

The Java Tutorial, Third Edition

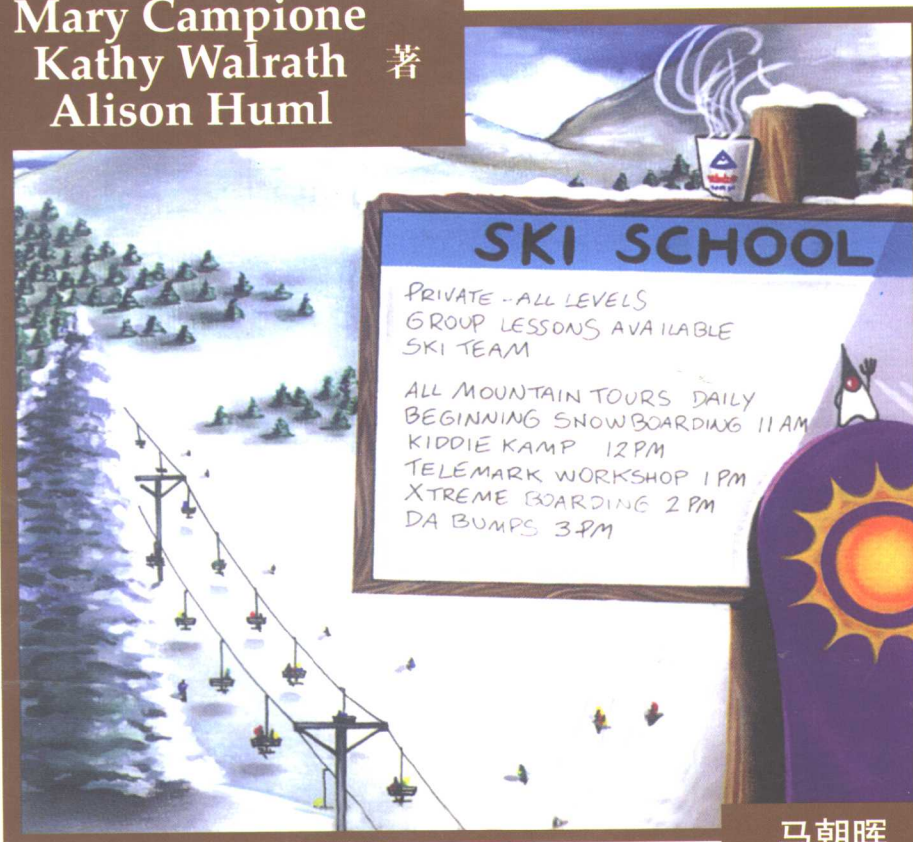
A Short Course on the Basics

Java

语言导学

(原书第3版)

Mary Campione
(美) Kathy Walrath 著
Alison Huml



马朝晖 等译

Sun 公司核心技术丛书

Java 语言导学

(原书第 3 版)

Mary Campione
(美) Kathy Walrath 著
Alison Huml
马朝晖 等译



机械工业出版社
China Machine Press

本书以面向任务的方法介绍 Java 语言,涉及了 Java 语言的基础知识和使用技巧。主要内容
包括:Java 语言基础、对象基础和简单数据对象、类和继承、接口和包、异常处理、线程、输入/输
出、Swing 用户界面等。本书的作者经验丰富,为初学者指引了易学快用的方法,本书大量实例
提供了 Java 最新应用方法(针对 Java 2 SDK 1.3 版),并提供了编程问题的解决方案。

Authorized translation from the English language edition, entitled The Java Tutorial, Third Edi-
tion: A Short Course On the Basics, 3 by Mary Campione, kathy Walrath and Alison Huml, pub-
lished by Pearson Education, Inc., publishing as ADDISON WESLEY Copyright 2001.

All Rights Reserved. No part of this book may be reproduced or transmitted in any form or by
any means, electronic or mechanical, including photocopying, recording or by any information storage
retrieval systems, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION NORTH A-
SIA LTD and CHINA MACHINE PRESS, Copyright 2002.

This edition is authorized for sale only in People's Republic of China (excluding the Special Ad-
ministrative Region of Hong Kong and Macau) .

版权所有,侵权必究。

本书版权登记号: 图字: 01 - 2001 - 4772

图书在版编目 (CIP) 数据

Java 语言导学 (原书第 3 版) / (美)坎皮恩 (Campione, M.), (美)沃拉瑟 (Walrath,
K.), (美)赫木尔 (Huml, A.) 著; 马朝晖等译. - 北京: 机械工业出版社, 2002.1
(Sun 公司核心技术丛书)

书名原文: The Java Tutorial, Third Edition: A short Course On the Basics
ISBN 7 - 111 - 09585 - 5

I .J... II .①坎...②沃...③赫...④马... III .JAVA 语言 - 程序设计 IV .TP312

中国版本图书馆 CIP 数据核字 (2001) 第 091265 号

机械工业出版社 (北京市西城区百万庄大街 22 号 · 邮政编码 100037)

