选框····· (189)	8.6 事件处理····· (201)
8.2.6 下拉式列表····· (190)	8.6.1 Event 对象····· (202)
8.2.7 列表····· (191)	8.6.2 如何实现事件处理 程序····· (202)
8.2.8 滚动条····· (192)	8.6.3 典型事件处理····· (202)
8.3 容器组件····· (194)	8.7 用组件构造用户接口····· (207)
8.3.1 面板····· (194)	8.7.1 将组件加到容器中···· (207)
8.3.2 框架····· (194)	8.7.2 Component 类提供 的功能····· (207)
8.3.3 对话框····· (195)	

8.8 组件在容器中的布局	(208)	小结	(217)
8.8.1 使用布局管理器	(208)	习题	(218)
8.8.2 示例程序	(215)		

第九章 Java 的多线程机制

9.1 什么是多线程机制	(222)	9.3.4 ThreadGroup 类	(230)
9.2 多线程实现机制	(222)	9.3.5 线程的同步	(231)
9.2.1 继承 Thread 类	(223)	9.4 线程通信	(232)
9.2.2 实现 Runnable 接口	(224)	9.4.1 主存读/写通信	(232)
9.3 Thread 类	(225)	9.4.2 管道流通信	(233)
9.3.1 Thread 类成员变量 及方法	(225)	9.5 线程死锁	(234)
9.3.2 线程状态	(226)	9.6 线程控制综合示例	(235)
9.3.3 Thread 优先级	(228)	小结	(242)
		习题	(242)

第十章 Java 网络编程

10.1 Java 网络编程基本概念	(243)	10.4.3 客户方实现	(260)
10.1.1 通信与协议	(243)	10.5 URL 类编程	(261)
10.1.2 套接口编程概述	(244)	10.5.1 URL 类简介	(261)
10.2 Java 网络包(java.net.)	(245)	10.5.2 构造 URL 类对象	(262)
10.2.1 套接口类(Socket)	(245)	10.5.3 构造 URL 类对象中 的异常	(263)
10.2.2 服务器套接口类 (ServerSocket)	(246)	10.5.4 URL 类获取 URL 特征 的主要方法	(263)
10.3 使用 TCP 协议的 Socket 网络 编程	(246)	10.5.5 URL 类获取 URL 对象 内容	(263)
10.3.1 一对一的 Socket C/S 通信	(246)	10.6 URLConnection 类的使用	(267)
10.3.2 综合示例:一对多通信 的实现	(251)	10.6.1 利用 URLConnection 类对象 读取 URL 内容	(267)
10.4 使用 UDP 协议 的 Socket 网络编程	(258)	10.6.2 利用 URLConnection 类对象 向 URL 对象发送服务请求 及参数	(268)
10.4.1 数据报套接口 (DatagramSocket) 类和数据包 (DatagramPacket) 类	(258)	小结	(270)
10.4.2 服务器方实现	(259)	习题	(270)

第十一章 Java 与数据库的连接——JDBC 技术

11.1 概述	(271)	11.2.1 JDBC 的基本功能	(271)
11.2 JDBC 的基本功能与特点	(271)	11.2.2 JDBC API 的特点	(271)

11.2.3 JDBC 与 ODBC 和其他 API 的比较	(272)	11.4.1 与数据库建立连接	(276)
11.2.4 JDBC 在数据库访问的两层 与三层模型上的应用	(273)	11.4.2 执行查询语句	(277)
11.3 JDBC API	(274)	11.4.3 检索结果集	(279)
11.3.1 使用方法	(274)	11.4.4 更新数据库操作	(280)
11.3.2 安全性问题	(274)	11.4.5 参数的输入和输出	(281)
11.3.3 JDBC 接口概貌	(275)	11.5 动态数据库访问	(282)
11.4 JDBC 应用	(276)	11.6 用 JDBC 访问数据库示例	(284)
		小结	(291)
		习题	(291)

