开源技术专家
OPEN SOURCE TECHNOLOGIST

Apress

The Definitive Guide to Grails

Grails 权威指南

国内首部Grails图书：
借助动态脚本语言Groovy，
在Grails项目创始人带领下应用Grails框架进行Java敏捷开发。

[美] Graeme Keith Rocher 著
张若飞 孙岚 郭会强 译
飞思科技产品研发中心 监制

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

开源技术专家
OPEN SOURCE TECHNOLOGIST

The Definitive Guide to Grails

Grails 权威指南

[美] Graeme Keith Rocher 著
张若飞 孙岚 郭会强 译
飞思科技产品研发中心 监制

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书译自Grails项目负责人Graeme Keith Rocher所著的“The Definitive Guide to Grails”一书，着重介绍了如何在Grails框架下使用Groovy语言进行敏捷的Web开发。本书详细讲解了Grails开发的全部过程，包括项目构架、控制器和视图、与关系数据库之间的ORM映射，以及与Ajax和Java平台的无缝集成。同时该书也揭示了Grails对Java Web开发领域极大的推动作用，英文原版自出版后屡受大师佳评。

本书分为 11 章，所介绍的内容主要包括：Grails 的目的与优势、Groovy 语言基础与高级特性、Grails 的工程项目结构、根据领域模型自动创建 Web 程序、在 Grails 中实现单元测试和功能测试、控制器的使用、Grails 的高级视图技术 GSP、在 Grails 中使用 Ajax 增强用户体验、Grails 的高级特性，以及与 Java 的无缝集成。

本书适合所有对动态语言感兴趣的读者阅读，不管是否有过从事 Perl、Ruby 还是 Python 项目开发的背景，只要希望能够对动态语言有更深入的理解，都可以从本书中获益。不过，本书主要还是面向那些已经掌握 Java 语言，希望找到更好、更快捷地开发 Web 框架的 Java 爱好者。

1-59059-758-3 The Definitive Guide to Grails by Graeme Rocher

版权贸易合同登记号 图字：01-2007-4143

图书在版编目（CIP）数据

Grails 权威指南 /（美）瑞切（Rocher,G.K.）著；张若飞，孙岚，郭会强译.—北京：电子工业出版社，2007.11
（开源技术专家）
书名原文：The Definitive Guide to Grails
ISBN 978-7-121-05201-9

I. G… II.①瑞…②张…③孙…④郭… III.软件工具－程序设计 IV.TP311.56

中国版本图书馆 CIP 数据核字（2007）第 159485 号

责任编辑：宋兆武 张 帆
印 刷：北京智力达印刷有限公司
装 订：北京中新伟业印刷有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编：100036
开 本：787×1092 1/16 印张：21.25 字数：502 千字
印 次：2007 年 11 月第 1 次印刷
印 数：5 000 册 定价：49.80 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

译者前言

Java 作为如今世界上使用最广泛的语言，经过十多年的发展，已经在各个领域尽显其优势，尤其在 Web 开发领域中，从众多的 Web 框架就可以看出其快速并日趋成熟的发展。在开源社区的努力推动下，Struts、Spring、Hibernate 都已经成为了行业中实际的标准。但是如今的 Java 开发者们仍然需要埋头于繁杂的配置文件中，这也正是之前 Ruby on Rails（简称 ROR）这样的敏捷性开发框架能够震撼 Java 社区的原因。许多 Java 开发者不得不重新开始学习 Ruby，以便投入 ROR 的怀抱。这使得 Java 社区终于开始反思，随即迎来了属于自己的脚本语言 Groovy 和敏捷的 Web 开发框架 Grails。

作为赢得 JAX（德国最重要的 Java 会议）2007 年创新大奖第一名的 Groovy，已经用三年多的时间在 Java 社区中树立了其坚实的根基。而对于 Rails 这个使 Ruby 一举成名的框架，Java 社区也顺水推舟，借助于 Groovy 的强大优势推出了 Grails。由于其整合了 Spring、Hibernate、SiteMesh 这些经过千锤百炼的框架，而且与 Java 的无缝集成也使得 Java 开发者可以轻松地转到敏捷开发的道路上来。此外，不仅 Eclipse、IntelliJ IDEA 都开始提供 Groovy 和 Grails 的开发插件，Oracle 和 IBM 等也都提高了对 Groovy 和 Grails 的关注程度，基于二者的应用也如雨后春笋般地出现了。

