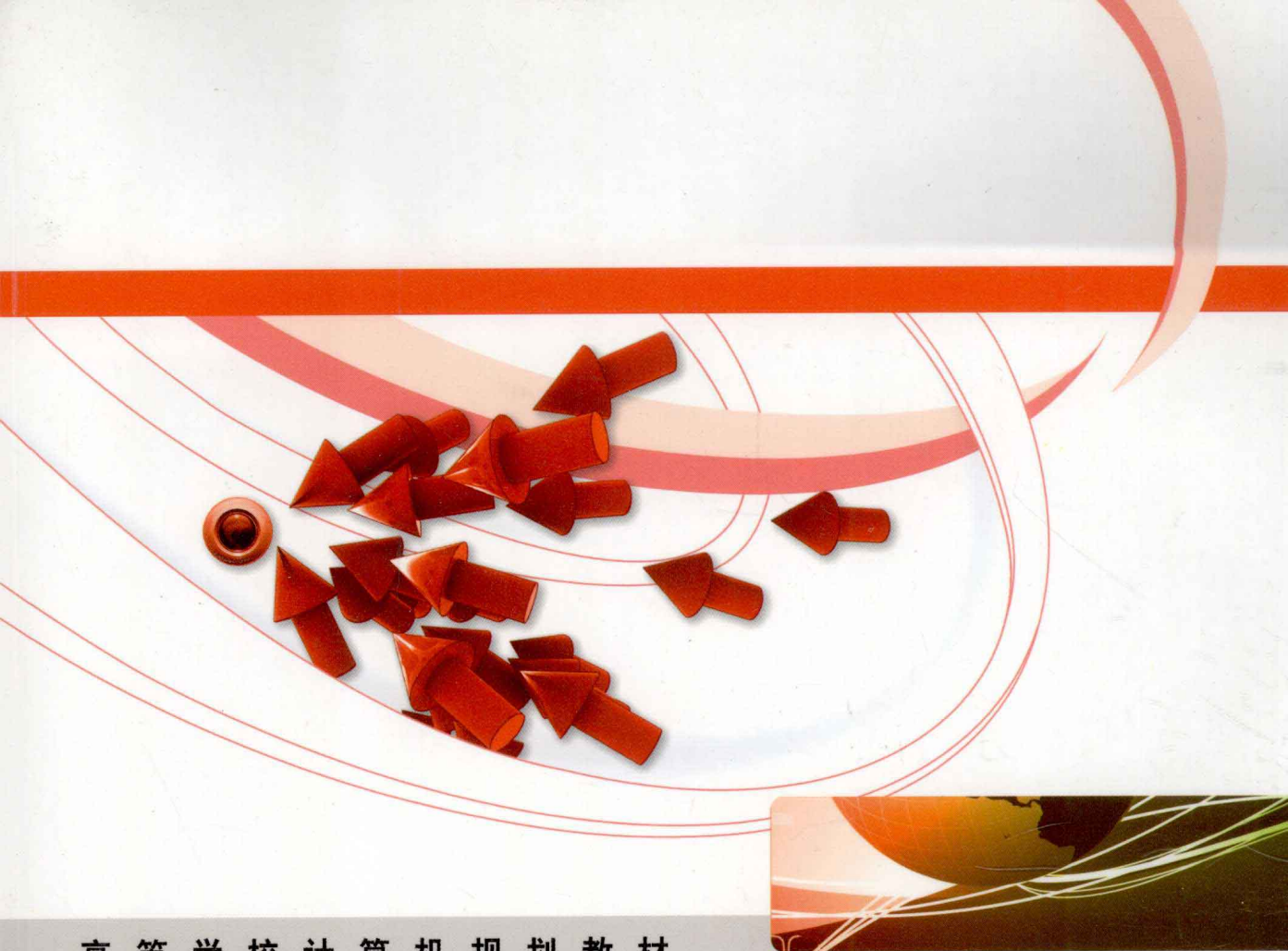




高等学校计算机规划教材

Java程序设计与开发

■ 张晓龙 主编 ■ 边小勇 刘 钊 赵红武 副主编



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

高等学校计算机规划教材

Java 程序设计与开发

张晓龙 主编

边小勇 刘 钊 赵红武 副主编

电子工业出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

本书以 Java 语言面向对象的编程理念为出发点,循序渐进地介绍 Java 语言的基本内容,包括基本语法、数组、字符串、类和对象、I/O 流和异常处理、用户界面类和 Applet、网络和多线程编程;同时还着重介绍几种 Java 核心技术及相应的开发实例,包括 JDBC、Servlet、JSP 和 EJB。通过融入 Java 面向对象的编程思想,使读者在学习理论知识的同时,初步掌握面向对象的系统分析、设计与开发能力,为大型软件的设计和开发奠定基础。本书配有 PPT、源代码、习题答案等教学资源。

本书可作为高等学校计算机及相关专业“Java 程序设计”课程的教材,也可作为软件开发人员、工程技术人员及其他相关人员的自学参考书或培训教程。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容
版权所有·侵权必究

