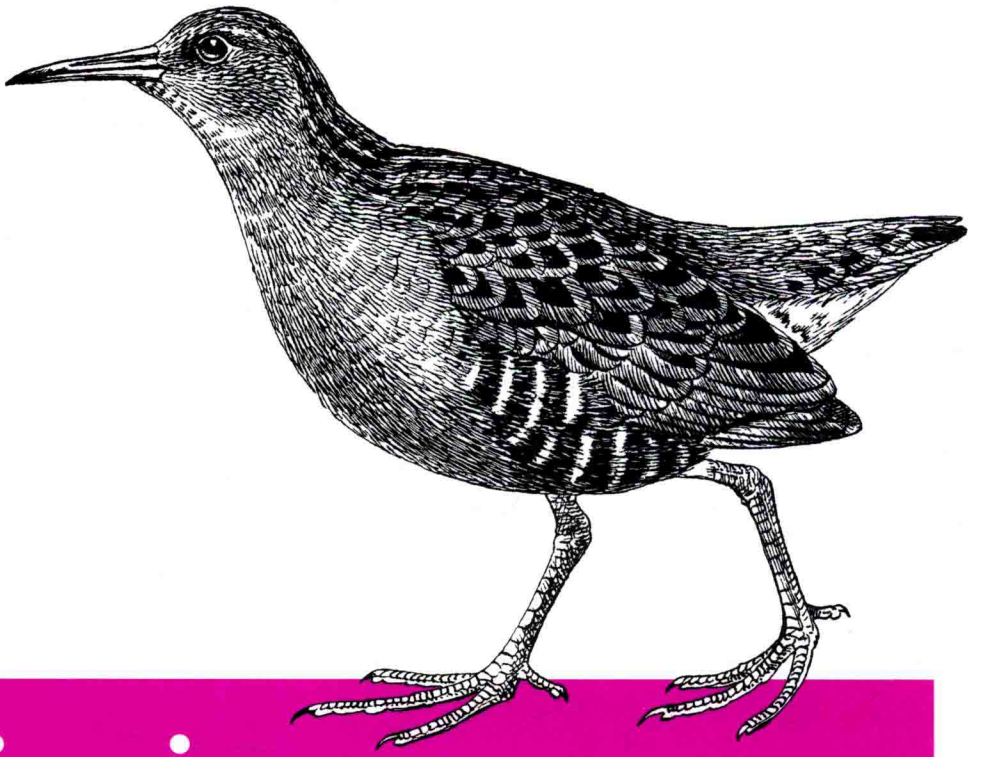
O'REILLY®



Living Clojure (中文版)

Living Clojure

Carin Meier 著
吴滠栩 译

中国电力出版社

Living Clojure（中文版）

Carin Meier 著

吴滠栩 译

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, 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

Living Clojure：中文版/（美）梅尔（Meier, C.）著；吴滠翎译. —北京：中国电力出版社，2016.2

书名原文：Living Clojure

ISBN 978-7-5123-8415-6

I. ①L… II. ①梅… ②吴… III. ①程序语言－语言设计 IV. ①TP312

中国版本图书馆CIP数据核字（2015）第240724号

北京市版权局著作权合同登记

图字：01-2015-6001号

封面设计/ Ellie Volckhausen，张健

出版发行/ 中国电力出版社（http://www.cepp.sgcc.com.cn）

地　　址/ 北京市东城区北京站西街19号（邮政编码100005）

经　　销/ 全国新华书店

印　　刷/ 北京丰源印刷厂

开　　本/ 787毫米×980毫米　16开本　14.125 印张　264 千字

版　　次/ 2016年2月第一版　2016年2月第一次印刷

印　　数/ 0001－3000册

定　　价/ 48.00元（册）

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

献给Jim Weirich（1956～2014），你对生活和学习的热爱鼓舞了如此多的人。

目录

第2部分 践行Clojure练习项目

前言

本书的灵感有两个来源。第一个，也是最直接的，是我多年来在Clojure上的学习、探索，以及最终的专职Clojure工作的经历。第二个，不那么直接，是我第一次坚持跑步的经历。我第一次单凭自己的尝试是一个彻底的失败。我以为大概我不是跑步的料。幸运的是，有人给我介绍了一个很棒的手机应用叫“从0到5公里”（Couch to 5k）。应用的创建者们认识到，人们没能达到自己的健身目标最常见原因是，他们过于图多和追求快。为了解决这个问题，他们开发出一个渐进的、持续8周的训练项目，旨在帮助用户达到不需中断连续跑步30分钟的目标。尽管这并不容易，我还是努力完成了这个项目并能坚持跑30分钟了。

