异步图书
www.epubit.com
MANNING

JavaScript 函数式编程指南

[美] 路易斯·阿泰西奥（Luis Atencio） 著
欧阳继超 屈鉴铭 译

JavaScript

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

JavaScript 函数式编程指南

[美] 路易斯 • 阿泰西奥（Luis Atencio） 著

欧阳继超 屈鉴铭 译

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

JavaScript函数式编程指南 /（美）路易斯·阿泰西奥（Luis Atencio）著；欧阳继超，屈鉴铭译. -- 北京：人民邮电出版社，2018.6
ISBN 978-7-115-46204-6

Ⅰ. ①J… Ⅱ. ①路… ②欧… ③屈… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2018)第045224号

版权声明

◆ 著　[美] 路易斯·阿泰西奥（Luis Atencio）
译　欧阳继超　屈鉴铭
责任编辑　吴晋瑜
责任印制　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
固安县铭成印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：14
字数：202 千字　2018 年 6 月第 1 版
印数：1－2 400 册　2018 年 6 月河北第 1 次印刷

著作权合同登记号　图字：01-2016-7580 号

定价：59.00 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

本书主要介绍如何通过 ECMAScript 6 将函数式编程技术应用于代码来降低代码的复杂性。

本书共三部分内容。第一部分“函数式思想”是为第二部分的学习作铺垫的，这一部分引入了对函数式 JavaScript 的描述，从一些核心的函数式概念入手，介绍了纯函数、副作用以及声明式编程等函数式编程的主要支柱；第二部分“函数式基础”重点介绍函数式编程的核心技术，如函数链、柯里化、组合、Monad 等；第三部分“函数式技能提升”则是介绍使用函数式编程解决现实问题的方法。

本书循序渐进地将函数式编程的相关知识铺陈开来，以理论作铺垫，并辅以实例，旨在帮助读者更好地掌握这些内容。如果读者是对面向对象软件有一定的了解，且对现代 Web 应用程序挑战有一定认识的 JavaScript 开发人员，那么可以从中提升函数式编程技能。如果读者是函数式编程的初学者，那么可以将本书作为入门书籍仔细阅读，为今后的学习夯实基础。

序

在本科和研究生阶段，我的课程安排专注于面向对象设计，并将其作为软件系统规划与架构设计的唯一方法。像许多开发人员一样，我的职业生涯也是从编写面向对象代码开始的，并且基于该编程范式来构建整个系统。

在整个职业生涯中，我密切关注并学习编程语言，不仅是因为想要学习一些很酷的知识，也因为我对每种语言的设计决策和设计哲学都很感兴趣。新的语言会对如何解决软件问题提供不同的观点，新的范式可以达到相同的效果。虽然面向对象的方法仍然是软件设计的主流工作方式，但是学习函数式编程能够拓宽视野，因为该技术既能够单独使用，也可以与其他设计范例并用。

函数式编程已经存在多年。尽管我听说过Haskell、Lisp、Scheme以及近年流行的Scala、Clojure和F#在表现力方面以及高效的平台上拥有优势，但起初我对此并不是很关心。随着时间的流逝，即使是传统上一直被认为很啰嗦的语言Java，也具有了一些让代码更简洁的函数式特性。最终，这项不起眼的技术变得让我无法抵挡。更令人难以置信的是，JavaScript这种大家都当成面向对象的语言，也可以作为函数式语言来使用了。事实证明，这正是JavaScript更强大、更高效的使用方法。我花了很长时间才发现这一点，所以希望能通过本书让你也意识到这一点，如此一来，你的JavaScript代码就不会变得过于复杂。

作为开发人员，我学会了如何使用函数式编程原则来创建模块化、表达性强且易于理解和测试的代码。毫无疑问，作为一名软件工程师，函数式编程让我脱胎换骨，所以我想记录下这些经验，将其放到一本书中。于是，我联系了Manning出版社，打算以Dart编程语言为基础来编写这本函数式编程的书。当时我正在使用Dart，并认为如果将它与我的函数式背景相结合，会产生一个非常有趣的未知领域。因此，我拟定了一个写作方案，并在一个星期后与出版社的人进行了沟通——我了解到Manning正在寻找人写一本关于JavaScript函数式编程的书。因为JavaScript也是我非常痴迷的语言，所以我毫不犹豫地抓住了这个机会。通过这本书，我希望能帮助你提升这方面的技能，并为你的发展带来新的方向。

前言

复杂性是一头需要驯服的巨兽，我们永远无法完全摆脱它，而它也将永远是软件开发的一部分。我曾尝试花费无数小时和无法估量的脑力试图了解一段特定的代码。函数式编程能够帮助你控制代码的复杂性，使其不会与代码库的大小成正比。我们正在编写越来越多的 JavaScript 代码。我们已经经历了小型客户端事件处理程序的构建、富客户端架构以及同构（服务器+客户端）JavaScript 应用程序的实现。函数式编程不是一种工具，而是一种可以同时适用于任何环境的思维方式。

