HZ BOOKS

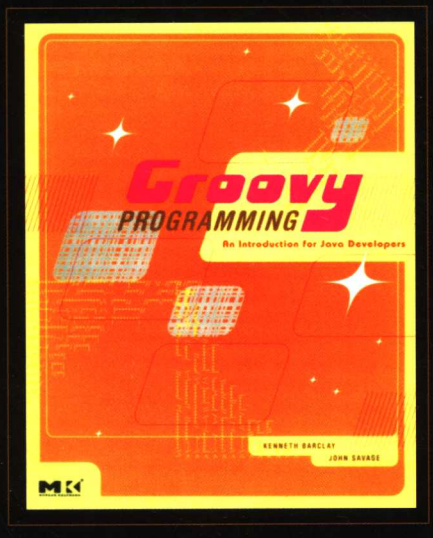
ELSEVIER

华章程序员书库

Groovy 入门经典

Groovy Programming

An Introduction for Java Developers



(英) Kenneth Barclay John Savage 著
龚波 张平 陈蓓 王琦 等译

“如果想全面了解有关脚本、面向对象或者动态语言等相关知识，本书是最适合的！”

Andrew Glover, President, Stelligent Incorporated

机械工业出版社
China Machine Press

Groovy 入门经典

Groovy Programming

An Introduction for Java Developers

华章程序员书库

Groovy 入门经典

Groovy Programming

An Introduction for Java Developers

(英) Kenneth Barclay John Savage 著
龚波 张平 陈蓓 王琦 等译

机械工业出版社
China Machine Press

本书详细介绍脚本语言Groovy，首先介绍Groovy语言的基本特性，包括讨论Groovy方法、程序闭包、列表、映射以及对类和继承的支持，然后介绍如何使用Groovy创建更加高级的应用程序，如使用Spring框架和Cloudscape/Derby关系型数据库管理系统来实现持久性，最后讨论模板和Web应用程序。

本书内容全面详尽，浅显易懂，易于选择性阅读。可以作为对Groovy语言感兴趣的计算机软件开发人员的参考书。

Groovy Programming: An Introduction for Java Developers
Kenneth Barclay, John Savage
ISBN: 0-12-372507-0
Copyright © 2007 by Elsevier Inc. All rights reserved.
Authorized Simplified Chinese translation edition published by the Proprietor.
ISBN: 981-259-991-6
Copyright © 2007 by Elsevier (Singapore) Pte Ltd. All rights reserved.
Printed in China by China Machine Press under special arrangement with Elsevier (Singapore) Pte Ltd. This edition is authorized for sale in China only, excluding Hong Kong SAR and Taiwan. Unauthorized export of this edition is a violation of the Copyright Act. Violation of this Law is subject to Civil and Criminal Penalties.

本书简体中文版由机械工业出版社与Elsevier(Singapore)Pte Ltd.在中国大陆境内合作出版。本版仅限在中国境内（不包括中国香港特别行政区及中国台湾地区）出版及标价销售。未经许可之出口，视为违反著作权法，将受法律之制裁。

版权所有，侵权必究。
本书法律顾问　北京市展达律师事务所

本书版权登记号：图字：01-2007-4489

图书在版编目（CIP）数据

Groovy入门经典/（英）巴克莱（Barclay, K.），（英）萨维奇（Savage, J.）著；龚波等译. -北京：机械工业出版社，2007.10
（华章程序员书库）
书名原文：Groovy Programming an Introduction for Java Developers
ISBN 978-7-111-22493-8

Ⅰ.G…　Ⅱ.①巴…　②萨…　③龚…　Ⅲ.程序语言－程序设计　Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第154315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：王春华
北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行
2008年1月第1版第1次印刷
186mm × 240mm · 22.75印张
定价：49.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

www.hzbook.com

填写读者调查表　加入华章书友会
获赠精彩技术书　参与活动和抽奖

尊敬的读者：

感谢您选择华章图书。为了聆听您的意见，以便我们能够为您提供更优秀的图书产品，敬请您抽出宝贵的时间填写本表，并按底部的地址邮寄给我们（您也可通过www.hzbook.com填写本表）。您将加入我们的“华章书友会”，及时获得新书资讯，免费参加书友会活动。我们将定期选出若干名热心读者，免费赠送我们出版的图书。请一定填写书名书号并留全您的联系信息，以便我们联络您，谢谢！