几周之后，在参加完一个社区用户组的活动后，我跟开发者们在一起。话题转到了Clojure上。我们中有些人在讨论是多么享受这门语言，可是有一个朋友却在抱怨他曾尝试学习Clojure，但最后仍不得要领。我发现，他的学法是想在一个周末就学完Clojure。图多、图快。学习如何跑步的问题，与学习一门新的语言中的问题，这两者看起来没有多大的不同。学习一门新语言的过程，即是学习以一种新的方式来思考，训练大脑以一种新的方式来处理输入和解决问题。这不是一蹴而就的。像跑步训练一样，在大量的实践之后，水到渠成。

本书融合了这些灵感。因此本书既是学习Clojure的入门，也是一个自成体系的练习项目，给你的大脑建立一种新的思考方式。

本书为谁而准备

你正在寻找一本平缓的Clojure入门书籍吗？你是否已经具有其他编程语言的背景？太好

了！这本书就是为你而准备的。我们将会涉及跟Clojure有关的编程概念，以及它们的独特设计。为了与函数式语言做比较，我们也会提及面向对象编程及其一些概念。在书中，我们将假设你已经对面向对象开发比较熟悉。因此，如果你熟悉另一门编程语言比如Ruby或Java，你应当感觉很好。如果你是一个编程初学者，本书最好跟一本介绍编程概念的入门书或参考书搭配使用。

如果你是一位语言专家，或者想深挖Clojure的每一个细节，那么本书很可能不适合你。我们将以“学习如何以Clojure的方式思考”为目标，把精力集中于这门语言的一些主要方面，而不是细枝末节上。在介绍Clojure生态系统中的一些工具或库的时候，同样遵循这一理念。我们力争把它写成这样的一本入门书：专注这种语言常见且务实的部分，提供全面而整体的视野，使你读完之后会感到自信，并准备开始践行Clojure，甚至以Clojure为工作。

如何使用本书

为了能够践行Clojure，最终胜任Clojure的工作，本书的结构围绕着对知识和实践的培养而建立的。正如本书的创意一样，本书由两部分组成。第一部分是Clojure编程语言的简单性和威力的一个观光之旅。它将涉及Clojure有用的库和常见用法。第二部分是践行Clojure练习项目。它将引领你完成一个按周来划分的练习项目，旨在帮助你能够上手，乃至成长为一名Clojure开发者所需要的实践、知识和工具。

本书的前一半将介绍这种语言，并伴随着一些代码示例。有一些重要的东西需要记住，这些在下面的几节中列出。

一定要动手实践这些示例

我知道想尽可能快地浏览完本书是一种诱惑。但实际上，输入这些代码示例，并亲眼见证奇迹，将有助于增进你的理解。花点时间亲自练习一下新概念和新命令。在前言的后面部分，我们将谈到如何建立你的环境以便运行Clojure。

不必感到压力大

我们在本书的前半部分将涉及大量的内容。这不是一下子都能消化了的。没关系。实际上，本书就是这么设计的。本书的后半部分是建立在总体印象之上的一个实践项目，因此基本的语言概念可以在使用中真正地得到消化吸收。到了你开始这个练习项目，问题开始冒出来的时候，你将拥有知道去哪里搜寻答案的所有工具和一般知识。

