希望
Beijing Hope Electronic Press
www.bhp.com.cn

IBM教育学院教育培养计划指定教材
英特尔软件学院教育培养计划指定教材

Java 软件开发

Java
Software
Development

张义 等 编著

科学出版社
www.sciencep.com

IBM教育学院教育培养计划指定教材
英特尔软件学院教育培养计划指定教材

Java 软件开发

Java
Software
Development

张义 等 编著

科学出版社
www.sciencep.com

·北京·

内 容 简 介

本书探讨运用 Java 进行应用程序开发，详尽地讲解了当前流行的应用程序开发工具——Java 语言的核心技术。

全书共分 23 章。内容主要包括 4 个部分，分别介绍了 Java 语言及其面向对象特性，Java 基础应用程序开发，Java 高级应用及网络应用开发，以及跨平台应用程序开发及 Eclipse 开发工具的使用等内容。

本书是广大 Java 软件设计、嵌入式及网络应用开发行业程序员的必备工具书，亦可作为高校、社会培训班教师教材。由于本书的专业性、实用性与易读性，现已被选为“IBM 教育学院”、“英特尔软件学院”教育培养计划指定教材。

需要本书或技术支持的读者，请与北京清河 6 号信箱（邮编：100085）发行部联系，电话：010-62978181（总机）转发行部、010-82702675（邮购），传真：010-82702698，E-mail：tbd@bhp.com.cn。

图书在版编目（CIP）数据

Java 软件开发 / 张义等编著. —北京：科学出版社，2009.

IBM 教育学院教育培养计划指定教材. 英特尔软件学院教育培养计划指定教材

ISBN 978-7-03-025495-5

Ⅰ. J… Ⅱ. ①张… Ⅲ. Java 语言—程序设计—职业教育—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2009）第 157917 号

责任编辑：刘 芯 / 责任校对：周 玉

责任印刷：密 东 / 封面设计：青青果园

科学出版社 出版

北京东黄城根北街 16 号

邮政编码：100717

http://www.sciencep.com

北京市密东印刷有限公司

科学出版社发行 各地新华书店经销

*

2009 年 11 月第 1 版 开本：787mm×1092mm 1/16

2009 年 11 月第 1 次印刷 印张：23.25

印数：1-2 000 册 字数：535 千字

定价：39.00 元

“IBM 教育学院教育培养计划指定教材”
“英特尔软件学院教育培养计划指定教材”
编 委 会

IBM 教育学院认证体系

IBM 教育学院认证体系是顺应 IT 认证市场规律，推陈出新的一项 IT 专业技能认证，拥有 IBM 教育学院认证资格的专业人士具有相应认证实际工作的基本能力和基本技能。

IBM 教育学院认证体系包括电子商务、Java 软件开发、软件测试、数据库管理、数据分析及数据建模、网络管理、IT 销售、IT 技术支持方向。

一、认证体系概述

IBM 教育学院认证体系提供了 8 个认证方向，它们所代表的专业水平相当于 IBM 认证电子商务师、IBM 认证软件开发员、IBM 认证数据库管理员、IBM 认证网络管理员、IBM 认证软件测试员、IBM 认证数据建模员、IBM 认证销售工程师、IBM 认证技术支持工程师。

- **IBM 认证电子商务师：**了解电子商务和信息技术的基础知识，掌握本专业知识的体系结构和整体概貌。主要内容包括：电子商务的基本概念和原理，电子商务的现状和发展，电子商务的特点、电子商务的类型、电子商务模型、计算机技术、程序设计、操作系统、编译系统、数据库系统、通信技术、网络技术、Internet、EDI 技术、电子支付技术、安全等技术的概述，电子商务系统的构成及其开发工具、电子商务整体解决方案与案例介绍。可从事网上信息交换与业务交流、网络营销、电子订单处理、网上采购、网页制作、网站后台管理等工作。
- **IBM 认证 Java 软件开发员：**具有 Java 软件开发的基本能力、掌握 Java 核心技术概念、掌握 Java 编码规则、掌握 JDBC 操作基础、熟悉理解 Java 远程方法调用、掌握 Java 网络编程基础。可从事中低端软件开发类工作。
- **IBM 认证数据库管理员：**具有数据库开发及管理的基本技能，熟悉并理解数据基本对象概念及操作，掌握数据库设计的基本原则。可从事数据库开发、数据库维护及管理工作。
- **IBM 认证网络管理员：**掌握负责规划、监督、控制网络资源的使用和网络的各种活动，以使网络的性能达到最优的技能。可以从事计算机网络运行、维护类工作。
- **IBM 认证软件测试员：**掌握软件测试的基本技能、熟悉并理解软件测试基本概念和测试的必要性、熟悉掌握测试用例的做成、熟悉掌握相关测试工具的使用。可从事软件测试类工作。
- **IBM 认证数据建模员：**掌握需求开发与需求管理的理念，建立正确的需求观，掌握需求工程总体框架；需求开发和需求管理的方法与使用原则；需求的业务需求、用户需求和功能需求三个层次之间的关系、作用、权利与责任；需求获取、分析、编写和确认的方法与手段；需求原型的管理和实现；建模技术和需求规格说明书的编写方法；变更控制、版本控制、需求状态跟踪和需求跟踪的技术和方法。可从事数据库需求分析、架构分析及数据库模型设计工作。
- **IBM 认证销售工程师：**掌握基本的销售技巧，提高学员对市场的敏感性和对市场的观察与分析力，具有独立管理和策划商品销售的能力。可从事和 IT 相关的各类销售工作。

