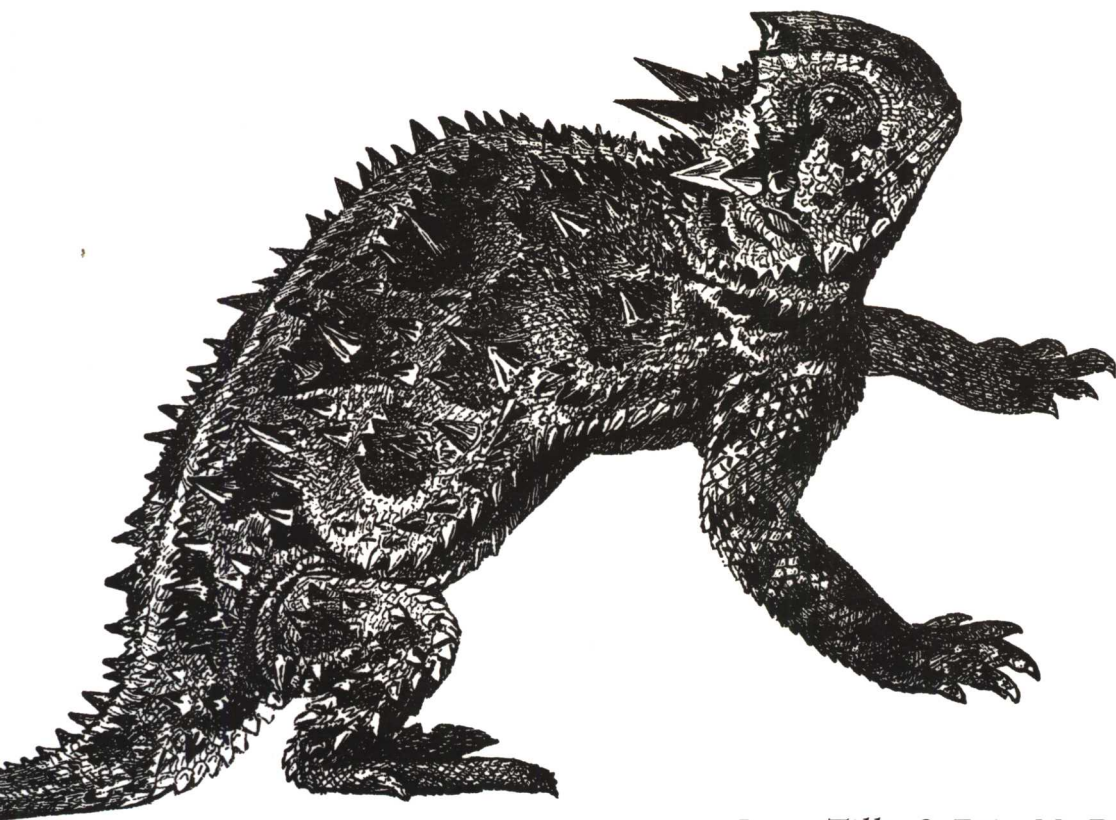


Ant: The Definitive Guide

Ant

权威指南



Jesse Tilly & Eric M. Burke 著
James Duncan Davidson 序

林琪 译

O'REILLY®

中国电力出版社

Ant 权威指南

Jesse Tilly & Eric M. Burke 著

James Duncan Davidson 序

林琪 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

Ant 权威指南 / (美) 泰利 (Tilly, J.) 等著; 林琪译. - 北京: 中国电力出版社, 2003

书名原文: Ant: The Definitive Guide

ISBN 7-5083-1417-4

I. A... II. ①泰 ... ②林 ... III. 软件工具, Ant IV. TP311.56

中国版本图书馆 CIP 数据核字 (2003) 第 023090 号

北京市版权局著作权合同登记

图字: 01-2002-3275 号

©2002 by O'Reilly & Associates, Inc.

Simplified Chinese Edition, jointly published by O'Reilly & Associates, Inc. and China Electric Power Press, 2002. Authorized translation of the English edition, 2002 O'Reilly & Associates, 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 & Associates, Inc. 出版 2002。

简体中文版由中国电力出版社出版 2002。英文原版的翻译得到 O'Reilly & Associates, 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 & Associates, 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 / Ant 权威指南

书 号 / ISBN 7-5083-1417-4

责任编辑 / 宋宏

封面设计 / Hanna Dyer, 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 24 印张 351 千字

