HZ BOOKS
华章科技

PEARSON

Web开发
技术丛书

Node.js实战

[美] Marc Wandschneider 著　　姚立　彭森材 译

Learning Node.js
A Hands-On Guide to Building Web Applications in JavaScript

- 由拥有20余年开发经验的Web技术权威、Google高级工程师兼畅销书作家撰写
- 系统讲解Node.js基础知识、核心概念和高级特性，同时包含大量案例和最佳实践，是系统学习和进阶修炼Node.js的实战教程

机械工业出版社
China Machine Press

Web开发
技术丛书

Node.js实战

Learning Node.js

A Hands-On Guide to Building Web Applications in JavaScript

[美] Marc Wandschneider 著　姚立 彭森材 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Node.js 实战 /（美）温施耐德（Wandschneider, M.）著；姚立，彭森材译 . —北京：机械工业出版社，2014.3
（Web 开发技术丛书）
书名原文：Learning Node.js: A Hands-On Guide to Building Web Applications in JavaScript

ISBN 978-7-111-45969-9

I. N… II. ①温… ②姚… ③彭… III. JAVA 语言 – 程序设计 IV. TP312

中国版本图书馆 CIP 数据核字（2014）第 033988 号

本书版权登记号：图字：01-2013-5980

Node.js 实战

（美） Marc Wandschneider 著

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：关 敏

印 刷：三河市杨庄长鸣印刷装订厂		版 次：2014 年 4 月第 1 版第 1 次印刷	
开 本：186mm × 240mm 1/16		印 张：14.5	
书 号：ISBN 978-7-111-45969-9		定 价：59.00 元	

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991 88361066 投稿热线：（010）88379604
购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

译 者 序

从 1995 年诞生至今，JavaScript 已经走过了 18 个年头，很难想象，当初 Brendan Eich 在 10 天内写出来的语言竟然会成为如今最热门的语言之一。

这些年，JavaScript 一直被当做 Web 浏览器端脚本使用，但在这个过程中，ECMA 标准也在不断地演变，不断地加入新的特性；同时，各个厂商的 JavaScript 实现（特别是 Chrome 的 V8）也在不断地提升性能和稳定性，这为后来 Node.js 的流行埋下了伏笔。

2009 年，Node.js 出现在大家的视野中，它在诞生伊始就获得了众多开发者的关注。由于使用了性能优越的 Chrome 浏览器 V8 引擎和全新的事件驱动、异步编程模式，它从一开始就显得“与众不同”。Node.js 是 Ryan Dahl 发起的开源项目，这些新特性给它带来巨大的性能提升，并且大量的第三方模块也让其开发效率得到了极大提高。如今，诸多大型互联网公司在其产品中广泛使用 Node.js，如 LinkedIn、淘宝等；同时大部分云计算平台也支持部署 Node.js 应用。由此足以看出，Node.js 是今后的发展趋势之一。

本书一共分为四大部分，由易入难，循序渐进，既适合 Node.js 新手入门，也能让使用过 Node.js 的开发者在阅读过程中收获“惊喜”。如果你是一名初学者，可以跟着书中的示例一步步成长；如果你是一名有经验的开发者，也可以从后面的章节中获取灵感。本书和其他介绍 Node.js 的书籍的最大不同在于，书中会包含大量的后端知识，如 shell 脚本等，这会让读者（特别是 Web 前端开发者）脱离原先的思维桎梏，从后端开发的角度来学习这个全新的开发平台。

本书由姚立、彭森材合作翻译完成。其中，彭森材翻译第 2 章、第 4 ~ 6 章、第 9 章和第 10 章；姚立翻译其余部分。在翻译本书的过程中得到了华章公司责任编辑关敏老师的帮助，她多方协调并给出中肯的建议，让译者收获颇丰，在此表示衷心感谢。