书名：　　书号：7-111-(　　)

姓名：	性别：☐男　☐女	年龄：	职业：
通信地址：		E-mail：	
电话：	手机：	邮编：	

1. 您是如何获知本书的：

☐朋友推荐　☐书店　☐图书目录　☐杂志、报纸、网络等　☐其他

2. 您从哪里购买本书：

☐新华书店　☐计算机专业书店　☐网上书店　☐其他

3. 您对本书的评价是：

技术内容	☐很好	☐一般	☐较差	☐理由_____
文字质量	☐很好	☐一般	☐较差	☐理由_____
版式封面	☐很好	☐一般	☐较差	☐理由_____
印装质量	☐很好	☐一般	☐较差	☐理由_____
图书定价	☐太高	☐合适	☐较低	☐理由_____

4. 您希望我们的图书在哪些方面进行改进？

_____

_____

5. 您最希望我们出版哪方面的图书？如果有英文版请写出书名。

_____

_____

6. 您有没有写作或翻译技术图书的想法？

☐是，我的计划是_____　☐否

7. 您希望获取图书信息的形式：

☐邮件　☐信函　☐短信　☐其他_____

请寄：北京市西城区百万庄南街1号　机械工业出版社　华章公司　计算机图书策划部收

邮编：100037　电话：（010）88379512　传真：（010）68311602　E-mail: hzjsj@hzbook.com

此为试读，需要完整PDF请访问：www.ertongbook.com

序　言

脚本语言不是新生事物。通常，脚本语言运行于Linux和UNIX操作系统环境下，可以完成很多shell脚本任务，比如软件自动安装和配置，平台定制，使用Python语言建立科学的应用程序原型，以及使用bash脚本完成一次性的命令行任务。有的脚本语言（如PHP）已经广泛地用于开发许多网站。事实证明，脚本语言适合于实现关键的业务应用程序。

在通常情况下，脚本语言是独立的平台，一般情况下不与其他平台进行交互。尽管在桥接（bridge）其他系统时也许存在绑定现象，但是这种集成未必都非常直观或者自然。Groovy试图解决这个问题，它是一种创新的语言，能够自然地通过相同虚拟机的Java环境进行交互。

Groovy使用简练的、易于理解的类Java语法，这样可以降低Java程序员学习Groovy语言的难度。除了语法类似于Java语言之外，Groovy给常见的JDK应用程序编程接口（Application Programming Interface）提供包装器API，进一步拓展Groovy的应用领域。借助Groovy，可以简化常见任务的实现，并集成元编程能力，以开发功能强大的新语言结构，或者轻松地处理已有的语言结构。

Groovy可以用于多种情况，比如作为shell脚本语言完成数据处理和文件操作任务，或者试验使用新的API。它也适合于创建强大的小型或者中型应用程序，能够充分利用Java库和组件。除此之外，另一个重要用法是在Java或者Java EE应用程序中嵌入Groovy，实现Java和Groovy的集成。这样做有助于编写和集中处理经常变化的业务逻辑，或者给应用程序架构提供可编程的配置管理能力。

尽管前两种用法很常见，但是我认为嵌入式应用是最吸引人的，也是最有前途的。现在，开发者始终使用模板引擎来定制和重构我们的视图，或者使用业务逻辑引擎来实现逻辑的外部化和集中化。对于一般的语言来说，除了有限的函数集外，程序员通常很难获得更多的开发支持。幸运的是，宿主于平台的脚本语言，比如Groovy语言，有助于解决这种功能需求障碍。Groovy及其后代Grails（一个功能强大的模型-视图-控制器Web框架）的成功就很能说明这个问题。Sun公司也认可这种向应用程序添加动态性的替代方法，在Java 6中包含新的Java 规范请求（Specification Request）：javax.script.* API能够把任何脚本或者动态语言无缝地嵌入到具有一致的编程API的Java应用程序中。

脚本语言的发展已经达到一定的成熟度，能够突破标准化主流平台的限制。当集成这些语言和平台时，你将会很快得到意想不到的欣喜。

Ken Barclay和John Savage都是值得尊敬的讲师，完全有资格向有经验的开发者和开发新手讲解Groovy。他们以清晰有序的方式介绍如何使用Groovy增加Java平台的魅力，以及如何充分利用Groovy的新功能。本书浅显易懂，即使开发新手也能使用，内容全面详尽，涉及Groovy语言的方方面面。

