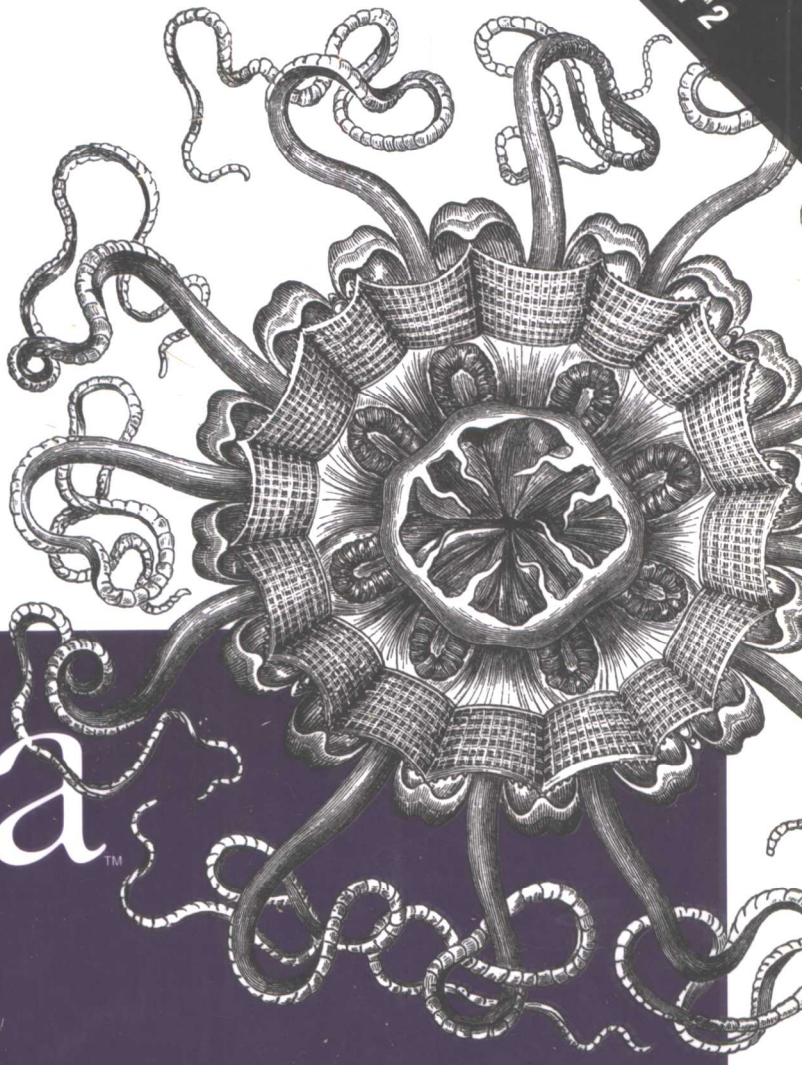
*Java Threads*

第二版
涵盖 Java™ 2



# Java™ 线程

*Scott Oaks & Henry Wong* 著
黄若波 程峰 译

O'REILLY®
中国电力出版社

# Java™线程

## 第二版

*Scott Oaks & Henry Wong* 著

黄若波 程峰 译

O'REILLY®

*Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo*

O'Reilly & Associates, Inc. 授权中国电力出版社出版

中国电力出版社

**图书在版编目（CIP）数据**

Java线程/（美）欧克（Oaks, S.），（美）王（Wong, H.）著；黄若波等译．-北京：中国电力出版社，2002

书名原文：Java™ Threads, second edition

ISBN 7-5083-1318-6

I. J... II. ①欧... ②王... ③黄... III. JAVA语言－程序设计 IV. TP312

中国版本图书馆CIP数据核字（2002）第110530号

北京市版权局著作权合同登记

图字：01-2002-2781号



书　　名/ Java线程（第二版）
书　　号/ ISBN 7-5083-1318-6
责任编辑/ 宋宏
封面设计/ Emma Colby、张健
出版发行/ 中国电力出版社（www.infopower.com.cn）
地　　址/ 北京三里河路6号（邮政编码100044）
经　　销/ 全国新华书店
印　　刷/ 北京市地矿印刷厂
开　　本/ 787毫米×1092毫米　16开本　22印张　319千字
版　　次/ 2003年5月第一版　2003年5月第一次印刷
印　　数/ 0001-5000册
定　　价/ 39.00元（册）

