The Pragmatic Programmers

Web Development with Clojure

Clojure Web 开发实战

[美] Dmitri Sotnikov 著

张恒 译

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

The Pragmatic Programmers

Web Development with Clojure

Clojure Web 开发实战

[美] Dmitri Sotnikov 著

张恒 译

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

Clojure Web开发实战 / (美) 肖特尼科夫(Sotnikov,D.) 著 ; 张恒译. -- 北京 : 人民邮电出版社, 2015.11
ISBN 978-7-115-39893-2

Ⅰ. ①C… Ⅱ. ①肖… ②张… 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188698号

版权声明

◆ 著 [美] Dmitri Sotnikov
译 张 恒
责任编辑 陈冀康
责任印制 张佳莹 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：13.75
字数：262 千字 2015 年 11 月第 1 版
印数：1-2 500 册 2015 年 11 月北京第 1 次印刷

著作权合同登记号 图字：01-2014-5439 号

定价：45.00 元

读者服务热线：(010)81055410 印装质量热线：(010)81055316
反盗版热线：(010)81055315

内容提要

Clojure 是一门 Lisp 方言。它通过函数式编程技术，直接支持并发软件开发，得到众多开发人员的欢迎，广泛应用于各个领域。Web 开发是 Clojure 的主战场之一。

本书专门探讨 Clojure 在 Web 开发领域的实际应用。通过阅读本书，读者既可以深入理解 Clojure Web 栈的专业知识，同时又能运用这些知识来轻松构建 Web 应用。全书共 7 章，详细介绍了 Clojure Web 开发的各个方面，附录部分介绍了 IDE 的选择、Clojure 快速入门以及相关的数据库技术。

本书适合各个层次的读者。如果具备一些函数式方面的编程经验，将对阅读本书有所助益，但这并不是必需的。如果你还没有真的用过 Clojure，可以快速掌握如何运用这门语言来解决 Web 开发中的实际问题。

对本书的赞誉

这是一本极好的书，我会强烈推荐身边所有的 Clojure Web 开发者阅读。

- Colin Yates，QFI 咨询事务所首席工程师、技术团队领袖

Clojure 这种语言棒极了，用它来开发 Web 应用简直是一种享受。对于诸多熟悉用 Clojure Web 开发工具库的人来说，这本书是份宝贵且及时的材料。

- Fred Daoud，资深 Web 开发者、《七周七 Web 框架》合著者

对于想用 Clojure 开发 Web 应用的人来说，Dmitri Sotnikov 会通过 Clojure 开发实践让你迅速上手。如果你已懂一些 Clojure 基本知识，但是还只在了解的阶段，那这本书太适合你了。

- Russ Olsen，Cognitect 副总裁、咨询服务

Sotnikov 阐明了如何借助 Clojure 的灵活性来搭建网站，使用顶尖的库来制作务实的站点。

- Chris Houser，《Clojure 编程乐趣》合著者

有了这本书，您将手操强大的函数式编程技术直接步入 Web 开发。随着渐学渐长，你还会为你的应用注入更多可扩展、可维护的特性，并且，这种 Clojure 开发经验还会延续到 JavaScript 客户端。

- Ian Dees，《Cucumber Recipes》作者

Dmitri 在书中通过穿插介绍语言特性的同时，成功解决了现在软件开发中遇到的真实问题。这说明你投入时间阅读本书将富有效率和价值。

- Brian Sletten，Bosatsu 咨询、《Resource-Oriented Architecture Patterns for Webs of Data》作者

本书通篇详细介绍用 Clojure 搭建网站应用，节奏轻快、直白。从第 1 章开始，你

将陆续接触真实的 Web 应用，接下来为其添加数据库、处理安全问题、JavaScript 等。不教条、不唠叨、不废话！就为了简洁、高效。本书从无到有提供一个货真价实的应用，使得你可以通过完善不断成长。

