

The  
Pragmatic  
Programmers

# Node.js 开发实战

Node.js 8 the Right Way

[美] Jim R. Wilson 著

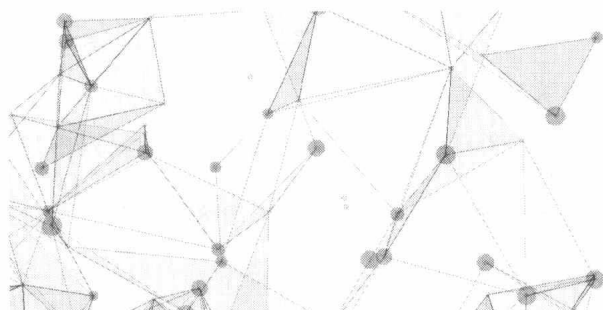
梅晴光 杜万智 陈琳 纪清华 段鹏飞 译



华中科技大学出版社  
<http://www.hustp.com>

# Node.js 开发实战

Node.js 8 the Right Way



[美] Jim R. Wilson 著

梅晴光 杜万智 陈琳 纪清华 段鹏飞 译

华中科技大学出版社  
中国·武汉

## 内 容 简 介

Node.js 是基于 Chrome V8 引擎的 JavaScript 运行环境,它采用事件驱动、非阻塞式 I/O 模型,具有轻量、高效的特点。Node.js 工作在前端代码与数据存储层之间,能够提高 web 应用的工作效率和响应速度。本书以最新版 Node.js 为基础,从实际案例出发讲解 Node.js 的核心工作原理和实用开发技巧,既注重实用性又具有深度,适合 JavaScript 程序员进阶学习。

Node.js 8 the Right Way © 2018 The Pragmatic Programmers, LLC. All rights reserved.

湖北省版权局著作权合同登记 图字:17-2018-315 号

### 图书在版编目(CIP)数据

Node.js 开发实战 / (美) 吉姆·威尔逊 (Jim R. Wilson) 著;梅晴光等译. —武汉:华中科技大学出版社,2018.11

ISBN 978-7-5680-4766-1

I. ①N… II. ①吉… ②梅… III. ①JAVA 语言-程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2018)第 249557 号

Node.js 开发实战

[美]Jim R. Wilson 著

Node.js Kaifa Shizhan

梅晴光 杜万智 陈琳 纪清华 段鹏飞 译

策划编辑:徐定翔

责任编辑:徐定翔

责任监印:赵月

出版发行:华中科技大学出版社(中国·武汉) 电话:(027)81321913

武汉市东湖新技术开发区华工科技园 邮编:430223

录 排:华中科技大学惠友文印中心

印 刷:湖北新华印务有限公司

开 本:787mm×960mm 1/16

印 张:19.5

字 数:447千字

版 次:2018年11月第1版第1次印刷

定 价:99.90元



华中出版

本书若有印装质量问题,请向出版社营销中心调换  
全国免费服务热线:400-6679-118 竭诚为您服务  
版权所有 侵权必究

# 前言

---

## Preface

近年来，软件开发领域发生了两大变革，Node.js 都处在变革的最前沿。

第一，异步编程技术应用越来越广泛。不论是等待大数据计算任务的完成，与客户端交互，操控无人机，还是响应 API 请求，你都会用到异步编程技术。

第二，JavaScript 运行环境已经成为通用的代码执行环境，它无处不在。浏览器、NoSQL 数据库、机器人、服务器中都能运行 JavaScript。

Node.js 已成为这两大变革不可或缺的组成部分，并且发挥了巨大作用。

## 为什么写这本书

### Why Node.js the Right Way

让我们把时间拨回到 2010 年 3 月，我在波士顿 NoSQL 会议上做了一个主题为“全栈 JavaScript”的小分享。那时我就意识到可以使用 JavaScript 实现系统架构中的所有技术栈，同时大幅降低系统的复杂度。

如果系统中所有部分都是使用 JavaScript 实现的，那么你将很容易解决系统匹配问题，轻松实现代码复用。Node.js 将成为连接前端用户界面和数据存储层的重要一环。