再次强调，学习以一种新的方式来思考需要时间，要有耐心！

不必担心括号

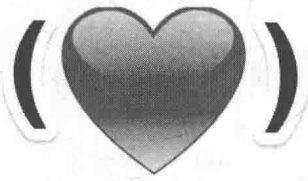
人们在初看Clojure时一个普遍的、首要的担心就是这些括号。

不必担心括号。

真的。

在你通过一个支持paredit模式（即总是会为你插入成对括号的模式）的好编辑器进入Clojure之后，括号的问题将不复存在。在后面的第4章中我们将涉及选择和设置一个编辑器。

括号结构带来的简单和优雅是Clojure的主要优势之一。实际上，一些人甚至觉得括号是给他们代码的拥抱。



还有一件事——玩得开心！

Clojure是一门用起来讨人喜欢的语言。学习一门新语言是一次令人兴奋的历险，处处是奇观。拥抱它吧！

在别的地方，代码示例有时候是枯燥的，晦涩的，满是无意义的数字，而且坦白地说，是很无聊的。本书力争避免这个问题，利用故事的力量来增强代码示例。在故事中我们的大脑会变得更活跃。实际上，有研究表明，当问题以讲故事的方式提出时，学习成效的确会得到提升。因而，我们准备召唤一个著名的故事来贯穿全书陪伴我们：Lewis Carroll（刘易斯·卡罗尔）的《爱丽丝梦游仙境》。

随着你探索Clojure，爱丽丝梦游仙境的故事也将会被放入到代码示例中。在你的旅途中，她的历险故事将时刻提醒着你要放松、微笑，以及玩得开心。

使用本书你需要什么

在开始一次旅行之前，我们需要整理手提箱并做好准备。很幸运，为开始我们的历险之旅，我们并没有太多需要做的。

安装Java

如果你的电脑还没有安装Java，你需要获取它。你可以检查看看是否已经设置好了，打开命令行提示符并输入：

```
java -version
```

你应能看到类似这样的输出：

```
java version "1.7.0_60"
Java(TM) SE Runtime Environment (build 1.7.0_60-b19)
Java HotSpot(TM) 64-Bit Server VM (build 24.60-b09, mixed mode)
```

如果你看到版本号是1.6或更高，你就可以继续了。如果不是，你可以从*http://www.oracle.com/technetwork/java/javase/downloads/index.html*下载最新版本。

注意： 等一下。为什么我需要Java？我要学的是Clojure。

Clojure运行于Java虚拟机（JVM）之上，这是一个成熟而且健壮的平台，很多大企业都使用。这是一个适合运行快速、可伸缩和高可靠程序的极好的环境。

你不需要懂得Java就可以写Clojure代码。可是，当你想要的时候你也可以使用Java的类和库，并与之交互。在先对这门语言稍作探索之后，你就会学到如何做到这些。

既然你已经安装好了Java，离你准备好就只差一件事了：Clojure REPL。

准备Clojure REPL

Clojure有一个重要的交互性工具叫做REPL，表示“read-eval-print loop”。一旦你进入了Clojure（读取－运算－打印循环）REPL，你就身处Clojure语言中了。你能够输入Clojure代码，按下回车键，REPL就会为你运算并打印出结果。这是你探索这门语言的钥匙。

让我们现在就试用一下Clojure REPL。贯穿全书，我们将使用REPL来试验和探索代码。在最前面几章，我们将仅在REPL中试验代码。随着我们进入到创建Clojure项目的第4章，我们将涉及不同的编辑器和工具。现在，我们将以最简单的方式来试用REPL，即使用Leiningen，这是一个创建Clojure项目的流行的工具。

遵循这些步骤，以便让REPL运行起来：

1. 访问Leiningen站点（*leiningen.org*），并按照提示下载Leiningen。
2. 创建一个名为wonderland的新项目，命令如下：

```
-> lein new wonderland
```

3. 进入新创建的项目，使用命令：

```
-> cd wonderland
```

4. 运行以下命令，启动Clojure REPL：

```
-> lein repl
```

你将会看到以下输出：

```
nREPL server started on port 65247 on host 127.0.0.1 - nrepl://127.0.0.1:65247
REPL-y 0.3.1
Clojure 1.6.0
    Docs: (doc function-name-here)
          (find-doc "part-of-name-here")
  Source: (source function-name-here)
 Javadoc: (javadoc java-object-or-class-here)
```

```
   Exit: Control+D or (exit) or (quit)
 Results: Stored in vars *1, *2, *3, an exception in *e

user=>
```