图书在版编目(CIP)数据

Java 程序设计与开发/张晓龙主编. —北京:电子工业出版社, 2010.9

高等学校计算机规划教材

ISBN 978-7-121-11656-8

I. ①J… II. ①张… 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(2010)第 162750 号

策划编辑:史鹏举

责任编辑:史鹏举 特约编辑:王 崧

印 刷:北京市李史山胶印厂

装 订:

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×1092 1/16 印张:16.5 字数:475 千字

印 次:2010 年 9 月第 1 次印刷

定 价:28.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010)88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010)88258888。

前 言

随着互联网的普及和发展,面向 Web 的程序设计和开发已成为软件编程的新模式,而 Java 日渐成为互联网编程的一种主流语言之一。自 20 世纪 90 年代早期以来,Java 在 IT 业的应用一直保持增长的趋势。高等院校的计算机和非计算机专业相继开设了 Java 程序设计课程。

本书的目的是使学生在理解 Java 语法特点的基础上,逐步掌握 Java 语言面向对象的编程要领,同时鼓励学生利用面向对象编程来解决实际问题。

本书以 Java 语言面向对象的编程理念为出发点,循序渐进地介绍 Java 语言的基本内容,包括基本语法、数组、字符串、类、抽象类、接口、网络和多线程编程;同时还着重介绍几种 Java 核心技术,包括 JDBC、Servlet、JSP 和 EJB。通过融入 Java 面向对象的编程思想,使读者在学习理论知识的同时,初步掌握面向对象的系统分析、设计与开发能力,为大型软件的设计和开发奠定基础。本书在讲解 Java 语言和 Java 核心技术的同时,辅以大量的实例和结果分析帮助读者快速理解概念的本质,进一步掌握 Java 语言及其技术的精髓。本书在实例的安排上,由点到面、由浅入深,并在面向对象的设计思想和技巧上进行深入思考和探讨。

本书第 1 章主要介绍 Java 语言的特点和优势,使读者对 Java 语言有一个概括性的了解。第 2 章介绍 Java 语言基础,主要包括数据类型、变量、表达式、流程控制语句、数组和字符串。第 3 章介绍 Java 类和对象,包括 Java 类的设计,对象创建和构造方法的执行过程;多态性,类、变量和方法的修饰符(访问和非访问修饰符);内部类,抽象类和接口,包及其作用。第 4 章介绍 Java 的 I/O 流和异常处理,包括 System I/O 类、Scanner 类、File 类、字节级输入输出流、字符级输入输出流、对象序列化、异常捕获和处理。第 5 章介绍 Java 图形用户界面和 Applet,使用 AWT 包和 Swing 包生成图形用户界面,同时使用多种布局管理器设计友好的用户界面。第 6 章介绍 Java 网络编程,包括基于高层 URL、基于套接字 Socket 和基于数据报 Datagram 等 3 种编程方式,以及使用文件 I/O 流辅助网络编程。第 7 章介绍 Java 多线程编程,包括线程的创建和启动、线程的优先级、状态切换和多线程的同步处理。第 8 章介绍 Java 数据库连接技术(JDBC),以及利用 JDBC 编写数据库访问程序的步骤。第 9 章介绍 Servlet 和 JSP,包括 Java EE Web 开发概念,JavaBean,Servlet 概念及其生命周期,Servlet 类的编写、配置和运行,JSP 及其编写、配置和运行,JSP 的两种设计模式。第 10 章介绍 RMI 和 EJB 技术,利用 RMI 机制设计 RMI 客户端和服务端程序的步骤,介绍了 EJB 规范、会话类 EJB 组件的创建和部署,以及编写 EJB 客户端程序。

本书配有 PPT、源代码、习题答案等教学资源,需要者可从华信教育资源网 <http://www.hxedu.com.cn> 免费注册下载。

本书适合作为计算机及相关专业的本科生、研究生“Java 程序设计”课程的教材,也可作为软件开发人员、工程技术人员及其他相关人员作为自学参考书或培训教程。

本书由张晓龙任主编。其中,第 1、3、5 章由张晓龙编写;第 2、4 章由赵红武编写;第 6、7 章由刘钊编写;第 8、9、10 章由边小勇编写。书中的所有实例都在 JDK 1.6 中调试通过。本书编写过程中,得到了武汉科技大学计算机学院人工智能与机器学习研究室成员的大力支持,在此,谨向他们表示衷心的感谢。

由于时间仓促,编者水平有限,书中难免存在疏漏和不足,恳请读者批评指正,使本书得以改进和完善。

编 者
2010 年 7 月

目 录

第 1 章 Java 语言概述	(1)
1.1 Java 语言及其产生	(1)
1.2 Java 的特点	(1)
1.3 Java 的体系结构	(3)
1.3.1 Java 的体系结构	(3)
1.3.2 Java 虚拟机	(3)
1.3.3 垃圾回收机制	(3)
1.3.4 代码安全检测	(4)
1.4 Java 的 3 种版本	(4)
1.5 Java 开发环境概述	(4)
1.5.1 JDK 的安装、配置和使用	(4)
1.5.2 JDK 工具箱	(6)
1.5.3 Eclipse 开发环境的安装、配置和使用	(6)
1.5.4 Java 程序的编写	(7)
1.6 本章小结	(9)
习题 1	(9)
第 2 章 Java 语言基础	(11)
2.1 Java 语言要素	(11)
2.1.1 标识符	(11)
2.1.2 关键字	(11)
2.1.3 分隔符	(12)
2.1.4 注释	(12)
2.2 基本数据类型	(13)
2.2.1 简单类型	(13)
2.2.2 引用类型	(14)
2.2.3 常量	(14)
2.2.4 变量	(15)
2.3 运算符和表达式	(16)
2.3.1 算术运算符	(16)
2.3.2 关系运算符	(17)
2.3.3 逻辑运算符	(17)
2.3.4 位运算符	(18)
2.3.5 赋值运算符	(19)
2.3.6 其他运算符	(20)
2.3.7 运算符的优先级与结合性	(21)