# O'Reilly & Associates公司介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O'Reilly & Associates公司授权中国电力出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly & Associates公司是世界上在UNIX、X、Internet和其他开放系统图书领域具有领导地位的出版公司，同时是联机出版的先锋。

从最畅销的《The Whole Internet User's Guide & Catalog》（被纽约公共图书馆评为二十世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly & Associates一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly & Associates是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly & Associates公司具有深厚的计算机专业背景，这使得O'Reilly & Associates形成了一个非常不同于其他出版商的出版方针。O'Reilly & Associates所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly & Associates还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly & Associates依靠他们及时地推出图书。因为O'Reilly & Associates紧密地与计算机业界联系着，所以O'Reilly & Associates知道市场上真正需要什么图书。

# 目录

# 前言

1995年冬天，所有程序员都注意到了Sun公司发布的第一个Java™ alpha版。Java有很多吸引程序员的特性（不仅仅限于Sun公司对Java的宣传）：Java是健壮的、安全的、与体系结构无关的、可移植的、面向对象的、简单的和多线程的。对于大多数程序员而言，最后两点看起来是矛盾的：一种多线程的语言能够简单吗？

实际上Java的线程系统是简单的，至少和其他的线程系统相比是这样的。Java线程系统的简单性使得它相当容易学习，即使不熟悉线程的程序员也可以轻松地掌握线程编程的基础知识。但是这种简单性是要付出代价的：Java中没有其他线程系统中的高级功能。但是程序员可以使用Java提供的简单功能来实现这些高级功能。这也正是本书的主题：如何使用Java提供的线程工具来进行简单的线程编程，以及如何在更复杂的程序中通过扩展这些工具来完成更高级的任务。

## 本书的读者对象

本书适合于那些希望学习在Java程序中使用线程的各种级别的程序员阅读。前几章讨论使用Java进行线程编程的问题，在这几章中，假设程序员不具有线程编程经验，因此处于基本级别。后面几章的级别要高一些，这是由此时使用的材料和对程序员经验所做的假设而决定的。那些没有线程编程经验的程序员应该按自然顺序阅读。

这种方法模拟了Java自身以及Java书籍的发展过程。早期的Java程序尽管有效，但是趋向于简单化：在Web页面上一个可以跳舞的动画效果是展示Java能力的最好广告，但是只表现了Java的皮毛。同样，关于Java的早期书籍也是趋向于提供对Java的全面说明，而只用一到两章来讲述Java的线程系统。

本书属于另一种类型的Java书籍：它仅仅关注一个主题，因此可以详细地解释Java线程系统的细节。本书读者的目标应该是编写更复杂的程序，以完全使用Java线程系统的能力。

尽管本书使用的材料不需要你原来就了解线程，但还是假设读者对于Java API方面的其他知识有一定的了解，并且可以编写简单的Java程序。

# 本书使用的版本

在Internet时代编写一本关于Java的书十分困难：书的基石是在不停地变化的。因此我们要选定一个基石，而这个基石就是Sun公司的JDK™ 2。也许在本书中没有包含Java 2以后的版本对于线程系统的一些改变。在本书中我们也会随时指出Java 2和以前版本的区别，因此使用Java 2以前版本的程序员也可以使用本书。

一些Java（包括嵌入浏览器中的和作为开发系统的）提供商都在为向Java线程系统提供一些附加的功能（这类似于我们在第五章到第八章通过Java线程系统的基本技术来提供一些附加功能）而互相竞争。这些扩展超出了本书的讨论范围：我们只关心Sun公司的JDK 2。只有当JDK在Unix平台和Windows平台上不一样时，我们才考虑这些平台的区别。我们将在第六章讨论在这些平台上Java线程调度的不一致性。

# 本书的组织结构

下面是本书的大纲，通过它可以了解本书的素材组织。在附录中讨论的问题要么是还不够成熟，要么是主要具有学术意义，只在极少数情况下才有用。

第一章，线程简介

本章介绍了线程的概念以及本书要使用的术语。

第二章，Java线程API

本章介绍了用来创建线程的Java API。

第三章，同步技术

本章介绍了一种简单的锁定机制，Java程序员可以用它来同步对数据和代码的访问。

第四章，等待和通知

本章介绍另外一种机制，Java程序员可以用它来同步对数据和代码的访问。

第五章，Java线程编程的例子

本章总结了前面几章的技术。不同于上面几章的是，本章是面向解决方案的：提供的例子教你如何将已经学到的基本线程技术综合起来使用，并且告诉你如何使用线程来有效率地进行设计。