责任编辑: 张鸿斌

北京忠信诚胶印厂印刷 · 新华书店北京发行所发行

2002 年 2 月第 1 版第 1 次印刷

787mm × 1092mm 1/16 · 28.25 印张

印数: 0 001 - 5 000 册

定价: 59.00 元 (附光盘)

凡购本书, 如有倒页、脱页、缺页, 由本社发行部调换

前 言

自从最初的 Java Development Kit 于 1995 年 5 月发布以来，Sun Microsystems 的工程人员为改进和增强 Java 平台付出了艰苦的努力。我们也相应地更新了本书的内容来反映这些工程人员的成果。

本书是第 3 版，针对 Java 2 Software Development Kit (SDK) 1.3 版。因为你可能不得不使用 Java 平台的较早版本编写或更新代码，所以本书也适用于 1.2 和 1.1 版。

除了添加了 1.3 版的信息，我们还增加了问题与练习以帮助你实践所学到的内容。为了帮助初学者避免许多常见的错误，附录 A 研究了编程问题及其解决方案。每节后面的小结也是本版中新增的。

与第 1 版和第 2 版一样，本书也以 Sun Microsystems 网站上的 Java 平台在线指南为基础。该网站地址为：<http://java.sun.com/docs/books/tutorial/index.html>。

与在线版本一样，本书反映了 Java 技术的最新发展。与在线版本不同的是，本书只关注初、中级程序员需要的 API。一旦你掌握了本书中的资料，就可以在此网站上研究 Java 平台的其他问题。

我们的目标一直是写一本轻松易学的面向任务的程序员指南，并且以大量实际例子帮助人们学习编程。

谁应该阅读本书

本书既适合初学者，也适合有经验的程序员：

- 新程序员通过从头到尾阅读本书可以得到最大的收获，包括按照第 1 章“起步”中的步骤说明编译和运行自己的第一个程序。
- 有过程式语言（比如 C）经验的程序员可能希望从 Java 编程语言的面向对象概念和特性开始学习。
- 有面向对象编程经验的程序员可能希望先学习更高级的内容，比如关于 applet、基本类或用户界面。

无论你是哪种程序员，都可以通过本书找到满足自己需要的学习途径。

如何使用本书

本书的设计方式使你既可以通读它，也可以在主题之间跳转。当另一个地方讨论某主题

时，你会看到到此位置的“链接”，其中指出了章节号。

本书中使用的所有样例代码都可以从网上和本书附带的光盘上得到。在每章的末尾有一节“代码样例”，其中的表格指出样例在光盘和网上的位置。

我们尽力使本书跟上最新的技术。要学习更新的信息，请访问以下 URL：

<http://java.sun.com/docs/books/tutorial/books/3e/index.html>

本书主要由马朝晖、陈美红翻译，参与翻译、录入、审校的还有刘丽珍、王建芬、杨帆、邹辉、潘浩、楼涵、董小蕾、王悦、李军、罗伟、鲍广华、瞿兰、陆明、宋丽、杨立军、李鸣、马晓云、成荣光等。

目 录

前言		2.1 什么是对象	34
第 1 章 起步	1	2.2 什么是消息	36
1.1 关于 Java 技术	1	2.3 什么是类	37
1.1.1 Java 编程语言	1	2.4 什么是继承	39
1.1.2 Java 平台	2	2.5 什么是接口	40
1.1.3 Java 技术可以做什么	3	2.6 如何将这些概念运用到代码中	41
1.2 Java 技术将如何改变我们的生活	5	2.6.1 ClickMe 的源代码和 applet 标记	42
1.3 第一步 (Win32)	5	2.6.2 ClickMe applet 中的对象	42
1.3.1 检查表	6	2.6.3 ClickMe applet 中的类	43
1.3.2 创建第一个应用程序	6	2.6.4 ClickMe applet 中的消息	43
1.3.3 创建第一个 applet	9	2.6.5 ClickMe applet 中的继承	44
1.3.4 错误解释 (Win32)	11	2.6.6 ClickMe applet 中的接口	45
1.4 第一步 (UNIX/Linux)	12	2.6.7 API 文档	46
1.4.1 检查表	12	2.7 小结	47
1.4.2 创建第一个应用程序	12	2.8 问题与练习	47
1.4.3 创建第一个 applet	16	2.8.1 问题	47
1.4.4 错误解释 (UNIX/Linux)	17	2.8.2 练习	48
1.5 第一步 (MacOS)	18	2.8.3 解答	48
1.5.1 检查表	18	2.9 代码样例	48
1.5.2 创建第一个应用程序	18	第 3 章 语言基础	49
1.5.3 创建第一个 applet	23	3.1 变量	49
1.5.4 错误解释 (MacOS)	25	3.1.1 数据类型	50
1.6 分析 HelloWorld	25	3.1.2 变量名称	52
1.6.1 对应用程序的解释	25	3.1.3 作用范围	52
1.6.2 对 applet 的剖析	28	3.1.4 变量初始化	54
1.7 问题与练习	31	3.1.5 final 变量	54
1.7.1 问题	31	3.1.6 变量的小结	54
1.7.2 练习	32	3.1.7 问题与练习: 变量	55
1.7.3 解答	33	3.2 操作符	55
1.8 代码样例	33	3.2.1 算术操作符	56
第 2 章 面向对象的编程概念	34	3.2.2 关系和条件操作符	60