2.3.8	表达式与语句	(21)
2.4	控制语句	(22)
2.4.1	选择结构控制语句	(23)
2.4.2	循环结构控制语句	(25)
2.4.3	转移语句	(27)
2.5	数组	(29)
2.5.1	一维数组	(29)
2.5.2	二维数组	(31)
2.5.3	多维数组	(32)
2.6	字符串	(32)
2.6.1	字符串类	(33)
2.6.2	常用方法	(33)
2.7	综合应用实例	(34)
2.8	本章小结	(37)
	习题 2	(37)
第 3 章	Java 类和对象	(39)
3.1	面向对象基础	(39)
3.1.1	类	(39)
3.1.2	对象	(40)
3.1.3	消息	(40)
3.2	继承	(40)
3.2.1	创建子类	(40)
3.2.2	成员变量和方法	(41)
3.2.3	对象的构造和初始化过程	(44)
3.2.4	上转型对象	(45)
3.2.5	关键字 this 和 super	(47)
3.2.6	对象的撤销和清理	(49)
3.3	多态	(50)
3.3.1	基本概念	(50)
3.3.2	方法重载	(50)
3.3.3	方法覆盖	(51)
3.3.4	变量覆盖	(53)
3.4	修饰符	(53)
3.4.1	访问修饰符	(53)
3.4.2	非访问修饰符	(57)
3.5	内部类和匿名内部类	(59)
3.5.1	内部类	(59)
3.5.2	匿名内部类	(61)
3.6	接口	(63)
3.6.1	接口的定义	(63)

3.6.2	接口与多重继承	(63)
3.6.3	接口的实现	(64)
3.6.4	接口中的变量和方法	(64)
3.7	包	(66)
3.7.1	包的作用	(66)
3.7.2	Java 中定义的包	(66)
3.7.3	自定义包的创建和使用	(66)
3.7.4	编译和生成包	(67)
3.8	综合应用实例	(68)
3.9	本章小结	(70)
习题 3		(71)
第 4 章	Java 的 I/O 流和异常处理	(73)
4.1	I/O 流概述	(73)
4.2	System I/O 类和 Scanner 类	(75)
4.2.1	System I/O 类	(75)
4.2.2	Scanner 类	(76)
4.3	文件类	(76)
4.3.1	使用 File 类访问文件系统	(76)
4.3.2	随机文件读写	(78)
4.4	字节级输入输出流	(79)
4.4.1	数据输入输出流	(79)
4.4.2	文件输入输出流	(80)
4.4.3	缓冲输入输出流	(81)
4.4.4	管道输入输出流	(81)
4.4.5	格式化输出流	(81)
4.5	字符级输入输出类	(82)
4.5.1	字符输入输出类	(82)
4.5.2	缓冲读写类	(83)
4.5.3	文件读写类	(86)
4.5.4	格式化写类	(88)
4.6	对象序列化	(89)
4.6.1	对象输入流	(89)
4.6.2	对象输出流	(90)
4.7	异常处理	(91)
4.7.1	Java 中定义的异常	(91)
4.7.2	异常类的层次结构	(92)
4.7.3	捕获异常	(92)
4.7.4	异常处理的嵌套	(96)
4.7.5	自定义的异常和使用	(97)
4.8	文件流综合应用实例	(98)

4.9 本章小结	(106)
习题 4	(106)
第 5 章 Java 用户界面类和 Applet	(108)
5.1 AWT 简介	(108)
5.2 AWT 容器类	(108)
5.3 基本控件类	(110)
5.4 图形类	(112)
5.5 事件类型及其处理	(112)
5.5.1 事件类	(113)
5.5.2 事件监听器	(113)
5.5.3 事件处理流程	(114)
5.6 布局管理器	(115)
5.6.1 FlowLayout	(115)
5.6.2 BorderLayout	(116)
5.6.3 GridLayout	(116)
5.6.4 CardLayout	(116)
5.6.5 GrdBagLayout	(117)
5.7 Swing 简介	(117)
5.7.1 Swing 组件的特点	(118)
5.7.2 Swing 包中的类	(118)
5.7.3 Swing 包中的控件类	(118)
5.8 Java Applet	(122)
5.8.1 Applet 简介	(122)
5.8.2 Applet 程序的编写	(124)
5.8.3 Applet 程序的编译和执行	(125)
5.8.4 Applet 的绘图和控制	(126)
5.9 GUI 设计综合应用实例	(128)
5.10 本章小结	(130)
习题 5	(130)
第 6 章 Java 网络编程	(132)
6.1 Java 网络基础	(132)
6.1.1 IP 地址	(132)
6.1.2 端口	(133)
6.1.3 套接字	(134)
6.1.4 数据报	(134)
6.1.5 传输协议	(135)
6.2 基于 URL 的高层次 Java 网络编程	(135)
6.2.1 从 URL 中读取内容	(135)
6.2.2 建立 URL 连接并读取内容	(136)
6.3 基于 Socket 的网络编程	(137)

