Broadview®
www.broadview.com.cn

[PACKT] open source*
PUBLISHING
community experience distilled

Node.js Design Patterns, Second Edition

Node.js设计模式
（第2版）

掌握Node.js强大的组件和模式，轻松创建模块化和可扩展的应用程序

[爱尔兰] Mario Casciaro [意] Luciano Mammino 著
冯康 孙光金 丁全 许程建 译

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Node.js Design Patterns, Second Edition

Node.js设计模式
（第2版）

[爱尔兰] Mario Casciaro [意] Luciano Mammino 著
冯康 孙光金 丁全 许程建 译

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

本书通过大量示例形象地阐述了 Node.js 的哲学思想和设计模式。内容主要由六部分组成：Node 核心思想、基础设计模式、异步控制流模式、流编程、Node.js 的传统设计模式和特有设计模式、通用编程的 Web 应用以及处理复杂实际问题的高级编程技巧。

这是一本值得深入品读的书籍，读者若具备一些软件设计的理论知识会有助于理解书中提出的概念。本书尤其适合于已经接触过 Node.js 并且想在效率、设计质量和可扩展性方面获得提升的开发者。

版权贸易合同登记号　图字：01-2017-0517

图书在版编目（CIP）数据

Node.js 设计模式：第 2 版/（爱尔兰）马里奥 · 卡西罗（Mario Casciaro），（意）卢西安诺 · 马米诺（Luciano Mammino）著；冯康等译. – 北京：电子工业出版社，2018.3
书名原文: Node.js Design Patterns, Second Edition
ISBN 978-7-121-33522-8
I. ① N…II. ① 马… ② 卢… ③ 冯…III. ① JAVA 语言 – 程序设计 IV. ① TP312.8

中国版本图书馆 CIP 数据核字（2018）第 012640 号

策划编辑：张春雨
责任编辑：牛　勇
印　　刷：三河市良远印务有限公司
装　　订：三河市良远印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：27.25　　字数：595 千字
版　　次：2018 年 3 月第 1 版
印　　次：2018 年 3 月第 1 次印刷
定　　价：108.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）51260888-819，faq@phei.com.cn。

译者序

当前，Node.js 在服务端语言中也占有了一席之地。它独特的语言魅力、良好的生态环境，以及在服务端中的不俗表现，使得许多人都被它深深地吸引。很多公司也逐渐开始在生产环境中使用 Node.js 提供服务。特别是前端开发者，他们大多拥有深厚的 JavaScript 技术背景，有了 Node.js 可谓如虎添翼。尤其在 Web 接入层方面，Node.js 大大提高了开发效率。同样是基于 V8 引擎的 Node.js，对于类似 React 服务端渲染的处理，也具有更多的优化选择。

JavaScript 属于弱类型语言，而 Node.js 是单线程运行，同时采用事件驱动、异步编程方式，这与类似 Java 的传统后端语言有很大的不同，这一点可以说是革命性的变化。由此也引发了一系列思考，如作者在书中提出的疑问，如何去设计我们的代码？什么是正确的开发方式？Node.js 与传统后端语言的差异，导致我们的思维方式、具体的实现都有不同。本书正是从这里入手，深度剖析 Node.js 中的设计模式、编码技巧和实践经验。

《Node.js 设计模式》在第 1 版中，就写了很多关于底层的内容，并且擅长使用简单的例子来描述一些流行库的实现及其原理，使人深受启发。升级后的第 2 版更加全面地阐述了 Node.js 的设计模式与编码问题。书中还大量使用一些著名项目中常用的技术和库作为演示，来具体阐述相关的问题，以及提出解决问题的完整思路。

我们团队在翻译本书的时候，也刚好是我们的 Node.js 业务爆发式增长的阶段。底层架构在不断地补充和完善，各个业务线正在展开。当然也面临很多的挑战——如何设计底层架构才能使其既具有灵活性又兼具扩展性，如何来规范化业务代码，如何正确处理模块间的依赖关系，等等。重读本书后，掩卷沉思，不觉对于项目的实际操作、后续优化等技术问题，又有了很多新的见解和启发性的思考。最后，感谢所有为本书的出版提供帮助的同事和编辑们！如果在翻译方面有错误和不足的地方，恳请读者批评指正。