本书内容既包含了 Node.js 入门知识，又涵盖了 Node.js 程序开发的深入实践。

## 学习 Node.js

像所有快速兴起的技术一样，Node.js 也有大量学习资源。遗憾的是，其中大部分都只是针对 web 应用的。

web 应用有非常广泛的影响，但它不是 Node.js 的全部。Node.js 不局限于提供 web 服务，本书将会介绍它在更多领域的应用。

无论你面对什么类型的需求，本书都会提供你所需要的知识，让你成为高效的 Node.js 程序员。

## 使用 Node.js

我喜欢 JavaScript，它有多种实现方式。开发者有很大的空间可以探索和实验，找到最佳的实现方式。

Node.js 开发社区、Node.js 开发规范，甚至 JavaScript 语言本身都在快速发展，本书的示例代码和开发建议都是基于当前的最佳实践，同时也充分考虑到未来可能发生的变化。

# 本书内容

## What's in This Book

本书针对的读者是想学习 Node.js 异步编程的中级开发者和高级开发者。阅读本书之前最好掌握一些 JavaScript 的知识，但不必精通。全书分为三个部分，在此先简单介绍这三部分的内容。

### 第一部分：开始接触 Node.js

第一部分(第 1 章至第 4 章)讲解 Node.js 的基础知识。你会学着写一些 Node.js 代码，使用原生的核心模块和外部模块实现一些简单功能，例如：与文件系统进行交互、启动子进程、管理网络连接，等等。

第 1 章介绍 Node.js 中的事件循环以及为什么 Node.js 能够在单线程的前提下支持高并发场景。本章还简要介绍了在后续章节中会提及的 Node.js 开发中的五个方面，以及如何安装 Node.js。

在第 2 章你会写更多的 Node.js 代码。如果你之前开发过服务端程序，那么应该接触过文件的读/写。我们将从这个常见需求出发，使用 Node.js 的文件模块创建异步的、非阻塞的文件处理工具。我们也会使用 Node.js 中非常常用的 `EventEmitter` 和 `Stream` 类处理数据，还会创建子进程并与之进行通信。

第 3 章详细阐述了 Node.js 的网络 I/O 开发。我们会开发 TCP 服务器，并创建一个与服务器通信的客户端。还会开发一个自定义类来发送消息，它的消息遵循基于 JSON 格式的消息通信协议。我们会使用 Node.js 中非常流行的测试框架 Mocha 开发单元测试。

第 4 章将把注意力转向第三方框架。你将会学习使用 `npm` 引入高效率低延迟的网络应用开发库 `ØMQ`（读作“Zero-M-Q”）。你将会学习使用 `ØMQ` 开发基于发布-订阅模式和请求-应答模式的网络应用，创建一系列互相配合的程序，还会学习进程管理工具。

## 第二部分：数据处理

第二部分（第 5 章、第 6 章）学习如何操作数据，为端到端的应用打好基础。先从处理数据文件开始，使用易于测试的方式处理数据。还会学习使用 Node.js 开发命令行工具，以及如何与 HTTP 服务进行交互。

第 5 章将启动一个贯穿第二部分和第三部分的项目。从一个叫 Project Gutenberg 的电子书在线服务下载代码。使用 `Cheerio` 模块解析数据文件和提取重要字段。使用 `npm`、`Mocha`，以及 `Chai` 断言库来搭建集成测试环境，还会学习使用 Chrome 浏览器的 `DevTools` 开展交互式的调试。

第 6 章学习将 Project Gutenberg 数据导入 `Elasticsearch` 索引。你将会使用到 `Commander` 模块开发一个叫 `esclu` 的命令行工具来完成数据的导入。由于 `Elasticsearch` 是基于 JSON 的 RESTful 数据存储，所以你会使用 `Request` 模块与它进行交互，还会使用 `jq` 这个非常强大的命令行工具来处理 JSON。

## 第三部分：从头开始创建应用程序

第三部分（第 7 章至第 9 章）把之前学习的知识综合起来开发一个 web 服务，这个服务处理从 API 调用到后端数据服务之间的逻辑。终端用户不直接发起 API 请求，所以需要实现一个美观的用户界面。最后实现 `session` 管理和认证，使用它

把界面和 web 服务结合起来。

Node.js 完美支持 HTTP 服务的开发，第 7 章讲解这方面的知识。使用 Express 做路由（Express 是一个非常流行的 Node.js web 框架）。深入阐述 REST 语义，并讲解如何使用 Promise 和 async 函数组织异步代码。最后学习使用 nconf 模块配置服务，使用 nodemon 模块维持服务运行。

第 8 章学习前端界面的开发。使用 webpack 将前端项目打包（webpack 是基于 Node.js 的构建工具）。使用 TypeScript 将代码转换成可以在浏览器执行的代码。TypeScript 是微软开发的一种语言，具有类型检查的特性，它提供的转译器可以将 TypeScript 转译成 JavaScript。