6.3.1	Socket 通信过程	(138)
6.3.2	创建客户端 Socket	(139)
6.3.3	创建服务器端 Socket	(139)
6.3.4	打开输入输出流	(139)
6.3.5	关闭流和 Socket	(141)
6.4	基于 Datagram 的网络编程	(141)
6.4.1	数据报和套接字	(142)
6.4.2	创建客户端 Datagram	(142)
6.4.3	创建服务器端 Datagram	(142)
6.4.4	打开输入输出流	(143)
6.4.5	关闭流和 Datagram	(143)
6.5	网络编程与应用实例	(144)
6.6	本章小结	(150)
	习题 6	(150)
第 7 章	Java 多线程编程	(152)
7.1	线程基础	(152)
7.1.1	线程概述	(152)
7.1.2	线程的状态	(152)
7.1.3	线程的优先级	(153)
7.2	线程的创建	(153)
7.2.1	通过继承 Thread 类创建	(153)
7.2.2	通过实现 Runnable 接口创建	(155)
7.3	多线程的创建	(156)
7.4	线程的调度	(159)
7.4.1	线程优先级的获取与设置	(159)
7.4.2	线程睡眠	(159)
7.4.3	线程让步	(159)
7.4.4	线程等待	(160)
7.4.5	线程唤醒	(160)
7.5	多线程的同步处理	(161)
7.5.1	多线程的互斥	(161)
7.5.2	synchronized 方法	(161)
7.6	多线程综合应用实例	(164)
7.7	本章小结	(168)
	习题 7	(168)
第 8 章	数据库编程	(170)
8.1	数据库基础	(170)
8.1.1	关系数据库	(170)
8.1.2	SQL 基本内容	(171)
8.1.3	ODBC 技术	(173)

8.2	JDBC 数据库连接技术	(174)
8.2.1	JDBC 概述	(174)
8.2.2	JDBC 的实现及其驱动程序	(174)
8.2.3	JDBC 的常用类和接口	(175)
8.3	Java 访问数据库的基本步骤	(176)
8.3.1	加载驱动程序类	(176)
8.3.2	建立 JDBC 连接	(177)
8.3.3	执行 SQL 语句	(179)
8.4	JDBC 高级特性	(180)
8.4.1	预备语句	(180)
8.4.2	可滚动和可更新的结果集	(180)
8.4.3	元数据	(182)
8.4.4	事务	(183)
8.5	创建数据库	(184)
8.6	JDBC 综合应用举例	(185)
8.7	本章小结	(187)
	习题 8	(188)
第 9 章	Servlet 与 JSP	(189)
9.1	Java EE Web 开发技术	(189)
9.2	Servlet 基础	(190)
9.2.1	Servlet 定义	(190)
9.2.2	Servlet 工作方式	(191)
9.2.3	Servlet API	(192)
9.2.4	Servlet 异常	(193)
9.3	配置 Servlet 工作环境	(193)
9.3.1	Servlet 开发环境	(193)
9.3.2	Servlet 类的配置	(194)
9.3.3	编写简单的 Servlet	(195)
9.4	JavaBean 基础	(196)
9.4.1	JavaBean 基本概念	(196)
9.4.2	JavaBean 的属性、方法和事件	(197)
9.4.3	一个简单的 JavaBean	(197)
9.5	JSP 概述	(198)
9.5.1	JSP 基本概念	(198)
9.5.2	JSP 技术的特点	(199)
9.5.3	JSP 的工作方式	(199)
9.5.4	JSP 和 Servlet 的区别	(200)
9.6	编写简单的 JSP	(200)
9.7	在 JSP 中访问数据库	(200)
9.8	JSP 的设计模式	(202)

9.8.1	JavaBean 与 JSP 的结合	(202)
9.8.2	Servlet 与 JSP 的结合	(203)
9.9	JSP 的语法要素	(203)
9.9.1	JSP 脚本元素	(203)
9.9.2	JSP 指令和动作元素	(205)
9.9.3	JSP 内置对象	(206)
9.10	Web 综合应用实例	(209)
9.11	本章小结	(222)
习题 9		(223)
第 10 章	RMI 和 EJB	(224)
10.1	RMI 基础	(224)
10.1.1	RMI 概述	(224)
10.1.2	RMI 的目标和体系结构	(224)
10.1.3	RMI 的工作机制	(225)
10.1.4	用 RMI 设计应用系统	(226)
10.2	RMI API	(227)
10.3	Java RMI 与其他分布式技术	(228)
10.4	RMI 部署问题	(229)
10.5	编写一个 RMI 应用系统	(229)
10.6	EJB 技术	(233)
10.6.1	EJB 基本概念	(233)
10.6.2	EJB 组件	(234)
10.6.3	EJB 容器	(235)
10.6.4	开发和部署 EJB	(236)
10.6.5	EJB 和其他分布式技术	(237)
10.6.6	基于 EJB 的 Web 应用程序实例	(238)
10.7	本章小结	(244)
习题 10		(245)
附录 A	Java 运算符及其优先级	(246)
附录 B	Java 类路径和类加载器	(247)
参考文献		(249)