本书的组织结构是：前面几章着重介绍Groovy语言的基本知识，后面的章节分门别类讨论与该语言相关的高级概念。除此之外，本书还提供很多内容丰富的附录，让读者可以了解相关主题的更详细信息。

本书每章的篇幅较小，易于选择性阅读；每章提供大量完整的代码范例、练习和解决方案。为尽可能清晰地展示如何使用Groovy语言，本书提供一个学习案例，随着每章的学习进度，这个学习案例逐渐演化，功能越来越复杂和完善。除此之外，增量开发和单元测试也是本书的一个重要主题，这是为了支持Groovy的动态特性。本书作者也认为Groovy是一种多模式语言。

作者基于多年的行业开发经验，认为Groovy语言适合作为学校语言课程，也适合作为有经验开发者的常备开发工具。

衷心希望你通过阅读本书能够很好地掌握Groovy，相信你不会感到遗憾的！

Guillaume Laforge
Groovy Project Manager
JSR-241 Specification Lead

翻译说明：本书主要由龚波和张平共同翻译，龚波负责全文审校，其他对本书翻译和审校做出贡献的人员还包括：陈蓓、王琦、田军、田丽韫、王高翔、牛献忠、张巧莉、龚志翔等。

前　言

本书是脚本语言Groovy的入门书。对于Java开发者来说，Groovy语言会使得编写Java平台的脚本和应用程序变得既快又容易。Groovy语言包含很多在其他脚本语言（诸如Python、Ruby和Smalltalk）中可见的语言特性。因为Groovy是一种基于Java的语言，Groovy语言编写的应用程序可以完全使用Java应用程序编程接口（API）。这意味着，Groovy可以与使用Java语言所编写的框架和组件完美集成。

脚本语言Groovy和系统编程语言Java相互补充，综合使用两者可以加速程序开发过程。比如，可以使用Java语言编写框架和组件，而把Groovy语言用作框架和组件的“黏合剂”。Groovy语言的易用性有利于大大扩展Groovy的应用范围。现在，组件架构、图形化用户界面（Graphical User Interface，GUI）、数据库访问以及因特网应用等日趋重要，这有利于Groovy脚本语言的应用和发展。

Groovy开发者可以充分利用脚本语言所特有的快速应用程序开发特性，Groovy适合处理涉及大量数据或者文件操作的任务，应用程序测试，或者在小型或者中型项目中替代Java语言。

Groovy的语法类似于Java编程语言的语法。这样可以大大缩短Java程序员学习Groovy语言的时间。Java平台的其他脚本语言通常都是基于早期的预处理器概念，这样就存在先天性的发展障碍。但是，Groovy语言“就是”Java语言，提供与Java平台更自然、更无缝地集成。

本书组织方式

本书的内容组织形式注重引导读者快速了解Groovy语言的基本元素，并且假定阅读本书的读者已经阅读过至少一本Java图书。对于本书最后几章，如果读者具有Swing、标准查询语言（Standard Query Language，SQL）、Spring、XML、Ant以及Web应用程序构建等方面的经验，则学习起来会更加自如。作者尽量保证每章目标的单一性和简要性。有些读者也许希望在一章中就可以学习到特定Groovy特性的全部内容，希望本书的这种组织方式能够实现读者的这个目标。同样，每章内容的简短性也有利于读者进行选择性学习，直接跳转到自己关注的内容章节。

本书中很多章都有对应的附录，以更加详细地介绍所讨论的主题。比如，第7章介绍定义和使用Groovy方法的基本概念。附录G更加深入地讨论这个概念，诸如方法的重载和递归，这些更加高级的话题却不是正文的核心内容。同样，这样做也有利于正文中内容简短单一。

本书中很多章也包含很多简短易懂的范例，以说明相应的语言概念。这些范例是完整的，建议读者在学习过程中尝试和试验这些范例。每章最后也有对应的练习，读者应该尝试做这些练习。本书的网站提供每章中范例的源代码，以及练习的解决方案。

本书的一个特色是提供一个不断演化的学习案例——管理和维护一个电子图书馆的借阅库。在本书中，每一章都会逐步完善这个学习案例，使用刚讨论的Groovy新特性来增强这个学习案例的功能。比如，在第11章中，学习案例增加了以前章所介绍的方法、程序闭包（closure），