关于作者

Mario Casciaro，软件工程硕士学位，软件工程师，企业家，对技术、科学和开源知识充满了热情。他在IBM开始了职业生涯，数年间先后参与很多不同产品的开发，例如Tivoli Endpoint Manager、Cognos Insight及SalesConnect。后来，他加入了一个成长中的SaaS公司——D4H Technologies，负责开发一款实时应急管理的前沿产品。现在，Mario是Sponsorama.com的联合创始人兼CEO，这是一个帮助在线项目募集企业赞助资金的平台。

Mario也是*Node.js DesignPatterns*（Node.js设计模式）第1版的作者。

关于翻译团队

翻译成员全部来自陆金所大前端团队，也是公众号“大前端工程师”的翻译小组成员，他们在公众号与知乎专栏里面也有很多新的技术文章的翻译。此次由寸志老师带队，大家一边在公司进行 Node.js 项目的推广实践，一边将实践的心得注入到本书翻译的理解，这是非常难得的结合，相信大家在读的过程中能体会到这一点。

致谢

致谢 1

当我在写这本书的第 1 版时，从来没有想过它会如此成功。在这里要感谢第 1 版的读者、购买了这本书的人、为这本书留下评论的人，以及在 Twitter 或其他在线论坛向朋友们推荐这本书的人。当然我的感激之情，还要献给第 2 版的读者们，也就是正在阅读这些文字的你，是你让我的努力有了意义。同时，我想要你和我一起来祝贺我的朋友，也就是第 2 版的共同作者 Luciano，他为第 2 版的更新和新增内容做了大量的工作。第 2 版所有的功劳都应该归功于他，我只是扮演了一个顾问的角色。写书并不是一个简单的工作，但是 Luciano 给我和 Packt 出版社的工作人员们留下了深刻的印象。他的献身精神、专业能力和技术能力，向我们证明了他可以达到任何想要达到的目标。我很荣幸能与 Luciano 一起工作，也期待今后新的合作。我也要感谢其他为此书付出努力的人：Packt 出版社的工作人员、技术评审（Tane 和 Joel），以及所有为此书提出有价值意见和见解的朋友们：Anton Whalley(@dhigit9)、Alessandro Cinelli (@cirpo)、Andrea Giuliano (@bit_shark) 和 Andrea Mangano (@ManganoAndrea)。谢谢所有给我无条件的爱的朋友和家人，最重要的是我的女朋友 Miriam，她是我所有冒险旅程的同伴，为我生命中的每一天带来爱和快乐。

Luciano Mammino 是一名软件工程师，1987 年出生。在那一年，任天堂在欧洲发布了《超级马里奥兄弟》，这是他最喜欢的视频游戏。他 12 岁就开始写代码，用的是他父亲旧式的英特尔 386 计算机，只有 DOS 操作系统和 QBASIC 语言解释器。在取得计算机科学硕士学位以后，他主要作为一名自由职业者在意大利一些成熟公司和初创公司从事 Web 开发工作。他的编程技巧获得提升主要是在这段时间。他曾在意大利的 start-up parenthesis 担任过 3 年的首席技术官，是爱尔兰 Sbaam.com 公司的创始人之一。之后他到柏林工作，成为 SmartBox 一名 PHP 高级开发工程师。他喜欢开发开源软件库，喜欢在工作中使用像 Symfony 和 Express 这样的框架。他坚信 JavaScript 的发展尚处于起步阶段，这项技术在将来会对大多数 Web 和

移动相关技术产生巨大的影响。因此，他把大部分的空闲时间都用来学习 JavaScript 和耕耘 Node.js。

致谢 2

首先最要感谢的是 Mario 能给予我机会和信任，让我和他一起撰写本书的第 2 版。这是一个非常棒的经历，希望这也是今后一系列合作的开始。

其次，要感谢 Packt 出版社团队，他们的工作惊人地高效。特别要感谢 Onkar、Reshma 和 Prajakta 三人，感谢他们为本书付出的努力和持有的耐心。也要感谢评审 Tane Piper 和 Joel Purra，他们在 Node.js 方面的经验对这本书质量提升有非常重要的作用。

我要给这些朋友一个大大的拥抱（当然还有很多啤酒），他们是 Anton Whalley（@dhigit9）、Alessandro Cinelli (@cirpo)、Andrea Giuliano (@bit_shark) 和 Andrea Mangano(@ManganoAndrea)。感谢他们一直以来对我的鼓励，以及分享的开发者经验和关于此书的深刻的见解。

另外要感谢的是 Ricardo、Jose、Alberto、Marcin、Nacho、David、Arthur 以及 SmartBox 的同事们，是他们让我爱上工作，鼓舞并激励我成为一个更好的软件开发工程师。相信没有比这更好的团队了。