第 1 章 Java 语言概述

Java 是由美国 Sun 公司开发的一种完全面向对象程序设计语言。Java 语言由 C++ 语言发展而来, 但比 C/C++ 语言拥有更多特性和优势, 如简单、高效、健壮、可移植、多线程等特点。Java 已成为最流行的编程语言之一。使用 Java 进行程序设计和开发极大地降低了编程的复杂性, 提高了程序的可移植性。本章将对 Java 的发展和特性做一个比较全面的介绍。

1.1 Java 语言及其产生

20 世纪 70 年代的计算机革命使人们对成熟计算机软件的需求大增, 以便利用日益增长的计算机能力来处理数据。20 世纪 80 年代之后, 程序员经历了程序设计语言的又一次变革。新问题不断出现, 这些问题在于, 程序变得越来越复杂, 越来越难以设计、编写和管理, 从而难以跟上计算机硬件的发展速度, 导致了一代程序语言的更替。从 20 世纪 60 年代末起, 一种从 Simula 67 和 Smalltalk 发展来的设计概念使程序设计语言向前迈出了革命性的一步。这个时期, 面向对象程序设计成为新技术的代名词, 流行于整个软件行业。

随着 www 的发展, 面向对象程序设计语言的发展过程中又出现了另一个萌芽。由于 C 和 C++ 程序都是与机器相关的, 如果不重新编译, 程序就不能在另一种类型的计算机上运行。为解决此问题, 研究人员将注意力集中于开发一种与机器无关的程序设计语言上。1991 年, Sun 公司的 5 位技术专家 James Gosling、Patrick Naughton、Chris Warth、Ed Frank 和 Mike Sheridan 花了 18 个月的时间开发了一种称为 Oak 的编程语言, 在 1995 年正式发布时, 更名为 Java。

1.2 Java 的特点

1. 简单性

Java 是一种简单的程序设计语言。Java 的设计者尽量把语言的规模结构缩小, 通过提供最基本的方法来完成指定的任务。程序员只需要理解一些基本概念, 就可以编写出合适的应用程序。Java 语言还选择性地继承了 C++ 语言中的语法规则和面向对象的基本机制, 放弃了 C++ 语言中十分烦琐和难以理解的内容, 如指针、运算符的重载、类的多重继承, 并增加了垃圾回收功能, 极大地简化了程序的内存管理工作, 降低了编程学习的难度; 同时 Java 编译器很小, 便于在各种机型上实现。

2. 面向对象

Java 语言是一种纯粹的面向对象编程语言, 提供类、接口和继承等方法。Java 语言中的所有设计集中于对象及其接口上, 它提供了简单的类机制和动态的接口模型。对象中封装了变量及方法, 实现了模块化和信息隐藏; 而类提供了对象的原型。通过继承机制, 子类可以使用父类提供的方法, 从而实现了代码的复用, 并对不断变化的需求有很强的适应性。

3. 对象性

Java 应用程序可以通过 URL 地址, 在访问网络上任何位置的对象时, 实现的效果几乎与本地访

问文件系统相同。Servlet 机制的出现,使 Java 编程效率有了很大提高,并且很多大型 Web Server 都支持 Servlet。利用 Java 来进行分布式网络开发是其主要应用之一。

4. 健壮性

Java 语言提供了严密的语法规则,在编译和运行时具有独特的检测措施,有效地降低了程序出错的可能性。例如,引用一个非法类型或执行一个非法类型操作,Java 将在编译时指出相应的错误。

Java 语言提供了面向对象的异常处理机制,在编译时捕获未被处理的异常,防止系统崩溃的发生。Java 语言还提供了内存管理功能,能够自动跟踪程序使用的所有资源类型。当内存资源不再使用时能够自动回收,无须用户程序单独进行处理,减少了编程人员在内存管理上的困惑。

5. 安全性

由于 Java 编译器输出的并不是可执行文件(*.exe),而是字节代码(bytecode),Java 的运行系统就是 Java 虚拟机,每个 Java 程序的运行都是在 Java 虚拟机的控制之下。因此,下载安全的 Java 虚拟机是对付病毒入侵和保障程序安全最重要的手段,同时对 Java 程序进行解释执行的方式也是保证 Java 程序安全性的一个重要原因。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,这样就防止了程序员使用指针等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出现错误。此外,在运行程序时有严格的访问权限检查。

6. 体系结构中立

Java 是一门编译语言,但它不直接生成硬件处理器的指令,而是生成一种与体系结构无关的字节码指令,这种字节码由 Java 虚拟机解释执行。只要安装了 Java 运行时环境,Java 程序就可以在任意处理器上运行。这些字节码指令对应于 Java 虚拟机中的表示,Java 解释器得到字节码指令后,对它进行转换,使之能够在不同的平台上运行。