- Sam Griffith Jr.，语言专家、任职于 Interactive Web Systems

简 介

这是一株盆景，优雅、高洁。之所以选择盆景作为本书的封面，是因为相同的品质在 Clojure 这门极具魅力的语言身上，同样展现得淋漓尽致。开发软件，就像是修剪一株盆景，只有精工细作，才能将其雕琢成型；也唯有工具趁手，方可体味其中的乐趣。而 Clojure，就是这个不可思议的工具。相信在读完本书之后，你也会这么认为的。

本书适合你吗？

本书适合各个层次的读者。如果具备一些函数式方面的编程经验，将对阅读本书有所助益，但这并不是必需的。如果你还没有真的用过 Clojure，那么阅读本书会是一个不错的起点，因为本书关注的，就是如何运用这门语言来解决实际问题。这意味着，我们仅仅需要少量语言特性，就能实现常见的 Web 应用。

为什么选择 Clojure？

作为一门轻巧的语言，Clojure 的首要目标是简洁并且准确。此外，作为一门函数式语言，它还格外强调不变性（Immutability），以及声明式编程（Declarative Programming）。正如你将在本书中看到的那样，这些特性使得编写清爽又正确的代码，竟会变得如此简单且自然。

编程语言各有千秋，关于它们之间孰优孰劣的争论也从未休止。有的语言，结构简单却表达冗长。也许你曾听人这样说过，表述啰唆一点并没什么大不了，理由是，只要两种语言都是图灵完备的，那么能用简洁语言表达出来的东西，用冗长一些的语言也可以表达出同样的含义，只不过是多出几行代码罢了。

然而，这种说法却忽略了关键所在。因为真正的问题并不在于能不能表达出来，而在于表达得好不好，直不直接。一门好的语言，能让你始终围绕着问题域去思考；而糟糕的语言，则迫使你不得不把问题转换为这门语言强加的概念。

后者，往往都是枯燥无味的。最终，你通篇都是样板代码，并且一再重复着这些早已做过无数遍的事情。如果我们总是不得不编写许多重复的代码，那多少也显得有些可悲了。

还有一些其他语言，它们并不冗长，甚至还提供了诸多用来解决各种问题的工具。可惜的是，绝大多数工具都未必能真正转化为更强的生产力。

语言具备的特性越多，你就越需要花费更多的精力来考虑如何有效地运用这门语言。我之前用过许多这样的语言，发现自己总会费尽心思地纠缠于众多特性之间，难以自拔。

于我而言，何谓理想的语言，就是我可以不假思索地使用它。当一门语言缺乏表达力，就会明显让我感觉到捉襟见肘。另一方面，当一门语言有太多特性的时候，我又会经常感到不知所措，甚至受其所扰。

用数学来进行类比，能记住一个可以推导其他公式的通用公式，总是好过死记硬背一大堆针对特定问题的公式。

Clojure 正是为此而生。它使得我们借助少量的几个通用模式，就可以轻松获得解决特定问题的方案。你只需要学习几个简单的概念，以及些许语法，就可以迅速将它们转化为生产力。这些概念可以用无数种方法加以组合，用来解决任何类型的问题。

为何选 Clojure 来构建 Web 应用？

Clojure 被广泛应用于各个领域，在其数以万计的使用者中，不乏银行和医院这样挑剔的用户。说 Clojure 是 Lisp 语系发展至今最流行的一门方言，也毫不为过了。尽管这门语言还很年轻，但它已经充分证明了自己。这份自信源自于它在生产系统中的表现，也源自于用户们排山倒海般的好评。

由于 Web 开发是 Clojure 的主战场之一，一些重要的库和框架也开始在这个领域崭露头角。一般来说，Clojure 的 Web 栈是基于 Ring[①]和 Compojure[②]的，其中 Ring 是 HTTP 基础库，而 Compojure 则在 Ring 的基础上，提供了路由机制。在接下来的章节中，你将会逐渐地了解并熟悉这个 Web 栈，并懂得如何有效地借助它们来构建你自己的 Web 应用。

① https://github.com/ring-clojure/ring
② https://github.com/weavejester/compojure

