

Beyond Java



超越 Java™

探讨程序语言的未来

O'REILLY®

東南大學出版社

Bruce A. Tate 著

O'Reilly Taiwan公司 编译



超越 Java

程序程序語言的本質

O'REILLY

專業技術出版社

Robert A. Van Der
Linden

with Emily Rosemond and Scott
Wasserman

超越 JavaTM

Bruce A. Tate 著
O'Reilly Taiwan 公司 编译

O'REILLY®

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

O'Reilly Media, Inc. 授权东南大学出版社出版

东南大学出版社

图书在版编目 (CIP) 数据

超越 Java™/ (美) 泰特 (Tate, B. A.) 著; O'Reilly
Taiwan 公司编译. —南京: 东南大学出版社, 2007.1
书名原文: Beyond Java™
ISBN 978-7-5641-0639-3

I. 超... II. ①泰... ②O... III. Java 语言—程序设计
IV. TP312

中国版本图书馆 CIP 数据核字 (2006) 第 155200 号

江苏省版权局著作权合同登记

图字: 10-2006-218 号

©2005 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Southeast University Press, 2006. Authorized translation of the English edition, 2005 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2005。

简体中文版由东南大学出版社出版 2006。英文原版的翻译得到 O'Reilly Media, Inc. 的授权。此简体中文版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有, 未得书面许可, 本书的任何部分和全部不得以任何形式重制。

书 名 / 超越 Java™

责任编辑 / 张烨

封面设计 / Ellie Volckhausen, 张健

出版发行 / 东南大学出版社

地 址 / 南京四牌楼 2 号 (邮编 210096)

印 刷 / 扬中市印刷有限公司

开 本 / 787 毫米 × 980 毫米 16 开本 13.25 印张 223 千字

版 次 / 2007 年 1 月第 1 版 2007 年 1 月第 1 次印刷

印 数 / 0001-4000 册

书 号 / ISBN 978-7-5641-0639-3/TP · 104

定 价 / 29.00 元 (册)

O'Reilly Media, Inc. 介绍

为了满足读者对网络 and 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 Media, Inc. 授权东南大学出版社, 翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media, Inc. 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 Media, Inc. 一直处于 Internet 发展的最前沿。

许多书店的反馈表明, O'Reilly Media, Inc. 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 Media, Inc. 具有深厚的计算机专业背景, 这使得 O'Reilly Media, Inc. 形成了一个非常不同于其他出版商的出版方针。O'Reilly Media, Inc. 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 Media, Inc. 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 Media, Inc. 依靠他们及时地推出图书。因为 O'Reilly Media, Inc. 紧密地与计算机业界联系着, 所以 O'Reilly Media, Inc. 知道市场上真正需要什么图书。

作者简介

Bruce A. Tate 是一位泛舟者、越野车骑士、父亲、作家及 Java 程序员，住在德克萨斯州的奥斯汀。他写了 5 本书，包括获得 Jolt 奖的《轻快的 Java》(O'Reilly 出版) 以及热卖的《Bitter Java》(Manning 出版)。他有 17 年的工作经验，包括在 IBM 工作、两次失败的创业以及开办了自己的叫做 J2Life, LLC 的独立顾问公司。

封面介绍

《超越 Java》的封面动物是一只猫鼬 (bassaris)。猫鼬是北美肉食动物，出现在墨西哥、德克萨斯州、加州。体积为家猫大小，外形接近浣熊和狐狸。

这种棕褐毛的动物有着黑白相间的尾巴，而尾巴和身体一样长。尾巴的长度可以让它在通过窄岩壁和大树枝时保持平衡，甚至可以横翻筋斗转 180 度。它的后脚可以转动 180 度，使它相当擅长在峭壁和树木中行走。

猫鼬是夜行动物，不具有侵略性。它居住在洞穴、岩缝以及树洞中，也曾在废弃的房子内、有人居住的房子的阁楼中被发现过。它会去露营地乱翻东西，有时候还会把东西带走，特别是亮晶晶的东西。它是敏捷的攀爬者，可以轻易地穿越树木和岩石。