7. 移植性

移植性即平台无关性,是指一个应用程序能够运行在不同的操作系统平台,也就是 Sun 公司设计 Java 的宗旨“Write once, Run anywhere”。此外,Java 还严格规定了各个基本数据类型的长度。平台无关性使 Java 应用程序可以运行在任何操作系统平台上,也是区别于其他高级语言最重要的标志。由于 Java 虚拟机支持平台无关性,不同的操作系统需要与之相符的 Java 虚拟机版本。Java 提供了可移植的类库,而且,Java 编译器是用 Java 语言编写的,运行时环境由标准 C 实现。因此,Java 系统本身也就具有可移植性。

8. 解释性

如前所述,Java 运行时环境可以在任何主机上将程序代码编译为 Java 字节码格式,而字节码本身带有编译时产生的信息,使得程序连接过程更为简单。在运行 Java 程序时,由 Java 解释器对这些字节码进行解释执行,执行过程中所需访问的类被动态载入到运行时环境中。

9. 高效性

由于 Java 字节码格式的设计专门针对机器码的转换,自动的寄存器分配与编译器对字节码的一些优化能生成高质量的代码。因此,Java 程序可以快速地转换成机器码,直接执行。随着 Java 虚拟机的改进和即时编译技术 JIT (Just In Time) 的出现,使得 Java 的执行速度有了很大提高。

10. 多线程

Java 的特点之一就是内置了多线程的类库。Java 的多线程机制中,每个线程都完成特定的功能

和任务,各线程间并发执行,互不影响,并能协调一致地处理和共享数据。通过使用多线程,程序员不需要采用全局的事件循环机制,就很容易实现网络上的实时交互行为。

11. 多态性

Java 的设计使它适合于一个不断变化的环境。它允许程序动态地装入运行过程中所需的类,并可以自由地向类库中加入新的方法和实例变量而不影响用户程序的执行。Java 通过接口来支持多重继承,相比严格的类继承,具有更强的灵活性和可扩展性。这一特点使 Java 非常适合于网络编程和软件开发,即使是更新类库也不必重新编译。

1.3 Java 的体系结构

1.3.1 Java 的体系结构

Java 的体系结构如图 1.1 所示。

程序员编写的代码被编译器转换成 Java 类文件。在通过 Java 运行时环境时,使得类文件被翻译成字节码文件。解释器与虚拟机构成了 Java 的运行环境,它完成字节码映射到本地平台指令。

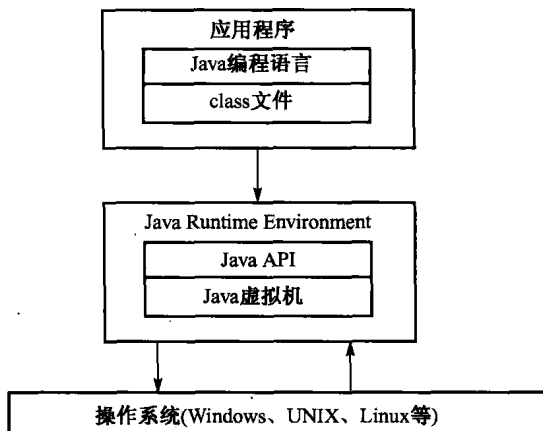


图 1.1 Java 的体系结构

1.3.2 Java 虚拟机

JVM(Java 虚拟机)是 Java Virtual Machine 的缩写。它是一个虚构出来的计算机,是通过在实际的计算机上模拟各种计算机功能来实现的。Java 虚拟机有自己完善的硬件架构,例如处理器、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与平台无关性,而使用 Java 虚拟机是实现这一特点的关键。一般的高级语言要在不同的平台上运行时,至少需要编译成不同的目标代码。而引入 Java 虚拟机后,它屏蔽了与具体平台相关的信息,使得 Java 编译程序只需生成能在 Java 虚拟机上运行的目标代码(字节码),然后,Java 虚拟机把字节码解释成具体平台上的机器指令执行。因此,Java 程序无须重新编译就可以跨平台运行。

JVM 在执行应用程序时具有以下 3 个典型特点。

① 多线程:Java 虚拟机采用多线程方式运行,各线程独立执行 Java 代码,并能够协调一致地处理共享数据。

② 动态性:Java 虚拟机在内存管理等功能方面全部采用动态方式,如动态分配数组的存储空间、动态创建对象、动态连接数据库等,这些动态特性使 Java 程序适合在网络上运行。

③ 异常处理机制:Java 虚拟机提供可靠的异常处理机制。

1.3.3 垃圾回收机制

在 Java 语言中,所有属性和方法都被封装在类中。当要创建类的对象时,这种动态的对象被分配到内存堆中。为了实现内存资源的合理使用,Java 有一个系统级的线程,定期跟踪内存的使用情况。此线程在虚拟机的空闲周期检测内存,对内存中已经死亡或者长时间没有使用的对象进行清除和回收。当然,Java 虚拟机在一定条件下才执行内存收集。例如可用内存较少、CPU 占用率较低,或者其他的策略。因此,编程人员无须考虑复杂的内存管理。