以及文件处理等内容。

本书第1部分包括第1～16章，介绍Groovy语言的基本特性。比如，讨论Groovy方法、程序闭包、列表、映射，以及对类和继承的支持。

自动化单元测试也是本书的重要讨论主题。Groovy的快速创建-运行循环使之非常适合于开发单元测试。Groovy充分利用工业标准JUnit框架，来简化单元测试的创建和使用过程。单元测试在Groovy中的应用集成了动态类型语言的灵活性和静态类型语言的安全性。为加强说明这一点，本书把单元测试作为大多数学习案例中不可或缺的一部分。

本书的第2部分包括第17～24章，讨论如何使用Groovy创建更加高级的应用程序。比如，使用Spring框架和Cloudscape/Derby关系型数据库管理系统来实现持久性。Groovy提供独特的生成器符号，以支持XML和GUI应用程序。本书最后讨论模板和Web应用程序。

本书约定

本书提供很多与所介绍主题相关的网站和参考书目。

一般情况下不区分“程序”和“脚本”的概念，两个术语可以替换使用。但是，在本书中，这两个概念始终专指Groovy脚本。

软件发布

作者搭建本书的一个支持性网站——http:// www.dcs.napier.ac.uk/~kab/groovy/groovy.html，其中包含本书中所有范例和学习案例的脚本，也提供每章最后练习的答案。

Groovy在不断发展变化着。作者希望通过这个网站让读者有机会了解最新的重要变更。因此，建议读者经常浏览这个网站，了解最新的信息。

感谢

本书作者非常感谢参与Groovy概念设计和开发，以及Java Specification Request (JSR-241) (http:// www.dcs.napier.ac.uk/~cs05/groovy/groovy.html) 的人们。本书的目的是宣扬Groovy语言。我们感谢始终改进Groovy的Guillaume Laforge (Groovy Project Manager)，感谢Andrew Glover (CTO, Vanward Technologies) 在IBM Developer网站发表有关Groovy的文章。同样，非常感谢Jon Kerridge教授 (School of Computing, Napier University, Edinburgh)，他始终鼓励和支持这个项目，并引导我们关注以前没有注意到的领域和挑战。

本书作者也感谢Denise Penrose、Tim Cox、Mary James、Christine Brandt以及他们在Morgan Kaufmann的同事，感谢Elsevier在本书出版过程中给予的帮助。最后，感谢技术评审专家Andrew Glover和Sean Burke，他们提供大量有价值的建议。但是本书中出现的任何错误都是本书作者的责任。

目　录

序言
前言

第1章　Groovy ……1
1.1　为什么使用脚本语言 ……1
1.2　为什么使用Groovy ……2
第2章　数值和表达式 ……3
2.1　数值 ……3
2.2　表达式 ……3
2.3　运算符优先级 ……5
2.4　赋值 ……5
2.5　自增和自减运算符 ……6
2.6　对象引用 ……6
2.7　关系运算符和等于运算符 ……7
2.8　习题 ……8
第3章　字符串和正则表达式 ……10
3.1　字符串字面值 ……10
3.2　字符串索引和索引段 ……10
3.3　基本操作 ……11
3.4　字符串方法 ……11
3.5　比较字符串 ……14
3.6　正则表达式 ……15
3.7　习题 ……16
第4章　列表、映射和范围 ……18
4.1　列表 ……18
4.2　列表方法 ……19
4.3　映射 ……21
4.4　映射方法 ……22
4.5　范围 ……23
4.6　习题 ……24
第5章　基本输入输出 ……26
5.1　基本输出 ……26
5.2　格式化输出 ……27
5.3　基本输入 ……28
5.4　习题 ……30
第6章　学习案例：图书馆应用程序（建模） ……31
6.1　迭代1：需求规范和列表实现 ……31
6.2　迭代2：映射实现 ……32
6.3　习题 ……34
第7章　方法 ……35
7.1　方法 ……35
7.2　方法参数 ……37
7.3　默认参数 ……37
7.4　方法返回值 ……38
7.5　参数传递 ……39
7.6　作用域 ……40
7.7　集合作为参数和返回值 ……42
7.8　习题 ……42
第8章　流程控制 ……44
8.1　while语句 ……44
8.2　for语句 ……46
8.3　if语句 ……47
8.4　switch语句 ……49
8.5　break语句 ……53
8.6　continue语句 ……53
8.7　习题 ……54
第9章　闭包 ……57
9.1　闭包 ……57
9.2　闭包、集合和字符串 ……60
9.3　闭包的其他特性 ……65
9.4　习题 ……68
第10章　文件 ……71
10.1　命令行参数 ……71
10.2　File类 ……71