猫鼬猎食的时间主要是晚上，对象主要是小型鸟类、啮齿类、蜥蜴、蛇、无脊椎动物以及水果，偶尔也食腐尸。水果是主食，这可以减少对水分的摄取。

某些地区会为了皮毛而猎捕猫鼬。它也常被驯为宠物，特别是在墨西哥的某些地区。它有一些不同的俗名，包括山猫、civit cat，墨西哥称其为 cacomixl。它的学名 bassaris 起源于希腊的狐狸。在希腊神话中，Dionysus (酒神) 穿着一件猫鼬，象征着新生命。

目录

前言	1
第一章 猫头鹰与鸵鸟	7
忽略是一种美德	8
水煮青蛙	10
新的水平线	16
预告	19
第二章 完美的风暴	20
风暴警告	20
C++ 的经验	24
拨云见日	29
暴怒的释放	32
事过境迁	38
向前走	38
第三章 皇冠上的宝石	40
语言和 JVM 的设计	42
因特网	46
企业集成	49

社区	51
打破迷思	53
第四章 打破玻璃	57
Java 的新工作描述	58
基本的 Java 限制	62
类型	64
基本类型	73
分手时的刻薄话	75
为什么不直接修改 Java?	78
第五章 游戏规则	80
Java 提高竞争门槛	81
企业集成	85
产生话题	87
语言特色	92
几个有潜力的语言	97
第六章 Ruby 简介	102
关于 Ruby	103
应用某些结构	111
Breaking It Down	125
第七章 Ruby on Rails	127
数字游戏	127
Rails 范例	133
查看内部	142
精华	145
第八章 延续服务器	149
问题	149
延续	152

延续服务器	156
Seaside	161
一个 Seaside 范例	164
那又怎样?	168
第九章 竞争者	170
主要竞争者	171
较小的竞争者	182
“下一个大东西”	188
索引	191

前言

2005年3月，我惊喜地收到通知，我写的一本《Better, Faster, Lighter Java》获得 Jolt Award 震撼奖。我在那本书中告诉 Java 开发者如何建立更好、比以往更快的程序。那本书在我心中的特殊地位将永远不会被取代。但是，这一路下来，有些新东西出现了，我必须动手去接触、去了解。

在这期间，我的一个顾客正在建立一个应用程序，包含复杂数据库模式 (schema) 以及基于 Web 的用户界面。我们用了 Spring 和 Hibernate 搭配 Web Work，对于轻量级的 Java 开发来说，这是很常见的组合，我们一般也算满意。但是有些事情困扰着我们：重复的事情太多、XML 配置 (configuration) 激增、改变的步调太快。一时兴起，我们试了 Ruby on Rails，这是一个高生产力、创新的框架，它正在席卷非 Java 的社区，而且也为 Java 的架构带来一些争论。使用 Rails 所获得的高生产力让我们感到震惊，所以后来我们把整个计划都搬过去了。

有些东西和我一拍即合。对于这种应用程序来说，Java 是个碍手碍脚的东西。将 Java 排除，我可以把程序代码减少到原来的 1/4，把 XML 减少到只剩十分之一，达到令人惊讶的生产力以及不错的性能。更棒的是，《Better, Faster, Lighter Java》(简称 BFLJ) 一书中所提到的概念依然可以在此采用。对于其他项目，如果需要 Java 所提供的社区和工具，我会用它来取代；如果不需要 Java，我可以把 BFLJ 的原则发挥到极致。我内心的水坝已经溃堤，思潮通过这本书倾泻而出，我要告诉大家我的想法。

几个月后，我找到了听众，在数千里外，离我家有19个小时的路程。在Java用户社区的面前，我感到局促不安，我在更大的场面中演讲过，但是这趟旅程却很不一样。这次挪威Java用户社区帮我支付奥斯陆（挪威首都）之行的花费，他们却无法在会场中听到我歌颂Spring、赞扬Hibernate、褒奖敏捷开发（Agile Development），而是会听到我说出“他们挚爱的宝贝很丑”，我是否要实话实说，我陷入了困境。Java为我带来许多好处：写了畅销书、得到Jolt奖、在不景气的经济形势下还能有相当不错的咨询业务，我当然希望Java列车继续开下去、不要停歇、永远根基稳固，我希望Java让我的生产力直上云霄，希望无尽的群众和脑力能解决Java今日所面对的所有棘手的问题，但是世间没有任何东西可以永恒维持。

