

· 省级精品课程配套教材 ·

Java

程序设计

青
讲

许焕新 丁宏伟 ◎ 主 编
刘丽华 李 丹 ◎ 副主编



讲解透彻，注重编程思想的论述



内容丰富，近200个实战案例



免费提供配套教学资源

清华大学出版社



Java 程序设计精讲

许焕新 丁宏伟 主编

刘丽华 李 丹 副主编

清华大学出版社

北 京

内 容 简 介

本书涵盖了 Java 平台标准版 (Java SE 6.0) 的全部基础知识和高级特性, 内容由浅入深, 主要包括 Java 程序设计基础、面向对象程序设计、数组、异常处理、断言、集合类、GUI 设计与事件处理、I/O 流、泛型、注解、多线程编程机制、Socket 网络编程、JDBC 数据库编程等。

本书结构严谨, 层次清晰, 语言生动, 对概念的论述精准而深刻, 并安排有大量实用的程序实例, 可以帮助读者迅速掌握使用 Java 语言进行桌面应用程序的开发。

本书可作为高等院校、应用型本科、高职高专计算机等相关专业的教材, 也可以作为在职人员的培训和学习参考书。

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 程序设计精讲/许焕新, 丁宏伟主编. —北京: 清华大学出版社, 2010.9

ISBN 978-7-302-23695-5

I. ①J… II. ①许… ②丁… III. ①Java 语言-程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2010) 第 165154 号

责任编辑: 贾小红

封面设计: 刘 超

版式设计: 王世情

责任校对: 王 云

责任印制: 杨 艳

出版发行: 清华大学出版社

地 址: 北京清华大学学研大厦 A 座

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

邮 编: 100084

社 总 机: 010-62770175

邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

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

印 刷 者: 北京密云胶印厂

装 订 者: 三河市新茂装订有限公司

经 销: 全国新华书店

开 本: 185×260 印 张: 25.25 字 数: 580 千字

版 次: 2010 年 9 月第 1 版 印 次: 2010 年 9 月第 1 次印刷

印 数: 1~4000

定 价: 39.80 元

产品编号: 038487-01

推 荐 序

Java 是由 Sun 公司开发的新一代编程语言。使用它可以自由地在不同机器、不同操作平台的网络环境中开发软件。

Java 语言广受欢迎，影响着一代又一代程序员。在学习 Java 的过程中，书籍是我们获取知识的第一来源。尽管 Java 技术已趋于成熟，然而市场上适合初学者学习的 Java 书籍仍然是凤毛麟角。因此，当我听说许焕新老师编写了一本与 Java 基础相关的教程时，我非常惊喜，但并无意外。他是乐知学院的一名教师，不仅拥有丰富的一线开发经验，而且熟知高校的课程体系，能把企业的开发实战经验融于教学之中。他的这本《Java 程序设计精讲》语言生动形象，很多晦涩难懂的概念都用比喻的手法进行了形象化讲解，让人易于理解并且记忆深刻，使初学者能充分感受到 Java 语言的魅力，领会到使用 Java 进行编程的快乐。相信大家选用它作为入门教程，能够在学习过程中少走很多弯路。

我期待能有更多具有实际开发经验的优秀高校教师来编写相关的教材，为 IT 教育的发展注入活力。

CSDN 教育部教学总监 徐明华

前 言

Java 语言自 1995 年 5 月由 Sun 公司发布以来,以极为迅猛的势头发展至今,现已不仅是一门高级编程语言,也是一种完备的技术体系和开发平台。Java 拥有卓越的技术特性、丰富的编程接口(类库)和多款功能强大的开发工具平台,在企业级应用开发领域中占有过半的市场份额,目前超过 25 亿台设备正在使用 Java 技术并获得所有主流 IT 厂商的大力支持。从目前的应用现状和发展前景来看,Java 已经成为软件开发从业人员的首选技术。

本书从最基础的内容开始,详细讲述了使用 Java 技术进行应用程序开发的方法。全书共分 17 章,内容由浅入深,全面涵盖了 Java 编程的基础知识及高级特性。在保证知识体系完备、脉络清晰、论述精准的同时,注重培养读者的动手能力,并结合相应的知识点编写了大量的示例。本书适合作为普通高等院校应用型本科、高职高专院校 Java 程序设计课程的教材,也可作为 Java 语言学习者的自学用书。学习本书之前应该熟悉计算机的基本操作,但不必具有编程经验。

各章节的主要内容如下:

第 1 章 Java 技术入门。本章主要介绍 Java 语言的发展历史、Java 技术特性、Java 平台核心机制以及 Java 开发环境的安装与配置,并通过一个简单示例使读者对 Java 程序的基本结构、运行过程有一个感性认识。

