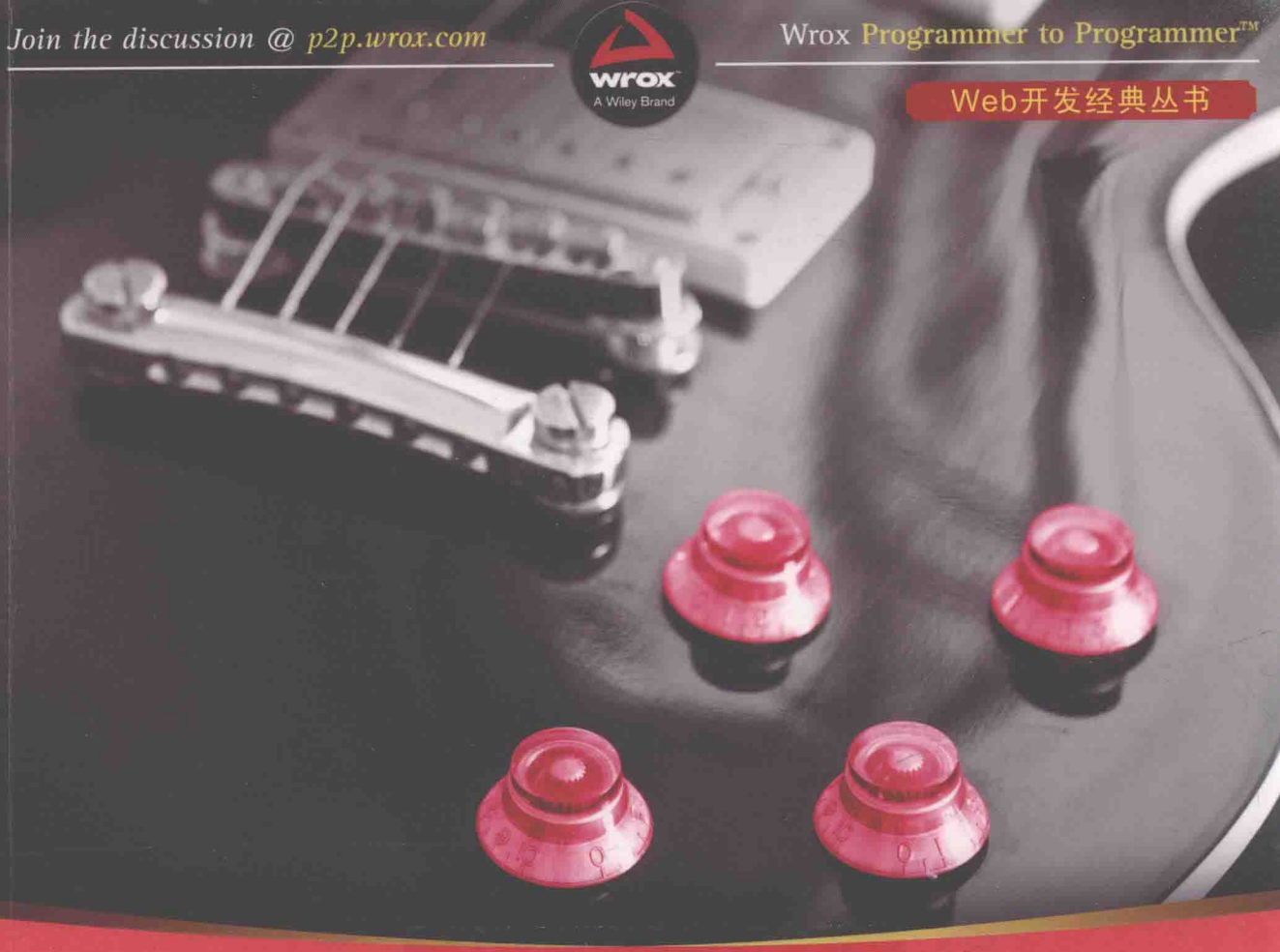
Join the discussion @ p2p.wrox.com

Wrox Programmer to Programmer™

Web开发经典丛书



Professional Clojure

Clojure高级编程

Jeremy Anderson
Michael Gaare
[美] Justin Holguín 著
Nick Bailey
Timothy Pratley
蒋 楠 译