3.2.3 移动和位操作符	63	4.2.6 在字符串中搜索字符或子字符串	109
3.2.4 赋值操作符	66	4.2.7 比较字符串和部分字符串	111
3.2.5 其他操作符	67	4.2.8 操作字符串	112
3.2.6 操作符的小结	68	4.2.9 修改字符串缓冲区	113
3.2.7 问题与练习: 操作符	70	4.2.10 字符串和编译器	115
3.3 表达式、语句和代码块	71	4.2.11 字符和字符串小结	115
3.3.1 表达式	71	4.2.12 问题与练习: 字符和字符串	117
3.3.2 语句	73	4.3 数字	118
3.3.3 代码块	74	4.3.1 数字类	119
3.3.4 表达式、语句和代码块的小结	74	4.3.2 将字符串转换为数字	121
3.3.5 问题与练习: 表达式、语句和 代码块	74	4.3.3 将数字转换为字符串	122
3.4 流程控制语句	75	4.3.4 对数字进行格式化	123
3.4.1 while 和 do-while 语句	76	4.3.5 用定制的格式对数字进行 格式化	125
3.4.2 for 语句	77	4.3.6 高级算术功能	126
3.4.3 if-else 语句	78	4.3.7 数字小结	131
3.4.4 switch 语句	80	4.3.8 问题与练习: 数字	131
3.4.5 异常处理语句	83	4.4 数组	132
3.4.6 分支语句	83	4.4.1 创建和使用数组	133
3.4.7 流程控制语句的小结	87	4.4.2 对象数组	135
3.4.8 问题与练习: 流程控制语句	89	4.4.3 数组的数组	136
3.5 代码样例	90	4.4.4 复制数组	138
第 4 章 对象基础和简单数据对象	93	4.4.5 数组小结	139
4.1 对象的生存周期	93	4.4.6 问题与练习: 数组	139
4.1.1 创建对象	94	4.5 代码样例	140
4.1.2 使用对象	98	第 5 章 类和继承	143
4.1.3 清除不被使用的对象	100	5.1 创建类	143
4.1.4 对象小结	101	5.1.1 声明类	143
4.1.5 问题与练习: 对象	101	5.1.2 声明成员变量	145
4.2 字符和字符串	102	5.1.3 定义方法	146
4.2.1 字符	103	5.1.4 为类提供构造器	149
4.2.2 字符串和字符串缓冲区	105	5.1.5 将信息传递给方法或构造器	150
4.2.3 创建字符串和字符串缓冲区	105	5.1.6 从方法返回值	152
4.2.4 得到字符串或字符串缓冲区的 长度	107	5.1.7 使用 this 关键字	154
4.2.5 通过索引从字符串或字符串缓冲区 得到字符	107	5.1.8 控制对类成员的访问	155
		5.1.9 理解实例和类成员	160

5.1.10 实例和类成员的初始化	162	7.2 捕获或指定要求	197
5.1.11 创建类的小结	164	7.3 捕获和处理异常	198
5.1.12 问题与练习: 创建类	164	7.3.1 try 块	199
5.2 管理继承	165	7.3.2 catch 块	200
5.2.1 覆盖和隐藏方法	165	7.3.3 finally 块	200
5.2.2 隐藏成员变量	168	7.3.4 结合	201
5.2.3 使用 super	168	7.4 指定方法抛出的异常	205
5.2.4 使用 Object 的后代	169	7.5 如何抛出异常	206
5.2.5 编写 final 类和方法	172	7.5.1 throw 语句	206
5.2.6 编写抽象类和方法	173	7.5.2 Throwable 类及其子类	207
5.2.7 管理继承的小结	175	7.5.3 创建自己的异常类	208
5.2.8 问题与练习: 管理继承	175	7.6 运行时异常——争论	209
5.3 实现嵌套的类	176	7.7 异常的优点	210
5.3.1 内部类	178	7.7.1 优点 1: 将错误处理代码与“常规” 代码分离	210
5.3.2 关于嵌套类的其他问题	180	7.7.2 优点 2: 将错误沿调用堆栈向上 传递	211
5.3.3 嵌套类的小结	180	7.7.3 优点 3: 对错误类型进行分组和 区分	213
5.3.4 问题与练习: 嵌套类	180	7.8 异常的小结	214
5.4 代码样例	181	7.9 问题与练习	214
第 6 章 接口和包	183	7.9.1 问题	214
6.1 编写和使用接口	183	7.9.2 练习	215
6.1.1 定义接口	184	7.9.3 解答	216
6.1.2 实现接口	185	7.10 代码样例	216
6.1.3 将接口作为类型使用	186	第 8 章 线程: 同时执行多个任务	217
6.1.4 接口不能改变	186	8.1 什么是线程	217
6.1.5 接口小结	187	8.2 使用 Timer 和 TimerTask 类	219
6.1.6 问题与练习: 接口	187	8.2.1 停止计时器线程	220
6.2 创建和使用包	188	8.2.2 重复执行任务	221
6.2.1 创建包	189	8.2.3 关于 Timer 的更多信息	222
6.2.2 命名包	190	8.3 定制线程的 run 方法	223
6.2.3 使用包成员	190	8.3.1 对 Thread 类进行子类化和覆盖 run	223
6.2.4 管理源代码文件和类文件	192	8.3.2 实现 Runnable 接口	225
6.2.5 创建和使用包的小结	194	8.3.3 决定使用 Runnable 接口	226
6.2.6 问题与练习: 创建和使用包	194	8.4 线程的生存周期	226
6.3 代码样例	195	8.4.1 创建线程	227
第 7 章 使用异常处理错误	196		
7.1 什么是异常	196		