第六章，Java线程调度

本章介绍了如何使用Java API来控制虚拟机对线程的调度，其中包括在虚拟机的不同实现中不同调度之间的区别。

第七章，Java线程调度例子

本章提供了扩展Java调度模型的例子，其中包括提供循环调度和线程池的技术。

第八章，和同步相关的高级主题

本章讨论了一些和数据同步相关的高级主题，其中包括和死锁相关的设计以及其他一些同步类（包括Java不直接提供但是其他平台具有的同步方法）。

第九章，多处理器机器上的并行化

本章讨论了如何在具有多处理器的机器上设计程序，使得其可以充分利用该机器的能力。

第十章，线程组

本章讨论了Java的ThreadGroup类。该类允许程序员控制和操作一组线程。Java中和线程相关的安全机制也是基于这个类的，因此也在本章进行了讨论。

附录一，其他主题

本附录包括了Java API中很少有人感兴趣的一些方法：处理线程堆栈的方法以及ThreadDeath类。

附录二，异常和错误

本附录说明了由线程系统使用的异常和错误的细节信息。

# 排版约定

等宽字体（constant width）表示：

- 代码例子：

```
public void main(String args[]) {
    System.out.println("Hello, world");
}
```

- 正文中的方法、变量、参数名以及关键字。

等宽黑体（**bold constand width**）表示：

- 为解决问题而对程序做出的修改：

```
public void main(String args[]) {
    System.out.println("Hello, world");
}
```

- 在一大段代码中重点表示的部分。

斜体（*italic*）表示URL或者文件名，也表示新引入的术语。

例如，本书的例子可以从如下地址下载：

*http://www.oreilly.com/catalog/jthreads2*

# 作者联系方法

我们尽量保证了本书的完整性和精确性。但是Java规范的改变、在不同平台上厂商实现的不同以及底层操作系统的不同使得我们不可能保证所有的例子都是完全正确的（这并不包括可能的文字错误）。因为Java也是在不停变化的，所以本书也是在不停前进的。

作者欢迎你对此书提出反馈意见，特别是指出错误和遗漏之处。你可以通过以下的email地址与我们联系：

*scott.oaks@sun.com*
*henry.wong@sun.com*

# 建议与评论

本书的内容都经过测试，尽管我们做了最大的努力，但错误和疏忽仍然是在所难免的。如果你发现有什么错误，或者是对将来的版本有什么建议，请通过下面的地址告诉我们：

美国：

O'Reilly & Associates,Inc.
101 Morris Street
Sebastopol, CA 95472

中国：

100080 北京市海淀区知春路 49 号希格玛公寓 B 座 809 室
奥莱理软件（北京）有限公司

询问技术问题或对本书的评论，请发电子邮件到：

*info@mail.oreilly.com.cn*

最后，您可以在WWW上找到我们：

*http://www.oreilly.com*
*http://www.oreilly.com.cn*

# 致谢

正如读者所知，仅靠作者是不可能完成一本书的。在此我们对如下人员所提供的帮助和鼓励表示深深的感谢。

Michael Loukides，你肯定了我们的想法，并且带领我们走过构思的阶段。David Flanagan，感谢你对我们草稿的有价值的反馈。Hong Zhang，感谢你在Windows线程方面的帮助。Reynold Jabbour和Wendy Talmont，感谢你们的支持。

另外，还要感谢我们的家人。James，你在Scott完成本书的过程中给予了支持和鼓励；Nini，感谢你在Henry编写本书时给他10%的时间独处，并且在其他的时间里给予他鼓励：感谢你所做的一切。

最后，我们还要感谢那些对第一版提出建议的读者。我们已经尽力回答你们提出的每一个问题。

本章内容：

- Java 术语
- 线程概述
- 为什么要使用线程
- 总结

# 第一章

# 线程简介

本书介绍的是如何在Java编程语言和Java虚拟机中使用线程。在Java中，线程这一主题是很重要的。这种重要性从Java语言中内置了大量与线程相关的特性就可以看出，并且线程系统的另外一些特性也是Java虚拟机所必需的。线程是使用Java时一个不可分割的部分。

线程不是什么新概念：在一段时间里，许多操作系统都通过提供的库来为C语言程序员提供创建线程的机制。而其他一些语言（如Ada）与Java一样，将对线程的支持内置到语言内部。但是，作为在特定的编程情况下才被用到的技术，线程一直被看成是一个非主流的编程主题。