10.3 习题 ……77
第11章 学习案例：图书馆应用程序（方法、闭包）……79
11.1 迭代1：需求规范和映射实现 ……79
11.2 迭代2：基于文本的用户交互界面的实现 ……83
11.3 迭代3：使用闭包实现 ……85
11.4 习题 ……88
第12章 类 ……89
12.1 类 ……89
12.2 复合方法 ……95
12.3 习题 ……97
第13章 学习案例：图书馆应用程序（对象）……99
13.1 需求规范 ……99
13.2 迭代1：最初的模型 ……99
13.3 迭代2：模型完善 ……101
13.4 迭代3：用户界面 ……106
13.5 习题 ……111
第14章 继承 ……113
14.1 继承 ……113
14.2 继承方法 ……115
14.3 方法重定义 ……117
14.4 多态性 ……117
14.5 抽象类 ……120
14.6 接口类 ……123
14.7 习题 ……126
第15章 单元测试（JUNIT）……130
15.1 单元测试 ……130
15.2 GroovyTestCase类和JUnit TestCase类 ……131
15.3 GroovyTestSuite类和JUnit TestSuite类 ……136
15.4 单元测试的角色 ……138
15.5 习题 ……141
第16章 学习案例：图书馆应用程序（继承）……142
16.1 需求规范 ……142
16.2 迭代1：多态性 ……143
16.3 迭代2：功能性需求演示 ……145
16.4 迭代3：提供用户反馈 ……149
16.5 迭代4：强制性约束 ……157
16.6 习题 ……160
第17章 持久性 ……162
17.1 简单查询 ……162
17.2 关系 ……164
17.3 更新数据库 ……165
17.4 表的对象 ……170
17.5 继承 ……172
17.6 Spring框架 ……173
17.7 习题 ……176
第18章 学习案例：图书馆应用程序（持久性）……177
18.1 迭代1：域模型的持久化 ……177
18.2 迭代2：持久性的影响 ……186
18.3 习题 ……192
第19章 XML构造器和解析器 ……193
19.1 Groovy标记 ……193
19.2 MarkupBuilder ……194
19.3 XML解析 ……196
19.4 习题 ……207
第20章 GUI构造器 ……208
20.1 SwingBuilder ……208
20.2 列表框和表格 ……215
20.3 Box类和BoxLayout类 ……220
20.4 习题 ……221
第21章 模板引擎 ……224
21.1 字符串 ……224
21.2 模板 ……224
21.3 习题 ……228
第22章 学习案例：图书馆应用程序（GUI）……229
22.1 迭代1：GUI原型 ……229
22.2 迭代2：处理器的实现 ……231
22.3 习题 ……238
第23章 服务器端编程 ……239
23.1 Servlets ……239
23.2 Groovlets ……240