8.4.2 启动线程	227	9.2.5 操作过滤器流	263
8.4.3 使线程不可运行	229	9.3 对象的串行化	270
8.4.4 停止线程	229	9.3.1 对对象进行串行化	271
8.4.5 isAlive 方法	230	9.3.2 为类提供对象串行化	272
8.5 理解线程优先级	231	9.4 操作随机访问文件	274
8.5.1 400 000 微米线程赛跑	231	9.4.1 使用随机访问文件	276
8.5.2 自私的线程	233	9.4.2 为随机访问文件编写过滤器	277
8.5.3 时间片	233	9.5 其他问题	279
8.5.4 线程优先级的小结	235	9.6 读和写的小结	279
8.6 线程的同步	235	9.7 问题与练习	280
8.6.1 生产者/消费者例子	235	9.7.1 问题	280
8.6.2 锁定对象	237	9.7.2 练习	280
8.6.3 使用 notifyAll 和 wait 方法	239	9.7.3 解答	281
8.6.4 运行生产者/消费者例子	241	9.8 代码样例	281
8.6.5 避免饿死和死锁	242	第 10 章 Swing 用户界面	283
8.7 线程的分组	243	10.1 SwingAPI 概述	283
8.7.1 线程组	244	10.2 示例 1: Hello World Swing	284
8.7.2 在组中显式地创建线程	244	10.3 示例 2: SwingApplication	286
8.7.3 得到线程的组	244	10.3.1 外观	286
8.7.4 使用 ThreadGroup 类	245	10.3.2 建立按钮和标签	287
8.8 线程的小结	248	10.3.3 处理事件	288
8.8.1 线程的包支持	248	10.3.4 在组件周围添加边框	290
8.8.2 线程的语言支持	249	10.4 示例 3: CelsiusConverter	290
8.8.3 线程的运行支持	249	10.4.1 添加 HTML	291
8.9 问题与练习	249	10.4.2 添加图标	292
8.10 代码样例	250	10.5 示例 4: LunarPhases	292
第 9 章 I/O: 读和写	252	10.5.1 复合边框	294
9.1 I/O 流概述	252	10.5.2 组合框	295
9.1.1 字符流	253	10.5.3 多个图像	296
9.1.2 字节流	254	10.6 示例 5: VoteDialog	296
9.1.3 理解 I/O 超类	254	10.6.1 单选按钮	297
9.2 使用流	255	10.6.2 对话框	298
9.2.1 如何使用文件流	257	10.7 布局管理	301
9.2.2 如何使用管道流	259	10.8 线程和 Swing	304
9.2.3 如何封装流	261	10.8.1 单线程规则	305
9.2.4 如何联结文件	261	10.8.2 如何在事件调度线程中执行代码	306

10.9 Swing 组件的可视索引	306	附录 A 常见问题及其解决方案	317
10.10 小结	313	附录 B 用于 Internet 的 applet	329
10.11 问题与练习	313	附录 C 集合	377
10.11.1 问题	313	附录 D 被废弃的线程方法	426
10.11.2 练习	314	附录 E 参考信息	432
10.11.3 解答	315		
10.12 代码样例	315		

第 1 章 起 步

本章将简单介绍 Java 技术。首先，我们解释什么是 Java 平台以及它可以做什么。接下来，一步步地讲解如何在 Win32、UNIX/Linux 或 MacOS 平台^①上编译和运行两个简单的程序。之后，我们看看这两个程序的代码，以便了解它们是如何工作的。本章结尾的问题和练习可以测试和扩展你的知识，最后是一个表格给出本章中使用的代码的下载说明。