- **技术支持工程师**：掌握 IT 技术，为企业计算机办公提供完整的解决方案和维护策略，并具有对新技术的敏感触觉，及时把握技术发展。可从事和 IT 相关的各类技术服务工作。

二、IBM 教育学院认证和途径

IBM 教育学院认证面向的是各大院校学生与 IBM 教育学院授权培训中心学员。要获得职业认证体系中的不同认证证书，都必须通过认证考试。

注：IBM 教育学院的技术认证，不需要考生预先具有任何认证证书，只需要通过相应的专项技术考试即可。

IBM 教育学院

前　言

Java 的升级从来都不是对旧版本的彻底淘汰，而是在原有基础上进行“扬弃”的改进，并保持对旧版的兼容，让用户可以迅速适应新的特性。

自 1995 年以来，Java 的芯片技术、数据库连接、Jini（基于 Java 的信息家电联网）和 Enterprise JavaBeans 都已经逐渐和现代生活息息相关。它的安全、跨平台和高性能都是其他语言所无法比拟的，相信将来 Java 还会给我们带来越来越多的惊喜，在 Java 的推动下，我们的生活会更加便捷，更加丰富多彩！

本书由浅入深地对 Java 编程规则及其应用进行了详细的讲解，全书由如下几个部分组成。

（1）Java 及其面向对象特性。第 1 章介绍了什么是 Java 和 Java 的由来；第 2 章介绍 Java 的基本语法知识；第 3 章介绍面向对象思想——继承、封装和多态；第 4 章介绍 Java 面向对象的设计；第 5 章介绍 Java 中类的高级特性。

（2）Java 标准应用。第 6 章介绍 Java 中的数据结构，首先使用伪码讲解，最后给出了 Java 的实现；第 7 章介绍了 Java 的异常处理机制；第 8 章介绍了输入输出系统；第 9 章介绍了小应用程序——Applet；第 10 章介绍了多线程机制；第 11 章介绍了基本的图形用户界面；第 12 章介绍了 AWT 的组件和事件处理机制；第 13 章介绍了 Swing 图形用户界面。

（3）Java 高级应用。第 14 章介绍了网络编程原理；第 15 章介绍了 JDBC 编程；第 16 章介绍了服务器小应用程序——servlet；第 17 章介绍了 Struts 和 Hibernate；第 18 章简单地介绍了 J2EE 的概念；第 19 章简单地介绍了 J2ME 的概念。

（4）其他。第 20 章介绍了 Java 的跨平台特性；第 21 章介绍了泛型程序设计；第 22 章介绍了 Java 编程规范。第 23 章介绍了使用 Eclipse 开发工具进行 Java 开发的详细步骤。

本书可供刚刚接触 Java 的读者作为 Java 的初级教材（详细阅读前 13 章），而对具有一定 Java 编程经验的程序员来说，如果要提高自己的编程水平，在 Java 的道路上走得更远，也可以参阅本书。

在写本书与改版过程中，北航软件学院杨晴红老师和计算机学院的朱成军博士付出了大量的心血，史惠卿、邢燕鹏、张义、柳丽、韩晓东、胡昌军、吴永辉、王翌、袁涛、初平、刘晶、白晓峰、黄雷、杨至超、刘鹏冲、康怡暖、高艳铭、张红艳、李宗花、戴朝晖、林桢、梁海涛、左晓宁等同志为本书提供了大量的例程和帮助，另外来自清华大学、北京大学、哈尔滨工程大学的三位博士孙红芳、邵金燕和彭倞也对本书提出了许多宝贵的意见，在此一并表示感谢。

由于时间匆忙，而且编者水平有限，书中的疏漏和不足之处，还望读者不吝指正。关于本书的技术问题，请 E-mail 至 henshui228@hotmail.com，在此深深感谢。