本书历经数月的翻译及审校，由张若飞、孙岚、郭会强（排名不分先后）三人共同翻译完成。其中郭会强翻译第 1、2 章，孙岚翻译第 3～7 章，张若飞翻译了文前、第 8～11 章并负责了全书的统一及初步校对工作。由于长期从事 Java 开发工作，我们深切地感受到 Spring、Hibernate 等框架在配置文件方面的烦琐，也曾被 ROR 的简捷所感染（虽然效率一直是其诟病），但仍舍不得完全抛弃使用多年的 Java 及其丰富的资源。当我们第一次接触到 Grails 的时候，一种期盼已久的感觉不由涌上心头——因为我们知道，它就是众多 Java 开发者所期待的东西。随着其不断的发展，我们也以各种方式对 Grails 及 Groovy 进行了宣传介绍：不仅在各自的博客上不断发布有关 Grails 和 Groovy 的技术文章，热心解答爱好者们提出的问题，也参与了部分 Grails 的开发工作，其中孙岚便参与了 Cache 部分的开发。在实际应用中，我们也积极采用 Grails，积累了大量的实践技巧和经验，并且取得了令人满意的效果。在 2007 年年底正式发行 Groovy1.1 和 Grails1.0 后，我们坚信 2008 年一定会是一个“Groovy 和 Grails 年”！

本书能与大家见面，首先要感谢策划编辑张春雨的不断支持与帮助，为我们尽心尽力解

决各种困难，并提出许多中肯的建议。感谢我们的家人、朋友的关怀与鼓励，是他们在这期间给了我们无微不至的照顾。感谢出版社的编辑、审校等的严谨校对，也感谢 Groovy 精英联盟群中荐言献策的朋友们及后期参与校对的众多优秀技术人员，是他们进一步地提升了本书的品质，因此我们相信这本书一定会成为读者学习 Grails 的最佳资料。在本书的翻译过程中，我们本着严谨、创新的态度，不仅完全保留了原书的所有内容，修正了原书中的错误，并且根据如今最新的 Grails 发布版本（0.5.6 版）及我们从实践应用中获得的经验和技巧更新了原书中的部分内容，力求使读者能够接受最新的知识。此外，我们还将与原书配套的代码进行了修正，读者可以在飞思的网站上进行下载。翻译此书，不仅因为我们本身热衷于 Groovy 和 Grails，同时我们也希望能将更新、更好的技术尽快地引进国内，引起更多人的关注，促进国内的技术发展，共同提高技术水平。鉴于时间仓促及技术和经验上的不足，本书中难免有错误和疏漏之处，恳请读者批评指正，不胜感激。

译　者

2007 年 8 月

前　言

Grails 是一个面向企业级应用、基于 MVC 模式的 Web 框架，其构建于 Spring、Hivernate、Quartz 和 SiteMesh 这些已被无数实际应用证实的、可扩展的开源框架之上。与以前的 J2EE 规范（Bruce Tate 称其为“大象”，意味着 J2EE 规范既强大又笨拙）相比，Grails 的目的在于帮助开发人员更快地创建 Web 程序。

或许从 Ruby on Rails、Django 或 TurboGears 这些新框架中得到了灵感，Grails 也按照“习惯优于配置”的原则来简化复杂的问题。不仅如此，Grails 使开发人员找回了开发 Web 程序的乐趣。Grails 的灵活性使得开发人员可以在几个小时之内就创建出 Web 系统的原型（ProtoType），这样就可以对底层的域模型（Domain Model）进行验证，或者进一步通过与客户讨论来确定需求，避免出现用户在开发的前几个月中只能等待的情况。