Sun Microsystems 公司提供的 Software Development Kit (SDK) 包括运行和编译程序所需的最小的工具集。严谨的开发人员应该使用一个专业的集成开发环境 (Integrated Development Environment, IDE)^②。IDE 的列表参见附录 E。

1.1 关于 Java 技术

到处都在谈论 Java 技术，但是 Java 技术到底是什么？下面两节解释 Java 编程语言和 Java 平台。

1.1.1 Java 编程语言

Java 编程语言是一种高级语言，它具有以下性质^③。

- 简单
- 面向对象
- 分布式
- 解释
- 稳固
- 安全
- 与体系结构无关
- 可移植
- 高性能
- 多线程
- 动态

在大多数语言中，要么编译程序，要么解释程序，才能在计算机上运行它。Java 编程语言的特殊之处在于程序既被编译又被解释。首先，使用编译器将程序翻译为一种称为 Java 字节编码 (bytecode) 的中间语言，这是由 Java 平台上的解释器解释的与平台无关的代码。解释器在计算机上分析并运行每条 Java 字节编码指令。编译只发生一次；而解释在每次执行程序时都发生。图 1-1 表示了此工作过程。

你可以将 Java 字节编码看作用于 Java 虚拟机 (Java Virtual Machine, Java VM) 的机器码指令。每个 Java 解释器，无论是开发工具还是可以运行 applet 的 Web 浏览器，都是一种 Java

① 如果你使用的平台没有在这里列出，那么怎么办？Sun Microsystems 公司维护一个用于其他平台的第三方端口列表：
<http://java.sun.com/cgi-bin/java-ports.cgi>。

② 实际上，Java 2 SDK Standard Edition v.1.3 与一个 IDE 捆绑，Forte for Java Community Edition。本书的光盘中包含这个版本。

③ 这些术语都在 James Gosling 和 Henry McGilton 所著的白皮书 “The Java Language Environment” 中做了解释，可以在
<http://java.sun.com/docs/white/langenv/index.html> 找到此白皮书。

VM 实现。

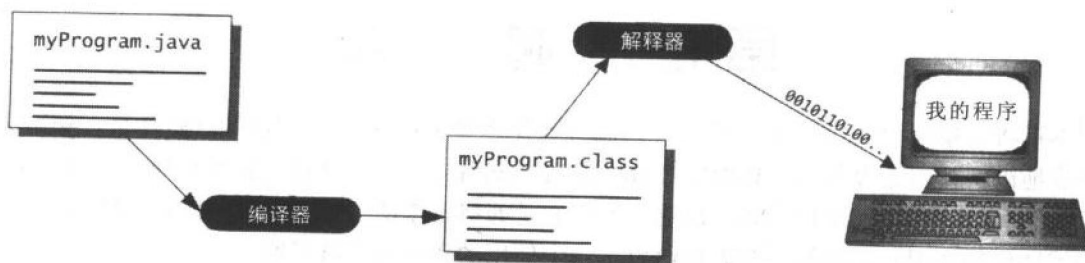


图 1-1 用 Java 编程语言写的程序先被编译，然后被解释

Java 字节编码有助于使“一次编写，处处运行”成为可能。你可以在任何有 Java 编译器的平台上将你的程序编译为字节编码。字节编码可以在任何 Java VM 实现上运行。这意味着只要计算机上有一个 Java VM，那么用 Java 编程语言写的同样的程序就能够在 Windows 2000、Solaris 工作站或 iMac 上运行，如图 1-2 所示。