第 9 章把用户界面和 web 服务连接起来，形成一种端到端的解决方案，使用 Express 中间件实现身份认证 API 和有状态的 session。学习使用 npm 的 shrinkwrap 选项保证系统不会受依赖模块更新的影响。

第 10 章讲解 Node-RED 智能可视化编辑器的用法，在 Raspbian 树莓派操作系统上设计和开发基于事件的应用程序。学习使用 Node-RED 快速开发 HTTP API。

附录 A 和附录 B 分别介绍 Angular 开发环境和 React 开发环境的设置，并学习通过 webpack 把它们跟 Express 整合起来。

## 本书不包含的内容

### What This Book Is Not

开始读本书之前，你应该知道哪些内容不会出现在本书里。

#### Node.js 百科全书

现在 npm 仓库里有超过 528000 个模块，而且以每天 500 个的速度在增加。Node.js 社区非常大，并且在高速增长，本书不可能包含所有这些内容。

书中提到的 Node.js 与非 Node.js 服务之间的调用非常复杂，我们的示例程序会涉及处理多个系统和用户之间的互相调用，涉及许多前后端技术。我们会简要介绍这些技术，让大家了解其全貌，但不会做太深入的探讨。

## MEAN

本书不会专门介绍特定的技术栈，例如 MEAN<sup>1</sup>（Mongo、Express、Angular、Node.js）。我们把注意力放在 Node.js 的学习和使用上。

我选择 Elasticsearch 而不选择 MongoDB 做数据库。据 RisingStack 在 2016 年的一份调查报告显示，Elasticsearch 在资深 Node.js 开发者中越来越流行<sup>2</sup>。

本书也不讲解前端框架。目前最流行的两个前端框架是 Facebook 的 React<sup>3</sup>和 Google 的 Angular<sup>4</sup>。大家自己可以多关注这两个框架。

## JavaScript 初学者指南

JavaScript 语言可能是当今最被误解的语言。虽然我们偶尔会讨论 JavaScript 语法（尤其是 ES6+语法），但本书不是 JavaScript 初学者指南。你需要掌握 JavaScript 的基本语法，比如你应能读懂下面的代码，能够理解其中的逻辑。

```
const list = [];
for (let i = 1; i <= 100; i++) {
  if (!(i % 15)) {
    list.push('FizzBuzz');
  } else if (!(i % 5)) {
    list.push('Buzz');
  } else if (!(i % 3)) {
    list.push('Fizz');
  } else {
    list.push(i);
  }
}
```

细心的读者会发现，这是 Jeff Atwood 在 2007 年提出的经典编程题 FizzBuzz<sup>5</sup>的答案。下面是另一个版本的答案，其中使用了更高级的 JavaScript 新特性。

```
'use strict';
const list = [...Array(100).keys()]
  .map(n => n + 1)
  .map(n => n % 15 ? n : 'FizzBuzz')
  .map(n => isNaN(n) || n % 5 ? n : 'Buzz')
  .map(n => isNaN(n) || n % 3 ? n : 'Fizz');
```

<sup>1</sup> <http://www.modulecounts.com/>

<sup>2</sup> <https://blog.risingstack.com/node-js-developer-survey-results-2016/>

<sup>3</sup> <https://facebook.github.io/react/>

<sup>4</sup> <https://angularjs.org/>

<sup>5</sup> <https://blog.codinghorror.com/why-cant-programmers-program/>



如果你看不懂上面这段代码也没关系，本书会讲解如何使用这些新特性。

### 给 Windows 用户的小提示

本书的示例代码是针对 Unix 操作系统编写的，我们使用标准输入/输出流，标准的数据传输方式，shell 脚本都在 Bash 中测试通过，也能在其他 shell 命令行工具运行。

如果你使用 Windows 操作系统，我建议你安装 Cygwin<sup>6</sup>，或者运行 Linux 虚拟机，这样可以尽量保证示例代码正常运行。

## 示例代码和格式约定

### Code Examples and Conventions

本书包含 JavaScript、shell 脚本和 HTML/XML 代码，大部分情况下，示例代码都可以在运行环境下直接执行。Shell 命令行语句以\$开头。

本书会讲解如何捕获异常，开发 Node.js 程序应该养成捕获异常的习惯，哪怕只是简单地捕获之后重新抛出，也应该把捕获逻辑写上。但是，为了提高代码的可读性和节省版面，书中有些示例代码没有写捕获异常的逻辑。请务必自己养成处理异常逻辑的习惯。