清华大学出版社

Web 开发经典丛书

Clojure 高级编程

Jeremy Anderson
Michael Gaare
[美] Justin Holguín 著
Nick Bailey
Timothy Pratley
蒋楠 译

清华大学出版社
北京

Jeremy Anderson, Michael Gaare, Justin Holguín, Nick Bailey, Timothy Pratley
Professional Clojure
EISBN: 978-1-119-26727-0

北京市版权局著作权合同登记号 图字：01-2016-9508

图书在版编目(CIP)数据

Clojure 高级编程/(美)杰里米·安德森(Jeremy Anderson) 等著；蒋楠 译. —北京：清华大学出版社，2017
(Web 开发经典丛书)
书名原文：Professional Clojure
ISBN 978-7-302-47111-0

Ⅰ. ①C…　Ⅱ. ①杰… ②蒋…　Ⅲ. ①JAVA 语言—程序设计　Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字(2017)第 103623 号

责任编辑：王　军　韩宏志
装帧设计：孔祥峰
责任校对：曹　阳
责任印制：王静怡

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　印　　张：16.75　　字　　数：397 千字
版　　次：2017 年 6 月第 1 版　　印　　次：2017 年 6 月第 1 次印刷
印　　数：1~3000
定　　价：49.80 元

产品编号：072433-01

译 者 序

作为一门植根于 JVM 平台的函数式编程语言，Clojure 近年来受到越来越多的关注。无论 C++11 还是 Java 8，都提供针对函数式编程的语法支持。由于不存在可变状态，函数是引用透明的，没有副作用，程序代码易于推理和调试。

无论学习哪种语言，“它能做什么”是不少用户都会问的问题。全球零售业巨头沃尔玛采用 Clojure 开发数据管理系统，自动化运维管理服务提供商 Puppet 采用 Clojure 创建 Trapperkeeper 框架，一站式后端云服务提供商 LeanCloud 采用 Clojure 构建存储 API、推送等核心功能。根据 Clojure 官方网站的统计，目前已有花旗银行、Facebook、LINE、Netfix、Red Hat 等近 300 家企业在自己的系统中采用了 Clojure 或 ClojureScript。此外，在空前重视人工智能的今天，使用 Clojure 进行数据挖掘也是一个热门话题。

任何一门语言都有其拥趸和质疑者，“PHP 是最好的语言”也并非总是笑谈。Clojure 的优点是简洁、专注和实用，但由于 JVM 较慢的启动速度，Clojure 在处理日常事务时或许不那么得心应手。加之 Clojure 和 JVM 平台的绑定是如此之深，要想真正掌握 Clojure，就需要深入理解 Java，这在一定程度上提高了 Clojure 的学习成本。

摆在读者面前的这本书并不厚，却凝结了业内多位专家的宝贵经验。作者尝试以简洁的语言和丰富的示例阐述 Clojure 的独到之处，并培养读者从函数式编程的角度思考问题。不过，本书的目标群体是具有一定经验的开发人员，缺少相关背景的读者可先参阅其他资料，以夯实基础。无论官方文档、论坛博客、主题演讲还是 GitHub，都是很好的学习资源。“一万小时定律”或许并非放之四海而皆准，但唯有不断实践，才能真正掌握一门语言。

非常感谢清华大学出版社的王军老师给予译者翻译本书的机会，他为本书中文版面世所付出的辛勤劳动令译者十分感动。由于译者水平有限，疏漏之处在所难免，恳请读者批评指正，也希望读者提出宝贵意见和建议。

蒋 楠

作者简介

Jeremy Anderson 就职于美国密歇根州的 Code Adept，这是一家提供高品质软件交付的咨询公司，业务涵盖软件开发、敏捷教导与培训服务。Jeremy 是一名 Clojure 爱好者，对多种 Clojure 库的开发都有贡献。Jeremy 对向用户提供编程培训极为热心，并作为志愿者在当地中学协助讲授计算机课程。

Michael Gaare 就职于美国一家提供金融技术服务的初创公司 NextAngles，担任平台技术负责人。从 2012 年起，Michael 就采用 Clojure 开发专业的 Web 服务、数据处理系统与各种库(而非框架)。Michael 爱好参加歌剧演出，大部分闲暇时间都与妻子和两个女儿度过。