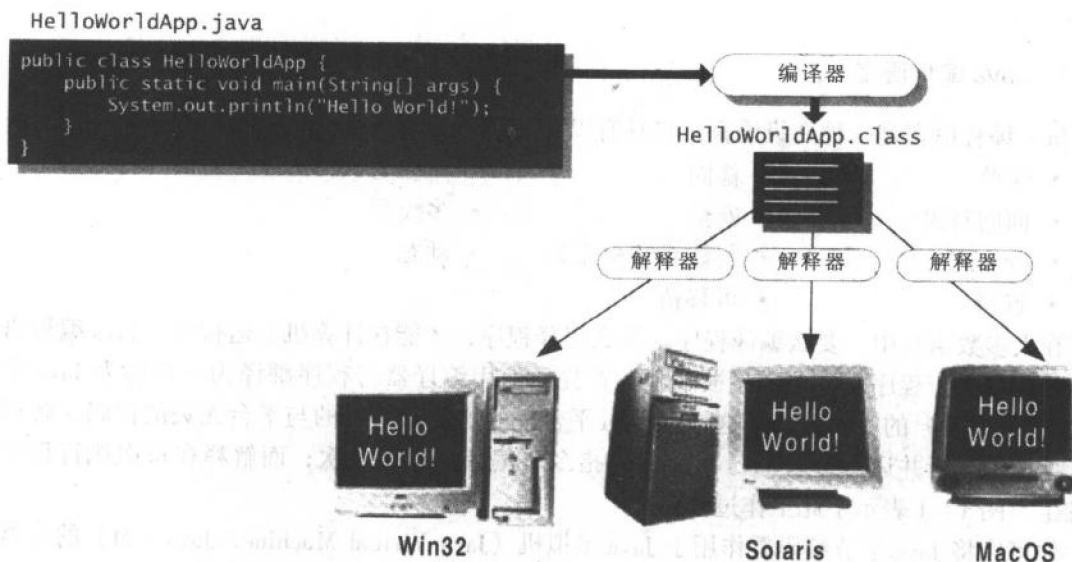


图 1-2 编写一次程序，它可以在几乎任何平台上运行

1.1.2 Java 平台

平台(platform)是程序在其中运行的硬件或软件环境。我们已经提到了一些最流行的平台,比如 Windows 2000、Linux、Solaris 和 MacOS。大多数平台可以被描述为操作系统和硬件的组合。Java 平台与大多数其他平台的不同之处在于,它是一种运行在其他硬件平台上的纯软件平台。

Java 平台有两个组件：

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

我们已经介绍了 Java VM。它是 Java 平台的基础，已经被移植到了各种硬件平台上。

Java API 是预先建立的软件组件的大型集合，它们提供许多有用的功能，比如图形用户界面 (graphical user interface, GUI) 部件。Java API 被分组为相关类和接口的库，这些库称为包 (package)。下一节将说明 Java API 中的一些包提供的功能。

图 1-3 描述了一个在 Java 平台上运行的程序。Java API 和虚拟机将程序与硬件隔离开。

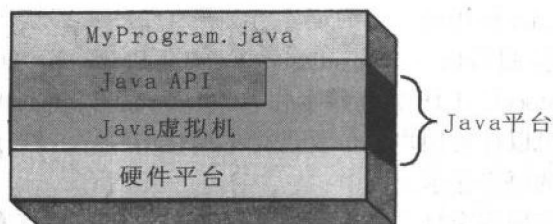


图 1-3 Java API 和 Java VM 将程序与硬件隔离开

本机代码是编译之后的在特定硬件平台上运行的代码。作为一种与平台无关的环境，Java 平台会比本机代码慢一点儿。但是，智能编译器、良好调整的解释器和即时 (just-in-time) 字节码编译器可以将性能提高到接近本机代码的程度，而不影响它的可移植性。

1.1.3 Java 技术可以做什么

用 Java 编程语言写的程序最常见的类型是 applet 和应用程序。如果你曾经在 Web 上冲浪，那么可能熟悉 applet。applet 是一种遵从某种约定、可以在支持 Java 的浏览器中运行的程序。要想看看运行中的 applet，请访问本书的在线版本中的以下页面：

<http://java.sun.com/docs/books/tutorial/getStarted/index.html>

在这里，你可以看到一个动画，Java 平台的吉祥物 Duke 正在向你挥手：



但是，Java 编程语言不只用于为 Web 编写可爱、有趣的 applet。通用而高级的 Java 编程语言还是一种强大的软件平台。通过使用丰富的 API，可以编写许多类型的程序。

应用程序是直接在 Java 平台上运行的独立程序。一种称为服务器 (server) 的特殊应用程

序为网络上的客户服务。服务器的例子有 Web 服务器、邮件服务器和打印服务器。

另一种特殊的程序是 servlet。servlet 差不多可以被看作在服务器端运行的 applet。Java servlet 的流行用途是建立交互式的 Web 应用程序，以替代 CGI 脚本。servlet 与 applet 的类似之处在于它们都是应用程序的运行时扩展。但是 servlet 不是在浏览器中运行，而是在 Java Web 服务器中运行，用于配置或调整服务器。

API 如何支持所有这些程序类型呢？它是通过提供大量功能的软件组件包来实现的。Java 平台的每种完整实现都提供了以下功能：

- 基本内容：对象、字符串、线程、数字、输入和输出、数据结构、系统属性、日期和时间等等。
- Applets：由 Java applet 使用的一组约定。
- 连网：URL、传输控制协议（Transmission Control Protocol, TCP）、用户数据报协议（User Datagram Protocol, UDP）套接字和 IP（Internet Protocol）地址。
- 国际化：帮助编写可以针对用户的位置进行本地化的程序。程序可以根据特定的地区自动调整并以适当的语言显示。
- 安全：包括低层和高层安全性，如电子签名、公共密钥和私有密钥管理、访问控制和证书。
- 软件组件：称为 JavaBeans 的组件可以插入现有的组件体系结构。
- 对象串行化（Object serialization）：通过远程方法调用（Remote Method Invocation, RMI）支持轻型的持久性和通信。
- Java 数据库连接（Java Database Connectivity, JDBC）：提供对各种关系型数据库的统一的访问。