此为试读，需要完整PDF请访问：www.ertongbook.com

23.3 GSP页面 ……245
23.4 习题 ……249
第24章 学习案例：图书馆应用程序（WEB） ……250
24.1 迭代1：Web实现 ……250
24.2 习题 ……253
第25章 后记 ……254
附录A 软件发布 ……255
A.1 Java开发工具 ……255
A.2 Groovy开发工具 ……255
A.3 ANT ……255
A.4 Derby/Cloudscape数据库 ……256
A.5 Spring框架 ……256
A.6 Tomcat服务器 ……256
A.7 Eclipse IDE ……256
A.8 本书源文件 ……256
附录B Groovy简介 ……258
B.1 简洁和优雅 ……258
B.2 方法 ……259
B.3 列表 ……260
B.4 类 ……260
B.5 多态性 ……261
B.6 闭包 ……262
B.7 异常 ……263
附录C 关于数值和表达式的更多信息 ……264
C.1 类 ……264
C.2 表达式 ……264
C.3 运算符结合性 ……264
C.4 定义变量 ……265
C.5 复合赋值运算符 ……266
C.6 逻辑运算符 ……266
C.7 条件运算符 ……267
C.8 数字字面值的分类 ……268
C.9 转换 ……268
C.10 静态类型 ……269
C.11 测试 ……270
附录D 关于字符串和正则表达式的更多信息 ……272
D.1 正则表达式 ……272
D.2 单字符匹配 ……273
D.3 匹配开始部分 ……273
D.4 匹配结尾部分 ……273
D.5 匹配零次或者多次 ……273
D.6 匹配一次或者多次 ……273
D.7 匹配零次或者一次 ……274
D.8 次数匹配 ……274
D.9 字符类型 ……274
D.10 选择 ……275
D.11 辅助符号 ……275
D.12 组合 ……276
附录E 关于列表、映射和范围的更多信息 ……278
E.1 类 ……278
E.2 列表 ……279
E.3 范围 ……279
E.4 展开操作符 ……280
E.5 测试 ……280
附录F 关于基本输入输出的更多信息 ……283
F.1 格式化输出 ……283
F.2 类Console ……285
附录G 关于方法的更多信息 ……287
G.1 递归方法 ……287
G.2 静态类型 ……289
G.3 实参协议 ……290
G.4 方法重载 ……290
G.5 默认参数值的不确定性 ……291
G.6 参数和返回值类型为集合的方法 ……292
附录H 关于闭包的更多信息 ……295
H.1 闭包和不明确性 ……295
H.2 闭包和方法 ……295
H.3 默认参数 ……296
H.4 闭包和作用域 ……296
H.5 递归闭包 ……297
H.6 状态类型 ……297
H.7 有关实参的约定 ……298
H.8 闭包、集合和范围 ……298

H.9 Return语句 ……299
H.10 测试 ……300
附录I 关于类的更多信息 ……303
I.1 属性和可见性 ……303
I.2 对象导航 ……307
I.3 静态成员 ……310
I.4 操作符重载 ……312
I.5 调用方法 ……313
I.6 习题 ……315
附录J 高级闭包 ……316
J.1 简单闭包 ……316
J.2 部分应用 ……318
J.3 组合 ……319
J.4 计算模式 ……320
J.5 业务规则 ……322
J.6 打包 ……324
J.7 列表简化 ……330
J.8 习题 ……332
附录K 关于构造器的更多信息 ……334
K.1 AntBuilder ……334
K.2 专用的构造器 ……341
附录L 关于GUI构造器的更多信息 ……345
L.1 菜单和工具条 ……345
L.2 对话框 ……350

第1章 Groovy

本章简要介绍Groovy。Groovy是增强Java平台的唯一的脚本语言。它提供类似于Java的语法，内置映射（Map）、列表（List）、方法、类、闭包（closure）以及生成器。Groovy语言具有动态的弱类型，以及对Java应用程序接口（API）的无缝访问，因此非常适合用于开发小型和中型应用程序。

1.1 为什么使用脚本语言

一般来说，和Java这样的系统编程语言相比，脚本语言（比如Groovy）具有更好的表示能力，能够提供更高的抽象等级。这通常会提供更快捷的应用程序开发能力，以及更高的编程生产力。但是，脚本语言和系统编程语言的目标是不同的。脚本语言用于把应用程序集成起来，而不是实现复杂的数据结构和算法。因此，为了保证实用性，脚本语言必须能够访问不同类型的组件。

通常，脚本语言不会替代系统编程语言，它们相互补充（Ousterhout, 1998）。一般来说，系统编程语言应该用于如下目的：

- 开发复杂的算法或者数据结构。
- 实现计算密集型应用。
- 操作大型数据集。
- 实现良好定义的、缓慢变更的需求。
- 是大型项目的一部分。

而脚本语言应该用于如下目的：

- 连接已有的组件。
- 处理经常变化的多种类型的实体。
- 具有图形化用户界面。
- 拥有快速变化的功能。
- 是小型或者中型项目的一部分。

相对于系统编程语言，脚本语言的主要长处是所需的编码工作量相对少。通常，系统编程语言的代码看起来非常复杂，难以维护。这是因为系统编程语言的代码需要大量的模板或者转换代码。

系统编程语言是强类型的，能够确保代码的安全性和健壮性。在强类型语言中，变量必须指定为一种类型，只能按照固定方式使用。尽管强类型特性使得大型程序的可管理性更好，并且允许编译器（静态地）检测特定类型的错误，但可能有时候起不到类型安全保护作用。比如，当事先很难或者不可能决定变量的类型时，强类型是没有用处的。当连接组件时，这种情况会经常发生。