由于借助于 Java 平台和开源框架，Grails 不仅能在装有开源 Servlet 容器的普通机器上运行 Web 程序，也能用于商业中大规模的集群服务器。这样企业可以放心地在软件和硬件上进行投资。而且开发团队无须经过长时间的培训和拥有长期的开发经验，就能在开发过程中充分发挥 Java、Spring 和 Hibernate 等技术的优点。虽然 Grails 将底层的实现框架封装了起来，但是当实际需要（如与遗留系统的集成）时依然可以由开发者来进行配置。

Graeme Rocher 和他的团队不仅在 Grails 中使用之前那些强大、成熟的框架，而且使用了 Groovy——与 Java 平台集成性最好的动态语言作为 Grails 的基础，这样 Grails 就能够充分利用 Groovy 具有的动态特性。之所以选择使用 Groovy 是由于 Groovy 与 Java 非常类似，这样 Java 开发人员可以很轻易地过渡到 Groovy 的开发中。我们不仅可以像在 Java 中一样编写 Groovy 代码，在深入了解 Groovy 后还可以使用其强大的动态特性，而这一切都得益于背后的 Java 平台和 Grails 框架。

之所以给这个框架起名“Grails”，是希望它能够成为所有开发者手中的 Holy Grails（圣杯）。Grails 通过极大地提高开发人员的开发效率，已经证明了这一目的不只是夸夸其谈。Graeme Rocher，Grails 项目的负责人及本书的作者，已经为开发这个强大的 Web 框架做了巨大的工作，并且也培养出了一个开放、团结的社区。在这本书中，他会带领我们逐步对 Grails 有一个清晰、透彻的了解。

《Grails 权威指南》是 Graeme Rocher 的另一本著作，他本人不但负责整个框架的开发工作，而且还编写了大量细致的在线文档。在这本书中，Graeme 会借助他在开发 Web 程序中的经验和知识，带领我们逐步掌握 Grails 框架和 Groovy 动态语言。

我们十分有幸能与 Graeme 一同开发 Grails 并完成这本书。

对于任何希望学习 Grails，并且愿和我们共同分享编程乐趣的读者来说，都应该仔细阅读这本书。

Guillaume Laforge
Groovy项目经理
Dierk König
《Groovy in Action》一书主要作者

关于作者

GRAEME KEITH ROCHER 是 Skills Matter（http://www.skillsmatter.com，专注于开源技术和敏捷软件开发的技术传播公司）的软件架构师与技术部负责人。他现在的职位是CTO，并负责Skills Matter公司的课件开发方案和总体技术指导。作为开源社区的多领域专家之一，他积极地在各种会议上进行有关Groovy、Grails以及Java动态语言等话题的演讲。

Graeme的职业生涯开始于与一个团队共同开发基于J2EE技术、可升级式的企业学习管理系统，其中他负责电子学习（E-Learning）部分的开发。随后他进入了数字TV领域，当面对不断更新的数字平台时，他决定使用敏捷方法来解决日益增长的复杂需求。这是Graeme第一次接触Groovy，随后他使用Groovy和Cocoon，在数字TV平台上实现了动态多通道容量管理系统。

由于看到了Web服务及其实现复杂功能的增长趋势，Graeme着手开发一个能够加以简化的项目，这就是Grails。Grails框架本身来源于Ruby on Rails，但是目标在于与Java的紧密集成。Graeme是现在Grails项目的负责人，而且还是Groovy JSR-241执行委员会的成员之一。

在Skills Matter之前，Graeme还在Knowledge Pool工作过。在那里他使用Groovy开发了一种名为Originate的快速电子学习开发工具。Originate使用Groovy将多种技术融合到一起，其中包括Visual Basic、Microsoft Word、XSLT、JavaScript和XSL-FO。

不过Greame最大的成就还是他的两个漂亮的孩子，Lexeia和Alex，以及他深爱的妻子Birjinia。

关于技术评审

GUILLAUME LAFORGE 是Groovy项目的官方项目主管，以及JSR-241（该规范负责Groovy脚本语言的标准化）的规范负责人。他将业余时间都花在开发一种全面的敏捷环境上，一年之前他开始讨论基于Groovy和Spring的Grails框架。