Java 平台还有用于 2D 和 3D 图形、可访问性、服务器、协作、电话、语音、动画等的 API。图 1-4 说明了 Java 2 SDK 中包含的内容。

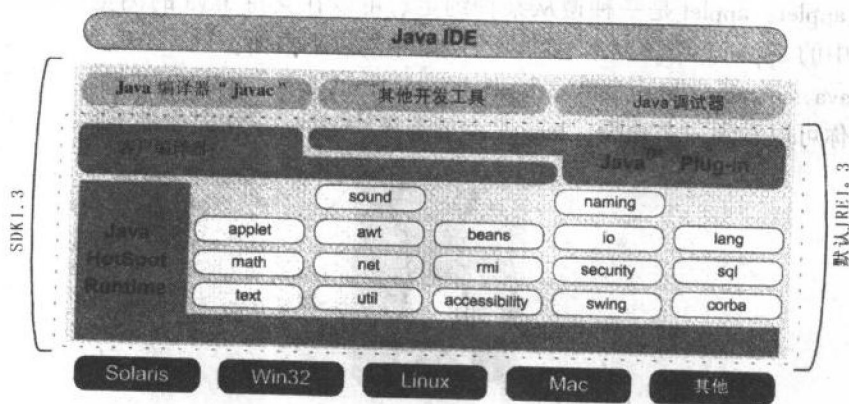


图 1-4 Java 2 SDK, Standard Edition v.1.3。Java 2 运行时环境（Java 2 Runtime Environment, JRE）包含 Java VM、Java 平台核心类和支持文件。Java 2 SDK 包括 JRE 和开发工具，比如编译器和调试器

本书讨论的 Java 编程语言是初级到中级程序员最经常使用的一部分核心 API。如果你需要的其他信息没有包含在本书中,那么可以研究另外两本书:《The JFC Swing Tutorial》和《The Java Tutorial Continued》。这两本书的内容可以在本书附带的光盘中找到,也可以从在线指南中得到:

<http://java.sun.com/docs/books/tutorial/index.html>

1.2 Java 技术将如何改变我们的生活

我们不能保证如果你学习了 Java 编程语言就能够得到荣誉、财富,甚至工作。但是它很可能使你的程序更好,而且你需要付出的努力比使用其他语言时少。我们相信 Java 编程语言会给你带来以下好处:

- 入门迅速: 尽管 Java 编程语言是一种强大的面向对象的语言,但是它容易学,特别是对于熟悉 C 或 C++ 的程序员。
- 编写更少的代码: 程序度量(比如类的数量、方法的数量等等)的对比说明,用 Java 编程语言写的程序比用 C++ 写的相同程序小四倍。
- 编写更好的代码: Java 编程语言鼓励使用更好的编程方法,而且它的垃圾收集功能有助于避免内存泄露。它的面向对象性质、JavaBeans 组件体系结构以及丰富的 API 使你重新使用其他人的代码并减少 bug。
- 更快地开发程序: 开发速度可能比用 C++ 编写相同的程序快一倍。为什么? 使用 Java 编程语言时写的代码更少,而且它比 C++ 简单。
- 以 100% 纯 Java 避免平台相关性: 通过避免使用用其他语言写的库,可以保持你的程序的可移植性。100% 纯 Java 产品认证程序有一个历史性的过程手册、白皮书、小册子以及类似资料的存储库,网址为 <http://java.sun.com/100percent/>。
- 一次编写,处处运行: 因为 100% 纯 Java 程序被编译为与机器无关的字节编码,所以它们可以在任何 Java 平台上以一致的方式运行。
- 更容易地发布软件: 可以很容易地从一个中心服务器升级某类程序(比如 applet)。applet 利用了允许动态(on the fly)装载新类而不必重新编译整个程序这个特性。

让我们从一个简单的程序“Hello World”开始学习 Java 编程语言。根据你使用的平台,你将需要阅读以下三节之一: 1.3 节“第一步 (Win32)”给出在 Windows 平台上编译你的第一个程序的详细说明, 1.4 节“第一步 (UNIX/Linux)”给出对于 UNIX 和 Linux 平台的说明, 1.5 节“第一步 (MacOS)”讨论 MacOS 平台。然后,不要错过 1.6 节“分析 Hello World”,其中解释了 Hello World 程序所演示的一些 Java 编程语言的基本特性。

1.3 第一步 (Win32)

下面的详细说明帮助你编写第一个程序。这些说明适用于 Win32 平台上的用户,包括 Windows 95/98 和 Windows NT/2000 (针对 UNIX 和 Linux 的说明见 1.4 节, MacOS 平台上的用户可以参考 1.5 节)。我们先提供编写第一个程序所需的内容检查表。接下来,讨论创建应