在Java中，情况是完全不一样的。如果不引入有关线程的技术，则只能编写最简单的Java程序。许多原来用C和C++编程的程序员可能从来没有考虑过学习线程，但随着Java的流行，他们也需要熟练地掌握线程编程了。

## Java 术语

首先，我们要定义一些本书使用的术语。在不同的资料中，许多和Java相关的术语是不一致的；在本书中，我们将尽量一致地使用这些术语。

*Java*

第一个术语是*Java*本身。正如我们所知，Java最初是作为一种编程语言出现的，因此许多人现在还认为Java仅仅是一种编程语言。但是Java不仅仅是一种编程语言：它还是API（Application Programming Interface，应用编程接口）规范和虚拟机规范。所以当我们提到Java时，我们所指的是整个Java平台。它是由Java编程语言、Java API和Java虚拟机规范组成的一个完整的编程和运行时环境。在很多情况下，当提到Java时，我们可以通过上下文清楚地分辨出所指的是Java编程语言、Java API还是Java虚拟机。要注意的是，本书所讨论的线程特性直接依赖于完整Java平台的所有组成部分。尽管现在已经可能将Java程序直接编译成汇编语言并且脱离Java虚拟机运行，但本书中要讨论的程序与这种可以直接运行的程序不同。

*虚拟机、解释程序和浏览器*

Java虚拟机（virtual machine）是Java解释程序（interpreter）的另一种说法。Java解释程序是通过解释Java编程语言定义的中间字节码来运行Java程序的。Java解释程序的三种最常见的形式是：程序员使用的解释程序（*java*），它通过命令行或者文件管理器来运行程序；最终用户使用的解释程序（*jre*），它是编程环境的子集，同时也是构成Java插件的基础。内置到许多流行的浏览器［例如Netscape Navigator、Internet Explorer、HotJava™以及JDK(Java Developer's Kit,Java开发者工具包)中包含的appletviewer］中的解释程序。所有这些形式的Java解释程序都仅仅是Java虚拟机的实现，因此我们用Java虚拟机来任指它们中的一个。术语“Java解释程序”特指那些通过命令行独立运行的Java虚拟机(其中也包括那些实行即时编译的虚拟机)；而术语“支持Java的浏览器”（或者更简单地说是“浏览器”）专用于对那些内置于Web浏览器里的虚拟机的讨论中。

在大部分情况下，至少从理论上来说，这些虚拟机是一致的。但是实际上，在这些虚拟机的不同实现之间，还是存在着一些很重要的差别，其中一个差别就来自于线程。在某些情况下，这些差别是很重要的。我们将在第六章中讨论这些差别。

*程序、应用程序和applet*

这些术语都是用来定义用Java编写的程序的。一般而言，它们被统称为程序。但是Java程序可以分成两种类型：一种是直接在Java解释器中运行的，另一种是在支持Java的浏览器中运行的（注1）。在绝大部分时间里，这两种类型的程序的差异是无关紧要的。此时，我们就用程序来称呼它们。但是当它们之间的区别很重要的时候，我们用术语“应用程序”来称呼那些独立运行的Java程序，而将那些在支持Java的浏览器中运行的Java程序称为applet。对于线程而言，应用程序和applet的不同仅仅体现在它们的Java安全模型方面。我们将在第十章讨论Java安全模型和Java线程之间的交互。

# 线程概述

现在就剩下一个术语没有定义了：线程到底是什么？线程（thread）其实是控制线程（thread of control）的简写。而控制线程，简单地说，就是在一个程序中与其他控制线程无关的能够独立运行的代码片段。

## 控制线程

控制线程听起来是一个复杂的技术术语，但其实是一个简单的概念。它是程序运行时的路径。它决定将要执行什么代码：是if块还是else块？while循环到底要运行多少次？如果我们从一个“to do”列表中取出任务来执行，就像计算机运行一个应用程序那样，则我们的执行步骤以及执行的顺序就是执行路径。而这个执行路径就是控制线程运行的结果。

有多个控制线程就如同从两个“to do”列表中执行任务一样。我们仍然以正确的顺序运行每一个“to do”列表中的任务，但是当我们对其中一个列表中的任务感到厌烦时，可以切换到另外一个列表中去运行，并且当我们在以后返回到第一个列表中时，可以回到刚才离开的地方继续执行。

注1：尽管可以写一个可以同时在解释器和浏览器中运行的Java程序，但是当它们实际运行时，这些区别还是适用的。