编　者

目　录

第 1 章　Java 及 Java 开发环境概述 1
1.1　Java 的诞生及其影响 1
1.2　Java 的特征 1
1.2.1　简单 2
1.2.2　面向对象 2
1.2.3　分布式 2
1.2.4　健壮 2
1.2.5　体系结构中立 3
1.2.6　可移植 3
1.2.7　解释执行 3
1.2.8　高性能 3
1.2.9　多线程 4
1.2.10　动态 4
1.3　Java 5.0 新特性 4
1.4　安装 Java 开发工具 5
1.4.1　JDK 的取得 5
1.4.2　安装并测试 5
1.5　JDK 开发工具包 9
1.5.1　Javac 9
1.5.2　Java 10
1.5.3　Javadoc 10
1.5.4　jar 13
1.5.5　Javah 14
1.5.6　Javap 15
1.5.7　appletviewer 15
1.5.8　jdb 16
1.5.9　native2ascii 17
1.5.10　extcheck 17
1.6　Java 集成开发环境简介 18
1.6.1　Eclipse 发展背景 18
1.6.2　Eclipse 工作台简介 18
第 2 章　Java 语言基础 23
2.1　Java 关键字和标识符 23
2.1.1　标识符 23
2.1.2　关键字 23
2.2　Java 数据类型、常量和变量 24
2.2.1　Java 数据类型 24
2.2.2　常量 24
2.2.3　变量 25
2.3　简单数据类型 26
2.3.1　整数类型 26
2.3.2　浮点类型 27
2.3.3　字符类型 28
2.3.4　布尔类型 29
2.3.5　枚举类型 29
2.3.6　综合举例 29
2.3.7　自动类型转换与强制类型转换 30
2.4　Java 运算符及表达式 31
2.4.1　Java 运算符简介 31
2.4.2　算术运算符 32
2.4.3　关系运算符 33
2.4.4　布尔逻辑运算符 33
2.4.5　按位运算符 34
2.4.6　赋值运算符 35
2.4.7　条件运算符 36
2.4.8　表达式及运算符优先级 36
2.5　数组 37
2.6　字符串 39
2.6.1　字符串的初始化 39
2.6.2　String 和 StringBuffer 类 40
2.6.3　StringBuilder 类 40
2.6.4　字符串的访问 40
2.6.5　修改字符串 41
2.7　Java 流程控制 41
2.7.1　条件语句 41
2.7.2　循环语句 42
2.7.3　转移语句 43
第 3 章　面向对象思想 45
3.1　结构化程序设计的方法 45
3.2　面向对象的编程思想 45
3.2.1　什么是对象 45
3.2.2　什么是面向对象 46

3.2.3 什么是类 46
3.2.4 学会抽象整个世界——实体、对象和类 46
3.2.5 面向对象方法——抽象的进步 47
3.3 面向对象的特点 48
3.3.1 继承 48
3.3.2 封装 49
3.3.3 多态性 49
第 4 章 Java 面向对象设计 50
4.1 类和类的实例化 50
4.1.1 定义类的结构 50
4.1.2 类的实例化 52
4.2 Java 内存使用机制 57
4.3 抽象类和接口 59
4.3.1 抽象类 59
4.3.2 接口 59
4.4 命名空间与包 61
4.4.1 包的基本概念 61
4.4.2 自定义一个包 61
4.4.3 源文件与类文件的管理 62
4.5 现有类的使用 62
4.5.1 访问权限 62
4.5.2 使用 import 导入已有类 64
4.5.3 静态导入 65
4.5.4 类的继承和多态 65
第 5 章 类的高级特性 68
5.1 静态变量和方法 68
5.1.1 静态变量 68
5.1.2 静态方法 69
5.2 常量、最终方法和最终类 70
5.2.1 常量 70
5.2.2 最终方法 70
5.2.3 最终类 70
5.3 抽象类和抽象方法的使用 70
5.4 接口的使用 71
5.4.1 接口的概念 71
5.4.2 定义接口 72
5.4.3 执行接口 73
5.4.4 使用接口 73
5.5 内部类的使用 74
5.5.1 使用内部类的共同方法 74
5.5.2 内部类 75
5.5.3 内部类属性 76
第 6 章 数据结构 77
6.1 抽象数据类型 77
6.2 基本数据结构 77
6.2.1 向量 77
6.2.2 线性表 78
6.2.3 堆栈 86
6.2.4 队列 89
6.2.5 树 90
6.2.6 图 93
第 7 章 Java 异常处理 96
7.1 异常机制简述 96
7.1.1 异常的概念 96
7.1.2 异常的分类 97
7.2 Java 异常体系 98
7.2.1 捕获异常 98
7.2.2 声明异常 100
7.2.3 抛出异常 100
7.2.4 自定义异常 101
第 8 章 Java 输入/输出系统 103
8.1 Java 输入/输出体系 103
8.2 字节流 105
8.2.1 InputStream 类 105
8.2.2 OutputStream 类 106
8.2.3 FileInputStream 类 106
8.2.4 FileOutputStream 类 108
8.2.5 ByteArrayInputStream 类 109
8.2.6 ByteArrayOutputStream 类 110
8.2.7 管道流 PipedInputStream 和 PipedOutputStream 类 111
8.2.8 过滤流 FilterInputStream 和 FilterOutputStream 类 112
8.3 字符流 112
8.3.1 Reader 类 113
8.3.2 Writer 类 113
8.3.3 FileReader 类 114
8.3.4 FileWriter 类 115
8.3.5 CharArrayReader 类 116
8.3.6 CharArrayWriter 类 116
8.3.7 PushbackReader 类 117