Justin Holguín 在美国波特兰的 Puppet Labs 担任软件工程师，负责 Clojure 后端服务的开发。Justin 热爱函数式编程，对高级类型系统、基于属性的测试等能够提高软件稳定性的技术情有独钟。

Nick Bailey 是一名 Clojure 爱好者，也负责 Clojure java.jmx 库的维护。Nick 在总部位于美国加州的 DataStax 担任软件架构师，使用 Clojure 开发用于管理分布式数据库的企业级软件。Nick 从 2010 年起开始接触 Clojure，并由此成为这门语言的拥护者。

Timothy Pratley 从 2008 年起开始使用 Clojure，是这门语言的贡献者和倡导者。Timothy 目前就职于美国旧金山的 Outpace Systems，负责开发基于 Clojure、ClojureScript 和 Clojure Android 的解决方案。Timothy 已有 15 年的专业软件开发经验，接触过许多编程语言、框架和数据库，热爱 Clojure、Datomic 数据库、结对编程(pair programming)[1]，喜欢思考。

[1]一种敏捷软件开发方法，两名程序员在一台计算机上共同工作，一人负责输入代码，另一人负责审查代码。——译者注

技术编辑简介

Justin Smith 是一名活跃于 Clojure 社区的全职 Clojure 开发人员，他将全部精力都投入到 Clojure 开发中。

Zubair Quraishi 来自丹麦，是一名 UX/设计师与营销黑客。在过去 20 年中，Zubair 建立的初创公司已有两家被收购，并投资了超过 30 个初创团队。Zubair 从 2011 年起开始接触 Clojure 和 ClojureScript。Zubair 曾在美国、欧洲、亚洲的多家初创公司和财富 500 强企业工作，他的博客是 www.zubairquraishi.com。

Alex Ott 就职于德国帕德博恩的 Intel Security(其前身为 McAfee)，担任软件架构师，负责信息安全方面的工作。从 2009 年 Clojure 1.0 版发布起，Alex 就使用这门语言开发多种原型、内部服务与开源项目(如 Incanter)。

Doug Knight 具有 18 年的专业编程经验，是 Microsoft 技术方面的专家。2014 年，Doug 加入总部位于美国华盛顿的 LivingSocial，开始接触 Ruby on Rails。从 2015 年起，Doug 开始在工作中使用 Clojure 进行开发。

致　谢

首先，Jeremy 感谢上帝赐予他能随心所欲做自己喜欢的事情的权利。其次，Jeremy 感谢家人对他的支持和理解，使他可以在晚上和周末将自己关在办公室里疯狂写作。Jeremy 还要感谢 Christina Rudloff 和 Troy Mott 在本书筹备过程中所做的辛勤工作，以及邀请他加入写作团队。正是由于所有作者的共同努力，本书才得以面世。最后，感谢所有技术审稿人在百忙之中抽出时间阅读书稿，并提出宝贵的意见和建议。

Nick 感谢所有为本书面世付出努力的人士：负责组织工作的 Troy 和 Christina、其他撰写和审校的作者以及提供宝贵反馈的技术审稿人。Nick 还要感谢 DataStax，使他有机会撰写这样一部专业的 Clojure 著作。

Michael 感谢 Lara、Charlotte 与 Juliette 对他的关爱、支持和理解，以及 Keith 提供的宝贵帮助。Michael 还要感谢 Christina 和 Troy 的耐心，并给予他撰写本书的机会。此外，感谢 Rich 为写作提供的有趣素材，与他合作实乃人生快事。

Justin 感谢家人对他的引导以及莫大的帮助，鼓励他投身写作和编程之中。Justin 还要感谢他在 Puppet Labs 的诸多好友和同事，他们的鼓舞让 Justin 坚定了学习 Clojure 的信心，通过不断努力，最终成为这门语言的专家。

Timothy 感谢最终合作伙伴 Shin Nee。此外，还要感谢读者对编程发展做出的贡献，以及 Clojure 社区创造了这样一个友善、有益和愉悦的生态系统。最后，Timothy 感谢父母为他的人生提供了诸多机会。