第 2 章 Java 语言基础。本章主要介绍 Java 语言对分隔符、标识符、关键字、注释、数据类型的具体规定,以及相关的编码惯例,它们是用 Java 进行程序设计的基础。

第 3 章 运算符、表达式和流程控制。本章详细介绍 Java 语言中的运算符和由若干操作数和运算符按照约定规则构成的表达式以及流程控制语句,为后续章节的学习提供了语言编程基础。

第 4 章 面向对象编程初步。Java 是面向对象的程序设计语言,提供了定义类、定义变量、方法等最基本的功能。如何用面向对象的观点去分析和解决问题是学习 Java 语言的重点,本章详细介绍了 Java 语言的引用数据类型——类以及与之相关的方法、对象等概念和使用方法。

第 5 章 面向对象编程进阶。继第 4 章之后,本章继续介绍面向对象程序设计的核心内容,其中包括初始化块、最终类和最终方法、继承、多态的概念及实现方法、抽象类和接口等。

第 6 章 数组。本章主要介绍 Java 中一维数组、多维数组的声明、创建、初始化以及数组元素的访问。

要掌握好 Java 语言并具有利用 Java 语言解决实际问题的能力,仅学习语法规则是不够的。本书从第 7 章开始介绍 Java 应用程序编程接口中的常用类的使用以及一些重要的编程技术。

第 7 章 精通常用的 Java 类。Java 应用程序编程接口是 Sun 公司开发的 Java 程序类

库, 提供给 Java 程序员使用的平台和工具, 利用这些类库中的类和接口可以方便地实现程序中的各种功能。本章重点介绍 Java 类库中常用类的使用。

第 8 章 Java 的异常处理。程序的稳定性和可靠性是 Java 语言设计的重要目标之一。Java 程序通过异常处理机制来加强程序对各种错误情况的处理能力, 从而提高程序的稳定性和可靠性。本章介绍异常的概念、异常处理机制等内容。

第 9 章 Java 中的泛型编程。本章主要介绍自 JDK 5.0 开始引入的一种 Java 语言新特性——泛型, 其实质是将原本确定不变的数据类型参数化, 作为对原有 Java 类型体系的扩充, 使用泛型可以提供 Java 应用程序的类型安全、可维护性和可靠性。

第 10 章 Java 中的集合类。本章主要介绍 Java 集合类的具体语法和使用方式, 重点学习 Java 在数据结构方面强大的编程功能。

第 11 章 GUI 程序设计。本章详细介绍如何利用 java.awt 包和 javax.swing 包下的组件进行图形用户界面的应用程序开发。

第 12 章 Java 流与文件操作。所有的程序都离不开信息的输入和输出, 程序通过输入/输出与外部信息进行交互, Java 采用“输入/输出流”实现输入/输出操作, 从“流”读取数据或向“流”写入输入。本章详细介绍数据流的概念以及 java.io 包中封装的丰富的 I/O 流类。

第 13 章 多线程。本章介绍了 Java 中的多线程机制、多线程的实现和线程间的同步等内容。

第 14 章 Java 数据库编程。JDBC 是实现 Java 同各种数据库连接的关键, 它提供了将 Java 和数据库连接起来的程序接口, 使用户可以以 SQL 的形式编写访问请求, 然后传给数据库, 其结果再由此以接口返回。本章详细介绍 Java 程序中如何使用 JDBC 实现数据库的连接与访问。

第 15 章 反射。所谓反射, 可以理解为在运行时获取对象类型信息的操作。在反射机制的帮助下, 程序员可以在运行时动态地获取对象类型信息等, 从而编写更加具有可移植性的代码。本章详细介绍 Java 中的反射机制, 并通过具体的实例介绍了如何使用 Java 中的反射机制。

第 16 章 注解。JDK 5.0 以上的版本引入了一项新特性: Annotation, 中文翻译成注解, 是用来为程序元素(类、方法、成员变量等)设置说明和解释的一种元数据, Java 开发和部署工具可以读取这些注解, 并以某种形式处理这些注解。本章将详细介绍什么是注解、JDK 中内置的注解、如何自定义注解、如何对注解进行注解以及如何程序中读取注解信息等内容。

第 17 章 Java 网络编程。Java 是一种平台无关的编程语言, 具有“一次编写、到处运行”的特点, 所以出现了非常适合 Java 的网络编程。可以说在网络编程方面, 没有任何一门语言比 Java 更优秀。本章详细介绍 Java 中的网络编程机制。