8.4 文件的读写操作 119
8.5 对象序列化及其恢复 122
8.5.1 Serializable 接口 122
8.5.2 ObjectOutputStream 类 122
8.5.3 ObjectInputStream 类 123
第 9 章 创建 Java Applet 126
9.1 Applet 类 126
9.2 Applet 概述 126
9.3 Applet 的使用技巧 127
9.3.1 波浪形文字 127
9.3.2 大小变化的文字 129
9.3.3 星空动画 137
9.3.4 时钟 141
第 10 章 多线程 143
10.1 多线程的概念 143
10.1.1 多线程 143
10.1.2 Java 中的多线程 144
10.1.3 线程组 144
10.2 线程的创建 144
10.2.1 通过实现 Runnable 接口创建线程 145
10.2.2 通过继承 Thread 类创建线程 145
10.2.3 两种线程创建方法的比较 146
10.3 线程的调度与控制 146
10.3.1 线程的调度与优先级 146
10.3.2 线程的控制 147
10.4 线程的状态与生命周期 148
10.5 线程的同步 149
10.6 线程的通信 151
10.7 线程池 154
第 11 章 图形用户界面 158
11.1 AWT 及其根组件 158
11.1.1 java.awt 包 158
11.1.2 根组件（Component） 158
11.2 容器（Container）和组件 159
11.3 布局管理器（Layout Manager） 160
11.3.1 FlowLayout 布局管理器 161
11.3.2 BorderLayout 布局管理器 161
11.3.3 GridLayout 布局管理器 164
11.3.4 CardLayout 布局管理器 165
11.3.5 GridBagLayout 布局管理器 167
11.3.6 null 布局管理器 169
第 12 章 AWT 基本组件及事件处理机制 170
12.1 AWT 基本组件 170
12.1.1 Component 类 170
12.1.2 AWT 事件模型 173
12.2 GUI 事件的处理 174
12.2.1 AWT 事件继承层次 174
12.2.2 AWTEvent 子类事件 175
12.2.3 监听器接口 176
12.3 几个简单组件 180
12.3.1 按钮（Button 类） 180
12.3.2 标签（Label 类） 180
12.3.3 文本组件（TextField 和 TextArea 类） 180
12.4 使用类适配器（Adapter）进行事件处理 181
12.5 使用匿名类进行事件处理 184
第 13 章 Swing 用户界面组件 186
13.1 Swing 组件库简介 186
13.1.1 JFC 和 Swing 186
13.1.2 Swing 包概览 187
13.1.3 Swing 和 AWT 的区别 187
13.1.4 示例程序 SwingApplication 188
13.2 Swing 组件及其容器 192
13.2.1 JComponent 类 192
13.2.2 AbstractButton 及其子类 195
13.3 JComboBox 和 JList 组件 206
13.4 JSlider 类——滑杆 213
13.5 JInternalFrame 类 215
第 14 章 网络通信程序设计 217
14.1 java.net 包 217
14.2 socket 编程 217
14.2.1 socket 基础知识 217
14.2.2 socket 机制分析 218
14.2.3 客户端编程 220
14.2.4 服务器端编程 222
14.2.5 服务器/客户端通信实例 223
14.2.6 Datagram Sockets 编程 226
第 15 章 Java 数据库访问机制——JDBC 232
15.1 JDBC 介绍 232
15.1.1 JDBC 的概述 232