由于时间仓促，加之译者水平有限，书中难免会有疏漏之处，希望读者能够提供反馈，不胜感激。

最后，我们要感谢家人，感谢他们的理解和支持，感谢他们在本书翻译过程中所做的一切。一条毛毯、一杯热茶，都是我们翻译的动力和源泉。感激、感恩！

前　言

欢迎来到 Node.js 的世界。Node.js 是专为编写网络和 Web 应用而生的全新平台，在过去的几年中，迅速积累了大量的人气并在开发社区拥有一大批拥趸。在本书中，我会介绍更多关于 Node.js 的知识，告诉你为什么它会如此迷人。我会教你如何快速上手并编写 Node.js 程序。很快，你会发现 Node.js 名字的叫法非常灵活，它经常被叫做 Node，甚至被称为“ node”。本书中我也会经常这么称呼。

为什么使用 Node.js

Node.js 的兴起有两个主要原因，我会在下面加以解释。

万维网

在过去，编写 Web 应用是一个相当标准化的流程。你有一个或多个服务器部署在机器上并监听某个端口（如 HTTP 的 80 端口），每当服务器接收到一个请求时，它会创建一个进程或者线程去处理和响应请求。处理这些请求会频繁地与外部服务进行通信，比如数据库、内存缓存、外部计算服务器或者仅仅是文件系统。当所有的工作最终结束之后，线程或者进程会返回到“可用”服务器池中，这样服务器就可以处理更多的请求了。

这是一个合理的线性过程，容易理解且极易编码。但是，有两个缺点一直困扰着这种模型：

1. 每一个线程或进程都会消耗一些资源。在一些机器中，对于使用 PHP 和 Apache 构建的应用，每个进程甚至会消耗高达 10 ~ 15MB 的内存。特别是在大型服务器不断运行并创建线程来处理请求的时候，每一个线程都会消耗一些资源来创建新的堆栈和执行环境。很快，服务器就会陷入没有内存可用的境地。

2. 在最常见的使用场景下，Web 服务器会与数据库、缓存服务器、外部服务器或文件系统通信，这会消耗大量的等待时间，直到这些服务处理完并返回响应。当服务器等待响应的时候，当前线程就会被“阻塞”，无法继续执行下去。因此，消耗的服务器资源和当前服务器进程或线程会被完全冻结，直到返回响应相关资源才会得到释放。

只有当外部组件返回响应之后，进程或线程才会继续完成处理，并把结果返回给客户端，然后重置并等待处理下一个请求。

因此，尽管这种模型非常容易理解和编码，但是一旦脚本花费大量的时间来等待数据库服务器结束查询，这种模型就略显低效——而这却是现代 Web 应用极其常见的应用场景。

当然，这种问题已经有了很多通用的解决方案。我们可以购买更强大、拥有更多内存的 Web 服务器；可以使用更小的轻量级服务器如 lighttpd 或者 nginx 来替代 Apache 这样功能强大的 HTTP 服务器；可以定制或者精简你喜欢的编程语言版本，比如 PHP 或 Python（事实上，Facebook 已经先行一步，并且开发出将 PHP 转换成原生 C++ 代码的系统，以达到最大执行速度和最优代码规模的目的）；甚至，可以通过购置更多服务器的方式来提高所能容纳的并发连接数。

前沿技术

多年来，Web 开发者一直在为优化服务器资源和提高并发数而努力；而这期间，其他一些有趣的事情已经悄然发生。

JavaScript，这门在 Web 浏览器中以客户端脚本语言而闻名于世（同时也经常遭人诟病）的古老（1995 年左右面世）语言，已经重新焕发生机。诸多现代 Web 浏览器重构了 JavaScript 的实现，并添加了许多新的特性，让它变得更加强大和稳定。随着诸多浏览器客户端类库的出现，如 jQuery、script.aculo.us、Prototype 等，JavaScript 编程变得有趣且富有成效。这些类库封装了原先臃肿的 API，并添加了很多有趣、动态的效果。