## 在线资源

### Online Resources

Pragmatic Bookshelf 网站<sup>7</sup>上有本书的相关资源，你可以下载所有示例代码，同时可以在上面找到在线论坛和勘误表。

最后，感谢你选择本书开启 Node.js 开发之路。

Jim R. Wilson  
2017 年 12 月

---

<sup>6</sup> <http://cygwin.com/>

<sup>7</sup> <http://pragprog.com/book/jwnode2/node-js-8-the-right-way>

# 目录

## Contents

第一部分 开始接触 Node.js .....	1
第 1 章 入门 .....	3
1.1 不限于 Web .....	3
1.2 Node.js 的应用范围 .....	4
1.3 Node.js 的工作原理 .....	6
1.4 Node.js 开发的 5 个方面 .....	8
1.5 安装 Node.js .....	9
第 2 章 文件操作 .....	11
2.1 Node.js 事件循环编程 .....	12
2.2 创建子进程 .....	16
2.3 使用 EventEmitter 获取数据 .....	18
2.4 异步读/写文件 .....	20
2.5 Node.js 程序运行的两个阶段 .....	24
2.6 小结与练习 .....	24
第 3 章 Socket 网络编程 .....	26
3.1 监听 Socket 连接 .....	27
3.2 实现消息协议 .....	32
3.3 建立 Socket 客户端连接 .....	34
3.4 网络应用功能测试 .....	36
3.5 在自定义模块中扩展 Node.js 核心类 .....	39
3.6 使用 Mocha 编写单元测试 .....	44
3.7 小结与练习 .....	50

第 4 章 创建健壮的微服务 .....	52
4.1 安装 ØMQ .....	53
4.2 发布和订阅消息 .....	58
4.3 响应网络请求 .....	61
4.4 运用 ROUTER/DEALER 模式 .....	65
4.5 多进程 Node.js .....	68
4.6 推送和拉取消息 .....	72
4.7 小结与练习 .....	75
第二部分 数据处理 .....	79
第 5 章 数据转换 .....	81
5.1 获取外部数据 .....	82
5.2 基于 Mocha 和 Chai 的行为驱动开发 .....	84
5.3 提取数据 .....	90
5.4 依次处理数据文件 .....	100
5.5 使用 Chrome DevTools 调试测试 .....	103
5.6 小结与练习 .....	108
第 6 章 操作数据库 .....	111
6.1 Elasticsearch 入门 .....	112
6.2 使用 Commander 创建命令程序 .....	114
6.3 使用 request 获取 JSON .....	120
6.4 使用 jq 处理 JSON .....	125
6.5 批量插入 Elasticsearch 文档 .....	128
6.6 实现 Elasticsearch 查询命令 .....	132
6.7 小结与练习 .....	139
第三部分 从头开始创建应用程序 .....	143
第 7 章 开发 RESTful Web 服务 .....	145
7.1 使用 Express 的好处 .....	146
7.2 运用 Express 开发服务端 API .....	147
7.3 编写模块化的 Express 的服务 .....	149
7.4 使用 nodemon 保持服务不间断运行 .....	153
7.5 添加搜索 API .....	154

7.6	使用 Promise 简化代码 .....	159
7.7	操作 RESTfull 文档 .....	165
7.8	使用 async 和 await 模拟同步 .....	168
7.9	为 Express 提供一个 async 处理函数 .....	170
7.10	小结与练习 .....	178
<b>第 8 章</b>	<b>打造漂亮的用户界面 .....</b>	<b>181</b>
8.1	开始使用 webpack .....	182
8.2	生成第一个 webpack Bundle .....	186
8.3	使用 Bootstrap 美化页面 .....	188
8.4	引入 Bootstrap Javascript 和 jQuery .....	192
8.5	使用 TypeScript 进行转译 .....	193
8.6	使用 Handlebars 处理 HTML 模板 .....	197
8.7	实现 hash 路由 .....	200
8.8	在页面中展示对象数据 .....	202
8.9	使用表单保存数据 .....	207
8.10	小结与练习 .....	211
<b>第 9 章</b>	<b>强化你的应用 .....</b>	<b>214</b>
9.1	设置初始项目 .....	215
9.2	在 Express 中管理用户会话 .....	219
9.3	添加身份验证 UI 元素 .....	222
9.4	设置 Passport .....	224
9.5	通过社交账号进行身份验证 .....	228
9.6	编写 Express 路由 .....	240
9.7	引入书单 UI .....	245
9.8	在生产模式下部署服务 .....	246
9.9	小结与练习 .....	250
<b>第 10 章</b>	<b>使用 Node-RED 进行流式开发 .....</b>	<b>252</b>
10.1	配置 Node-RED .....	252
10.2	保护 Node-RED .....	254
10.3	开发一个 Node-RED 流 .....	255
10.4	使用 Node-RED 创建 HTTP API .....	259
10.5	处理 Node-RED 流中的错误 .....	269