第十二章 JBuilder 可视化开发环境的使用

12.1 JBuilder 开发环境	(292)	12.2.5 给 FontChooser 附加一个 菜单项事件	(312)
12.1.1 JBuilder 开发环境	(292)	12.2.6 给 JColorChooser 附加一个 菜单项事件	(313)
12.1.2 创建新工程	(293)	12.2.7 添加一个清除文本区 的菜单事件处理器	(314)
12.1.3 编辑源代码	(294)	12.2.8 添加一个文件选择 对话框	(315)
12.1.4 设计用户界面	(294)	12.2.9 添加从文件中读出文本 的代码	(315)
12.1.5 编译和运行工程	(295)	12.2.10 添加一个保存文件 的菜单项	(316)
12.1.6 用 UML 使代码 可视化	(296)	12.2.11 添加测试文件是否被修改 的代码	(318)
12.1.7 程序调试	(296)	12.2.12 激活工具栏按钮	(320)
12.1.8 单元测试	(296)	12.2.13 将事件处理与文本区 联系起来	(322)
12.1.9 程序打包	(298)	12.2.14 在文本区添加一个 右击弹出菜单	(323)
12.1.10 由 API 源文件生成 Javadoc (Java 文本)	(298)	12.2.15 在窗口的标题栏显示 文件名和状态	(324)
12.1.11 团队开发	(299)	12.2.16 将 Text Editor 应用程序 打包成一个 JAR 文件	(326)
12.1.12 开发数据库应用	(300)	12.3 编译、运行和调试	(332)
12.1.13 开发 Web 应用	(300)	12.3.1 打开例子 DebugTutorial	(333)
12.1.14 开发企业 JavaBeans (EJB)	(301)	12.3.2 修改语法错误	(333)
12.1.15 开发采用 XML 的应用	(301)		
12.1.16 使用帮助	(301)		
12.2 JBuilder 开发环境的应用实例——用 Java 开发的文本编辑器	(302)		
12.2.1 创建工程	(303)		
12.2.2 添加一个文本区	(306)		
12.2.3 创建菜单	(309)		
12.2.4 添加字体选择 对话框	(310)		

12.3.3 纠正编译错误	(335)	12.3.6 改正 oddEven()方法中	
12.3.4 改正修复 subtractValues()		的错误	(346)
方法中的错误	(338)	小结	(349)
12.3.5 改正 divideValues()方法中		习题	(349)
的错误	(343)	参 考 文 献	(350)

第一章 计算机语言概述

在计算机技术发展过程中,出现过各种各样的程序设计语言。从程序设计语言发展过程来分,可分为机器语言、汇编程序设计语言、高级程序设计语言(如 BASIC 语言、FORTRAN 语言、PASCAL 语言、C 语言、C++、Prolog 语言、LISP、Java 语言等)和数据库管理语言等。

1.1 机器语言与汇编语言

仅由硬件组成的计算机只能接受由“0”和“1”组成的二进制信息。要计算机执行一定的操作,就要编写一系列的二进制代码。这种不需翻译即由计算机直接执行的指令叫做机器指令。这些指令的集合叫做机器语言。每一条机器指令都是一串二进制代码,因此,要记住每一指令及其含义是十分困难的,编写出来的程序难以阅读,而且由于它完全依赖于硬件系统,不同的机器有不同的指令系统,因此,它不具有兼容性。一台机器上编制的程序在另一台机器上根本无法运行,一个问题要在多个机器上求解,就必须重复地编写多个应用程序。这种程序直观性差,难以编写、调试、修改、移植和维护,由于这一问题的存在,人们期待用更接近于自然语言与数学语言的语言代替机器语言,汇编语言应运而生。

汇编语言是一种面向机器的程序设计语言,它用符号表示机器指令,例如用 ADD 代替机器语言中的加法运算。这种语言编写的程序不能直接运行,要经过汇编程序翻译成机器语言才能运行,一般来说汇编语言指令与机器语言指令之间是一一对应的。由于汇编语言一般都是为特定计算机或计算机系统设计的,因此,它虽然比机器语言好学、便于记忆,比用机器码编写程序省事了一点,但仍然没有解决语言对硬件的依赖关系。

1.2 高级程序设计语言

40 多年以来,计算机高级程序设计语言与计算机硬件、计算机应用、计算机网络相依存,一同得到了空前的发展。历史上、甚至当今仍然比较流行的高级语言有以下数种。

1. FORTRAN 语言

FORTRAN 语言是世界上最早出现的高级语言,它所支持的数据类型比较丰富,特别适合于科学和工程中的数值计算。

一个 FORTRAN 程序由一个主程序与若干个子程序组成。主程序与每一个子程序都是一个独立的程序单位,称为一个程序模块。在 FORTRAN 语言中,子程序是实现模块化的有效途径。