为简化组件连接任务，脚本语言被设计成弱类型。这意味着，在不同环境下，变量可以以多种方式使用。但是，当代码被实际执行时，才会检测变量是否被非法使用。比如，尽管

Groovy在编译时（静态地）检查程序的语法，（动态地）检测方法调用是否正确发生在运行时。最终结果是，正确编译的Groovy脚本在运行时也许会抛出异常，甚至导致非正常结束。

弱类型并不意味着代码是不安全的，或者不健壮。极限编程（Beck, 2004）已经成为一种软件开发方法。这个方法注重测试，使用全面的单元测试方案（Link, 2003）来驱动开发过程。通过在不同环境下执行所编写的代码，就可以保证代码的安全性和健壮性。当开发Groovy脚本时，单元测试应该是基础的开发过程。实际上，开发经验已经证明，在弱类型语言中，综合运用弱类型和单元测试通常比传统系统编程语言的强类型检测更好（请参考http://www.mindview.net/WebLog/log-0025的相关文档）。这样的话，就同时拥有弱类型的灵活性和单元测试的全面保障。

1.2 为什么使用Groovy

Java编译器会产生可以在Java虚拟机上运行的字节码。Groovy类和Java是二进制兼容的。这意味着，Groovy编译器产生的字节码与Java编译器产生的字节码是完全一样的。因此，对JVM而言，Groovy和Java是完全一样的。这就等于说，Groovy能够完全使用各种Java API，诸如用于数据库开发的JDBC（Fisher et al., 2003），以及用于开发GUI应用程序的Swing（Topley, 1998）。

Groovy的目标是把大量开发者需要做的工作让语言本身来实现。比如，当往GUI添加一个按钮时，只需要提供当按钮被单击时要执行的代码，而无需给这个按钮添加一个事件处理器作为实现特定接口的类的实例。Groovy就是这样做的。

Groovy是一种面向对象的脚本语言，其中涉及的所有事物都是对象，这一点不像Java语言。这样就可以实现语言语法的一致性。Groovy也是动态类型语言，类型标记存在于对象中，而不是由引用它的变量来决定。这样做的结果是，Groovy不要求声明变量的类型、方法的参数或者方法的返回值。这样一来，就可以大大缩短代码规模，并允许程序员把类型决定时间推迟到代码运行时。

通过提供概念“属性”（property），Groovy也尝试统一类中的实例字段和方法。属性概念可以消除实例字段（attribute）和方法之间的差别。结果是，客户端可以把一个属性认为是实例字段及其获取器/设置器（getter/setter）方法的组合。

重要的数据结构，比如Map和List，都是Groovy语言内置的。可以使用Groovy脚本直接表示一个List对象或者Map对象。对于开发新手来说，直接实现List和Map对象会让编程任务更加简单。List和Map对象都提供interator（迭代器）方法，比如each，可以简化处理这些集合中每个元素的过程。可以使用一个closure（闭包）来声明处理过程，闭包是表示一个代码块的对象。这是个非常有价值的结构，可以被变量引用，带参数，被作为参数传入方法或者其他闭包，也可以是类的实例字段。在Groovy编程中，闭包具有举足轻重的地位。

层次性数据结构，比如XML，也可以直接使用Groovy生成器所生成的Groovy脚本来表示。借助于XPath（http:// www.w3.org/TR/xpath20/）中的标记，Groovy可以快速地表示这些结构的路径，以及引用不同部分的方法。同样，迭代器和闭包提供处理它们的机制。

通常，Groovy生成器适用于任何被嵌套的树型结构。比如，它们可以用于描述使用多种组件组装而成的图形化应用程序。闭包可以充当组件（比如菜单项和按钮）的事件处理器。标准查询语言（SQL）的处理过程也是规范统一的。迭代器方法（比如eachRow）与闭包一起可以用来表示如何处理数据库表中的数据行。

第2章　数值和表达式

本章将集中讨论如何使用Groovy来处理基本的数值类型。在这个过程中，尤为重要的是，必须认识到Groovy是一门面向对象的语言。这也就是说，Groovy中每一个事物最终都会被当作某些类的一个实例对象。举例来说，在Groovy中，大家非常熟悉的整数123实际上是Integer类的一个实例。为使用一个对象来实现相关功能，必须调用所属类声明的某个方法。因此，在Groovy环境下，可以使用123.abs() 表达式调用abs方法来获得整数对象123的绝对值。同样，如要获得123的下一个整型值(124)的话，可以调用获取后续值的next方法，如123.next()所示。