为方便教学, 本书配有电子课件和示例源代码, 读者可发至邮箱 superxhx@163.com 索取。

本书由河北软件职业技术学院许焕新、丁宏伟任主编, 负责全书的构思和统稿, 刘丽华、李丹任副主编。第 1 章~第 3 章由丁宏伟编写, 第 4 章~第 6 章由刘丽华编写, 第 7 章

前 言

由赵占坤编写，第 8 章由薛玉倩编写，第 9 章~第 14 章由李丹编写，第 15 章~第 17 章由许焕新编写。特别感谢世纪乐知（北京）网络技术有限公司（CSDN）总裁蒋涛先生、乐知学院徐明华老师对本书的大力支持。

由于作者水平所限，书中难免有疏漏之处，欢迎各位同行和广大读者对本书提出建议和修改意见（E-mail: superxhx@163.com）。

编 者

2010 年 8 月

目 录

第 1 章 Java 技术入门	1
1.1 Java 概述	1
1.1.1 Java 发展简史	1
1.1.2 Java 技术特性	2
1.2 Java 平台核心机制	4
1.2.1 Java 平台	4
1.2.2 Java 程序的运行过程	5
1.3 Java 开发环境的安装与配置	6
1.3.1 下载和安装 Java 开发工具集	6
1.3.2 配置环境变量 Path	7
1.4 第一个 Java 应用程序	8
1.4.1 创建源文件 HelloWorld	8
1.4.2 将源文件编译为字节码文件	9
1.4.3 运行程序	10
1.4.4 诊断编译错误和运行时错误	11
1.4.5 Java 应用程序的结构	12
1.5 本章小结	13
1.6 上机实践	13
第 2 章 Java 语言基础	14
2.1 分隔符	14
2.2 标识符	15
2.3 关键字	15
2.4 注释	17
2.5 数据类型	18
2.5.1 数据类型的分类	18
2.5.2 基本数据类型	19
2.5.3 常量和变量	19
2.6 编码惯例	22
2.6.1 命名惯例	22
2.6.2 文件组织	23
2.6.3 分隔	23
2.6.4 声明语句	25
2.6.5 其他	25
2.7 本章小结	26
2.8 上机实践	26
第 3 章 运算符、表达式和流程控制	28
3.1 运算符和表达式	28
3.1.1 算术运算符	28
3.1.2 关系运算符	30
3.1.3 逻辑运算符	31
3.1.4 位运算符	32
3.1.5 赋值运算符	35
3.1.6 条件运算符	36
3.1.7 表达式	37
3.1.8 表达式中的数据类型转换	38
3.2 流程控制语句	41
3.2.1 分支语句	42
3.2.2 循环语句	48
3.2.3 控制循环结构	55
3.3 标准输入/输出	59
3.3.1 什么是控制台	59
3.3.2 读取控制台输入	60
3.4 本章小结	61
3.5 上机实践	62
第 4 章 面向对象编程初步	63
4.1 类和对象	63
4.1.1 类的定义	63
4.1.2 对象的使用	66
4.1.3 this 的使用	69
4.2 方法	71
4.2.1 方法的参数传递	72
4.2.2 形参长度可变的方法	76
4.2.3 递归方法	77
4.2.4 方法重载	78

4.3 变量.....	80	第 6 章 数组.....	125
4.4 隐藏和封装.....	83	6.1 一维数组.....	125
4.4.1 理解封装.....	83	6.1.1 一维数组的声明.....	126
4.4.2 使用访问控制符.....	83	6.1.2 数组对象的创建和使用.....	126
4.4.3 package 和 import.....	86	6.1.3 数组对象的 length 属性.....	129
4.5 本章小结.....	90	6.1.4 数组的静态初始化.....	130
4.6 上机实践.....	90	6.1.5 foreach 循环.....	131
第 5 章 面向对象编程进阶.....	91	6.2 多维数组.....	132
5.1 类的继承.....	91	6.2.1 二维数组的声明.....	133
5.1.1 继承的特点.....	91	6.2.2 二维数组对象的创建和 初始化.....	133
5.1.2 重写父类的方法.....	93	6.2.3 二维数组元素的访问.....	135
5.1.3 父类实例的 super 引用.....	94	6.3 数组排序.....	136
5.1.4 调用父类构造方法.....	96	6.4 本章小结.....	137
5.2 子类对象和父类对象的转换.....	98	6.5 上机实践.....	137
5.3 静态初始化块.....	100	第 7 章 精通常用的 Java 类.....	138
5.3.1 使用初始化块.....	100	7.1 Java API 的概念.....	138
5.3.2 静态初始化块.....	102	7.2 java.lang.Object 类——Java 类 的共同父类.....	139
5.4 单例类.....	104	7.3 字符串相关类型.....	142
5.5 final 修饰符.....	105	7.3.1 java.lang.String 类.....	142
5.5.1 final 变量.....	105	7.3.2 java.lang.StringBuffer 类.....	146
5.5.2 final 方法.....	109	7.4 java.lang.System 类与 java.lang.Runtime 类.....	148
5.5.3 final 类.....	110	7.4.1 System 类.....	148
5.6 抽象类.....	110	7.4.2 Runtime 类.....	148
5.6.1 抽象方法和抽象类.....	111	7.5 封装类.....	149
5.6.2 抽象类的作用.....	113	7.6 日期相关类型.....	152
5.7 接口.....	114	7.6.1 Date 类.....	153
5.7.1 接口的概念.....	114	7.6.2 Calendar 类.....	154
5.7.2 接口的定义.....	114	7.6.3 DateFormat 类.....	156
5.7.3 接口的继承.....	116	7.7 数学相关类.....	158
5.7.4 使用接口.....	117	7.7.1 基本数学功能类 Math.....	158
5.7.5 接口和抽象类.....	118	7.7.2 数据格式化工具类 NumberFormat/ DecimalFormat.....	159
5.8 内部类.....	119	7.8 本章小结.....	163
5.9 匿名类.....	121		
5.10 静态内部类.....	122		
5.11 本章小结.....	124		
5.12 上机实践.....	124		