2. ALGOL 语言

ALGOL 语言是另一个早期研制出来的高级语言。它有严格的文法规则。ALGOL 是一个

分程序结构的语言。分程序的结构可以是嵌套的,也就是说,分程序内可以含有别的分程序。过程也可以看成是一个分程序,这个分程序可以在别的分程序中被调用。

3. COBOL 语言

COBOL 语言是专门为处理商业事务而设计的一种通用语言,使用于商业和管理领域。其特点是针对商务和管理过程中要处理大量记录形式的数据的特殊需要,侧重于数据项和输入、输出记录的处理。

COBOL 语言把数据描述引入语言,数据描述与具体计算机无关,这一点后来又对数据库管理系统的发展产生重大影响。至今仍在商务处理领域得到一定的应用。

4. PASCAL 语言

PASCAL 语言是最早出现的结构化程序设计语言,在 PASCAL 语言中分程序和过程这两个概念合二为一,统一为过程,PASCAL 过程可以嵌套和递归。用它编写出来的程序结构性好,程序易读。这种语言广泛用于一些计算机课程的教学,也用于科学计算。

5. C 语言

C 语言是 20 世纪 70 年代发展起来的一种通用的结构化程序设计语言,特别是在系统软件开发上应用很广,它具有高级语言的诸多特点,也具有汇编语言控制硬件的能力,还可以比较方便地定义其他语言编写的程序。

字符、整数和浮点数是 C 语言的基本数据对象,用户可以构造指针、数组、结构和联合等复杂的数据类型。C 语言是一种较低级的语言,它提供了指针和地址操作的能力。C 语言提供书写结构良好的程序所需的控制结构。C 语言与 UNIX 操作系统紧密相关,UNIX 操作系统及其上的许多软件都是用 C 语言编写的。

上述语言主要擅长描述计算机问题的解决过程,所以叫它们是面向过程的语言。

6. 面向对象程序语言 C++

C++ 语言是在 C 语言的基础上发展起来的,与 C 语言兼容,但是,大大扩充、改善了 C 语言的性能。在 C++ 语言中,最主要的是增加了类功能,使它成为面向对象的程序设计语言。一般认为,面向对象程序语言至少包含下面一些概念:

(1) 对象 对象是客观存在的事物,它具有特征和行为两方面属性。对象是相互关联的,通过这种关联来相互影响(改变对象的特征)。面向对象语言把对象特征和行为封装在一起,对象内部才能对对象的“私有数据”进行操作,对象外部只能按提供的公开的操作,才能查询和修改对象的特征。这样,对象特征的具体表示和操作的具体实现都是隐蔽的。

(2) 类 类是对象共同特征和行为的描述。对象是类的实例。

(3) 继承 对象有特性,但是对象更有共性,共性存在于特性之中。继承反映了上述对象共性与特性的关系。子类继承父类就是说,子类具有父类的特征和行为。子类又可以对父类的特征和行为重定义或部分修改,达到代码重用的目的。

7. 面向逻辑的语言 Prolog

Prolog 语言擅长描述解决问题的规则,问题的求解是运用这些规则推理的结果。人们把这种语言叫做面向逻辑的语言,它在人工智能领域得到了广泛的应用。

8. 典型的函数型程序语言 LISP

函数是一种对应规则(映射),它使其定义域中每一个值和值域中惟一的值相对应。由于用函数程序设计语言书写的程序是利用自变量的值来计算函数的值,所以它没有副作用。

9. 纯面向对象程序设计语言 Java

Java 是一种跨平台的、适合于分布式计算机环境的面向对象的编程语言。它具有可移植、稳定、简单、高性能、动态执行等特性。

(1) Java 语言完全面向对象 C++ 等面向对象的程序设计语言是由原来面向过程的语言改造而来,而 Java 是纯面向对象的程序设计语言。

(2) 移植性 程序的可移植性指的是程序不经修改而在不同硬件或软件平台上运行的特性。Java 支持两种层次的可移植性:源代码级可移植性和字节码级的可移植性。

(3) 稳定性 Java 不支持指针数据类型,它提供了字节码的检查机制,Java 还提供了自动的“内存垃圾”收集功能,这些机制大大提高了 Java 的稳定性和安全性。