15.1.2 JDBC——底层 API 232
15.1.3 JDBC 的设计过程 233
15.1.4 JDBC 和 ODBC 的比较 233
15.2 关系数据库和 SQL 234
15.2.1 关系数据库 234
15.2.2 关系数据库的应用模型 235
15.2.3 结构化查询语言 236
15.3 JDBC 应用程序编程接口 239
15.3.1 JDBC 的类 239
15.3.2 DriverManager 239
15.3.3 JDBC 驱动程序的类型 240
15.4 JDBC 编程基础 241
15.4.1 JDBC 访问数据库 241
15.4.2 创建一个数据源 241
15.4.3 数据库 URL 243
15.4.4 建立与数据源的连接 244
15.4.5 发送 SQL 语句 245
15.4.6 处理查询结果 245
15.5 基本 JDBC 应用程序 246
15.5.1 JDBC 在应用程序中的应用 246
15.5.2 JDBC 在 Applet 中的使用 247
15.6 JDBC API 的主要界面 250
15.6.1 Statement 250
15.6.2 ResultSet 251
15.6.3 PreparedStatement 252
15.6.4 CallableStatement 254
15.7 事务管理 255
15.7.1 保存点 256
15.7.2 批量更新 256
15.8 高级连接管理 257
第 16 章 servlet 259
16.1 servlet 综述 259
16.1.1 什么是 servlet 259
16.1.2 servlet 的生命周期 260
16.1.3 servlet 与其他开发技术的比较 261
16.1.4 servlet 的应用范围 263
16.1.5 配置 servlet 的开发的环境 264
16.2 servlet 编程 265
16.2.1 HTTP 协议介绍 265
16.2.2 简单程序 servlet 267
16.2.3 会话跟踪 270
16.2.4 Servlet 协作 273
第 17 章 Struts 与 Hibernate 入门 275
17.1 MVC 框架 275
17.1.1 MVC 模式 275
17.1.2 基于 Web 应用的 MVC 模式 276
17.2 Struts 结构和处理流程 276
17.3 Struts 组件 277
17.3.1 Web 应用程序的配置 277
17.3.2 控制器 278
17.3.3 struts-config.xml 文件 279
17.3.4 Action 类 279
17.3.5 视图资源 279
17.3.6 ActionForm 279
17.3.7 模型组件 280
17.4 Hibernate 简介 280
17.4.1 第一个 Hibernate 程序 280
17.4.2 关联映射 289
第 18 章 J2EE 基础 295
18.1 J2EE 综述 295
18.1.1 J2EE 的主要特征 295
18.1.2 J2EE 的架构 296
18.1.3 J2EE 应用场景描述 297
18.2 客户端层技术 298
18.2.1 客户端层的问题 298
18.2.2 客户端层的解决方案 299
18.3 Web 层技术 300
18.3.1 Web 层的目的 300
18.3.2 Web 层的解决方案 301
18.4 EJB 层技术 304
18.4.1 EJB 组件结构 304
18.4.2 EJB 层的目的 306
18.4.3 EJB 层的解决方案 306
第 19 章 J2ME 概述 309
19.1 J2ME 综述 309
19.2 CLDC 介绍 310
19.2.1 CLDC 类库介绍 311
19.2.2 MIDLET 介绍 312
19.2.3 MIDlet 界面 313
19.3 CDC 概述 314
第 20 章 Java 跨平台特性 315
20.1 可移植性 315

20.1.1 源代码可移植性 315
20.1.2 CPU 可移植性 315
20.1.3 操作系统可移植性 316
20.2 解决国际化问题 316
20.2.1 Java 类包 317
20.2.2 参数化解决方法 318
20.2.3 处理提示和帮助 318
20.3 编写跨平台 Java 程序的注意事项 320
第 21 章 Java 泛型程序设计 322
21.1 简单泛型类的定义 322
21.2 泛型方法 323
21.3 类型变量的限定 324
21.4 泛型代码和虚拟机 325
21.4.1 翻译泛型表达式 326
21.4.2 翻译泛型方法 327
21.4.3 遗留代码调用 328
21.5 约束与限定 329
21.5.1 基本类型 329
21.5.2 运行时类型查询 329
21.5.3 异常 329
21.5.4 数组 330
21.5.5 泛型类型的实例化 330
21.5.6 静态上下文 331
21.5.7 擦除后的冲突 331
21.6 泛型类型的继承规则 332
21.7 通配符类型 333
21.7.1 通配符的超类型限定 333
21.7.2 无限定通配符 335
21.7.3 通配符抓取 335
21.8 反射和泛型 338
21.8.1 使用 Class<T>参数进行类型匹配 339
21.8.2 虚拟机中的泛型类型信息 339
第 22 章 Java 编码规范 342
22.1 概述 342
22.2 基本原则 342
22.2.1 取个好名字 342
22.2.2 三种 Java 注释 343
22.3 成员方法 343
22.3.1 方法命名 343
22.3.2 注释 344
22.3.3 编写清晰、易懂的代码 345
22.3.4 小技巧 345
22.4 成员变量 346
22.4.1 普通变量的命名 346
22.4.2 窗口组件的命名 346
22.4.3 常量的命名 346
22.4.4 注释 346
22.5 类和接口 346
22.5.1 类和接口的命名 346
22.5.2 注释 347
22.6 Java 源文件范例 347
第 23 章 使用 Eclipse 进行 Java 程序开发 349
23.1 建立 Java 项目 349
23.2 建立 Java 类别 350
23.3 代码功能 350
23.3.1 Code Completion 350
23.3.2 Code Assist 351
23.4 执行 Java 程序 352
23.5 Java 实时运算簿页面（Java Scrapbook Page） 353
23.6 自定义开发环境 355
23.6.1 程序代码格式 355
23.6.2 程序代码产生模板 356
23.6.3 Javadoc 批注 357
附录 A Java 关键字 359
附录 B Java 站点资源 361