7.9 上机实践.....	163	10.3.3 LinkedList 类.....	197
第 8 章 Java 的异常处理.....	164	10.4 集及其实现类.....	199
8.1 异常处理的基础知识.....	164	10.4.1 集中的主要方法.....	199
8.1.1 异常处理的类层次.....	164	10.4.2 HashSet 类.....	199
8.1.2 未捕获“异常”.....	165	10.5 “键-值”对型的集合.....	200
8.2 异常处理机制.....	166	10.5.1 为什么要使用“键-值”对型 的集合.....	201
8.2.1 使用 try-catch-finally 语句捕获 和处理异常.....	167	10.5.2 “键-值”对的典范—— Hashtable 类.....	202
8.2.2 throw 语句.....	171	10.6 枚举器与数据操作.....	204
8.2.3 throws 语句.....	172	10.6.1 访问集合类的“不确定性” 难题.....	204
8.2.4 finally.....	174	10.6.2 枚举器接口.....	204
8.3 自定义异常类.....	175	10.6.3 枚举器“分离”思想的 启示.....	205
8.4 断言.....	176	10.7 本章小结.....	206
8.4.1 断言的开启和关闭.....	176	10.8 上机实践.....	206
8.4.2 断言的使用.....	177	第 11 章 GUI 程序设计.....	207
8.4.3 什么时候使用断言.....	179	11.1 Java GUI 设计.....	207
8.5 本章小结.....	180	11.1.1 抽象窗口工具包.....	207
8.6 上机实践.....	180	11.1.2 组件和容器.....	207
第 9 章 Java 中的泛型编程.....	181	11.1.3 布局管理器.....	212
9.1 集合类的泛型.....	181	11.2 GUI 事件处理.....	218
9.2 使用自己定义的泛型.....	183	11.2.1 Java 事件和事件处理 机制.....	218
9.3 类型通配符.....	184	11.2.2 事件类.....	220
9.4 受限类型参数.....	185	11.2.3 监听接口.....	221
9.5 泛型方法.....	187	11.2.4 事件适配器.....	223
9.6 本章小结.....	188	11.2.5 内部类和匿名类在 GUI 事件 处理中的作用.....	225
9.7 上机实践.....	188	11.2.6 多重监听器.....	228
第 10 章 Java 中的集合类.....	189	11.3 字体和颜色的控制.....	229
10.1 集合类与数据容器.....	189	11.3.1 Color 类.....	229
10.1.1 在项目中自定义数据类型的 难处.....	189	11.3.2 Font 类.....	231
10.1.2 Java 中的集合类.....	190	11.4 AWT 常用组件.....	232
10.2 Collection 接口和 Map 接口.....	190	11.4.1 标签和按钮.....	232
10.3 列表.....	192	11.4.2 文本组件.....	233
10.3.1 Vector 类.....	192		
10.3.2 先进后出的 Stack 类.....	194		