最后一行就是等待你输入的REPL提示符。试着输入以下内容并按回车键：

```
(+ 1 1)
```

你应该看到2作为结果显示在下一行：

```
nREPL server started on port 65361 on host 127.0.0.1 - nrepl://127.0.0.1:65361
REPL-y 0.3.1
Clojure 1.6.0
    Docs: (doc function-name-here)
          (find-doc "part-of-name-here")
  Source: (source function-name-here)
 Javadoc: (javadoc java-object-or-class-here)
    Exit: Control+D or (exit) or (quit)
 Results: Stored in vars *1, *2, *3, an exception in *e

user=> (+ 1 1)
2
user=>
```

REPL只是读取你的输入，运算它，并为你打印出结果。

贯穿本书，我们都将按此来组织代码，而且代码的返回值将显示为带两个分号后跟着一个箭头，如下所示：

```
(+ 1 1)
;; -> 2
```

在Clojure中分号表示一个注释。当到了运算这段代码的时刻，那一行分号以后的任何内容都会被忽略。如果你使用的是本书的电子版，就会很方便，你可以很容易地把整段示例复制粘到你的REPL中，并亲自试验它。

现在你已有了强大的REPL在手。我们将在第1章中使用它来开始Clojure探索之旅。

排版约定

本书使用以下的排版约定：

斜体

表示新术语、URL、email地址、文件名以及文件扩展名。

等宽字体（`constant width`）

用于代码清单，以及在段落内引用程序元素，例如变量或函数名、数据库、数据类型、环境变量，以及关键字。

加粗等宽字体（**`constant width Bold`**）

显示应由用户逐字输入的命令或其他文本。

斜体等宽字体（*`constant width italic`*）

显示应替换为用户提供的值或由上下文来确定的值。

提示：这个图标表示提示或建议。

注意：这个图标表示一般说明。

警告：这个图标表示警告或提醒。

使用示例代码

补充材料（代码示例、练习等）可从*https://github.com/gigasquid/wonderland-clojure-katas*下载。

这本书为的是帮助你完成你的工作。一般来说，如果是书中所带的示例代码，你可以把它用于你的程序和文档。你不需要联系我们以求得允许，除非你要重新发布大量的代码。例如，写一个使用了好几块本书代码的程序不需要求得允许。销售或发行来自O'Reilly 图书的代码示例的光盘就需要求得允许。引用本书文字或示例代码来回答一个问题不需要求得允许。把本书相当数量的示例代码纳入你的产品文档中就需要求得允许。

我们欢迎引用时注明出处，但不强求。引用通常包含书名、作者、出版社，以及ISBN。例如，“Living Clojure by Carin Meier (O'Reilly). Copyright 2015 Carin Meier, 978-1-491-90904-1.”

如果你觉得对示例代码的使用超出了合理范围或者以上允许的范围，可以联系我们：*permissions@oreilly.com*。

Safari® Books Online

Safari Books Online（safaribooksonline.com）是一个按需出版的数字图书馆，发行在技术和商业领域世界领先的作者的专业内容，既有图书也有视频。

技术专家、软件开发者、Web设计者，还有商业和创意专业人士都使用Safari Books Online作为他们研究、问题解决、学习，以及认证培训的首要资源。

Safari Books Online为企业、政府、教育等组织还有个人提供了多种定制和定价方案。

会员可以在一个完全可搜索的数据库中访问数千本图书、培训视频和手稿。数据库来自于很多出版社，例如O'Reilly Media、PrenticeHallProfessional、Addison-WesleyProfessional、MicrosoftPress、Sams、Que、PeachpitPress、FocalPress、CiscoPress、JohnWiley&Sons、Syngress、MorganKaufmann、IBMRedbooks、Packt、AdobePress、FTPress、Apress、Manning、NewRiders、McGraw-Hill、Jones&Bartlett和CourseTechnology，此外还有其他数百家。关于Safari Books Online的更多信息，请访问我们的网站：*https://www.safaribooksonline.com/*。