我要深深地感谢我的家人，是他们在我人生中一直不断地鼓励我。感谢我的母亲，她是我生命动力和灵感的不竭源泉。感谢父亲的所有教诲、鼓励和建议，我很想和你交流，真的很想念你。谢谢我的兄弟姐妹，Davide 和 Alessia，在我痛苦和开心的时候都有你们在我身边，让我感觉我是大家庭的一员。

谢谢 Franco 和他的家人，支持我许多最初的想法，并给我分享他们的智慧和生活经验。

感谢我的“书呆子”朋友 Gianluca、Flavio、Antonio、Valerio、Luca，和他们在一起度过了一段美好时光，并且在这本书的创作过程中给予了我很多鼓励。

还要感谢“不那么书呆子”的朋友 Damiano、Pietro 和 Sebastiano。在 Dublin 的时候他们总是带给我很多的欢笑。

最后，也是最重要的，要感谢我的女朋友 Francesca。谢谢你无条件的爱，以及对我每一次冒险的支持，甚至是非常疯狂的冒险。期待与你共谱人生新篇章！

关于审稿者

Tane Piper 是英国伦敦一位全栈开发工程师。他曾在多家机构和公司从事软件工作，至今已有 10 年以上开发经验。期间，他曾使用多种开发语言，例如 Python、PHP 和 JavaScript。他自 2010 年便在工作中使用 Node.js，并且在 2011 年和 2012 年间在英国和爱尔兰的几次会议中，他最早参与探讨服务端 JavaScript 应用。他还是 jQuery 项目早期的贡献者和倡导者。目前，他在伦敦一家咨询公司工作，主要提供一些 React 和 Node 方面的创新解决方案。在专业工作之外，他热衷于潜水，还是位业余摄影师。

我非常感谢我的女朋友 Elina，她让我近两年的生活一帆风顺，并鼓励我去负责这本书的审稿工作。

——Tane Piper

Joel Purra 在他十几岁的时候便开始玩电脑，那时候他只是把电脑当作另一种视频游戏设备。不久，他在用电脑玩最新游戏的时候，会把它们都拆成零件（有时弄坏了，然后想办法修好）。是游戏让他在十几岁的时候就接触编程，当时有一款叫 Lunar Lander（平衡飞船）的游戏引发了他创建数字化工具的兴趣。后来随着家里接入了互联网，他开发了他的第一个电子商务网站，从此开始了他的事业。种种经历使他在很小的时候便开始了职业生涯。17 岁时，乔尔开始在一家核电站的学校学习计算机编程和一个能源科学项目。毕业后，他凭借他的研究能力成为瑞典陆军第二位中尉通信专家。后来他到 Linköping 大学继续读书，取得了信息技术与工程理科硕士学位。1998 年后，他在一些创业公司和成熟公司工作。从 2007 年开始，他成为了一名顾问。Joel Purra 在瑞典出生，长大，读书。作为一名自由职业的开发人员，他很享受这种弹性的生活方式。他作为一个背包客已经旅行了五大洲，并在国外生活了几年。他是一位不断寻求挑战的学者，建立和开发能广泛公用的软件是他的目标之一。你可以访问他的个人网站：`http://joelpurra.com/`。

我要感谢开源社区提供的用于构建大、小软件系统的积木。社区中不乏一些自由职业顾问。我们就好像站在巨人的肩膀上。记得早提交，常提交！

——Joel Purra

前言

很多人认为Node.js的出现是Web开发领域十年内最大的变化，它就像是游戏规则的改变者。之所以被喜爱不仅是因为技术上的出众能力，同时也因为它带给Web开发新的思维方式。

首先，Node.js应用是使用JavaScript语言编写的，而JavaScript又是唯一被绝大多数Web浏览器原生支持的编程语言。该特性使得单语言应用栈以及服务端、客户端代码共享成为可能。Node.js本身也促进了JavaScript语言的兴起和发展。人们意识到，在服务端使用JavaScript并不像在浏览器端使用它那样糟糕，并且人们将慢慢喜欢上它的编程思维和它混合的天性，即面向对象和函数式编程的结合。

其次，单线程和异步架构也是Node.js带来的革命性变化。除了性能和可扩展性方面的明显优势外，其改变了开发者处理程序并发和并行的方式。队列取代了互斥锁，回调函数和事件机制取代了多线程，因果关系取代了同步性。