第 1 章　Java 及 Java 开发环境概述

同其他成功的计算机语言一样，Java 在继承了其他语言先进特性的基础上，又提出了一些创新性的概念。在本书中，将详细地为读者介绍 Java 各方面的内容。

本章主要介绍什么是 Java 语言，它的产生背景、发展历程与影响，以及它的一些基本特性与 Java 技术的应用。

1.1　Java 的诞生及其影响

1994 年，当 Web 如火如荼发展的时候，随着 Internet 的发展，以网络为中心的计算机普及，客观上需要一种独立于平台、代码的可移动的计算技术，Java 恰恰填补了这一空白，这把 Intel 垄断的软硬件市场打开了一个巨大的缺口，引发了一场软件开发的革命。

Java 连同 Internet、WWW 改变着应用软件的开发和使用方式，一切都围绕着网络，围绕着跨平台进行。Word、Excel 等传统的信息处理工具都必然走向萎缩，因为它们是单机时代的产物。信息的价值在于使用和共享，Internet 和 Web 是信息使用和共享最快捷、最便宜的方式，Word 将演化成为 Web 写作工具，Excel 则将演化成 Web 上的电子表格。

1996 年初 Sun 正式发布了 Java 的第 1 版。但是人们很快意识到 Java1.0 不能用来进行真正的应用开发。尽管 Java1.0 提供了 applet 技术来实现 Web 架构下的信息实时更新，但它却没有提供打印功能。后来的 1.1 版弥补了其中大部分明显的缺陷，大大改进了反应能力。1998 年的 JavaOne 会议发布了 Java1.2，该版本取代了早期过于简单的 GUI，使它的图形工具箱更加精细且具有可伸缩性，更加接近“一次编写，随处运行”的承诺。

除了“标准版”之外，还推出了两个其他的版本：用于进行嵌入式编程 J2ME 与进行服务器端处理的“企业版”。本书对这两个版本都进行了说明，但重点是讲述标准版。

标准版的 1.3 与 1.4 版本对最初的 Java2 进行了某些改进，并扩展了标准类库，提高了系统性能。关于 5.0 版的特性，将在后面进行详细说明。

1.2　Java 的特征

总的来说，Java 具有以下特点：

- 简单（Simple）。
- 面向对象（Object-oriented）。
- 分布式（Distributed）。
- 健壮（Robust）。
- 安全（Secure）。
- 体系结构中立（Architecture-neutral）。
- 可移植（Portable）。

- 解释执行（Interpreted）。
- 高性能（High performance）。
- 多线程（Multi-threaded）。
- 动态（Dynamic）。

提示：Java 白皮书可在地址 Http://java.sun.com/doc/language_environment 中找到。

1.2.1 简单

Java 语言是一种面向对象的语言，它通过提供最基本的方法完成指定的任务，只需理解一些基本的概念，就可以用它编写出适合于各种情况的应用程序。程序小也是简单的一种特性，使得代码能够在小型机器上执行。

Java 删除了许多极少被使用、不容易理解和令人混淆的 C++功能，这些功能在我们的使用经验中只能带来麻烦。删除的功能主要包括运算符重载（operator overloading）、多重继承（inheritance）以及广泛的自动强迫同型（automatic coercions）。重载是指以一个辨识元参照多重项目，Java 语言也提供重载函数，不过它重载的对象是方法（method）而非变量或运算符。甚至可以说，Java 的语法就是 C++的清错版本。

在 Java 中增加了自动内存垃圾收集（auto garbage collection）功能，用于简化 Java 程序工作，但同时也让系统变得稍为复杂了一些。储存管理（storage management）是使 C/C++应用程序变得复杂的常见原因，即关于内存的分配与释放。Java 语言的自动垃圾收集功能（周期性地释放未被使用的内存）不仅简化了程序设计工作，而且能大幅度地减少小错误（bugs）数量。