本书旨在说明如何通过 ECMAScript 6 将函数式编程技术应用于代码。本书以渐进、稳定的速度呈现，涵盖了函数式编程的理论和实践两个方面，还为高级读者提供了更多信息，以帮助他们深入了解一些更高级的概念。

本书内容结构

本书分为三部分内容，指导读者学习从基础到函数式编程的更先进的应用。

第一部分“函数式思想”描绘了函数式 JavaScript 的高层次景观。它还讨论了如何像函数式程序员一样函数式地使用和思考 JavaScript 的核心。

- 第 1 章介绍了后续章节包含的一些核心的函数式概念（便于跨越到函数式），介绍了函数式编程的几个主要支柱，包括纯函数、副作用和声明式编程。
- 第 2 章为初级和中级 JavaScript 开发人员准备了练习场，高级的读者也可借此机会复习。本章还介绍了基本的函数式编程概念，为第二部分讨论的技术作铺垫。

第二部分“函数式基础”着重于核心函数式编程技术，包括函数链、柯里化、组合、Monad 等。

- 第 3 章介绍了函数链，并探讨了如何使用递归和高阶函数组合成程序，如 map、filter 和 reduce。其过程会使用到 Lodash.js。
- 第 4 章介绍流行的提高代码模块化程度的技巧和组合。使用诸如 Ramda.js 之类

的函数式框架。组合是编排整个 JavaScript 解决方案的黏合剂。

- 第 5 章带读者深入了解函数式编程的更多理论领域，并在错误处理的上下文中对 Functor 和 Monad 进行了全面并循序渐进的讨论。

第三部分“函数式技能提升”讨论了使用函数式编程解决现实世界挑战的优势。

- 第 6 章揭示了函数式程序易于进行单元测试的原因，还引入了一种严密的自动测试模式（**称为基于属性测试**）。
- 第 7 章介绍了 JavaScript 函数求值的内存模型。本章还讨论了有助于优化函数式 JavaScript 应用程序执行时间的技术。
- 第 8 章介绍了 JavaScript 开发人员在处理事件驱动和异步行为时经常遇到的一些主要挑战，讨论了函数式编程如何提供优雅的解决方案，以通过使用 RxJS 实现的称为**响应式编程**的相关范例，来降低现有命令式解决方案的复杂性。

本书面向的读者

本书是针对对面向对象软件有基本了解，以及对现代 Web 应用程序挑战具有一定认识的 JavaScript 开发人员编写的。JavaScript 是一种无处不在的语言，如果你需要函数式编程的介绍，并喜欢熟悉的语法，那么完全可以充分利用本书，而不是去学习 Haskell（如果想要以更轻松的方式入门 Haskell，本书不是最好的资源，因为每种语言都有自己的特性，直接学习其实是最好的理解）。

本书通过对高阶函数、闭包、函数调用、组合以及新的 JavaScript ES6 特性（如 lambda 表达式、迭代器、生成器和 Promise）的介绍，帮助初级和中级程序员提高他们的 JavaScript 技能。高级开发人员也将从中领略到 Monad 和响应式编程的解读，从而可以运用创新的方法，来完成处理事件驱动和异步代码的艰巨任务，并充分地使用 JavaScript 平台。

如何使用本书

如果读者是初级或中级 JavaScript 开发人员，并且刚刚接触函数式编程，请从第 1 章读起。如果读者是一名高级 JavaScript 程序员，那么可以简要阅读第 2 章，然后从第 3 章的函数链和整体函数式设计读起。

函数式 JavaScript 的更高级用户通常已经理解纯函数、柯里化和组合，因此可以快速浏览第 4 章，并从第 5 章开始学习 Functor 与 Monad。

示例和源代码

本书中的代码示例使用 ECMAScript 6 JavaScript 编写，它可以在服务器（Node.js）或客户端上运行。一些示例需要 IO 和浏览器 DOM API，但没有考虑浏览器的兼容性。期望读者已经有在 HTML 页面和控制台的基础级互动的经验。代码对浏览器没有特定要求。