11.4.3 选择组件.....	239	第 13 章 多线程.....	300
11.4.4 滚动组件.....	245	13.1 多线程简介.....	300
11.4.5 菜单组件.....	248	13.1.1 进程和线程.....	300
11.4.6 对话框.....	253	13.1.2 多线程优势.....	302
11.5 Swing 初步.....	260	13.2 多线程实现.....	302
11.5.1 Swing 概述.....	260	13.2.1 继承 Thread 类实现多线程.....	302
11.5.2 典型 Swing 组件.....	262	13.2.2 实现 Runnable 接口.....	305
11.6 本章小结.....	272	13.2.3 使用 Timer 和 TimerTask 组合.....	306
11.7 上机实践.....	272	13.3 线程的生命周期与线程 状态.....	308
第 12 章 Java 流与文件操作.....	273	13.3.1 线程优先级.....	309
12.1 数据流的基本概念.....	273	13.3.2 线程调度.....	310
12.1.1 输入流与输出流.....	273	13.4 多线程问题及处理.....	316
12.1.2 缓冲流.....	274	13.5 本章小结.....	319
12.2 Java 的标准数据流和输入/ 输出流.....	274	13.6 上机实践.....	320
12.2.1 Java 的标准数据流.....	274	第 14 章 Java 数据库编程.....	321
12.2.2 java.io 包中的数据流及 文件类.....	276	14.1 数据库简介.....	321
12.3 字节流.....	277	14.1.1 关系型数据库.....	321
12.3.1 InputStream 类和 OutputStream 类.....	277	14.1.2 SQL 语言.....	321
12.3.2 文件字节输入/输出流类 FileInputStream/ FileOutputStream.....	278	14.2 JDBC 基本概念.....	322
12.4 文件操作.....	282	14.2.1 什么是 JDBC.....	322
12.4.1 File 类.....	282	14.2.2 JDBC 数据库驱动程序.....	323
12.4.2 文件过滤器.....	285	14.3 JDBC 编程常用的类和 接口.....	324
12.4.3 随机文件操作.....	287	14.4 JDBC 编程访问数据库.....	327
12.5 字符流.....	289	14.4.1 使用 JDBC 访问数据库的 基本过程.....	328
12.5.1 Reader 类和 Writer 类.....	289	14.4.2 执行 SQL 语句.....	331
12.5.2 字符文件流 FileReader 和 FileWriter 类.....	290	14.5 获取数据库元数据.....	333
12.5.3 字符缓冲流 BufferedReader 和 BufferedWriter 类.....	291	14.6 可滚动和可更新结果集.....	335
12.6 对象序列化.....	295	14.7 预处理语句.....	341
12.7 本章小结.....	299	14.8 调用存储过程.....	343
12.8 上机实践.....	299	14.9 本章小结.....	344
		14.10 上机实践.....	344
		第 15 章 反射.....	345
		15.1 反射概述.....	345

15.1.1 Java 中的反射机制	345	16.4 对注解进行注解	368
15.1.2 Class 类	346	16.4.1 目标 Target	368
15.2 使用 Java 反射机制	347	16.4.2 类型 Retention	369
15.2.1 获取类信息	348	16.4.3 文档 Documented	370
15.2.2 创建对象	351	16.4.4 继承 Inherited	370
15.2.3 调用方法	353	16.5 利用反射获取注解信息	371
15.2.4 访问成员变量的值	354	16.6 本章小结	373
15.2.5 操作数组	355	16.7 上机实践	373
15.3 反射应用举例	357	第 17 章 Java 网络编程	374
15.3.1 properties 文件的处理	357	17.1 网络基础知识	374
15.3.2 应用反射实现松耦合	359	17.1.1 网络基本概念	374
15.4 本章小结	361	17.1.2 IP 地址和端口号	376
15.5 上机实践	361	17.2 Java 的基本网络应用	377
第 16 章 Java 注解	362	17.2.1 InetAddress 类的使用	377
16.1 注解概述	362	17.2.2 URL 类的使用	378
16.2 JDK 内置的基本注解类型	363	17.2.3 URLDecoder 类和 URLEncoder 类	379
16.2.1 重写 Override	363	17.3 基于 Socket 的网络编程	381
16.2.2 警告 Deprecated	364	17.4 本章小结	387
16.2.3 抑制警告 SuppressWarnings	365	17.5 上机实践	387
16.3 自定义注解类型	366		

第 1 章 Java 技术入门

Java 是由 Sun 公司于 1995 年推出的一种全新的、跨平台、适用于分布式计算环境的纯面向对象编程语言。本章将向读者介绍 Java 技术的基本情况, 主要包括 Java 语言的发展简史、Java 技术的关键特性、Java 平台核心机制以及 Java 开发环境的安装与配置, 并通过一个简单的示例, 讲解如何编写、编译和运行一个简单的 Java 应用程序。

本章学习要点如下:

- ☑ Java 语言的发展简史。
- ☑ 描述 Java 技术的关键特性。
- ☑ 描述 Java 虚拟机的功能。
- ☑ Java 开发环境的安装与配置。
- ☑ 编写、编译和运行第一个 Java 应用程序。

1.1 Java 概述

Java 语言自 1995 年 5 月由 Sun 公司发布以来, 便凭借其易学易用、功能强大的特点得到了广泛的应用, 发展势头极为强劲。目前, 它已不仅仅是一门高级编程语言, 也是一种完备的技术体系和开发平台。Java 拥有卓越的技术特性、丰富的编程接口(类库)、多款功能强大的开发工具平台, 在企业级应用开发领域中占有过半的市场份额, 超过 25 亿台设备正在使用 Java 技术并获得所有主流 IT 厂商的大力支持。从目前的应用现状和发展前景来看, Java 已经成为软件开发从业人员的首选技术。

1.1.1 Java 发展简史

1990 年, Sun 公司成立了一个由 James Gosling (Java 之父) 领导的“Green 计划”项目组, 准备为下一代智能家电(如电视机、微波炉、电话、机顶盒)编写一个通用控制系统。

该团队最初考虑使用 C++ 语言来编写嵌入式系统, 但后来发现 C++ 太复杂, 安全性差, 不适合这类任务, 于是 Sun 的首席科学家 Bill Joy 与 Gosling 决定开发一种全新的语言——Oak。这是一种用于网络的精巧而安全的语言, Sun 公司对其寄予了厚望, 但起初并非一帆风顺, 始终不温不火。

1994 年, 互联网和浏览器的出现给 Oak 语言带来了新的生机, Oak 小组完成了第一个 Java 语言的网页浏览器——WebRunner, Java 开始向 Internet 进军。当时 Oak 这个商标已被别人注册, 于是只得将 Oak 更名为 Java。

1996 年 1 月, Sun 公司发布了 Java 的第 1 版, 并从此开始提供并持续维护完备的 Java

开发工具集 (Java Development Kit, JDK)。Java 1.0 版本除了语言规范本身, 主要包含两部分——开发工具集 (JDK) 和运行时环境 (Java Runtime Environment, JRE)。

1997 年 2 月, Sun 公司发布了 Java 1.1 版。相对于 JDK 1.0, JDK 1.1 引入了内部类和即时编译 (Just-In-Time, JIT) 技术。

1998 年 12 月, Sun 公司发布了一个里程碑式的版本——Java 1.2。该版本中出现了许多革命性的变化, 这些变化一直沿用至今并对 Java 发展产生了极为深远的影响。首先, 从 1.2 版本开始 Sun 公司将 Java 改名为更具吸引力的 Java 2 (意为第二代的 Java); 其次, 原来的开发工具集 (JDK) 也更名为 J2SDK (Java 2 Software Development Kit); 最主要的是, 从 Java 1.2 开始 Sun 公司将 Java 版本一分为三, 即标准版、微缩版和企业版。

- ❑ 标准版 (Java 2 Standard Edition, J2SE): 是整个 Java 技术的核心和基础, 可以开发适用于 PC 机上运行的程序, 也是其他两个版本的基础。
- ❑ 微缩版 (Java 2 Micro Edition, J2ME): 主要用于移动设备 (如手机) 和信息家电等有限存储的设备。
- ❑ 企业版 (Java 2 Enterprise Edition, J2EE): Java 技术中应用最为广泛的部分, 提供了企业应用开发相关的完整解决方案。

各版本都有自己的开发工具集, 从 JDK 1.2 开始 Sun 公司大约每两年推出一个 JDK 的新版本。

2000 年 5 月, Sun 公司发布了 J2SE 1.3 版。此版本主要改进了类库和对本地资源的访问, 支持 XML, 并采用了新的 Hotspot 虚拟机。

2002 年 2 月, Sun 公司发布了 J2SE 1.4 版。此版本进一步改进了 Hotspot 虚拟机的性能, 并引入了新的语言特性“断言” (Assert)。

2004 年 10 月, Sun 公司发布了 Java SE 5.0 版。此版本原被命名为 1.5 版, 后又将版本号改为 5.0, 以说明其较以前版本的巨大改进。在 Java SE 5.0 中, 增加了诸如泛型、for-each 循环语句、可变数目参数、注释、自动装箱和拆箱等功能。

2006 年 11 月, Sun 公司发布了 Java SE 6.0 版。Java SE 6.0 不仅在性能、易用性方面得到了前所未有的提高, 而且还提供了如脚本、全新的 API 的支持。此外, Java SE 6.0 还专为 Vista 做过针对性设计, 它在 Vista 上将会拥有更好的性能。在推出 Java SE 6.0 的同时, Java SE 7.0 项目也已经启动。