版 次 / 2003 年 8 月第一版 2003 年 8 月第一次印刷

印 数 / 0001-3000 册

定 价 / 52.00 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

序	1
前 言	7
第一章 Ant 入门	15
文件和目录	15
Ant 的构建文件	16
运行 Ant	19
Ant 命令行参考	22
构建文件轮廓	24
继续学习	25
第二章 安装和配置	26
发 布	26
安 装	27
配 置	36

第三章 构建文件	40
为什么用 XML?	41
Ant 构建块	42
一个示例工程及构建文件	47
构建文件执行处理	63
Ant 并非脚本语言	75
构建文件授权问题	78
 第四章 Ant DataType	 81
已定义 DataType	82
XML 属性约定	82
argument DataType	85
environment DataType	88
filelist DataType	90
fileset DataType	93
patternset DataType	96
filterset DataType	98
path DataType	100
mapper DataType	102
 第五章 用户编写任务	 108
定制任务的需要	109
Ant 的任务模型	110
任务生命期	120
通过分析看示例: jar 任务	124
关于任务的其他内容	143
 第六章 用户编写监听者	 147
BuildEvent 类	148
BuildListener 接口	150

一个例子: XmlLogger	152
并行问题	156
第七章 核心任务	158
任务总结	159
常用类型和属性	162
工程和目标	165
核心任务参考	167
第八章 可选任务	261
任务汇总	261
可选任务参考	265
附录一 Ant 的未来	351
附录二 Ant 解决方案	355
词汇表	369

序

必须承认，我从没有想过像 Ant 这样一个小小的构建工具能够发展得如此迅速，而且在 Java 开发者社区中会取得如此显著的地位。在我编写第一版 Ant 时，它只不过是有助于解决我所遇到的跨平台构建问题的一个简单工具。现在它已成长壮大起来，而且正得到世界各地成千上万开发人员的使用。其间的奥秘在哪里呢？这样一个小程序何以最终被如此众多的人所采纳呢？也许由 Ant 的发展历程可稍见端倪。

早在加入 Apache 的 CVS 服务器之前，Ant 就已经被编写出来。1998 年中期，我在 Sun 公司负责创建 Java Servlet 2.1 规范以及相应的参考实现。我将此参考实现命名为 Tomcat，它是一种全新的代码基，这是因为原来的参考实现所基于的是 Java Web 服务器的代码，这是一种由 JavaSoft 迁移到 iPlanet 的商业产品。而新的实现必须是 100% 纯 Java 的。

为了使参考实现得到 100% 纯 Java 的证书，即便我们是 Sun 公司的内部人员，而且使用 Java 平台工作，也必须向 Key Labs（一个独立的证书公司）表明它可以在三种不同平台上运行。为了确保 servlet 参考实现能够随处运行，我选择了 Solaris、Windows 和 Mac OS。另外，我不仅希望 Tomcat 能够在这三种平台上运行，而且希望它在所有这三种平台以及 Linux 上还能够构建和开发。我尝试使用了 GNU Make、shell 脚本、批处理文件，以及各种各样其他的工具。每种方

法都存在其自己的问题。这些问题均源于一点，即所有现有的工具都无一例外地借鉴了C程序的构建。将这些做法应用于Java确实也能工作，但是很慢。即使Java程序本身可以很好地完成工作，与Java虚拟机相关的启动开销也相当大。而且，当Make为每个需要编译的文件创建一个新的VM（虚拟机）实例时，编译时间会基于一个工程中的源文件数成线性增长。

我尝试了许多方法来编写一个make文件，从而对于需要重编译的工程，使得其中的所有源文件一次即传递给javac。但是，无论我怎样努力，无论我采用了多少种Make向导工具，仍然无法得到一种能够在多个平台上采用同一方式工作的方法。make文件中的!`&$%#`制表符格式实在令我太厌烦了。即便我是Emacs的忠实支持者，我仍然无法忍受一个工具需要用Emacs来编写其文件，以确保不会出现非有意放置的空格（注1）。

那是在欧洲参加完一个会议的返航途中，我终于彻底厌倦了，对于创建在任何环境下都能采用同样方式工作的make文件，我已不打算再去尝试了。我决心“建立”一个自己的工具：这个工具应当能够检查一个工程中的所有Java源文件，将它们与所有已编译的类相比较，并把需要编译的源文件清单直接传给javac。另外，它还应完成其他的一些工作，例如将所有的类填入一个JAR文件中，并复制另外一些文件从而建立该软件的一个可发行版本。为了确保在每个得到支持的平台上都能采用相同方式工作，我决定用Java来编写这个工具。