目 录

第 1 章

起步

在简介部分，我们谈到了在编写应用程序时，采用函数式编程风格能够获得诸多好处。当然，想要学会一门语言，仅仅通过阅读是远远不够的，只有亲手编写一些代码，你才能获得真切的体验。

在本章中，我们将会介绍如何开发一个简单的留言簿应用，用户可以使用它给他人留言。通过它，我们能够了解 Web 应用的基本结构，并且尝试一些高效的 Clojure 开发工具。如果你是一个 Clojure 新手，那我建议你先跳到“附录 2 Clojure 入门”，快速了解一下 Clojure 的基本概念和语法。

1.1 环境设置

Clojure 需要 Java 虚拟机（JVM，Java Virtual Machine）才能运行，此外，你还需要一份 1.6 或是更高版本的 Java 开发工具包[①]（JDK，Java Development Kit）用于开发。Clojure 是作为一个 JAR 包来分发的，你只需简单地将其包含在工程的 class-path 中即可。你可以使用任何常规的 Java 工具来构建 Clojure 应用，比方说 Maven[②]或者 Ant[③]。不过，我强烈建议你使用 Leiningen[④]，它是专为 Clojure 定制的。

① http://www.oracle.com/technetwork/java/javase/downloads/index.html
② http://maven.apache.org/
③ http://ant.apache.org/
④ http://leiningen.org/

使用 Leiningen 管理工程

借助 Leiningen，你可以建立、构建、测试、打包和部署工程。也就是说，它能为你提供工程管理方面的一站式服务。

Maven 是一个非常流行的 Java 依赖关系管理工具，而 Leiningen 就相当于 Clojure 世界中的 Maven。重点是，Leiningen 与 Maven 兼容，因此它可以毫无障碍地访问那些得到精心维护，且存放着海量 Java 类库的存储中心。此外，Clojure 的库通常可以在 Clojars[①]这个存储中心找到。所以，默认情况下 Leiningen 是启用了 Clojars 的。

使用 Leiningen，你不用手动去下载那些在工程中需要用到的库。你只需要简单地声明一下工程的顶级依赖，剩下的事情 Leiningen 就会帮你自动搞定。

Leiningen 的安装实在是小菜一碟，只需要从官方主页[②]上下载并执行安装脚本即可。

不如动手试试看。我们会通过执行下列命令，来下载这个脚本，并创建一个全新的 Clojure 工程：

```
wget https://raw.github.com/technomancy/leiningen/stable/bin/lein
chmod +x lein
mv lein ~/bin
lein new myapp
```

由于这是我们第一次运行 lein 这个命令，它做的第一件事情是安装它自己。一切顺利的话，你将会看到下面的输出：

```
Generating a project called myapp based on the 'default' template.
To see other templates (app, lein plug-in, etc), try `lein help new`.
```

一个新的文件夹 myapp 就创建好了，里面是应用程序的骨架。应用程序的代码存放在 src 文件夹中。其中有另外一个 myapp 文件夹，这个文件夹中只有一个文件，名为 core.clj。文件内容如下：

① https://clojars.org/

② http://leiningen.org/#install

```
(ns myapp.core)

(defn foo
  "I don't do a whole lot."
  [x]
  (println x "Hello, World!"))
```

请注意命名空间的声明，与其文件夹结构是相匹配的。由于命名空间 core 位于 myapp 目录当中，所以它的名字就是 myapp.core。

Leiningen 工程文件一瞥

在工程文件夹 myapp 里有一个 project.clj 文件。这个文件包含了应用程序的描述信息，你可以仔细观察一下，就会发现这个文件是用标准的 Clojure 语法编写的，描述了应用的名称、版本、网址、许可证信息和依赖项，如下所示。

```
(defproject myapp "0.1.0-SNAPSHOT"
:description "FIXME: write description"
:url "http://example.com/FIXME"
:license {:name "Eclipse Public License"
          :url "http://www.eclipse.org/legal/epl-v10.html"}
:dependencies [[org.clojure/clojure "1.5.1"]])
```