(4) 简单性 Java 的避免使用 C++ 等语言的操作符重载、多继承和数据类型自动转换等,这些都有利于语言简单化。

(5) 高性能 Java 通过支持多线程、高效的字节码来提高语言性能。

(6) 动态特性 Java 的动态特性是其面向对象设计的延伸。这种特性使得 Java 程序能够适应不断变化的执行环境。Java 通过“滞后联编”,支持类库升级之后,相应的应用程序不必重新编译,也一样可以利用升级后类库的新增功能。Java 的动态特性还体现在其对动态数据类型和动态协议的支持上。利用一种特殊的 Applet,即内容句柄,编程人员可很方便地支持新的数据类型。Java 的动态性的价值是使程序员真正拥有“即插即用”(Plug - and - Play)的软件模块功能。

(7) 分布式 Java 支持数据分布和操作分布。

传统的程序设计过程是抽象思维过程,只有程序设计完成后,通过上机调试,方知程序的正确与错误。可视化程序开发环境,使程序一边开发,一边看到程序运行的结果,大大提高了软件的开发效率。可视化程序开发是以面向对象程序设计为基础的。可视化面向对象的程序设计方法得到普遍的认可。许多传统的非面向对象语言都经过扩充、改造成为支持可视化的面向对象的程序设计语言。PASCAL 演化为 Delphi, BASIC 演化为 VB, C 语言演化为 VC++、C++ Builder 以及 JBuilder 等。这些可视化集成开发环境成为当前软件开发环境的主流。

1.3 语言翻译程序

计算机只能识别、运行本身的机器语言(0、1 构成的指令系统),另一方面,人们又希望使用高级程序设计语言描述要计算机解决的问题。因此,如何使高级程序设计语言程序映射成等价的机器语言程序成为关键,语言翻译程序就是起这种作用(如图 1.1 所示)。

按语言程序的翻译方式不同,具体可分为解释型翻译程序和编译型翻译程序。

解释型翻译程序在语言程序翻译时,它读入源程序一句,翻译一句,执行一句,这样反复操作直到最终完成。BASIC 语言是典型的解释型语言。

编译型翻译程序也叫编译程序,它在翻译语言程序时,加工整个源程序,最终翻译成机器语言,然后交给计算机执行。编译程序有利于目标程序的优化,有利于提高目标程序的运行速度。FORTRAN 语言、C 语言、PASCAL 语言都是编译型语言。

有时人们形象地把解释型翻译程序比做“口译”,把编译程序比做“笔译”。由此可见,由于

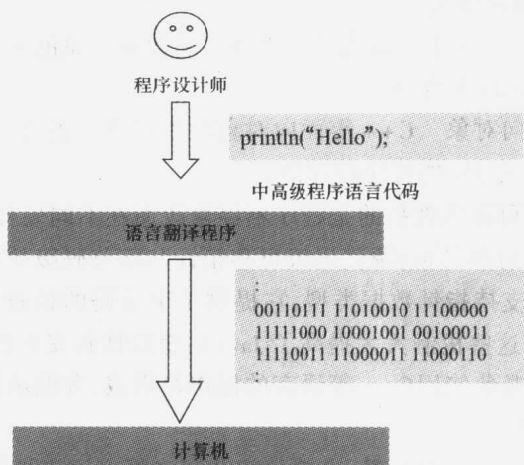


图 1.1 语言翻译程序的作用示意图

计算机有了一种语言的翻译程序,使计算机能“明白”用这种语言编写的程序。因此,配置了某种语言的翻译程序的计算机又被人们称为这种语言的虚拟机。不言而喻,语言翻译程序与操作系统和计算机硬件有关,由翻译程序产生的目标程序也与操作系统和计算机硬件有关。不同操作系统下的同一种语言的语言翻译程序是不一样的,同一操作系统下(如 Windows 环境)硬件不一样(如 x86 和 PowerPC),同一种语言的翻译程序也是不一样的(如图 1.2 所示)。