1.1.2 Java 技术特性

1. 简单性

Java 语言的简单性主要体现在以下 3 个方面:

- ❑ Java 的编程风格非常类似于 C++, C++ 程序员或学过 C++ 的读者可以借助 C++ 编程知识快速掌握 Java。
- ❑ Java 语言摒弃了 C++ 中容易引发程序错误的内容, 如指针、人工分配和回收内存、无条件转移语句以及无节制的强制类型转换等。
- ❑ JDK 中包含了丰富实用的应用程序接口 (Application Programming Interface, API),

即预先准备好的、提供各种常用功能的代码模块，在此基础上开发者可极大地提高开发效率。

2. 面向对象

Java 语言是一种纯粹的面向对象的语言。面向对象是现代编程语言的重要特性之一，其核心是对数据进行封装和对程序代码的模块化组织，在更大程度上支持代码复用。历史的经验已经表明，面向对象技术极大地提高了人们的软件开发能力，现在很难想象还使用纯粹的面向过程的语言去开发大型、复杂的项目。

3. 分布式

使用 JDK 中的 API，Java 程序可以便捷地实现网络通信，包括基于 HTTP/FTP 等协议的普通数据传输、基于 SMTP/POP3 等协议的邮件收发、程序间的协作；利用远程方法调用（Remote Method Invocation, RMI）可以实现网络环境中的分布式计算。

4. 多线程

多线程技术是在一个应用程序内部再细分而成多个子任务，这些子任务作为多个顺序控制流并发运行，既可以相互独立（可以进行独立启停控制），又可以在一定程度上共享数据和代码，并能够被自动调度、交替运行，极大地提高了 Java 程序的运行效率。

5. 动态性

虽然 Java 编译器在编译时的静态检查是极为严格的，但 Java 语言本身和运行时环境却采用了动态链接的方式，即一个 Java 类只有在用到时才会被载入。对现有 Java 类的任何修改都将立即体现到所有用到该类的程序中，被载入的类可以来自于本地甚至网络。动态链接这一特性也显著地减小了 Java 程序的规模，使之便于发布和通过网络传输。

6. 体系结构中立和可移植性

Java 语言主要的设计目标就是可以在异构的网络环境中运行，即同一 Java 程序可以不做任何改动就运行在不同的硬件平台和各种操作系统上。为确保 Java 程序的可移植性，Java 语言中所有数据类型都是定长的，例如其中 int 型数据占有的存储空间永远是 32 位，而与程序运行所在的计算机处理器结构和操作系统无关，这样可以避免同一程序在不同的机器上运行得到不同结果，或在一台机器上运行正常但在另一台机器上却出现数据溢出错误。

7. 健壮性

为了能够开发高可靠性的软件，Java 语言提供了广泛的编译时检查和运行时检查机制。例如，不匹配的强制类型转换、使用未经初始化的变量等都通不过编译检查，内存空间的分配和回收完全由系统负责而不必由程序员干预，取消了指针类型和指针运算等。Java 程序中的大多数问题均可由编译器和运行时环境快速发现，而 C++ 程序则经常因为这类错误难于发现和准确定位而搞得开发人员焦头烂额。

8. 安全性

Java 程序要运行在网络环境中，其安全性尤为重要。Java 应用程序在运行时对数据的访问处理权限受到严格控制，如不允许访问其他程序内存范围中的数据，也不允许网络程序修改本地数据，字节码指令在执行前还将经过一次安全性检验，以防止存在受限制操作。

1.2 Java 平台核心机制

1.2.1 Java 平台

在计算机科学中，支撑程序运行的硬件或软件环境被称为平台（Platform），也可以将平台理解为操作系统和底层硬件的结合体。目前，主流的平台包括 Microsoft Windows、Linux、UNIX、Sun Solaris 及 Apple Mac OS 等。

各种平台都有其特有的指令格式，例如不同的处理器或运行在同一处理器上不同的操作系统，所支持的指令格式都是不通用的，进而导致了 Windows 平台的可执行文件（.exe 文件）不能在 UNIX 平台上运行，反之 UNIX 平台的可执行文件（.bin 文件）也无法在 Windows 平台上运行，这种情况我们称之为平台相关。

和大多数平台不同，Sun 公司的 Java 平台是一种纯软件的平台，它运行在其他基于硬件的平台（如 Windows）之上。Java 平台主要由两部分组成：

- ❑ Java 虚拟机（Java Virtual Machine, JVM）。
- ❑ Java 应用程序编程接口（Java Application Programming Interface, API）。