如何联系我们

欢迎向出版社提出有关本书的意见和问题：

美国：

O'Reilly Media，Inc.
1005 Gravenstein Highway North
Sebastopol，CA 95472

中国：

北京市西城区西直门南大街2号成铭大厦C座807室（100035）
奥莱利技术咨询（北京）有限公司

本书有一个网页，在上面我们列出了勘误、代码示例和其他有用的信息。可以从这里访问*http://bit.ly/living_clojure*。

要评论或提问关于本书的技术问题，请发邮件到 *bookquestions@oreilly.com*。

关于图书、课程、研讨会和新闻的更多信息，请浏览我们的网站：

http://www.oreilly.com
http://www.oreilly.com.cn。

致谢

写一本书是我所做过的最有挑战的项目之一。要是我独自一人的话肯定完成不了。在这里我想对每一个帮助过我的人说声谢谢，无论是直接的还是间接的。

首先要感谢的是我的编辑Meghan Blanchette。跟一个新手作者一起工作并指导一本书的创作不是件简单的任务。她耐心又能鼓励人，总是督促我表达编程概念时既要清晰而又逻辑清楚。她心里总是想着读者的利益。谢谢你。你真优秀。

感谢所有本书的审阅者，你们的反馈是无价的。特别要感谢 Colin Jones、 Gabriel Andretta和Elliot Hauser，还有Luigi Montanez。

感谢由于他们的贡献或授权使得本书的出版成为可能的人们。特别是Alan Malloy、Alex McNamara和David Byrne，还有Anthony Grimes，他们对我的第一个Clojure开源项目给予了很大帮助，这真让人愉快。还有，要特别感谢他们允许我在本书的练习项目中使用4Clojure问题。同样要特别感谢Phil Hagelberg，不仅仅因为Leiningen和Clojars，还因为他在第5章上给予的指导。感谢Michael Klishin、James Reeves、Francesco Bellomi、Zachary Kim和Reid McKenzie，还有其他人，他们宽容地允许本书使用他们的屏幕截图。

感谢Cincinnati函数式程序员组，特别是Creighton Kirkendall、Joe Herbers和Benjamin Kyrlach，这些年你们帮助我建立了这个用户组。你们帮助为Clojure和其他函数式语言提供了一个强大的社区，就在我们的家乡。跟你们一起开始这个旅程我是多么高兴。

感谢Cincinnati的开发者社区。你们都是我的朋友，永远感激你们的支持和鼓励。还有，所有那些周五早晨咖啡组的常客们，你们也有助于我保持头脑清醒。

感谢Clojure社区，形成了这么好的一个成长和学习Clojure的地方。感谢Rich Hickey、Stu Halloway和Justin Gehtland，还有所有Cognitect 的人，他们开发了伟大的软件。感谢Alex Miller，他是一个能力出色的家伙，给社区带来了多个美妙的研讨会。感谢Michael Fogus和Chris Houser，他们写了《The Joy of Clojure》一书，因为此书我第一次进入了Clojure，并持续从中获得灵感。感谢Clojure社区的所有人：David Nolen、Stuart Sierra、Alex Baranosky、Alan Dipert、Ambrose Bonnaire-Sergeant、Jen Smith、Sam Aaron、Jonathan Graham、Eric Normand、Aaron Brooks、Brenton Ashworth、Luke VanderHart、Bodil Stokke、Bruce Durling、Chas Emerick、Tavis Rudd、Chris Ford、Craig Andera、Michael Nygard、Yodit Stanton、David Pollak、Daniel Higginbotham、Ryan Neufeld、Nada Amin、William Byrd、Anna Pawlicka、Kyle Kingsbury、Zach Tellman、Zack Maril，还有Outpace Systems所有我的Clojure朋友们，以及所有其他被我漏掉名字的人们。