1.3.4 代码安全检测

字节码的执行要经过 3 个步骤：首先，类加载器(ClassLoader)负责把类文件加载到 Java 虚拟机中。此过程会检验该类文件是否符合类文件规范；其次，字节码校验器(Bytecode Validator)检查该类文件的代码中是否存在某些非法操作；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再执行。

Java 虚拟机在上面的操作中使用了“沙漏”模型，即把 Java 程序的代码和数据都限制在一定的内存空间执行，不允许程序访问该内存之外的内存空间。这种访问过程不仅仅是本地的，也可以是远程的。如果是 Applet 程序，还不允许访问客户端机器的文件系统。

1.4 Java 的 3 种版本

Sun 公司要实现“Write once, Run anywhere”的目标，就必须提供相应的 Java 运行平台。目前，Java 运行平台主要分为标准版(Java SE)、企业版(Java EE)和微缩版(Java ME)3 个版本。

① Java SE——Java 标准平台。Java SE 提供了标准的 JDK 开发平台，利用该平台可以开发 Java 桌面应用程序和低端的服务器应用程序，也可以开发 Java Applet。

② Java EE——Java 企业平台。可以构建企业级的服务器应用。Java EE 平台包括了 Java SE 平台，并增加了一些扩展类库，以便支持目录管理、交易管理和企业级消息处理等功能。

③ Java ME——Java 小型平台。Java ME 是一种小巧的 Java 运行时环境，包含了 Java SE 中一部分类库，主要用于嵌入式的消费产品中，如呼机、智能卡、手机、PDA、机顶盒、移动电话、掌上电脑或其他无线设备等。

这 3 个版本之间的关系如图 1.2 所示。

Java EE 几乎包含了 Java SE 的功能，然后在 Java SE 的基础上添加了很多新的功能。

Java ME 主要是 Java 的功能子集，加上了一部分额外添加的功能。

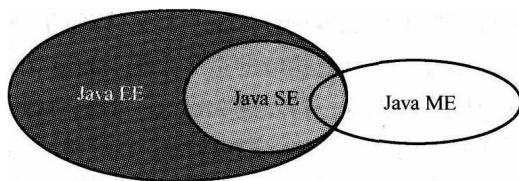


图 1.2 Java 3 个版本之间的关系

1.5 Java 开发环境概述

1.5.1 JDK 的安装、配置和使用

1. 安装 JDK

Sun 公司提供了一套免费的开发、运行时环境，可从以下站点下载 Java2 SDK(Java 2 Standard Develop Kit)，即 <http://java.sun.com/j2se/> 处可获得最新版本的 JDK(Java Develop Kit)。安装的时候可以选择安装到任意的硬盘驱动器上，例如，安装到 C:\Program Files\Java\jdk1.6 目录下。通常在 JDK 目录下有 bin、jre、lib 等子目录，如图 1.3 所示。

其中 JDK 主目录和 jre 子目录等主要目录的内容如下。

- bin 子目录：JDK 开发工具集中用于开发工具的可执行文件。系统的 path 环境变量应该包含这个目录项。
- lib 子目录：开发工具所使用的文件。其中，tools.jar 包含了在 JDK 中工具和实用工具支持的非核心类库。还有 dt.jar，这是 BeanInfo 文件设计时的压缩包。

- jre 子目录: JDK 开发工具所使用的 Java 运行时环境的根目录。这个运行环境是一个 Java 平台的实现, 这个目录使用 java.home 系统属性来表示。
- jre\lib 子目录: Java 运行时环境所使用的核心类库、属性设置和资源文件。例如, 自举类 rt.jar, 包含了 Java 平台的核心 API。
- jre\lib\ext 子目录: 用于扩展 Java 平台默认的安装目录。

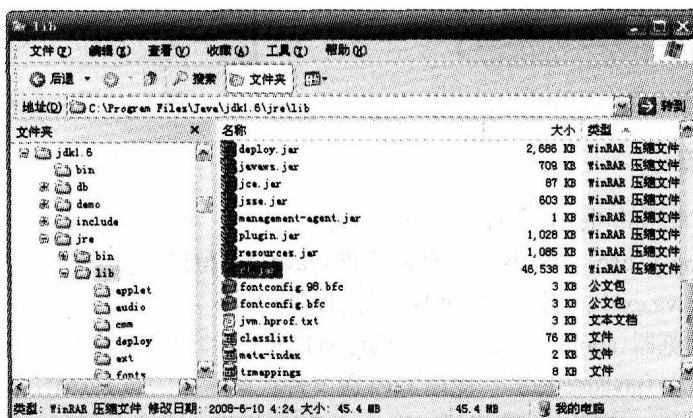


图 1.3 JDK 安装目录结构

2. 配置环境变量

在安装完 JDK 之后, 必须配置类路径 classpath 和环境变量 path, JDK 才能够正常运行。对于 Windows 2000/2003/XP, 右击桌面上“我的电脑”, 图标, 在弹出的快捷菜单中选择“属性”命令, 则弹出“系统属性”对话框, 如图 1.4 所示, 选择“高级”选项卡, 然后单击“环境变量”按钮, 在“环境变量”窗口中编辑 classpath 和 path。