通过修改这个 project.clj 文件，能让我们控制应用程序的方方面面。例如，我们可以通过添加:main 关键字，将 myapp.core 命名空间下的 foo 函数设置为应用的入口点：

```
(defproject myapp "0.1.0-SNAPSHOT"
:description "FIXME: write description"
:url "http://example.com/FIXME"
:license {:name "Eclipse Public License"
          :url "http://www.eclipse.org/legal/epl-v10.html"}
:dependencies [[org.clojure/clojure "1.5.1"]]
;;this will set foo as the main function
:main myapp.core/foo)
```

此时我们就可以通过执行 lein run 这个命令来运行应用了。由于 foo 函数要求传入一个参数，我们只得遵命行事：

```
lein run First
First Hello, World!
```

在前面这个例子中，我们创建的应用非常简单，只有一个依赖项：Clojure 运行时。如果我们直接以此为基础来开发 Web 应用的话，就免不了要编写大量的样板代码，才能让它运行起来。下面就让我们看看如何利用 Leiningen 的模板，来创建一个开箱即用的 Web 应用吧。

Leiningen 的模板

当把模板的名称提供给 lein 脚本时，就可以根据其对应的模板来初始化工程骨架。其实模板自身也不过是使用了 lein-newnew 插件[①]的 Clojure 工程罢了。稍后我们将看到如何创建自己的模板。

眼下，我们将会使用 compojure-app 模板[②]来初始化下一个应用。执行 lein 脚本时，模板的名称是作为参数传给 new 关键字的，紧接其后的是工程名称。为了创建一个 Web 应用，而不是之前那样的默认工程，我们只需执行以下命令即可：

```
lein new compojure-app guestbook
```

这样 Leiningen 就知道创建留言簿应用时，应该使用 compojure-app 模板了。此类应用需要启动一个 Web 服务才能运行。其实这很容易，我们只需要使用 lein ring server 来替代 lein run 即可。

当我们运行这个应用时，控制台会输出如下信息，与此同时还会弹出一个打开了应用主页的浏览器窗口。

```
lein ring server
guestbook is starting
2013-07-14 18:21:06.603:INFO:oejs.Server:jetty-7.6.1.v20120215
2013-07-14 18:21:06.639:INFO:oejs.AbstractConnector:
StartedSelectChannelConnector@0.0.0.0:3000
Started server on port 3000
```

喔，现在我们已经知道如何创建和运行应用了，接下来不妨考虑一下应该选用什么样的编辑器。

你多半已经留意到，Clojure 代码中有大量的括号。保持它们起止对应很快就会

① https://github.com/Raynes/lein-newnew
② https://github.com/yogthos/compojure-template

成为一种挑战，所幸 Clojure 编辑器会替我们收拾这个摊子，否则会令人产生严重的挫败感。

事实上，这些编辑器不仅仅能平衡括号，其中的一些甚至能够感知其结构。这就意味着编辑器能够理解一个表达式是从什么地方开始，又到什么地方结束的。因此，我们可以根据逻辑上的代码块来导航和选取，而非简单针对文本行号。

在本章中，我们将会选用 Light Table[①]来开发留言簿应用。获取并运行 Light Table 是非常容易的，这样我们就能尽快投入到代码的编写中了。然而，它的功能还比较有限，在较大的工程中，你对此可能有较深的体会。“附录 1 选择 IDE”中还有对其他开发环境的讨论。