数小时后，我就有了一个实用的工具。它非常简单和原始，只包括为数不多的类。其中用到了java.util.Properties的功能作为其数据层。它确实可以工作，而且相当不错，我的编译时间下降了一个数量级。回到美国后，我又在Solaris、Linux和Mac OS上对它做了测试，在所有这些平台上它均能很好地工作。当时它最大的问题就是其所能完成的工作仅限于编译文件和复制文件，而且此功能是硬编码的。

注1： 据称，make文件格式原来的设计者在第一周之后便认识到这些制表符可能会成为一个问题。但是当时他已经有了数十个用户，因此不想再破坏其兼容性。

数周之后，我向我的朋友 Jason Hunter 展示了这个名为 Ant 的工具（注 2）（之所以如此命名，是因为尽管它是一个小东西，却能够做大事，这与“蚂蚁”有着异曲同工之处），Jason 是 O'Reilly 公司出版的《Java Servlet Programming》（译注 1）一书的作者。Jason 认为，作为一个工具它已像模像样，但并不觉得它会有多重要的作用。当时我正在考虑使用 Java 的反射功能来提供一种简洁的方法对 Ant 的功能加以扩充，从而使程序员可以编写其自己的任务来实现扩展，也就是说，在我提到这种想法之前，Jason 并不认为 Ant 有太大的意义。而在此之后他的疑问顿消，于是我有了第一位 Ant 用户兼 Ant 的热心传播者。Jason 还有一个非凡的能力，他能够很快地在软件的任何一部分中找出 bug，并帮助我解决了不少问题。

有了反射层之后，我又编写了更多的任务，这样 Ant 对 Sun 公司的其他小组也能助一臂之力。不过，构建文件的格式稍显繁琐。特性（property）文件并未确实做到层次式分组，随着任务的引入又带来了目标（任务集合）的概念。我采用了许多不同方法来解决这一问题，不过同样又是在一次从欧洲的返航途中有了解决问题的灵感。这一解决方案即为令“工程-目标-任务”层次遵循 XML 文档层次结构。以往的反射工作需要将 XML 标签名与任务实现相关联，而该解决方案还可以减轻这项工作的负担。

最后，在横越大洋上空时，我完成了最棒的代码编写。海拔高的位置是否对产生灵感有所帮助，对此我一直感到疑惑。也有可能是到欧洲的旅行为我带来了创造力，这一点只有更多的实践才能得到验证。

正如我们所知，Ant 已经推出。你在当前使用的各版本 Ant 中所见的一切（包括好的和不好的），都是当时所做决定的结果。可以肯定，尽管自此已经做了许多修改，但是基础仍保持不变。2000 年末，连同 Tomcat 一同加入到 Apache 的 CVS 工具库中的基本上即为此源代码。其后我又转向其他一些工作，主要是作为 Sun 公司 Apache 软件基金会（Apache Software Foundation）的代表，以及从事 XML 规范（如 Sun 公司的 JAXP 和 W3C 的 DOM）方面的工作。

注 2： 另外，ANT 还代表“Another Neato Tool”的字母缩写。我知道这样解释有点傻，但事实确实如此。

译注 1： 本书中文版《Java Servlet 编程》已由中国电力出版社引进出版。

令人难以置信的是，世界各地的人们都开始谈论 Ant。最早发现它的是那些在 Apache 上使用 Tomcat 的人。然后他们开始向他们的朋友介绍，而这些朋友又向他们的其他朋友推广，如此继续。此时，更多的人们已经了解到了 Ant，并且使用率甚至多于 Tomcat。在 Apache 上开发和使用 Ant 的人已经形成了一个强大的群体，而在此发展道路上也已经对该工具做了许多改进。人们现在使用它来构建各种形式的工程，从很小的应用到极其庞大的 J2EE 应用都包括在内。

我得知 Ant 被载入史册是在 2001 年的 JavaOne 大会上。当时我正在参加一个主题介绍，其中要演示一个主要数据库软件公司所推出的新的开发工具。演示者展示了通过在方框之间进行连线可以何等容易地设计软件，然后单击了构建（Build）按钮。在显示屏幕上展现的居然是每个 Ant 用户所熟识并视为基础的那些方括号。对此我是如此震惊，同时简直无法相信。

Ant 用户的数目仍在增长。显然我所开发的这个小东西已经为全世界 Java 开发人员所共享。而且还不仅仅限于 Java 开发人员。最近我还发现了一个 NAnt，这是 Ant 思想针对 .NET 开发的一种实现（注 3）。