最后也是最重要的一点，Node.js拥有一套完整的生态系统：npm包管理器、不断增长的模块数量、热情活跃的开发社区，以及基于简单、实用主义和极端模块化而产生的独特文化。

然而，因为这些特性，Node.js开发给人一种与其他服务端语言开发非常不一样的感受，刚开始接触Node.js的开发者，会经常困惑于如何有效地解决一些最常见的设计和代码编写问题。常见的问题有："如何组织代码？""设计这个系统的最好方法是什么？""怎样使我的程序更加模块化？""我该怎样高效实现大量的异步调用？""我该如何确保我的程序随着规模增大会一直稳定运行，不会崩溃？"或者更简单的问题，"Node.js开发的正确方式是什么？"幸运的是，Node.js已经成为一个非常成熟的开发平台，以上大部分问题都能通过设计模式、被证明有效的编码技巧或者他人提供的经验来解决。本书的目的就是指导你学习并掌握Node.js开发的一些设计模式、编码技巧和实践经验，告诉你解决这些常见问题的有效方法并教会你如何从这些方法出发，解决你自己遇到的特定问题。

通过阅读本书，你将掌握以下这些内容：

- Node.js 的开发方式

  如何使用正确的思维方式去解决一个 Node.js 开发设计问题。比如你会学习到，传统设计模式在 Node.js 开发中的不同体现，或者如何设计提供单一功能的模块。

- 一整套解决常见 Node.js 设计和编码问题的设计模式

  你会学习到一整套像“瑞士军刀”一样功能多样、实用的设计模式，并且你能即学即用，解决日常遇到的程序开发和设计问题。

- 如何编写模块化、高效率的 Node.js 程序

  你将会了解开发大规模并且结构组织合理的 Node.js 程序的基本方法，并能运用这些方法去解决不属于现有设计模式范畴的新问题。

在本书中，你会看到一些真实项目中用到的库和技术，比如 LevelDb、Redis、RabbitMQ、ZMQ 及 Express 等。这些会用来作为示例阐述某个设计模式或者方法，除了让例子更加实用外，它们同时会让你对 Node.js 的生态系统以及它解决问题的一套方法有所了解。

无论你正使用或打算在你的工作、非正式项目或者开源项目中使用 Node.js，认识和使用众所周知的设计模式和技术能够让你通过一种通用的语言和他人共享你的代码和设计，不仅如此，这还会帮助你更好地了解 Node.js 的未来，以及知道如何为其发展贡献自己的一份力量。

各章介绍

第 1 章，*欢迎来到 Node.js 平台*，本章通过讲解 Node.js 本身核心的设计模式来介绍 Node.js 程序的设计，包括 Node.js 的生态系统、编程思想，以及 Node.js V6 版本、ES2015 和 Reactor 模式的简单介绍。

第 2 章，*Node.js 基础设计模式*，开始介绍 Node.js 异步编程和设计模式，讨论和比较了回调函数与事件触发器 (观察者模式)。本章还介绍了 Node.js 的模块系统和相关模块的设计模式。

第 3 章，*异步控制流模式之回调函数*，介绍了系列用于有效处理 Node.js 中的异步控制流的模式和技术。这一章将教你怎样使用纯 JavaScript 和异步库来缓解“回调地狱”的问题。

第 4 章，*异步控制流模式之 ES2015+*，介绍了 Promise、Generator 和 async-await 的异步控制流的探索进展。

第 5 章，*流编程*，深度挖掘 Node.js 中最重要的模式之一：流。本章将向你展示如何处理数据流交换及如何将它们组合成不同的布局。

第 6 章，设计模式，本章涉及一个有争议的话题：Node.js 的传统设计模式。介绍了最流行的传统设计模式，并展示了它们在 Node.js 中的应用。同时也介绍了一些 JavaScript 和 Node.js 中独有的新设计模式。

第 7 章，连接模块，分析了将多个模块关联到一个应用程序中的不同解决方案。在本章中我们将学习几个设计模式，例如依赖注入容器和服务定位器。

第 8 章，通用 JavaScript 的 Web 应用程序，探讨了现代 JavaScript Web 应用最有趣的功能之一：前、后端代码共享。本章我们将学习通用的 JavaScript 基本原则，通过使用 React、Webpack 和 Babel 来构建一个简单的 Web 应用程序。