同时，各大浏览器群雄割据的时代业已到来，谷歌的 Chrome、Mozilla 的 Firefox、苹果的 Safari 和微软的 IE 浏览器一起争夺浏览器之王的桂冠。作为其中的一部分，这些公司加大了在 JavaScript 上的研发比重，从而让 Web 应用可以更可靠地依赖脚本，并且拥有更强大的动态效果。特别是谷歌公司的 Chrome 浏览器使用的 V8 JavaScript 引擎，性能尤为卓越，并且是开源软件，可供任何人使用。

天时地利人和，JavaScript 的机遇随着开发人员的全新网络（Web）应用开发方式而出现。这，就是 Node.js 的新生。

Node.js 的前世今生

2009 年，一位名叫 Ryan Dahl 的研究员（后来他加入 Joyent，该公司是一家坐落于加利福尼亚州的云计算和可视化服务公司）一直在寻求开发一种专为 Web 应用提供类似于 Gmail 推送服务的功能，但是一直没有找到一个完全合适的解决方案。最后，他把目光落到了 JavaScript 上，因为这门语言缺乏健壮的输入 / 输出（I/O）模型（这也就意味着他可以自己写一个全新的 I/O 系统），同时还有性能优越的 V8 引擎可供编程使用。

受 Ruby 和 Python 社区中类似项目的启发，他最终选择了 Chrome 的 V8 引擎和一个叫做 libev

的事件处理类库，并且很快推出了第一版的 Node.js 系统。Node.js 的主要理论和创新就是它完全构建在事件驱动、非阻塞的编程模型之上。简而言之，你完全不会（或者极少）编写线程阻塞的代码。

一个 Web 应用——为了处理请求并返回响应——一般需要执行数据库查询。Web 应用处理请求，并会在返回响应之后告诉 Node.js 如何处理。同时，应用代码也可以开始处理其他传入的请求，实际上，它可以处理任何需要处理的任务，比如清理系统数据或分析数据。

通过这种应用处理请求和工作方式的改变，可以简单地编写能够同时处理几百甚至几千个请求的 Web 服务器，同时也不用消耗太多的资源。Node 是在单进程上运行的，并且 Node 代码一般也是单线程执行，因此，相较于其他平台而言，Node 对资源的需求就小得多。

但是，这种执行速度和能力是有一些瑕疵的。因此，必须充分认识到它们，然后就可以开始小心翼翼地使用 Node 进行工作了。

首先，这种全新的模型机制不同于以往你所接触过的模型，因此你可能会感到困惑。在充分理解这些核心概念之前，你可能会对这些 Node.js 代码感到奇怪。本书中的大部分篇幅会讨论编程人员在迎接 Node 异步、非阻塞编程挑战时所使用的核心模式，并指导你如何开发自己的 Node 程序。

这种编程模型的另一个限制是：它真的是在为处理大量不同任务的应用程序服务，会同很多不同的进程、服务器或服务通信。当 Web 应用需要连接到数据库、缓存服务器、文件系统、应用服务器或其他服务时，Node.js 便会大放异彩。但是另一方面，实际上它并不是那些需要做长时间精密计算的服务器的最佳运行环境。因此，单进程、单线程的 Node 模型在处理一个给定的请求时，如果该请求需要花费大量的时间生成一个复杂的密码摘要（password digest）或处理图像，就会带来问题。在那些需要更多的计算密集型工作的情况下，我们必须小心谨慎地处理应用程序使用资源的情况，甚至可以考虑把这些任务移植到其他平台，并把它们作为第三方服务供 Node.js 程序调用。

最后，Node.js 是一个值得信赖的全新平台，并且正在积极发展中。它还没有（截至 2013 年 2 月）发布 1.0 版本，同时 Node.js 的版本更新非常频繁，甚至到了让人眼花缭乱的地步。