在图 1.5 中, 添加一个系统环境变量 classpath, 其值为:

classpath=.;%java_home%\lib\dt.jar;%java_home%\lib\tools.jar

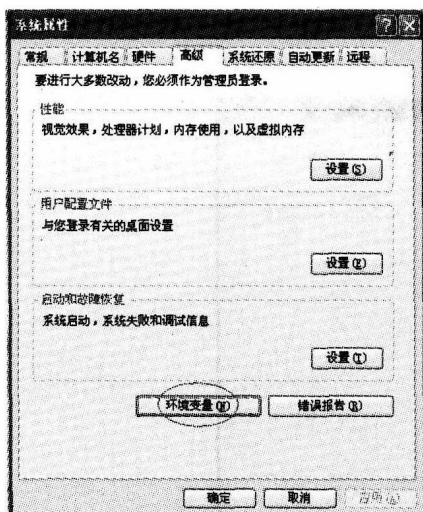


图 1.4 配置系统属性

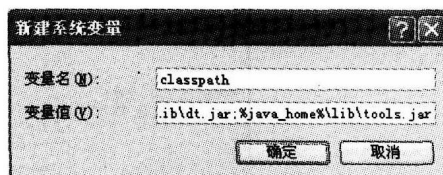


图 1.5 设置系统变量 classpath

在图 1.6 中, 添加一个系统环境变量 java_home, 其值为:

java_home= C:\Program Files\Java\jdk1.6

在图1.7中,向已经存在的系统环境变量 path 新增加一项,其值为:

%path%;c:\jdk1.6\bin

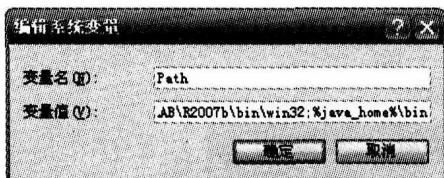


图 1.6 设置系统变量 java_home

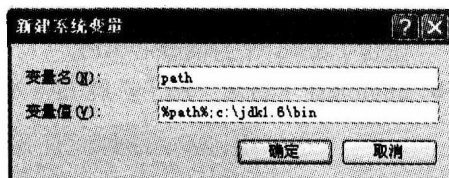


图 1.7 设置系统变量 path

1.5.2 JDK 工具箱

以下将对 JDK 中常用的可执行程序进行简要介绍。

- (1) Java 编译器(javac.exe): 将 Java 源代码文件(.java)编译成字节码文件(.class)。
- (2) Java 解释器(java.exe): 用来直接解释执行 java 字节码。
- (3) Java Applet 浏览器(appletviewer.exe): 执行 HTML 文件中内嵌的 Java 小程序的 Java 浏览器。
- (4) Java 文档生成器(javadoc.exe): 使用 Java 源文件代码生成一组 HTML 格式的相关文件。
- (5) Java 调试器(jdb.exe)是一个除错器,为 Java 程序提供一个命令行调试环境,便于对 Java 解释的类进行除错。
- (6) Java 反汇编器(javap.exe): 反汇编一个 Java 字节代码文件,返回类的相关成员。

1.5.3 Eclipse 开发环境的安装、配置和使用

Eclipse 是一个开放的,集 Java 程序编辑、编译和运行于一体,功能强大的 Java IDE 开发环境。用户可以从官方网站 <http://www.Eclipse.org> 上下载最新版本 Eclipse 3.6。为了下载和安装上的便利,本文以集成 myEclipse+Eclipse 的软件包 MyEclipse_6.0.1GA_Eclipse3.3.1_Full.exe 作为 IDE 开发环境,压缩包的大小是 250MB。其中,myEclipse 作为 Eclipse 的插件实现了与 Eclipse 的无缝连接,编程人员可以直接编写 Java 程序。此外,用户也可以直接下载 eclipse-SDK-3.3.1-win32.zip 进行安装。

安装好之后,从“开始”菜单找到程序项 MyEclipse 6.0,单击应用程序项 MyEclipse 6.0.1 即可进入 MyEclipse 主界面,如图1.8所示。

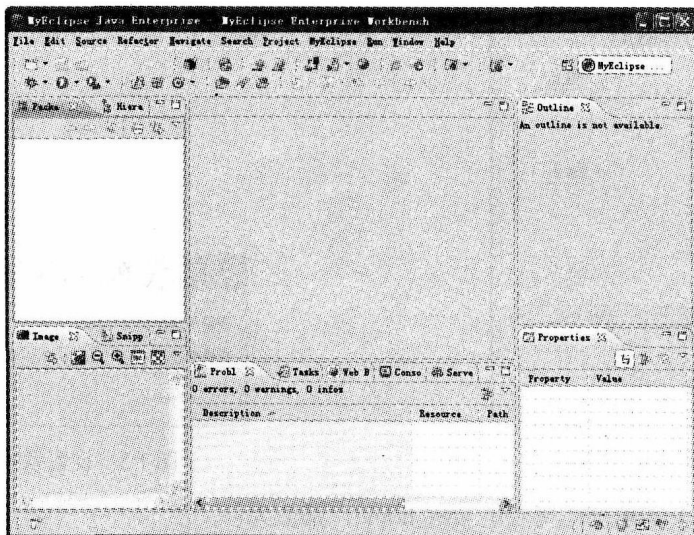


图 1.8 MyEclipse 主界面