他与另一名富有热情的Groovy开发者Dierk König共同编写了《Groovy in Action》一书，并已由Manning出版社出版。

在他的职业生涯中，Guillaume是OCTO Technology公司的软件架构师和开源顾问。OCTO Technology是法国一家关注于软件架构和信息系统的技术咨询公司。

DIERK KöNIG 是一名高级程序员、技术顾问及指导员。他是《Groovy in Action》一书的主要作者，热衷于Groovy和Grails。由他创立并管理的开源项目Canoo WebTest深受欢迎。Dierk通常在国际会议上和主要杂志上发表演讲和文章。他作为创始合伙人和执行委员会成员，工作于瑞士巴塞尔（Basel）的Canoo Engineering AG。

作者致谢

首先，也是最重要的，我要感谢我的妻子Birjinia，感谢她的美丽、聪明、无尽的爱和支持，她是最好的妻子。同样要感谢的还有我的孩子，Alex和Lexeia。很多次当我写这本书的时候，她们都想把我从笔记本电脑旁拽走，但她们还是耐心地等我写完。感谢我的母亲，我的父亲（我想念你），我的姐姐，以及Birjinia家的所有成员，你们对我而言是那么的重要。

对于Apress出版社，感谢Steve Anglin最初的建议，并鼓励我写一本关于Grails的书籍。虽然后来也有其他出版社询问此事，不过在此还是要感谢Steve和Apress出版社对Grails发展的远见卓识。

同样要感谢的还有一起工作过的Apress制作团队，包括Jennifer Whipple和Ami Knox（我的责任编辑），Nancy Sixsmith（校对员），Lori Bring（制作编辑），还要特别感谢容忍我推迟交稿的Kylie Johnston（项目经理）。

不用说，还要感谢本书的技术评审Guillaume和Laforge（Groovy项目负责人），以及在编写本书过程中提供了深刻见解和建议的Dierk König（《Groovy in Action》一书的主要作者）。

相信如果没有那些一如既往为Grails贡献力量的人们，这一切都是不可能的。在这里要再次感谢Interface21 的Steven Devijver和Guillaume，感谢他们在Grails还只是有一些想法的时候就在邮件中表示了对我的支持。

感谢Marc Palmer提供了合理的意见、机智的论点以及不断为Grails做出的贡献。感谢Micha Kujszo在Quartz项目上的贡献。再次感谢Dierk和Canoo（http://canoo.com）提供了Grails持续集成服务器与测试架构。感谢John “Tug” Wilson和Jochen “blackdrag” Theodorou等Groovy团队成员，是他们及时的支持才使得Grails项目成为可能。

感谢Hermit Design（http://www.thehermitdesign.co.uk）的Arthur Smit，他设计了最初的Grails图标及Grails中很多其他的图案，以及本书的封面。如果你需要制作富有创造性的商标或者任何图片，他将是合适的人选。

同时还要感谢Grails社区一如既往的支持。感谢所有Grails的用户，尤其是Sven Haiges，他制作了Grails的演示视频，以及在InfoQ等网站了写了无数介绍Grails文章的Jason Rudolph。

追溯过去，我还要感谢原先Futuremedia团队的核心成员们，他们教会了我很多东西，即使现在也是一样。感谢其中的Jeremy Aston、Piers Geyman、Phil Horton、Nathan Summers等人，并要特别感谢Pete Bergin(RIP)，我们想念你。

再次感谢Oracle的Tugdual Grall极力在Oracle和JavaOne、JavaPolis等重要会议上对Grails

的宣传和支持。最后，我要感谢Skills Matter的Wendy Devolder，感谢她对Grails的远见以及允许我在她美妙的公司工作。

自我介绍

在20世纪90年代后期，我正在使用早期的J2EE技术，例如EJB1.0和Servlet框架来开发大规模的企业学习管理系统。到处都在如火如荼地宣传Java，几乎当时每个主要IT出版物的封面都有EJB或者Java的字样。