1.2.2 面向对象

“面向对象”程序设计方法充分地证明了自己的价值。作为新一代的编程语言，Java 继承了这一宝贵特性，并将其做了扩充与发展。

Java 语言的设计集中于对象及其接口，它提供了简单的类机制以及动态的接口模型。对象中封装了它的状态变量以及相应的方法，实现了模块化和信息隐藏；而类则提供了一类对象的原型，并且通过继承机制，子类可以使用父类所提供的方法，实现了代码的复用。

1.2.3 分布式

Java 是面向网络的语言。它能轻易地处理 TCP/IP 协议，这使得在 Java 中比在 C/C++中更容易创建网络连接，用户可以通过 URL 地址在网络上方便地访问其他对象。Java 应用程序可以借由 URL 再通过网络开启和存取对象，就如同存取一个本地（local）文件系统一样简单。

1.2.4 健壮

Java 在编译和运行程序时，都要对可能出现的问题进行检查，以消除错误的产生。它提供自动垃圾收集器进行内存管理，防止程序员在管理内存时容易产生的错误。通过集成的面向对象的例外处理机制，在编译时，Java 提示出可能出现但未被处理的例外，帮助程序员正确地进行选择以防止系统的崩溃。另外，Java 在编译时还可捕获类型声明中的许多常见错误，防止动态运行时不匹配问题的出现。

许多病毒程序常使用的一个技巧就是通过巧妙地运用地址变量(如指针)获取计算机的资源，这就是为什么 Java 语言设计者决定放弃指针的重要原因之一，这也消除了安全方面的一个重要隐患。

类的内存分配和布局由 Java 环境透明地完成。因为程序员不访问内存布局，不能确切地知道内存的实际布局，这也使得病毒很难入侵 Java 程序的内部数据结构。

1.2.5 体系结构中立

Java 解释器生成与体系结构无关的字节码指令，只要安装了 Java 运行时系统，Java 程序就可在任意的处理器上运行。这些字节码指令对应于 Java 虚拟机中的表示，Java 解释器得到字节码后，对它进行转换，使之能够在不同的平台上运行。

1.2.6 可移植

许多类型的计算机和操作系统都是连接到 Internet 上的。要使运行于各种各样的平台上的计算机都能动态下载同一个程序，就需要有能够生成可移植性代码的方法。与平台无关的特性，使 Java 程序可以方便地被移植到网络上的不同机器。同时，Java 的类库中也实现了与不同平台的接口，使这些类库可以移植。另外，Java 编译器是由 Java 语言实现的，Java 运行时系统由标准 C 实现，这使得 Java 系统本身也具有可移植性。

Java 的可移植性取决于其中立的代码结构。代码只需编写一次，字节码可以发布给不同的平台，不需要任何修改就可以运行。其他语言中常见的一个移植问题就是基本数据类型（如整数、字符和浮点数）所占比特数不同。

例如，在 C++中，int 可以是一个 16 位整数，也可以是一个 32 位整数。而在 Java 中的 int，无一例外地都是 32 位的整数，不管它是在 UNIX、OS/2、Macintosh 或者 Windows 上运行。使用标准的整数定义可以避免类似上溢或下溢之类的错误，而这些错误对于基本数据类型大小（假定不一致时）是经常出现的。

1.2.7 解释执行

Java 编译器并不生成可执行程序的机器语言指令。相反，它生成一种中间代码，成为字节码。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时信息，使得连接过程更加简单。再者，由于其链结比较倾向于逐步增量与减量过程，因此发展程序更快、更精密。由于编译期间的信息属于字节代码资料流的一部分，因此可以在运行期间携带更多的信息。这正是链结器类型检查的基础，它也让程序更容易执行除错。Java 解释器和这种抽象机模型就称为 Java 虚拟机（Java Virtual Machine，JVM）。

1.2.8 高性能

和其他解释执行的语言（如 BASIC、TCL）不同，Java 字节码的设计使之能够容易地直接转换成对应于特定 CPU 的机器码，从而得到较高的性能。虽然解释过的字节代码性能已相当不错，不过有些情形下还是要求程序达到更高执行效能。字节代码可以动态地（runtime）为执行应用程序的特定 CPU 解释成机器码。这对习惯使用一般编译器与动态载入器（loader）的设计者而言，有点类似于将最终的机器码产生器放到动态载入器之内。字节代码格式在设计上顾及了机

器码的产生，因此实际的机器码产生程序相当简单。产出的机器码是有效的，编译器自动分配寄存器，而在产出字节代码时也会进行一些优化。

然而，Java 仍比 C 语言慢 20 倍，但对大多数交互式应用程序来说，Java 的速度已经足够了。如果想要 Java 的速度与 C 相当，必须使用 JIT（Just In Time）编译器。翻译为机器指令的字节码后，其性能与 C/C++程序就相差无几了。