第 9 章，高级异步编程技巧，本章展示怎样使用直接可用的解决方案来解决一些常见的编码和设计问题。

第 10 章，扩展和架构模式，介绍扩展 Node.js 应用的基本技术和模式。

第 11 章，消息传递与集成模式，提出了最重要的消息传递模式，介绍如何构建和集成使用 ZMQ 和 AMQP 的复杂的分布式系统。

你需要为本书准备什么

为了试验代码，需要安装 Node.js 第 6 版（或更高版本）和 npm 3(或更高版本)。一些例子还要求使用转码器，例如 Babel。还需要熟悉命令提示符，了解如何安装 npm 包，还要了解怎样运行 Node.js 应用。还需要有一个文本编辑器来编写代码和一个现代浏览器进行测试。

适合读者

本书适合于已经接触过 Node.js 并且想在效率、设计质量和可扩展性方面获得提升的开发者。由于本书也包含一些基本概念，因此你只需要通过一些基本例子了解相关技术即可。中级 Node.js 的开发者也会从本书有所收获。

具备一些软件设计理论背景知识也会有助于理解本书提出的概念。

本书假定你有 Web 应用开发、JavaScript、Web 服务、数据库和数据结构的相关知识。

约定

在本书中，你会发现许多文本样式，这些样式用于区分不同种类的信息。下面是一些这些样式的例子和它们表示的含义。

代码块设置如下：

```
const zmq = require('zmq')
const sink = zmq.socket('pull');
sink.bindSync("tcp://*:5001");

sink.on('message', buffer => {
  console.log(`Message from worker: ${buffer.toString()}`);
});
```

当希望读者特别注意代码块的特定部分时，以粗体显示该部分：

```
function produce() {
  //...
  variationsStream(alphabet, maxLength)
    .on('data', combination => {
    //...
    const msg = {searchHash: searchHash, variations: batch};
    channel.sendToQueue('jobs_queue', new Buffer(JSON.stringify(msg)));
    //...
    })
  //...
}
```

任何命令行输入或输出设置如下：

```
node replier
node requestor
```

新术语和**重要词汇**会以黑体（汉字）或粗体（英文）显示。

目录

第 1 章　欢迎来到 Node.js 平台 1

Node.js 的哲学思想 2

小核心 2

小模块 2

小接触面 3

简单和实用 3

认识 Node.js 6 和 ES2015 4

let 和 const 关键字 5

箭头函数 6

类语法 8

增强的对象字面量 10

Map 和 Set 集合 11

WeakMap 和 WeakSet 集合 13

模板字面量 14

其他 ES2015 特性 14

Reactor 模式 15

I/O 是缓慢的 15

阻塞 I/O 15

非阻塞 I/O 16

事件多路分解器 17

Reactor 模式简介 19

Node.js-libuv 的非阻塞 I/O 引擎 20

Node.js 的秘诀 21

总结 21

第 2 章　Node.js 基础设计模式 23
回调模式 24
CPS（Continuation Passing Style） 24
同步或异步 26
Node.js 回调约定 31
模块系统及其模式 34
揭示模块模式 34
Node.js 模块解释 35
模块定义模式 42
观察者模式 49
EventEmitter 类 49
创建和使用 EventEmitter 50
传播错误 51
使任何对象可观察 51
同步和异步事件 53
EventEmitter 与回调 54
组合回调和 EventEmitter 55
总结 55

第 3 章　异步控制流模式之回调函数 56
异步编程的困难 56
创建一个简单的网络蜘蛛 57
回调地狱 59
使用纯 JavaScript 60
回调规则 60
应用回调规则 61
顺序执行 63
并行执行 68
有限制的并行执行 73
async 库 77
顺序执行 78
并行执行 81
有限制的并行执行 81
总结 83

第 4 章　异步控制流模式之 ES2015+ 84
promise 84
什么是 promise 85
Promises/A+ 实现 87
Node.js 风格函数的 promise 化 88
顺序执行 90
并行执行 93
有限制的并行执行 93
在公共 API 中暴露 callback 和 promise 95
generator 97
generator 基础 97
generator 的异步控制流 100
顺序执行 104
并行执行 106
有限制的并行执行 108
使用 Babel 的 async await 111
安装和运行 Babel 112
比较 113
总结 114

第 5 章　流编程 115
流的重要性 115
缓冲和流 116
空间效率 117
时间效率 118
组合性 121
开始学习流 122
流的分类 122
可读流 123
可写流 127
双向流 (Duplex stream) 132
变换流 132
使用管道拼接流 135
使用流处理异步流程 137
顺序执行 138
无序并行执行 139
无序有限制的并行执行 143