正因为如此，在Groovy环境中，如果想得到整数123和整数456的算术之和的话，可以通过调用Integer类的“+”方法来实现，比如123.+(456)表达式。Integer对象456是该方法的参数。当然，和在学校学到的算术技巧相比，这种做法一点也不直观。幸运的是，Groovy同样也支持运算符重载（参见附录I）。这样，+方法就能够作为一个二元操作符出现在其两个参数之间，表达式123+456将更符合人们的习惯。然而，对象调用方法通常最好的做法是，将另外一个对象作为该方法的参数。事实上，在这个例子中，Groovy环境下更为实用的方法就是像123.plus(456)一样调用plus方法。

本章所涉及的算术操作都力求使用简单直观的风格。然而，在学习中，用恰当的形式来描述才是最重要的，本书将在附录C中更详细论述这点。

2.1　数值

Groovy支持整数和浮点数。整型值没有分数部分。浮点数是包含十进制小数部分的十进制数。

整数可能是正数、负数或者零。12345、−44和0是常见的整数形式。就像前面说到的那样，它们都是Integer类的实例。

有小数部分的数值是BigDecimal类的实例。浮点数的例子如1.23，−3.1415926。请注意，浮点数必须避免以小数点开始，防止出现混淆，比如必须使用0.25，而不是.25来表示浮点数0.25。同样，它的负数也必须使用−0.25来表示。

附录C会更深入讨论这些简单数值的类。

2.2　表达式

Groovy提供大量适用于数值类型的运算符，包括常见的算术运算符、比较运算符、位运算符，以及其他各种类型的运算符。表达式（expression）通常用来描述某些计算行为，由运算符和操作数组成。算术运算符（arithmetic operator）包括加法（+）、减法（−）、乘法（*）和除法（/）。Groovy同样也支持取模运算符，用百分号表示（%）；取模运算用来计算两个整数相除的余数。表2-1列举适用于整数的各种算术运算符。

表2-1 整数运算符

表达式	调用方法	结果	表达式	调用方法	结果
5 + 3	5.plus(3)	8	5/3	5.divide(3)	1.6666666667
5 − 3	5.minus(3)	2	5%3	5.mod(3)	2
5*3	5.multiply(3)	15			

请注意，两个整数的除法运算通常会产生一个浮点数，即使其结果可能是一个整数。举例来说，表达式6/3的结果是浮点数2.0，而不是整数2。

上面的这些运算符也适用于两个浮点数的运算，对应结果如表2-2所示。接下来的表会说明两个浮点数的取模操作。

表2-2 浮点数运算符

表达式	调用方法	结果	表达式	调用方法	结果
5.0 + 3.0	5.0.plus(3.0)	8.0	5.0*3.0	5.0.multiply(3.0)	15.0
5.0 − 3.0	5.0.minus(3.0)	2.0	5.0/3.0	5.0.divide(3.0)	1.6666666667

同上，除了取模运算符之外，Groovy的算术运算也适用于整数和浮点数的组合。表2-3给出一些范例。

表2-3 混合运算

表达式	调用方法	结果	表达式	调用方法	结果
5 + 3.2	5.plus(3.2)	8.2	5 * 3.2	5.multiply(3.2)	16.0
5.6 + 3	5.6.plus(3)	8.6	5.6 * 3	5.6.multiply(3)	16.8
5 − 3.2	5.minus(3.2)	1.8	5 / 3.2	5.divide(3.2)	1.5625
5.6 − 3	5.6.minus(3)	2.6	5.6 / 3	5.6.divide(3)	1.8666666667

如上述几个表所示，不管整数组合，还是整数和浮点数组合，除法运算通常都会得到相同的结果。下面所有组合的结果都是2.6：

```
13.0 /5
13 /5.0
13 /5
```

为了获得两个整型值相除的整数部分，必须调用intdiv方法：

```
13.intdiv(5)
```

该表达式的结果是整数2。

使用取模运算符(%)可以得到两个整数操作数相除的余数，因此，

```
13%5    结果值为3
15%5    结果值为0
```

请注意，对一个浮点数求模，或者对一个含有浮点数参数的整数求模都是非法的。因此，表达式13.0%5.0、13.0%5、13%5.0都会提示mod方法被错误地调用。

此为试读，需要完整PDF请访问：www.ertongbook.com