用程序的步骤、创建 applet 的步骤以及对你可能遇到的错误消息的解释。

1.3.1 检查表

要编写你的第一个程序，需要：

1) Java 2 SDK, Standard Edition: 本书附带的光盘上包含 Java 2 SDK 软件。你可以将这个平台下载到你的计算机，或者访问 <http://java.sun.com/products/> 以寻找最新的版本^①。

2) 文本编辑器：在这个例子中，我们将使用记事本（Notepad），这是 Windows 平台上包含的简单的文本编辑器。要启动记事本，请进入 Start 菜单并选择 Programs > Accessories > Notepad。如果你使用另一种文本编辑器，那么参见下面的说明。

注意 你可能会考虑使用一个 IDE 协助你编写程序。Java 2 SDK, Standard Edition v.1.3 捆绑了一个 IDE，Forte for Java, Community Edition。本书的光盘中包含这个版本。

1.3.2 创建第一个应用程序

程序 HelloWorldApp 将显示一句问候语“Hello World!”。要创建此程序，应该完成以下步骤：

- 创建源代码文件：源代码文件包含用 Java 编程语言写的文本，你和其他程序员可以理解这些代码。可以使用任何文本编辑器创建并编辑源代码文件。
- 将源代码文件编译为字节编码文件。编译器得到你的源代码文件并将文本翻译为 Java VM 可以理解的指令。编译器将这些指令转换为一个字节编码文件。
- 运行字节编码文件中包含的程序：你的计算机上安装的 Java 解释器实现了 Java VM。这个解释器得到你的字节编码文件，并将指令翻译为你的计算机可以理解的指令来运行它们。

1. 创建源代码文件

要创建源代码文件，有两种选择。你可以将文件 HelloWorldApp.java^② 保存到你的计算机上，这样可以避免进行大量的输入工作。然后直接进入第 2 步“编译源代码文件”。或者按以下这些指示办。

首先，启动记事本。在新文档中输入以下代码：

```
/**
 * The HelloWorldApp class implements an application that
 * displays "Hello World!" to the standard output.
 */
public class HelloWorldApp {
```

① 在版本 1.2 之前，Sun Microsystems 公司提供的 Software Development Kit (SDK) 被称为“JDK”，即 Java Development Kit。

② 在这个例子中，HelloWorldApp.java 可以从光盘和网上得到。文件在光盘和网上的位置参见 1.8 节“代码样例”。

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

输入代码时要小心 准确地输入这里的所有代码、命令和文件名。编译器和解释器是大小写敏感的，所以大小写必须保持一致。换句话说，HelloWorldApp 不等于 loworldapp。

然后，将此代码保存到一个文件。从菜单栏选择 File > Save As。在 Save As 对话框中进行如下操作：

- 使用 Save in 下拉列表指定保存文件的文件夹（目录）。在这个例子中，文件夹是 C 驱动器上的 java 文件夹。
- 在 File name 文本框中，输入“HelloWorldApp.java”，要包括双引号。双引号迫使文件被保存为 .java 文件，而不是“.txt”文本文件。
- 从 Save as type 下拉列表选择 Text Document。

完成时，此对话框应该与图 1-5 一样。现在点击 Save 按钮并退出记事本。

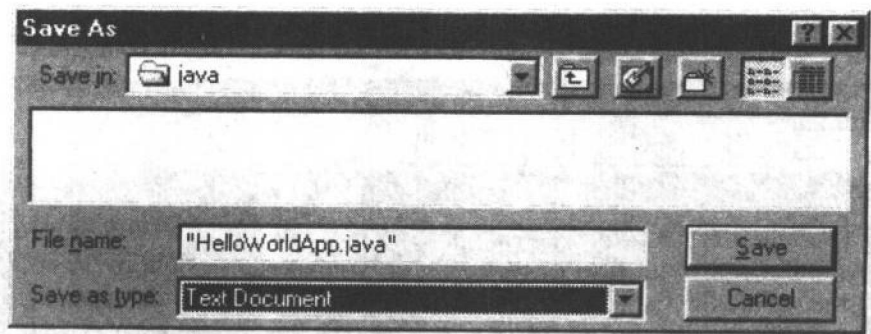


图 1-5 以正确的 .java 扩展名保存 HelloWorldApp.java 文件

2. 编译源代码文件

从 Start 菜单选择 MS - DOS Prompt (Windows 95/98) 或 Command Prompt (Windows NT/2000) 应用程序。应用程序启动时，界面如图 1-6 所示。

提示显示当前目录。当你在 Windows 95/98 上打开 MS - DOS 提示时，当前目录常常是 C 驱动器上的 WINDOWS (如图 1-6 所示)；对于 Windows NT 是 WINNT。要编译你的源代码文件，应该将当前目录改变为你的文件所在的目录。例如，如果你的源代码目录是 C 驱动器上的 java 目录，那么在提示下输入以下命令并按回车：

```
cd c:\java
```

现在，提示应该改变为 C: \ java。要改变为另一个驱动器上的目录，就必须多输入一个