本书大量使用了诸如 Lodash.js、Ramda.js 等函数式的 JavaScript 库。读者可以在附

录中找到文档和安装信息。

本书包含大量用于展示函数式技术的代码清单，并在适当的情况下比较了命令式和函数式设计。读者可以在 Manning 官方网站和 GitHub 上找到所有代码示例。

本书体例

本书中使用了以下约定：

粗体字用于引用重要术语。

`Courier` 字体用于表示代码清单，以及元素和属性、方法名称、类、函数和其他编程工件。

代码清单中会有一些代码注释，以突出重要的概念。

作者简介

Luis Atencio（@luijar）是美国佛罗里达州劳德代尔堡的 Citrix Systems 公司的一名软件工程师。他拥有计算机科学学士学位和硕士学位，现在使用 JavaScript、Java 和 PHP 平台进行全职开发和构建应用程序。Luis 积极参与社区活动，并经常在当地的聚会和会议中发表演讲。他在 luisatencio.net 上发布关于软件工程的博客，并为杂志和 DZone 撰写文章，同时还是《RxJS in Action》的共同作者。

作者在线

读者可免费访问由 Manning 出版社运营的专有网络论坛，可以在论坛对本书发表评论、询问技术问题，并从作者和其他用户那里获得帮助。注册后可在该页面获取如何进入论坛，如何寻求帮助以及论坛上的行为规则等信息。

Manning 只提供读者之间以及读者和作者之间进行有意义的对话的平台。作者对其具体参与程度不承担任何责任，且作者在线的贡献是自愿的（而且是无偿的），因此我们建议读者尽可能提出一些具有挑战性的问题，以获得作者关注。只要本书在销，读者将一直可以访问作者在线论坛及所有讨论。

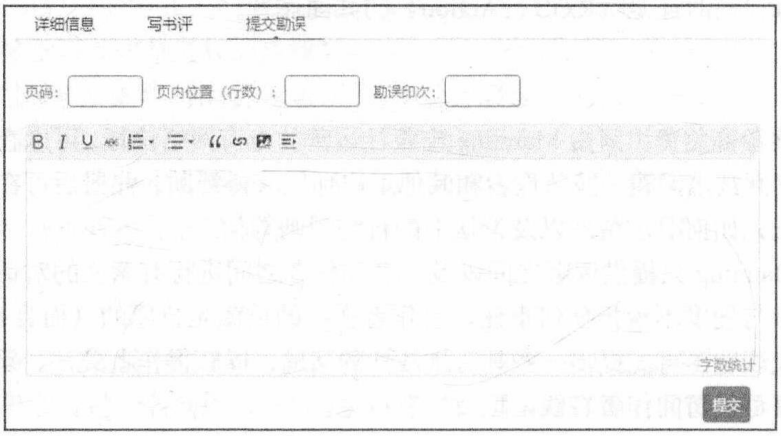
资源与支持

本书由异步社区出品，社区（https://www.epubit.com/）为您提供相关资源和后续服务。

提交勘误

作者和编辑尽最大努力来确保书中内容的准确性，但难免会存在疏漏。欢迎您将发现的问题反馈给我们，帮助我们提升图书的质量。

当您发现错误时，请登录异步社区，按书名搜索，进入本书页面，单击“提交勘误”，输入勘误信息，单击“提交”按钮即可。本书的作者和编辑会对您提交的勘误进行审核，确认并接受后，将赠送给您异步社区的 100 积分（积分可用于在异步社区兑换优惠券、样书或奖品）。



扫码关注本书

扫描下方二维码，您将会在异步社区微信服务号中看到本书信息及相关的服务提示。

与我们联系

我们的联系邮箱是 contact@epubit.com.cn。

如果您对本书有任何疑问或建议，请您发邮件给我们，并请在邮件标题中注明本书书名，以便我们更高效地做出反馈。

如果您有兴趣出版图书、录制教学视频，或者参与图书翻译、技术审校等工作，可以发邮件给我们；有意出版图书的作者也可以到异步社区在线提交投稿（直接访问 www.epubit.com/selfpublish/submission 即可）。

如果您是学校、培训机构或企业，想批量购买本书或异步社区出版的其他图书，也可以发邮件给我们。

如果您在网上发现有针对异步社区出品图书的各种形式的盗版行为，包括对图书全部或部分内容的非授权传播，请您将怀疑有侵权行为的链接发邮件给我们。您的这一举动是对作者权益的保护，也是我们持续为您提供有价值的内容的动力之源。

关于异步社区和异步图书

“异步社区”是人民邮电出版社旗下 IT 专业图书社区，致力于出版精品 IT 技术图书和相关学习产品，为作译者提供优质出版服务。异步社区创办于 2015 年 8 月，提供大量精品 IT 技术图书和电子书，以及高品质技术文章和视频课程。更多详情请访问异步社区官网 https://www.epubit.com。

“异步图书”是由异步社区编辑团队策划出版的精品 IT 专业图书的品牌，依托于人民邮电出版社近 30 年的计算机图书出版积累和专业编辑团队，相关图书在封面上印有异步图书的 LOGO。异步图书的出版领域包括软件开发、大数据、AI、测试、前端、网络技术等。