前　言

Clojure 一瞥

Clojure是一种动态、通用的程序设计语言，既有脚本语言易于学习和交互开发的特点，又具备适合多线程编程的高效和强健的基础架构。虽然Clojure属于编译语言，却是完全动态的，所有特性都能在运行时得到支持。借助可选的类型提示和类型接口，Clojure可以方便地访问Java框架，确保在调用时不会出现Java反射。

Clojure 是一种 Lisp 方言，继承了 Lisp“代码即数据”的设计理念以及功能强大的宏系统。总体而言，Clojure 属于函数式编程语言，包括丰富的不可变和可持久化数据结构。当需要处理可变状态时，Clojure 通过软件事务内存与响应式 Agent 系统，确保实现清晰、正确、多线程的设计。

——Rich Hickey，Clojure 作者

上述介绍来自 Clojure 的作者 Rich Hickey，点出了这门语言的本质。许多人将 Clojure 等同于函数式编程，不过与其前身 Lisp 类似，Clojure 实际是一门通用语言，能支持任何编程范式。

然而，Clojure 的确具有浓郁的函数式特点。它侧重于不可变值和可持久化数据结构，为函数式编程提供有力支持。读者或许对 Clojure 也能用于面向对象编程表示惊讶，本书将对此进行讨论。

目标读者

本书是为有经验的程序员准备的。读者应掌握至少一门编程语言，并了解 Clojure 的基本语法和概念，做好在更高层次上运用 Clojure 的准备。本书旨在实现读者从 Clojure 初学者向 Clojure 开发者的蜕变。学习 Clojure 不仅涉及全新的语法，所用的工具和构造可能与读者之前的体验大相径庭。

程序源代码

读者可以从 Wiley 网站 www.wiley.com/go/professionalclojure 下载源代码，也可以访问 GitHub 上的 Clojure 项目：https://github.com/backstopmedia/clojurebook。

一门功能强大的程序设计语言不仅可以引导计算机执行既定的任务，也是一个能体现开发人员进程思路的框架。

——《计算机程序的构造和解释》(SICP)[1]

本书假定读者已具备程序设计和 Clojure 的基础知识，但尚未达到精通 Clojure 的程度。本书涵盖的范围较广，包括改变读者的思考方式和编程习惯，如何将 REPL 集成到用户的开发程序中，以及采用 Ring 和 ClojureScript 构建实际的应用。

本书架构

本书将提供若干实际的示例，以帮助读者将所学的Clojure知识运用到日常的编程工作中，而非纸上谈兵式的空谈。

第 1 章

该章将介绍 Clojure 在程序设计中的独特视角。读者将了解 Clojure 有别于其他语言的原因，如为何默认使用不可变性、Clojure 为何也能用于面向对象编程等。

第 2 章

该章将介绍精通 REPL 所需的各种知识，以及 REPL 与实际应用交互时的技巧和技术。读者将学习如何从 REPL 运行代码和测试，以及编写便于从 REPL 重载而不需要重启的代码。

第 3 章

该章将讨论采用 Compojure 创建 Web 服务，以及所涉及的相关概念(如路由、处理器、中间件等)。读者将开发一个完整的 Web 服务，并学习与部署应用有关的各种技术。

第 4 章

该章将讨论 Clojure 中的测试，并重点介绍 clojure.test 测试库。读者将学习在各种常见测试场景中采用的技术，以及评估代码质量的工具。

第 5 章

该章将讨论如何使用 ClojureScript 开发一个类似于 Trello 的任务管理系统，并介绍如何在服务器端与客户端应用之间实现函数共享。

第 6 章

该章将介绍 Datomic 数据库，并讨论后者如何将不可变的概念应用到数据库中。读者

[1]英文书名为 *Structure and Interpretation of Computer Programs*，1984 年出版，1996 年修订为第 2 版。该书是一部经典的计算机著作，曾长期作为麻省理工学院(MIT)电子工程与计算机科学系的专业基础课教材。——译者注