使用 Light Table

Light Table 不需要安装，下载完成后即可直接运行。

Light Table 的外观相当简洁。默认情况下，它仅在编辑器窗格中显示了几行欢迎信息，如图 1-1 所示。

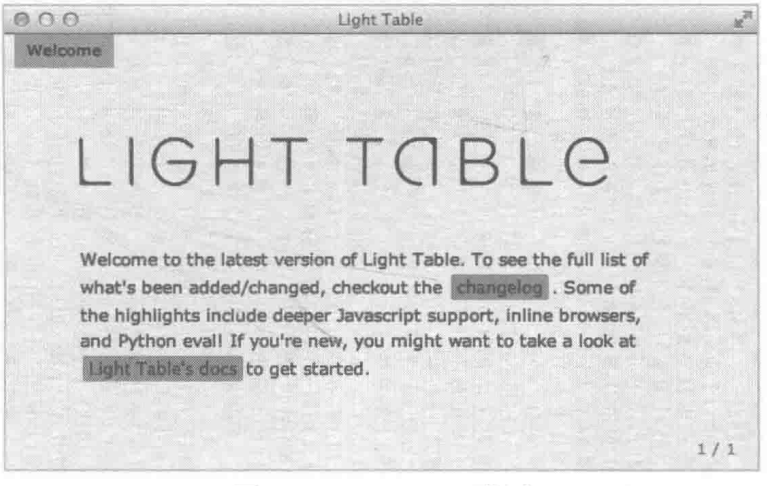

图 1-1 Light Table 工作区

为了显示 workspace 面板，我们可以在菜单中选择 View →Workspace，或是按下 Ctrl+T（Windows/Linux）组合键或 Cmd+T（OS X）组合键。