异步社区

微信服务号

致谢

写书并不是一件容易的事，需要经过许多人不懈的合作，才能从一页页手稿汇聚成最终呈现到你面前的一本书。

Manning 出版社编辑团队的工作非常出色，他们发挥了极其重要的作用，确保图书的质量达到了我们双方的预期。我发自内心地感谢他们每一个人。没有他们，这本书是不可能完成的。特别感谢 Marjan Bace 和 Mike Stephens 相信我能够胜任本书作者；感谢 Marina Michaels 给了我“地图”和“手电筒”，以引导我走出如迷宫般的写书的挑战；感谢 Susan Conant 给我上了如何编写技术书的第一课，让我的写作走上正轨；感谢 Bert Bates 给了我最初的创意火花，以及他在教授编程时的惊人见解；感谢团队的所有编辑和制作人员，包括 Mary Piergies、Janet Vail、Kevin Sullivan、Tiffany Taylor、Katie Tennant、Dennis Dalinnik 和许多在幕后工作的人。

非常感谢 Aleksandar Dragosavljevic 带领的技术审阅小组：Amy Teng、Andrew Meredith、Becky Huett、Daniel Lamb、David Barkol、Ed Griebel、Efran Cobisi、Ezra Simeloff、John Shea、Ken Fukuyama、Peter Edwards、Subhasis Ghosh、Tanner Slayton、Thorsten Szutzkus、Wilfredo Manrique、William E. Wheeler、Yiling Lu 以及那些才华横溢的论坛贡献者。他们找出了各种技术错误、术语错误和文字差错，并给出了相应的建议。论坛的每一轮审查过程和每一个反馈都使得该手稿更加完美。

在技术方面，特别感谢担任本书技术编辑的 Dean Iverson 以及担任本书技术校对的 Daniel Lamb。同时感谢 Brian Hanafee，他对整本书进行了深入的评估。他们是我见过的最好的技术编辑。

最后感谢我的妻子一直以来的支持，感谢我的家人每天都在帮助我变得更好，感谢他们不计较我在写书的时候没有经常与他们联系。此外，感谢我的同事购买了早期版本的章节。我很荣幸能与这些了不起的人一起工作。

目录

第一部分　函数式思想

第二部分　函数式基础

第三部分 函数式技能提升

第一部分

函数式思想

也许读者构建专业应用程序的大部分经验都与面向对象语言有关。读者可能通过阅读其他书籍、博客、论坛和杂志文章听说过函数式编程，但却从来没有编写过任何函数式代码。别担心，这正是笔者所想到的。笔者也曾在面向对象的环境中完成了大部分开发工作。编写函数式代码并不困难，但学会函数式的思考、放弃旧习惯才是真正的挑战。本书第一部分的主要目标是为第二部分和第三部分讨论的函数式技术奠定基础。

第 1 章讨论了什么是函数式编程，以及需要以什么样的心态来迎接它，同时还介绍了基于纯函数、不可变性、副作用和引用透明性等概念的一些重要技术。这些技术能够形成函数式代码的主干，并将帮助读者更轻松地走近函数式编程。此外，这也将成为后面章节中许多代码设计的指导原则。

第 2 章揭示了 JavaScript 作为函数式语言的另一面。由于 Javascript 是主流语言且广泛存在，因此这是一门理想的、可用来教授函数式编程的语言。如果读者不是一名高级 JavaScript 开发人员，本章将帮助你快速了解学习函数式 JavaScript 的必备基础，例如高阶函数、闭包和作用域规则。

第1章 走近函数式

本章内容

- 函数式思想
- 什么是函数式编程以及为什么要进行函数式编程
- 不变性和纯函数的原则
- 函数式编程技术及其对程序设计的影响

> 面向对象编程（OO）通过封装变化使得代码更易理解。
> 函数式编程（FP）通过最小化变化使得代码更易理解。
>
> ——Michael Feathers (Twitter)

如果你正在阅读这本书，那么很可能你已经是一名拥有面向对象或结构化设计工作经验的 JavaScript 软件开发人员，但你对函数式编程很感兴趣。或许你曾经尝试过学习它，但并不能在工作或个人项目中成功地应用它。这样的话，你的主要目标是增强开发技能，提高代码质量，那么本书可以帮助你实现这一目标。

Web 平台的快速演进和浏览器的不断进化以及最重要的——用户的需求，给如今的 Web 应用的设计带来了意想不到的变化。人们期望 Web 应用给人的感觉应该更像本地的桌面应用，或是具有丰富且响应式的部件的移动应用。这样的期望自然而然地迫使 JavaScript 开发人员能够更广泛地去思考各种解决方案，并适时地采用那些可能提供最优解决方案的编程范式和最佳实践。

作为开发人员，我们总是更喜欢那些拥有简洁应用结构并可以增强软件扩展性的框架。然而代码库的复杂性仍然超出预期，这使得我们去重新审视这些代码的基本设计原