当我们在按照已有的知识使用EJB时，却发现它打算用Servlet来解决我们所有的问题（当时甚至没有一种视图技术）的想法是错误的，但是行业依然在支持它。

不过现在时代已经不同了。如今，Java和J2EE这两个词已经被人们所淡忘，而业界又开始宣扬一些更复杂的概念，例如SOA和ESB等。从我的经验来说，我认为开发者一直面临的任务就是如何编写更少的代码实现相同的功能。像早期一些开发社区采用的J2EE规范并不能做到这点。如果一个框架或规范过于复杂，需要我们编写大量的重复代码，那就应该尽早放弃使用。为什么我们一定要编写这么多重复的样板代码（Boilerplate Code）呢？我相信一定有解决的办法。

实际上，开发者总是在不经意间影响着技术的发展。为什么有这么多开发者比Web服务的SOAP更喜欢REST？为什么更愿意使用Hibernate而不是EJB的持久层？为什么更愿意使用Spring而不是JNDI来实现控制反转（Inversion of Control）？最后，简单总是能战胜复杂。

当然，使用Spring和Hibernate会比使用传统J2EE方法好得多。实际上只要可能，我就会尽量去使用WebWork这样一些成功的项目。但是，这样面临的问题就是我必须将更多的精力放在如何解决环境的配置问题上，而不能集中精力去解决需求上的问题。毕竟我作为一个开发者在开发上应该效率更高，也希望将更多的时间花在自己喜欢的技术与学习新的、感兴趣的技术上。

在2003年，我发现了Groovy。我一直都非常喜爱动态语言更松散的控制方式，再加上以前也使用过Perl、Visual Basic和JavaScript，因此在研究WebWork的源代码之后，我很快可以用Groovy来编写MVC控制器（在WebWork术语中称为操作，Action）。

Groovy很适合编写控制器代码，用来通过服务实现业务逻辑委托（Delegate）和控制显示相应的视图。正当我打算花更多的时间来研究它的时候，以Ruby on Rails为首的各种动态语言框架席卷而至。

可惜，这场风暴来得有些晚了。Java语言本身、整个社区、工具、框架甚至思想都已经形成并且整合到了一起。Java发展的速度令人惊讶，对于在商业培训有过多年工作经验的我来说，没有看到一点Java发展减缓以及违背潮流的信号。但是，Java也有自身的缺点，并且我希望能编写更少的代码。在2005年夏天，我、Steven Devijver和Guillaume Laforge在Groovy邮件列表中讨论之后，诞生了创建Grails的想法。

实际上，J2EE的许多规范本身并没有错误，只是抽象的级别太低。Struts、WebWork和最近出现的JSF框架都为了解决这个问题，但是Java及其静态类型制约了这点。相反，用Groovy可以实现更高级别的抽象。在使用Groovy实现控制器之后，现在该是将它用于实现各层的时候了，其中包括从控制器到标签库（Tag Library），从持久层到视图等。

使用Groovy元编程（Metaprogramming）可以创建相当简单、准确的API。出于“编写更少的代码”和“保持与Java友好”这两点考虑，Grails不管是在自定义领域特定语言（Domain-Specific Language）还是在编译糅合（Mixin）时，都选择使用动态运行和动态编译。

不过，Groovy和Grails并不像其他动态语言框架一样，打算取代Java。相反，它们最初设计是为了能与Java协同工作，Java实际处于二者的核心地位。在Grails中，开发者可以选择使用动态类型性来实现某些功能，也可以出于安全角度使用静态类型。

Grails也并不是工作时的唯一工具，它只是为简化日常的烦琐操作提供了选择，当需要时也可以实现整个Java平台能够提供的强大功能与灵活性。因为我个人非常喜欢使用Grails，并且作为Grails社区的一员，希望你也能喜欢这本书。

本书适于的读者对象

Grails属于基于动态语言实现的框架之一，从这点上来讲，所有对动态语言感兴趣的人，不管是否有过Perl、Ruby还是Python的背景，只要希望能够对动态语言有更深入的理解，都可以从本书中获益。

如果你的项目正在用Java开发，那么Grails能提供一些其他框架无法提供的特性，它们也许正是你苦苦寻找的解决办法。不过，这本书主要还是面向那些本身已经掌握Java语言，希望找到更好的Web框架的Java爱好者。