要是我原先知道 Ant 会取得如此巨大的成功，最初可能会花费更多时间来增强其功能，使它不仅仅像现在这样只作为一个简单工具。不过，这样也许会将其最初使之脱颖而出的一些特性埋没。Ant 可能会设计得过于复杂。如果我花费太多时间来使之能够完成超出我的需求之外的工作，它就有可能成为一个过于庞大的工具，而且也可能过于“笨重”以至于难以使用。这种情况在软件行业屡见不鲜，特别是目前所提出的许多 Java API 更是如此。

Ant 成功的秘诀可能就在于它原本没有打算成功。它只是针对许多人所遇到的一个明显问题而提出的一个简单解决方案。作为解决这一问题的幸运儿，我感到颇为自豪。

你手中的这本书将指导你如何使用当前的 Ant。Jesse 和 Eric 将教你如何高效地使用 Ant，以及如何对其扩展，并告诉你将如何使用各种不同任务（包括内置任务

注 3： 可在 <http://nant.sourceforge.net/> 上找到 NAnt。

和广泛使用的可选任务)。另外,对于某些Ant设计决策,作者还将提供一些技巧从而避免其本身的缺陷。

在由他们接手为你提供有力的指导之前,我还有最后一句话要告诉你:只要可能,就应自行解决自己的切肤之痛。如果已有的工具不能满足你的需求,就应当另找一个工具。如果不存在这样的工具,那么就创建一个。而且要确保与大家共享。其他数以千计的人可能有着与你类似的难题。

—— James Duncan Davidson

旧金山,加利福尼亚州,2002年4月

前言

对于许多基于Java的工程而言,对所有Java™源文件进行编译已不再是构建这些工程所需的惟一步骤。对于典型的HelloWorld程序、书中的例子以及简单applet,源文件的编译就已经足够了。但是还有一些复杂的基于Java的工程,如Web应用或基于Swing的程序(如JBuilder),要求做更多的工作。必须根据资源控制得到最新的资源;未由Java编译器自动处理的依赖关系也需要得到管理;各种类必须被捆绑并交付到多个位置,有时是作为JAR或WAR文件进行交付;某些Java技术,诸如EJB(Enterprise Java Bean,企业Java Bean)和RMI(Remote Method Invocation,远程方法调用)类,则需要单独的编译和代码生成步骤,这些均并非由Java编译器完成。shell脚本和GNU Make通常是完成这些任务的首选工具,从“完成工作”的角度来说,这些工具可以很好地达到目的,但从长远来看,它们却是不太好的选择。

虽然GNU Make可以提供许多功能,但在易用性方面,却存在许多缺陷。makefile有其自己的语言语法,这就要求编写makefile的人具备此项专门的知识。GNU Make还缺乏平台无关性,因此对于同一个makefile,需要维护和分发多个版本(每一个版本对应于一个目标平台)。由于shell脚本和GNU Make(要记住,GNU Make只是现有shell基础上的一个语言扩展)所固有的性质,使得对于任何一个非专家级用户来说,在操作系统之间(甚至shell之间)进行迁移都是很困难的,

甚至是不可能的。如果使用 GNU Make，那么对于当前基于 Java 的工程，遵循这种做法所需的时间和维护开销是相当高的，这种情况可谓屡见不鲜。

Sun 公司对其所有 SDK 工具均提供了 Java 版本。诸如 *javac* 等可执行程序只是执行 Java 代码的包装器。其他厂商的工具，如 BEA 公司的用于 WebLogic 的 EJB 编译器、JUnit 和 Jakarta 工具以及库均采用 Java 编写。GNU Make 只能从命令行调用可执行程序。例如，为了调用一个 Java 类，GNU Make 必须使用 *java* 命令来调用 JVM，再将类名作为命令行参数进行传递。Make 不能在程序中使用 Java 工具的任何库，如异常和错误对象等，而这些库则允许更为灵活的构建过程。对于用 Java 编写的工具（如 WebLogic 的 *ejbc* 编译器），它们可以与同一 JVM 中可用的其他对象（如 Ant 任务对象）共享来自于异常和错误的信息。较之于命令行返回代码和事后的错误消息字符串解析而言，这样做可以改进构建过程。

GNU Make 所存在的问题和用 Java 编写构建工具的可能性促使 James Duncan Davidson 写出了 Ant。Ant 将 Java 编译器作为一个类运行，而不是来自命令行的一个调用。保持在 JVM 中则允许用特定的代码处理错误，还可对 Sun 公司通过其编译器提供的结果采取操作。Ant 用 XML 作为其构建文件语法，因此只是会增强开发人员和工程管理人员的技能，而不是令其因学习新知识而过于疲劳。Ant 对构建过程加以扩展，使其不只是运行程序，因此称之为构建环境（environment）比构建工具更为恰当。