在我的演说中，我先陈述Java成功的理由，然后论及它的严重问题，展示了一些可供选择的语言和框架。在整个演说中，我指出时机已经成熟，该是另一个技术出现的时候了。当我对这群好客的人演说时，我回答问题，也观察人的表情，一些人看起来具有敌意，有的人则是很受伤，但是大部分的人则是很能理解，或者有一些恐惧。他们了解我的中心想法。对于我们惯用Java来做的事，有些其他语言的框架可以做得更好。在某些情况下，生产力的差异实在太太大，值得我们好好地研究新的技术。

这场演说以及问答超时很长时间，但是没有人离席，令我惊讶的是，他们都能接受。演说完毕后，我们出去参观奥斯陆，一个具有敌意的听众几乎整晚都围绕着我，抛出一个又一个困难的问题：

- 为何我们不能改变Java以修正这些缺失？
- 你所展示的其他语言和框架有没有足够的商业支持？
- 它们支持分布式事务、Web Service、XML吗？
- 怎样找到有这种技术的程序员？又该如何培训自己的程序员？

这些问题都是真正需要注意的问题，这些问题也显示出进入新语言时必须跨越的重重障碍。问问题的人是位绅士，但是他无法完全隐藏自己烦躁不安的情绪以及根深蒂固的思想，他认为进入下一个成功的语言需要付出相当高的代价，并认为在可见的将来我们还是会继续使用Java写程序。他可能没错，但是我已经开始发现Java语言的局限性和许多框架的缺点。在某些问题的解决上，Java已经不再具有高生产力了，我改用一些其他的技术，且获得了成

功。虽然语言可以存活大半世纪以支持旧系统，但是我知道没有语言可以永远保持领先地位，Java统治的时代将会结束，这不是“会不会”的问题，而是“什么时候”的问题。

谁需要读这本书？

当C++褪色不再让人注意，我的许多好友都伤得很重。他们没有注意到，空气中已经有了不同的气息，巨大的改变瞬间来到。虽然我写这本书会失去许多，但是我写这本书的目的是希望这样的事情不要再发生。如果你不希望在一觉醒来之后忽然发现自己已经被淘汰了，那么你需要读这本书。

如果你认同我的说法，你可以开始试着建立你自己的技能。你可以下载一些我讨论到的框架，学习一些语言，这本书会告诉你成功的语言所具备的要点。如果我很幸运，找到了些未来可能的赢家，那么以后改朝换代时你就不會措手不及了。

如果你不认同我的说法，你可以用最好的语言、最好的框架、最好的技术来改进你今日用Java在做的事。新的框架像PHP、C Omega for .NET、Ruby on Rails 将会偶尔出现在你的周围，你需要知道它们，了解如何评估它们。

所以，不管你是否认同我的说法，你都是赢家。该是注意的时候了，该是看着水平线的时候了，要看得比Java更远，超越Java。

排版约定

由于本书保留了部分术语的原貌，为了避免读者的混淆，我们运用了一些字体变化，让读者能轻易辨别词汇本身的含义。

等宽字 (*Constant width*)

用于范例代码、类名、方法名及对象。

等宽斜体字 (*Constant width italic*)

用于指示程序中应以实际值替换的项目。

斜体字 (*Italic*)

用于文件名、目录、强调重点及第一次使用到的术语。

等宽黑体字 (Constant width bold)

用于文本中的用户输入以及范例中显示的输入及输出。也用于在代码中强调重点以及指示有注释的文本块。

某些有缩写的术语在后文再次出现时，会视情况以缩写出现，不一定会使用较长的中文译词。此外，若是在内文中列举多个项目时，为了醒目起见，会以粗体字来表示。

使用程序代码范例

帮助你完成工作是本书的目的。一般来说，你可以在你自己的程序和文档中使用本书的程序代码，你不需要先联系 O'Reilly 公司，除非是明显进行重制的行为。例如，写程序的时候使用到本书的片段并不需要我们的许可，销售本书的范例光盘就需要得到我们的许可。回答问题时引用书中的范例也不需要我们的许可，但是在你的文档中大量使用本书的程序代码就需要我们的许可。