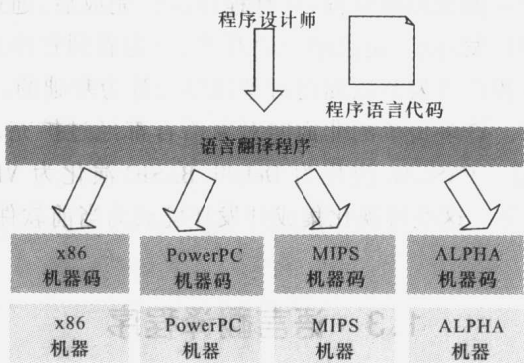


图 1.2 语言翻译程序与机器硬件有关

1.4 虚拟机、Java 虚拟机与 Java 运行环境

由上可见,编译程序与操作系统和计算机硬件有关,为了提高编译程序的可移植性,人们提出了虚拟机的理论。虚拟机好似通用的计算机,有自己的指令系统,但本身没有实际的硬件。为了使虚拟机代码可以执行,必须有虚拟机实时运行支持系统把虚拟机代码转换成相应硬件机器的代码(如图 1.3 所示),然后加以执行。有了虚拟机,编译程序首先把语言程序翻译成虚拟机代码,这样的编译程序可移植性就大大提高了。虚拟机代码与机器无关,它不仅可以

在本机上执行,也可以通过网络传输到其他配制了相应的虚拟机实时运行支持系统的网点上执行。这一特征,显示了虚拟机在网络时代的无限生机。

计算机网络的发展对计算机语言不断提出新的要求,特别是要求语言具有可移植性、安全性,这正是 Java 成为网络应用开发主流语言的原因。

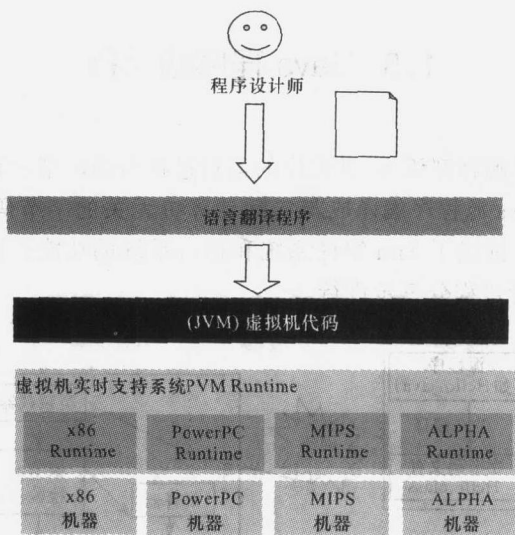


图 1.3 虚拟机原理示意图

JVM (Java Virtual Machine, Java 虚拟机) 是一种虚拟机,从结构上看它与实际的计算机相似,它的作用是使得一台实际的机器能够运行 Java 字节码程序。Java 字节码是 Java 源程序编译后的程序,它不能被计算机直接执行,但它可以被所有的 Java 虚拟机执行。这就是 Java 字节码程序可以在网络上移植的原因。

Java 程序必须有自己的运行环境 (Java 平台)。一个 Java 运行环境包括实际计算机、适配器、Java 虚拟机、Java 基本软件和 Java 应用程序接口,如图 1.4 所示。JVM 是 Java 运行环境的核心,JVM 的下层是和实际计算机的接口,称为适配器,不同类型的计算机其适配器是不同的。Java 的基本软件也称为基本类,而 Java API (Application Program Interface, 应用程序接口) 是已编译好的程序代码库,可以直接使用它们,以节约编程的时间。事实上,Java 的基本类和 API 的规模并不固定,许多 Java 平台对这两部分内容进行了补充,即除了基本类外,还有扩展类。

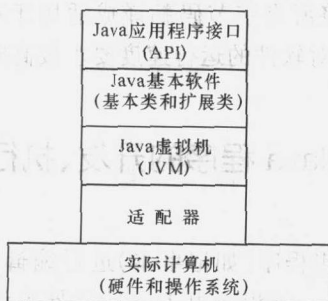


图 1.4 Java 运行环境

由此可见,Java 虚拟机与操作系统和计算机硬件无关。Java 经过编译后的字节码程序可以在网络上传输到任意支持 Java 虚拟机的网点机上运行。

利用 Java 语言可以开发两种形式的应用程序:Java 应用程序(Java Application)和 Java 小应用程序(Java Applet)。Java 小应用程序是借助浏览器运行的程序。