本书的结构

本书涵盖了初识 Ant 的人所需的全部知识。对于 Ant 专家来说，本书可作为一本参考书，其中提供了 Ant 核心任务的详细定义；讨论了 Ant 的主要特性；提供了用 Ant 来管理工程的一些最佳实践；还解释了某些 Ant 问题的解决方法。

第一章，Ant 入门。这一章逐步介绍了一个非常基本的 Ant 构建文件示例，其目的是使你能够很快上手。我们展示了如何创建目录、编译代码以及生成一个 JAR 文件，但是对于每一项工作如何进行的具体细节则未做深究。这一章还包括了 Ant 命令行使用的详细信息。最后得到一个构建文件的大致轮廓，以此作为初始模板。

第二章，安装和配置。这一章介绍了如何得到、安装以及在 Windows 和 Unix 平台上配置 Ant。我们列出了在这些开发平台上发现的一些缺陷，并提供了解决方法和解决方案。

第三章，构建文件。这一章介绍了一个示例工程上下文中的 Ant 构建文件例子。我们对此构建文件的主要部分和结构加以了剖析和描述，还对一些问题做了解释，如 Ant 引擎的一般流程和 Ant 使用 XML 的好处等，并强调了构建文件的主要部分。

第四章，Ant DataType。这一章详细描述了各种 Ant DataType。尽管在前面的章节中已经用到过 DataType，但这里才对其做深入研究。我们介绍了如何使用环境变量和如何传递命令行参数以进行处理，还说明了如何利用各种文件和模式。

第五章，用户编写任务。这一章涵盖了 Ant 的一个最佳功能，即能够对 Ant 进行扩展。有此编写扩展的功能，就能够处理特定工程可能需要的任何事情。作为一个意外收获，你还可以在将来的工程中重用这些任务，对于你所付出的努力，你的收益将并不仅限于最初的实现。你的任务还可以得到共享并公开发布，这样即使是你不认识的人也能够由于你的工作而受益。

第六章，用户编写监听者。这一章介绍了如何设计和开发你自己的构建事件监听者。据此，你就可以编写类来完成一些操作，这些操作所基于的是与构建文件处理相关的流程。操作的范围很广，从特定复杂任务完成时的发送邮件，到将同样的事件重定向到一个集中的“构建监听框架”等均属于此范畴。正如用户编写任务一样，可能的用户编写监听者也是无法计数的。这一章还包括对监听者的进一步扩展：用户编写日志工具 (logger)。利用这些日志工具，可以改进甚至替换 Ant 默认的日志系统。

第七章，核心任务。这一章是对全部核心 Ant 任务的详尽参考。对于每个任务，在此都可以看到一个描述、一组支持此任务的 Ant 版本以及对所有任务属性的定义。从这里还可以得到有关任务使用的有用示例。

第八章，可选任务。这一章的形式类似于第七章，是对 Ant 丰富的可选任务所提供的参考。

附录一，Ant 的未来。这一部分所讨论的正如其标题所示。我们介绍了 Ant 未来的方向和将出现的新功能，另外还提供了有关建议，你可通过采取所建议的步骤来避免使用很快就要过时的功能。

附录二，Ant 解决方案。Ant 可用于解决不同的构建问题，这一部分则深入研究了其中所采用的更为常见的方法。除此以外，我们还谈到如何使用带有级联工程结构的构建文件。这些工程结构有一个主工程目录以及许多子工程子目录。每个子工程包括其自己的构建文件，而且主工程有一个主构建文件，能够构建所有子工程。

本书的读者对象

本书主要面向 Java 开发人员，特别是那些开发企业级 Java 应用的人员。另外，如果某些人需要一个健壮的构建工具，并要求这种工具不只是调用命令行编译器和实用程序，那么这本书对其尤为适用。对于大型工程的构建管理人员（以及负责构建管理的工程管理人员）来说，这本书也很有用。

预备知识

对于本书的大多数内容，仅需要对 Java 和 XML 有基本的了解即可。有关为 Ant 编写扩展的章节则要求你还应对 Java 继承和接口有明确的理解。Ant 最适合用于构建和部署基于 Java 的工程。有些 Ant 任务可以提供功能来编译和运行其他语言，如 Perl、Python、C 和 C#，尽管这些任务也可用，但本书所强调的是 Java 在 Ant 中的使用。

平台和版本

作为 Apache 的 Jakarta 项目下的一个开源项目，Ant 得到“每晚（nightly）”代码修订和构建。这些每晚构建即创建了 Ant 的“非稳定版本”。主要维护人员时常会以一个每晚构建作为发布版，并宣布其功能和稳定性。在写这本书时，共有 5 个