将学习如何在 Datomic 中进行数据的建模和提取，并运用所学的知识创建一个能支持任务管理系统(在第 5 章开发)的数据库。

第 7 章

该章将讨论 Clojure 的性能，以及如何提高 Clojure 代码的执行速度。利用某些技巧，可以让 Clojure 代码的执行速度与 Java 代码一样快。

所需工具

俗话说："工欲善其事，必先利其器"。幸运的是，学习和使用本书示例仅需要 Java、Leiningen 以及一种优秀的文本编辑器。

Java

目前，大部分计算机都预装了 Java。不过为运行本书中的示例，读者应确保安装 Java 的最新版本。书中的代码示例在 JDK 1.8.0_25 下编写并测试通过。请浏览 Oracle JDK 下载页面[2]中的文档，以获得在不同平台下载和安装 JDK 的信息。

Leiningen

根据官网[3]的描述，Leiningen 是对 Clojure 贡献最大的项目。对具有 Java 背景的读者而言，Leiningen 之于 Clojure，如同 Maven 之于 Java，只是少了各种 XML。用户可以从容构建 Clojure 项目，避免陷入令自己心急火燎的窘境。Leiningen 能管理项目依赖项，并声明性地描述项目和配置，还提供代码分析、自动化等大量实用的插件。Leiningen 可以有效提高 Clojure 的使用体验。

幸运的是，配置和运行 Leiningen 相当简单。读者应安装 Leiningen 的最新版本，截至本书写作时，其最新版本为 2.5.3。请访问 Leiningen 网站，以获得在不同编程环境下配置 Leiningen 的信息。

编辑器

完成 Leiningen 的安装后，应选择一种优秀的文本编辑器，以提高编写 Clojure 代码的效率。如果读者已有熟悉的编辑器，继续使用它们即可。但是，如果所用的编辑器不支持括号平衡、REPL 集成、语法高亮、代码缩进等基本功能，应考虑换用以下编辑器。

Emacs

Emacs 为众多资深程序员所青睐，它与 Lisp 的历史一样悠久。尽管上手不易，但 Emacs

[2]参见 http://www.oracle.com/technetwork/java/javase/downloads/index.html

[3]参见 http://leiningen.org

被广泛认为是一种功能极强大的编辑器，其可扩展性无出其右。目前有许多有助于改善学习难度的自定义 Emacs 配置，如 Emacs Prelude[4]。后者合理的默认配置，可以用于包括 Clojure 在内的多种语言的开发。

LightTable

LightTable[5]最初是 Kickstarter[6]上的一个项目，现已发展为一个集代码编辑器、REPL、文档浏览器于一身的 IDE。LightTable 功能强大，目前在 Clojure 社区备受推崇。

Cursive (IntelliJ)

如果读者熟悉某种 JetBrains IDE 的使用，会很高兴看到这款名为 Cursive 的 IntelliJ 插件[7]。Cursive 不仅能很好地与 nREPL 进行集成，也提供强大的重构支持，以及调试和 Java 互操作性等功能。

Counterclockwise (Eclipse)

熟悉 Eclipse 的读者可以考虑使用 Counterclockwise[8]，后者既是一款 Eclipse 插件，也可以作为一个独立的编辑器使用。Counterclockwise 具备与上述几种编辑器相同的特性，以及 REPL 集成、内联代码评估等功能。

本书约定

为帮助读者在阅读本书的过程中获取最多信息，并随时了解当前处理的事项，本书使用了许多约定。

注意：

列出注释、提示、暗示、技巧或对当前讨论的弦外之音。

本书通过两种方式来显示代码：

- 对于大多数代码示例，使用没有突出显示的等宽字体来表示。
- 对在当前上下文中特别重要的代码，用粗体字来强调显示。

[4]参见 https://github.com/bbatsov/prelude

[5]参见 http://lighttable.com

[6]著名的众筹平台，众多初始项目可以通过 Kickstarter 筹集所需资金。LightTable 曾获评 Kickstarter 十大最有技术含量的项目。——译者注

[7]参见 https://cursive-ide.com

[8]参见 http://doc.ccw-ide.org

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果你在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