在线资源

与本书相关的信息，包括范例程序代码、相关资源链接、勘误表，都可在本书的中文网页取得：

<http://www.oreilly.com.cn/book.php?bn=978-7-5641-0639-3>

批评和建议

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 Media, Inc.
1005 Gravenstein Highway North
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进行评论，请发电子邮件到：

info@mail.oreilly.com.cn

最后，您可以在以下站点找到我们：

<http://www.oreilly.com>

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

第一章

猫头鹰与鸵鸟

我所认识的某些泛舟者都有死前的愿望，他们奋不顾身地投入 Class V（最高级）的泛舟活动。他们被水中的障碍物弄得翻船似乎只是早晚的事，障碍物会把船困住，而河水的力量会把船牢牢吸在水中。他们就像鸵鸟，把头埋进沙子中就以为危险不见了。

还有另一类泛舟者。当我开始泛舟时，我会侦察周围一切。我会待在 Class II+（初学者）的小波浪中仔细观察，并在开始前设置好安全绳索。我在河中的时间经常会不够，被迫跳进后面的区域，以在落日前完成这一切。现在，我很少离开船去侦察小急流。在某些地方，这并不实际，于是我改用“追击”的泛舟技巧（这是在美国东南部陡窄河流中发明出来的技巧），来增加我的机会。我不会用这种方式泛舟，因为我喜欢危险，事实上，我的直觉很准，可以感受到危险最可能出现的地方。我用这种方式泛舟，因为这让我把焦点集中在最需要的侦察上。这类泛舟者是猫头鹰。

结论是这样的：我常常忽略后果不严重的危险，或者发生频率很低的危险，因为把注意力放在这方面并不明智。适当地管理风险可能需要花相当多的精力、金钱、时间，且让我面临额外的风险。让我们回到猫头鹰和鸵鸟的问题上，正常情况下，这两者有很大的差别，但是偶尔猫头鹰会过度自信，或者对风险评估有了小错误，且说服他们自己没有先侦察就进行危险的事，这样的情况就曾经发生在我身上。我已经在这条小溪泛舟几百次了，但有一些状况会改变，像下雨后河面涨高。猫头鹰和鸵鸟只有一线之隔，有时候，难以分辨究竟是鸵鸟还是猫头鹰。身为一个泛舟者，即使我决定忽略某条河在某

种状况下的某种危险，有时候我还是会回头去评估风险。这就是这本书的主题。

忽略是一种美德

在许多方面，泛舟和编程很相似。我学了许多很棒的技巧，通过忽略大部分的问题，我可以具有超高的生产力。幸运的话，这些问题完全不会发生。这样的态度可以载舟亦可以覆舟。许多邮局职员以及最低薪资的快餐店员工也都学过，用这样的技巧对付他们的麻烦（也就是他们的顾客）是有效的，他们是鸵鸟。如果你靠近一点，你会发现一些选择性的、智能的“忽略应用程序”，这是猫头鹰的强项。我发现通常编程中最大的“问题”仅是潜在的问题，如果你读过我写的任何书，你就会知道我教导“不要太早对抗危险”，也教导响应一个受欢迎的敏捷守则 YAGNI, “You ain't gonna need it.”（你才不需要它）。我通常忽略那些保证会让我节省时间的吹牛的框架，我相信我的直觉，采用较简单的解决方案。

我发现 Java 能够做我需要做的一切事，所以我没有花太多时间看 Java 以外的东西。忽略是一种恩赐，我知道有些语言比较动态，并且可能更具生产力，但是最终，我还是认定 Java 会是赢家。它有太多框架可以用了，什么都可以做，从运行核反应堆系统到家电设备的嵌入式控制器，有些很棒的框架甚至还可以免费使用它们。我需要做什么事，都可以很容易找到适合的 Java 开发者。我知道人们已经用 Java 解决相当多的问题，知道采用 Java 可以让我的顾客觉得很安心。总而言之，Java 的社区和宽度不管面对什么事都胜过其他方案。所以，我不看别的东西，我也很高兴这么做，因为这可以让我竭尽全力建立 Java 顾问业务，并让我的顾客满意，而不用对每个新问题研究各种做法。

当一个占有优势的语言或技术正处于黄金时期时，忽略其他的解决方案也就是一种恩赐。图1-1展示了我的想法，当一个新的语言到达且其威力超越 Java 或 C++ 的时候，你可以忽略它一段时间，但是如果你没有精确地判断技术生命周期，你会被淘汰而自己却不知道。忽然间，你的竞争者已经飞奔到你的前面了，他们的生产力更高，产品质量更好，有更多顾客。当你进入过渡时期，最好提高警觉。