顾名思义，Java 虚拟机是由软件虚拟的计算机，是 Java 平台的核心。它也有自己的指令格式和可执行文件，即字节码（Byte Code）指令和字节码文件。需要强调的是，Java 虚拟机在运行时并不能直接操控硬件，例如它不能直接控制 CPU 或直接访问物理内存，而是通过调用底层基于硬件的平台（如 Windows）的功能来实现。Java 平台的工作原理如图 1-1 所示。

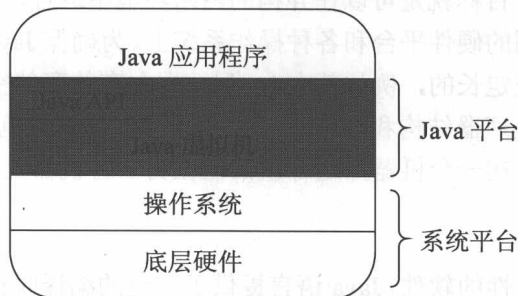


图 1-1 Java 平台工作原理

可以看出，Java 程序之所以能够实现跨平台运行，是因为它根本就不直接运行在任何底层平台上，而是需要在哪里运行，就在哪里（如 Windows）事先准备好自己的 Java 平台，这只是安装和配置一个软件而已，然后就可以“随处运行”。

除虚拟机之外,Java 平台的另一个核心技术是自动垃圾回收机制(Garbage Collection)。我们知道,任何语言的程序在运行时都需要占用一定的内存空间来保存数据,当不再需要这些数据时应及时清理回收这些无用内存空间以备将来其他程序使用,人们称这种无用内存空间的回收操作为“垃圾回收”。在其他语言如 C++ 中,主要由程序员负责内存空间的人工分配和回收,一般情况下由于数据量较大、数据使用时间较长或其作用范围较大,分配和回收内存的代码通常位于不同的程序段,进而经常出现不再使用的内存空间未能及时回收或忘记回收,这种情况被称为“内存泄露”,会导致系统因可用内存不足而瘫痪。

另外,还应掌握另一个常用术语——Java 运行时环境(Java Runtime Environment, JRE)。可以认为 JRE 是 JDK 的一个子集,用于解释执行编译后的 Java 程序(“.class”文件)。JRE 包含 Java 虚拟机、Java 核心类库及支持文件,但不包含编译器等其他工具。如果用户只需要在自己的计算机上运行而不是开发 Java 程序,可从 Sun 公司网站上单独下载 JRE。

1.2.2 Java 程序的运行过程

为了更好地了解 Java 程序的运行过程,在此有必要介绍一下编译(Compile)这个概念。所谓“编译”就是一种转换处理——将程序代码从一种指令格式转换为另一种指令格式,以使之能在特定平台/环境中运行,即能够被识别和处理。编译器(Compiler)则是能够提供编译功能的软件程序。

Java 语言编写的程序代码首先以纯文本文件形式保存,文件的扩展名(后缀)为“.java”,这些程序文件被称为“源文件”(Source File),其中的程序代码也称“源代码”(Source Code);然后由编译器将源文件编译为字节码(Byte Code)文件,其扩展名为“.class”。

字节码文件是 Java 编译器专门针对 Java 虚拟机生成的,其中的指令格式(字节码指令)可以由 Java 虚拟机识别和处理,因而也可认为字节码文件就是 Java 平台中的可执行文件。而对于其他平台来讲,字节码文件的指令格式是与平台无关的。

在运行时,Java 虚拟机中的运行时解释器(Runtime Interpreter)模块专门负责字节码文件的解释执行——运行时解释器先将字节码指令解释成所在底层平台(如 Windows)能够识别、处理的指令格式,即本地机器码,然后再委托/调用底层平台的功能来执行。这里的解释执行类似于国际会议中的同声翻译,是逐条进行的,即解释一条执行一条。Java 程序运行原理如图 1-2 所示。

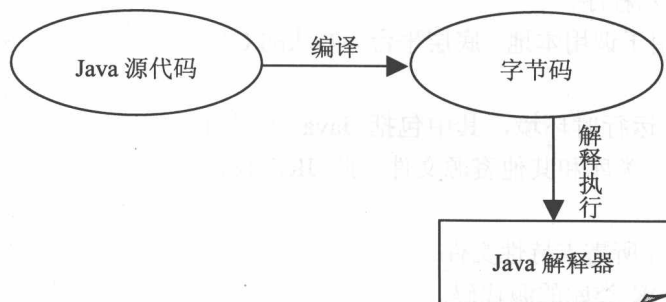


图 1-2 Java 语言运行机制