请给 wkservice@vip.163.com 发电子邮件，我们就会检查你的反馈信息，如果是正确的，我们将在本书的后续版本中采用。

要在网站上找到本书英文版的勘误表，可以登录 http://www.wrox.com，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，单击 Book Errata 链接。在这个页面上可以查看到 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/misc-pages/booklist.shtml。

p2p.wrox.com

要与作者和同行讨论，请加入 p2p.wrox.com 上的 P2P 论坛。这个论坛是一个基于 Web 的系统，便于你张贴与 Wrox 图书相关的消息和相关技术，与其他读者和技术用户交流心得。该论坛提供了订阅功能，当论坛上有新的消息时，它可以给你传送感兴趣的论题。Wrox 作者、编辑和其他业界专家和读者都会到这个论坛上来探讨问题。

在 http://p2p.wrox.com 上，有许多不同的论坛，它们不仅有助于阅读本书，还有助于开发自己的应用程序。要加入论坛，可以遵循下面的步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接。

(2) 阅读使用协议，并单击 Agree 按钮。

(3) 填写加入该论坛所需要的信息和自己希望提供的其他信息，单击 Submit 按钮。

(4) 你会收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

注意：

不加入 P2P 也可以阅读论坛上的消息，但要张贴自己的消息，就必须加入该论坛。

加入论坛后，就可以张贴新消息，响应其他用户张贴的消息。可以随时在 Web 上阅读消息。如果要让该网站给自己发送特定论坛中的消息，可以单击论坛列表中该论坛名旁边的 Subscribe to this Forum 图标。

关于使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作情况以及 P2P 和 Wrox 图书的许多常见问题。要阅读 FAQ，可以在任意 P2P 页面上单击 FAQ 链接。

源代码

在读者学习本书中的示例时，可以手工输入所有的代码，也可以使用本书附带的源代码文件。本书使用的所有源代码都可从本书合作站点 http://www.wrox.com/。登录站点 http://www.wrox.com/，使用 Search 工具或使用书名列表就可以找到本书。接着单击本书细目页面上的 Download Code 链接，就可以获得所有的源代码。还可以直接访问 www.wiley.com/go/professionalclojure 或 https://github.com/backstopmedia/clojurebook 下载本书源代码。也可以访问 www.tupwk.com.cn/downpage，输入本书中文书名或中文 ISBN，下载各章的源代码。此外，可扫描本书封底的二维码，直接下载。

注意：

由于许多图书的标题都很类似，所以按 ISBN 搜索是最简单的，本书英文版的 ISBN 是 978-1-119-26727-0。

下载代码后，只需要用自己喜欢的解压缩软件对它进行解压缩即可。另外，也可以进入 http://www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

目　录

第 1 章

保持初学者的心态

本章内容

- 理解命令式编程和函数式编程的区别
- 学习从函数式编程的角度思考问题
- 介绍 Clojure 对待面向对象编程的独特视角

若保持通透的头脑，就能随时准备好去接受，并对所有的可能性敞开大门。初学者的脑中总是存在无限可能，而专家的脑中却只有零星几个。

——铃木俊隆[1]

对于在过去 30 年中流行的大部分编程语言来说，它们的共通之处超过了相互之间的差异。事实上，一旦掌握了一门语言，学习另一门语言就并非难事。开发人员只需要掌握两门语言在语法上的细微差别，并了解前一门语言所不具备的某些新特性。由于当今不少最流行的语言存在诸多相似之处，一名“掌握多门语言的程序员”并不鲜见。

然而，Clojure 与大部分目前流行的语言完全不同，它从 Lisp 编程语言发展而来。与基于 C 的语言相比，Clojure 的语法和编程风格大相径庭。读者必须舍弃“所有语言都是相通的”这一先入为主的观念，才能更好地学习 Clojure 以及 Lisp 家族的其他语言。

请读者忘记所有先前的编程经验，将 Clojure 当作入门的第一门编程语言。唯有如此，才能领会这门语言的精髓。否则，读者只是学习了若干新语法，写出来的 Clojure 代码风格将偏向 Java/C/Ruby，而背离了 Clojure 原本的设计思想。学习 Clojure/Lisp 甚至将影响读者使用其他语言的方式，特别是在 Java 8 和 Scala 变得越来越流行的今天。

---

[1]铃木俊隆(1905-1971)，将禅宗思想介绍到西方的日本僧人，全球最有影响力的禅宗大师之一。其著作《禅者的初心》(*Zen Mind, Beginner's Mind*)备受苹果公司前 CEO 史蒂夫·乔布斯的推崇。——译者注