10.6 小结 .....	276
附录 A 配置 Angular 开发环境 .....	277
附录 B 配置 React 开发环境 .....	282
索引 .....	285
翻译审校名单 .....	300

# 第一部分

## 开始接触 Node.js

### Getting Up to Speed on Node.js

Node.js 是强大的 JavaScript 服务端开发平台。

第一部分将从简单的命令行程序讲起，然后逐渐深入微服务开发。在此过程中，你将学习如何把代码导出为模块，如何使用 npm 中的第三方模块，以及如何使用最新的 ECMAScript 语言特性。



# 第 1 章

---

## 入门

### Getting Started

有一句编程名言：功能是资产，代码是负债<sup>1</sup>。

在学习本书的 Node.js 程序时，请牢记这句名言，避免不必要的代码，尽量使用已有的成果。

不过，本书会教你自己实现一些功能，而不管这些功能是否已经被其他人实现过。因为只有自己动手做过，并且了解其中的原理，才能更好地利用它。

我们会循序渐进地学习。为了熟悉开发环境、语言特性、基础 API，我们会使用最基础的方式进行开发。等你打好基础后，再使用第三方模块、库、服务来代替之前写的代码。

你会发现使用第三方库很方便，而自己开发实现这些功能也是很有意义的事情。一旦你认识到这一点，就说明你已经成功脱离了初学者的行列。

## 1.1 不限于 Web

### Thinking Beyond the web

围绕 Node.js 的讨论大多局限于 web，人们常常忽略了它在其他领域的作用。我们使用图 1.1 来展示 Node.js 的应用范围。

---

<sup>1</sup> <http://c2.com/cgi/wiki?SoftwareAsLiability>



把所有计算机程序想象成栖息在大海里的生物，功能相似的程序栖息在相近的地方，功能不同的程序则相隔较远。图 1.1 是程序大海里的一个小岛，叫 I/O 密集之岛。

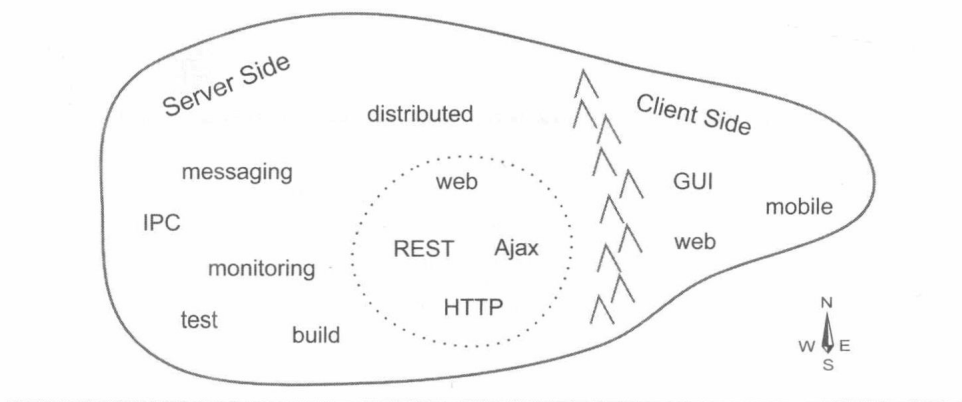


图 1.1 I/O 密集之岛

客户端程序在岛的东侧，包括各种 GUI 工具、购物应用程序、手机端应用程序和各种 web 应用程序。客户端程序直接与用户进行交互，它们通常需要等待用户输入。

服务端程序在岛的西侧，这片广袤的地区都是 Node.js 的领域。

在服务端领域深处，有一片叫 web 的区域，其中包含传统的 HTTP、Ajax、REST。那些吸引了大家注意力的网站、应用程序和 API 栖息在这里。

现在的情况是人们过分强调 Node.js 在 web 开发领域中的作用，而实际上 Node.js 的应用范围比这广得多，本书会带你慢慢探索。

## 1.2 Node.js 的应用范围

### Node.js's Niche

自从 JavaScript 1995 年发明以来，它一直被用于解决前端到后端的各种问题。

图 1.2 展示了 Node.js 的应用范围。