1.5 Java 程序的运行

Java 程序是半编译半解释型语言,其程序的运行过程与编译型和解释型语言都不同。

首先,编译程序将 Java 源程序编译成与实际计算机无关的字节码,然后 Java 运行系统解释并执行字节码。图 1.5 描述了 Java 编译系统和运行系统的功能。图中,运行系统解释 Java 小应用程序 Applet 的执行过程分三步进行。

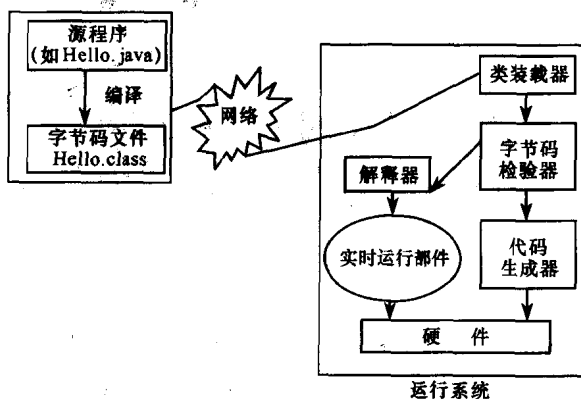


图 1.5 Java 编译和运行系统功能图

第一步,由“类装载器”具体完成字节码的装载,装载时运行系统确定程序的内存分配。

第二步,“字节码检验器”对字节码进行安全性检查,这种检查可以排除字节码可能存在的违法访问权限、不规范数据类型和非法调用等问题。

第三步,完成字节码的翻译和执行。这里有两种途径:一种是走运行系统左边的路径,通过“解释器”将一个个字节码翻译成机器码,然后由“实时运行部件”立即将机器码送硬件执行,这是一种解释工作方式,边翻译边执行,Java 系统一般采用这种方式;另一种是走运行系统右边的路径,通过“代码生成器”先将所有字节码翻译成适用于本系统的机器码,然后送硬件执行,这是一种编译型工作方式,当对软件的运行速度要求较高时使用编译型工作方式。

1.6 Java 程序的开发、执行流程

Java 的源程序可以用任何编辑程序(如记事本)进行编辑,然后以纯文本方式存储在文件中,文件必须以 java 作为后缀。Java 源程序由 Java 编译器编译成字节码,存储在以 class 为后缀的文件中,最后由运行系统解释和执行,如图 1.6 所示。

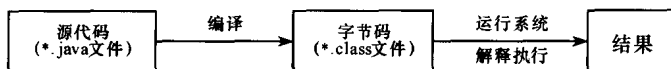


图 1.6 Java 程序的开发、执行流程

目前流行的 Java 开发环境(即工具)有多种。从教学目的看,JDK 环境有利于学生理解整个程序各部分之间的关系,有利于培养学生程序设计思想。JBuilder 环境方便开发大程序,但是往往一开始人们难于把握整个程序各部分之间的关系。本书在前七章以利用 JBuilder 6 与 JDK 1.2 兼容的部分功能开发 Java 程序为主,以后各章过渡到 JBuilder 6 环境下开发。

小 结

用户可以通过图标、菜单和命令使用计算机,这是最简单的人机交互方式。程序设计语言通常分为低级程序设计语言和高级程序设计语言,具体有面向过程的语言、面向逻辑的语言、函数式设计语言和面向对象的语言。语言程序必须由该语言的翻译程序映射成机器语言才能执行。语言翻译程序分为解释型翻译程序和编译型翻译程序。

Java 是纯面向对象的程序设计语言,是半编译半解释型语言。Java 由于其可移植性、安全性、分布性和高性能性,成为网络应用开发的主流语言。JDK 1.2、JBuilder 6 是典型的 Java 程序开发环境,可视化集成开发环境 JBuilder 6 主要适用于开发大型 Java 软件。

习 题

1. 叙述通常使用的计算机的主要接口。
2. 叙述计算机语言的主要分类。
3. 叙述不同语言翻译程序的特点。
4. 什么是虚拟机? 什么是 Java 运行环境?
5. 叙述 Java 程序开发的步骤。