Grails以框架的形式和简单优雅的方式，解决了Java世界中长时间困扰大家的问题。但是，这不表示本书的内容也很简单。

本书在使用Groovy语言的高级特性和实际示例时可能会有一些难度。我会带你逐步了解Groovy等类似动态语言在Web领域的方方面面，例如从使用Ajax技术的视图层到使用大量领域模型（Domain Model）的持久层。对于有经验的Java开发人员来说，像闭包（Closure）、生成器（Bulider）和元编程等Java中没有的概念，一定会让你耳目一新。

不过，虽然在内容和示例上有一些难度，但是解决方法都很容易，从中读者也可以学到开发Web应用程序的新方法。

本书的结构

本书一共分为11章，从概述和介绍开始逐步过渡到最后一章的高级特性。读者可以从第1章开始阅读，了解Grails的目的及其魅力所在。

在第2章，我们会介绍Groovy语言，对于以前没有接触过Groovy的读者来说，可以在这章对Groovy有大概的了解。虽然在这一章中会介绍一些Groovy的高级特性，但是在本书的后续章节中将通过实例使读者能够更好地掌握这些特性。

第3章介绍了Grails的工程项目结构，在这一章中你会创建第一个Grails项目，并且在接下来的章节中都会用到。在第4章会介绍什么是领域模型，以及Grails如何借助于Hibernate等先进ORM技术来实现强大的领域驱动（Domain-Driven）编程模型。在这一章你会创建一个领域模型，并且理解它与底层持久化模型之间的关系。

第5章相信会令你很兴奋，因为你将会学习如何根据领域模型自动创建Web程序。这一章会介绍“脚手架（Scaffolding）”的概念，它不仅能够帮助你提高工作效率，还是你学习Grails的非常好的工具。

在第6章，我们会带你一起在Grails中实现单元测试和功能测试，其中会使用闭包修整（Closure Currying）和模拟对象（Mock Object）等概念来编写高效的测试代码。

然后会在第7章深入探讨如何使用控制器（Controller），以及在第8章中会介绍Grails的高级视图技术——GSP。

第9章中，你会学到如何在Grails中使用Ajax来解决实际中的问题，并改进我们示例程序的可用性。

第10章会介绍Grails的一些高级特性，例如如何集成服务及如何使用作业调度。最后在第11章会介绍Grails中最关键的一点——与Java的无缝集成。

约定

本书中使用了多种语言，包括HTML、XML、JavaScript、Groovy及Java。每个示例都会进行详细的讲解，并且使用固定宽度的Courier字体。而且同时会保持在全书中使用命名约定，使示例尽可能地表达清楚。

很多情况下为了适合于页面的空间，我们对源代码的格式进行了调整，增加了额外的行并按照通常情况修改了代码的缩进方式。为了增加代码的清晰度，也省去了一些不必要的代码。在代码块之间会使用省略号“...”表示省略代码的位置。

必备条件

本书中的示例需要先安装Grails才能运行。本书中使用Grails 0.3（译者注：已经按照0.5.6版进行了更新），最新的版本可以从http://grails.org/Download下载。此外Grails需要Java Virtual Machine的支持，所以需要在安装Grails之前先安装JDK 1.4或以上版本。

读者还可以选择安装应用程序服务器，如Tomcat，以及数据库服务器，如MySQL。这些不是运行Grails必需的，因为Grails已经默认内置了服务器和数据库。不过，如果要部署Grails

程序至少应该安装数据库服务器。

下载代码

本书中示例的源代码可以从Apress网站http://www.apress.com的Source Code/Download部分下载。示例中使用了三种不同的持久化技术。

联系作者

Graeme 是开源社区的积极成员，非常欢迎任何建议和交流。可以通过 E-mail：graeme.rocher@gmail.com 或者其博客（http://graemerocher.blogspot.com）联系他。此外，也可以通过http://grails.org/Mailing+lists上的邮件列表（Mailing list）来发送邮件通知。

目　录