① http://www.lighttable.com/

如图 1-2 所示，我们可以在 workspace 的 folder 标签页中打开留言簿工程。

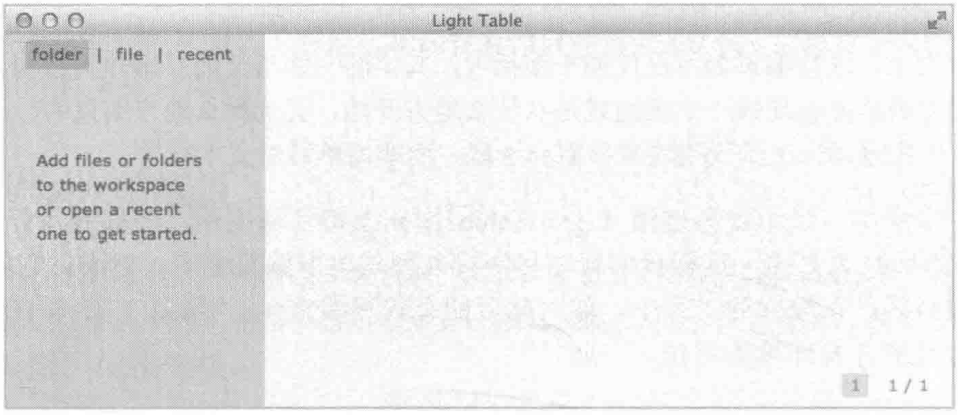

图 1-2　打开工程

一旦工程被选中，我们就可以浏览整个工程树，并选择我们想要编辑的文件，如图 1-3 所示。

现在，开发环境已经就绪，看起来我们终于可以为留言簿应用添加一些功能了。

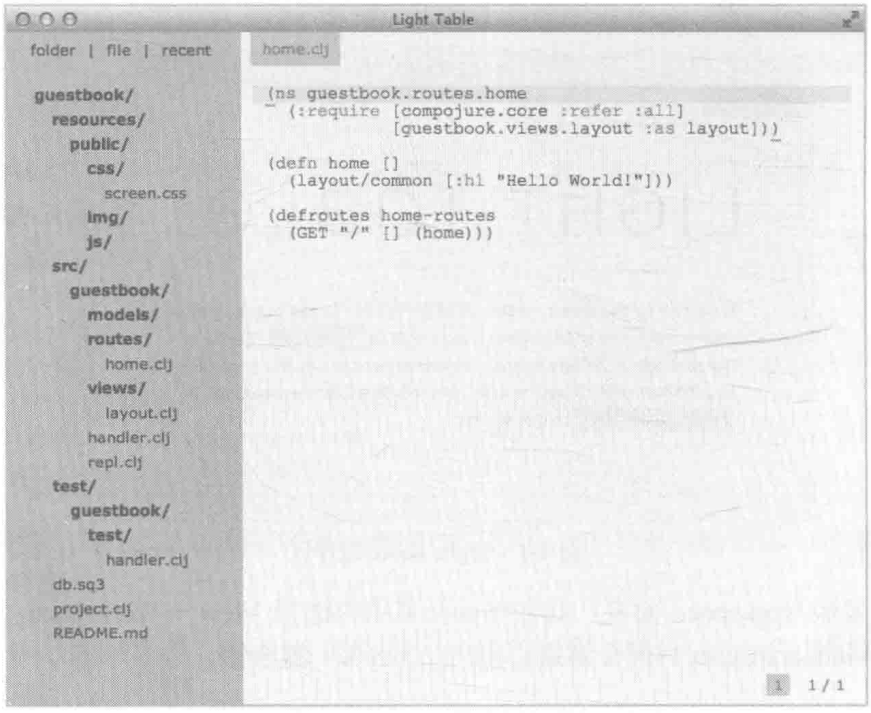

图 1-3　Light Table 的工程

1.2 你的第一个工程

你的留言簿应该已经在控制台运行了，可以通过 http://localhost:3000/来访问。在控制台终端按下 Ctrl+C，就能停止它的运行。既然我们已经在 Light Table 的工作区打开了这个工程，不妨就直接在编辑器中运行它吧。

我们现在要更进一步，创建一个“读取—求值—打印循环”（REPL，Read-Evaluate-Print Loop），将 Light Table 连接至我们的工程。菜单 View→Connections 可以打开连接标签页。如图 1-4 所示，让我们点击标签页中的 Add Connection 按钮。

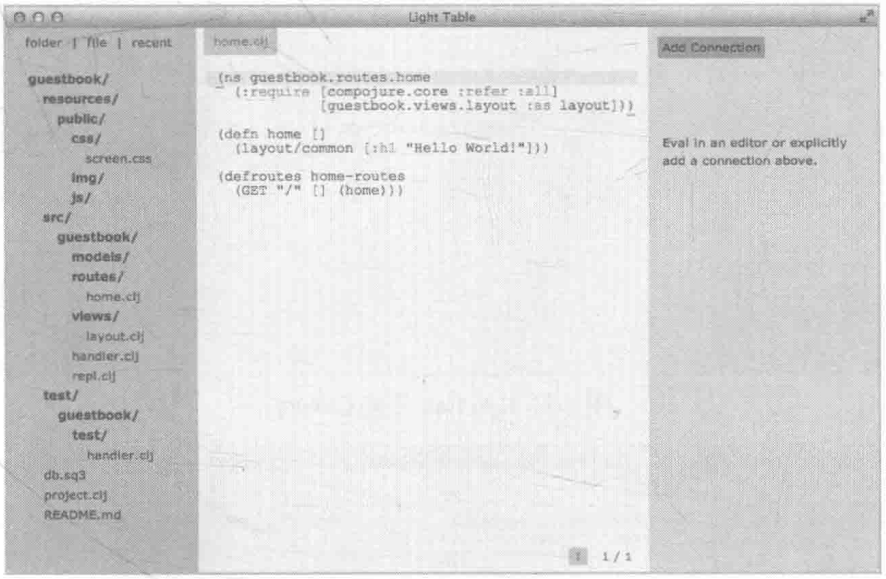

图 1-4 Light Table 的连接

此时，会弹出一个列表，列出了几种不同的连接选项。如图 1-5 所示，接下来选择 Clojure。然后，让我们找到留言簿工程所在的文件夹，并且选中 project.clj 文件。

一旦我们的工程与 Light Table 建立了连接，我们就可以直接在编辑器中对代码进行求值了。

说不如做，你可以立刻挑选一个函数，然后按下 Ctrl+Enter（Windows/Linux）组合键或是 Cmd+Enter（OS X）组合键。如果我们选择的是 home 函数，那么打印出来的内容应该是这样：

```
#'guestbook.routes.home/home
```