1.2.9 多线程

多线程（multithreading）是一种应用程序设计方法。不幸的是，要设计一个一次同时处理许多事件的程序，比设计传统的单一线程的 C/C++程序要复杂得多。Java 是仅有的几种语言本身就以多线程的形式支持多任务的语言之一（另外有 Ada 语言）。多线程机制使应用程序能够并行执行，而且同步机制保证了对共享数据的正确操作。通过使用多线程，程序设计者可以分别用不同的线程完成特定的行为，而不需要采用全局的事件循环机制，这样很容易实现网络上的实时交互行为。

1.2.10 动态

Java 是一种动态的语言，就是说它在需要时才加载相应的类。程序运行时，通过检查与类相关的运行类型信息，可以确定一个对象属于哪个类。Java 的这一特性使它适合于一个不断发展的环境。在类库中，可以自由地加入新的方法和实例变量而不会影响用户程序的执行。并且 Java 通过接口支持多重继承，使之比严格的类继承具有更灵活的方式和扩展性。此外，Java 的这一特点对于小应用程序的健壮性也十分重要，因为在实时系统中，字节码内的小段程序可以动态地被更新。

1.3 Java 5.0 新特性

5.0 版是自 1.1 版以来第一个对 Java 语言作出重大改进的版本（这一版本最初被命名为 1.5 版，在 2004 年 JavaOne 会议之后，版本数字升至 5.0）。经历多年的研究，这个版本添加了泛型类型（generic type，类似于 C++模板），其挑战性在于，添加这一特性无需对虚拟机做任何修改。

另外，还有几个来源于 C#的很有用的语言特性：for each 循环、自动打包和元数据。语言的修改总会引起兼容性问题，然而由于这几个新特性的诱人优点，程序员应该很快会接受这些变化。表 1-1 展示了 5.0 版本相对于以前版本的新增特性。

表 1-1 Java 语言发展过程

版 本	语言新特性	类和接口数目
1.0	语言本身	211
1.1	内部类	477
1.2	无	1524
1.3	无	1840
1.4	断言	2723
1.5	泛型类、for each 循环、可变元参数、自动打包、元数据、枚举、静态导入	3270

1.4 安装 Java 开发工具

如果你使用的是一台 PC 机的话，那么最好安装 Windows 98/2000/NT/XP 操作系统，因为本书所介绍的内容也是在这些环境下运行的。在不同的操作系统下，Java 开发环境的安装也有区别。

另外，在实际应用中，推荐的最低系统配置是 Intel Pentium 处理器，至少 128Mbyte 内存以及至少 50Mbyte 的剩余硬盘空间。目前最常用的 Java 开发包是 JDK（Java Development Kit，Java 开发包）5.0，本书介绍的实例开发都是基于这个版本。

1.4.1 JDK 的取得

如果你的系统已经安装了最新版本的 Java 开发环境，可跳过这一部分。

现在常用的 Java 开发环境是 JDK5.0 版本，并且一般应用基于 Java2 SDK 的标准版（J2SE），所以只需下载 J2SE 的版本即可。可以在 Sun 公司的网站上（http://java.sun.com）找到。

1.4.2 安装并测试

下面让我们测试一下安装后的运行环境，同时也可以让大家了解一下 Java 程序。先来看一个最简单的例子：

```
public class HelloWorldApp {
 public static void main (String args[]) {
  System.out.println("Hello World!");
 }
}
```

本程序执行后输出下面一行信息：

```
Hello World!
```

在程序中，首先用保留字 class 来声明一个新的类，其类名为 HelloWorldApp，它是一个公共类（public）。整个类定义由大括号{}括起来。

在该类中定义了一个 main()方法，其中 public 表示访问权限，指明所有的类都可以使用这一方法；static 指明该方法是一个类方法，它可以通过类名直接调用；void 则指明 main()方法不返回任何值。

对于一个应用程序来说，main()方法是必需的，而且必须按照如上格式定义。Java 解释器在没有生成任何实例的情况下，以 main()作为入口执行程序。

在 Java 程序中可以定义多个类，每个类中又可以定义多个方法，但是最多只能有一个公共类，main()方法也只能有一个，作为程序的入口。main()方法定义中，括号()中的 Stringargs[]是传递给 main()方法的参数，参数名为 args，它是类 String 的一个实例，参数可以为 0 个或多个，每个参数用“类名参数名”指定，多个参数间用逗号分隔。

main()方法的实现（大括号中），只有一条语句：

```
System.out.println ("Hello World!");
```

它用来实现字符串的输出，这条语句与 C 语言中的 printf 语句和 C++中 cout<<语句具有相同的功能。现在让我们运行一下该程序。首先把它放到一个名为 HelloWorldApp.Java 的文件中。

注意： 文件名应和类名相同，因为 Java 解释器要求公共类必须放在与其同名的文件中。