为了减少这些频繁的更新所引起的不确定性和烦恼，开发者已经使用标签来标识系统各个部分的不同稳定级别，从不稳定（Unstable）到稳定（Stable）再到锁定（Locked）。系统中稳定或锁定的部分几乎不会改动；如需改动，社区会通过大量的讨论以确定是否会产生太多问题。在你阅读本书的过程中，我们会指出哪些地方不太稳定，并提供相应的解决方法，从而减少因 API 改变而带来的危险。

好消息是，Node.js 已经有一个庞大而活跃的用户社区和一堆邮件列表、论坛及用户组，它们致力于促进平台的发展并提供力所能及的帮助。通过简单的谷歌搜索，就可以在几秒钟之内回答你 99%的问题，所以，请不要害怕 Node.js！

本书读者

本书的读者要喜欢编程，并且要熟悉至少一门主流编程语言的功能和语法，比如 Java、C/C++、PHP 或者 C# 等。读完本书之后，你不必成为一名专家，但你一定要高于“Y 天精通 X 语言”的水平。

如果像我一样，你可能已经写过一些 HTML/CSS/JavaScript 代码，从而和 JavaScript“打过交道”。但是你可能并不是非常熟悉 JavaScript，只是单纯地从博客或者邮件列表中把代码复制出来。事实上，由于笨重的用户界面和令人沮丧的浏览器不匹配问题，你可能在一开始就对 JavaScript 没有好感。但是不用害怕——在本书第一部分结束之后，你对这门语言的糟糕印象从此会变成过去时。我希望你能放松心情，面露微笑，轻松愉快地编写你的第一个 Node.js 程序。

你还需要了解 Web 应用的基本工作原理：浏览器向远程服务器发送 HTTP 请求；服务器处理请求并返回表明成功或者失败的标识，返回对象中有可能会附带一些数据（比如渲染页面用的 HTML 代码或者请求返回的 JSON 数据）。在过去，你可能会在连接数据库服务器并发送查询请求后，等待返回数据集等。而在使用示例和程序来描述一些全新的概念的时候，我会详细讲解并帮助大家理解这些全新或者生僻的概念。

如何使用本书

本书实质上就是一本新手教程。我会尽量使用代码去解释原理，避免使用冗长的篇幅和晦涩的语言。因为我觉得好的解释说明应该是生动有趣的，如果你感兴趣的话，我还会告诉你一些资源或者一些其他的文档（当然这些并不是必需的）。

本书一共分为四大部分。

第一部分　基础篇——首先，你会学习如何安装并运行 Node；然后，会进一步了解 JavaScript 这门语言以及在 V8 和 Node.js 中使用的扩展；最后，会编写你的第一个 Node.js 应用。

第二部分　提高篇——在本篇中，你会开始学习开发更强大且有趣的应用服务器。同时，我会教你编写 Node.js 程序时会用到的一些核心概念和最佳实践。

第三部分　实战篇——在本篇中，你会看到一些强大的工具和模块，利用它们，你可以编写自己的 Web 应用，比如 Web 服务器的中间件及与数据库服务器的通信。

第四部分　进阶篇——最后，我会以一些高级特性的介绍结束本书，比如如何在生产环境中运行应用，如何测试代码以及如何使用 Node.js 编写命令行实用程序。

在阅读本书的过程中，最好花些时间打开编辑器，输入这些代码，然后看看这些代码在你当前的 Node.js 版本下是如何运行的。或者是在看书的时候，编写并开发你自己的小程序。在阅读本书的过程中，你会开发一个小小的相册应用，希望能给你编写其他应用提供一些灵感和想法。

下载源代码

本书示例的源代码可以在 github.com/marcwan/LearningNodeJS 上找到。希望你能下载代码并运行一遍。如果条件允许，不要放弃亲手把代码敲出来的机会，可以多尝试尝试。

GitHub 上托管了功能齐全的示例代码，并且已经使用最新版的 Node.js，在 Mac、Linux 和 Windows 下测试过。如果最新版的 Node 需要更新源代码，我会在 GitHub 上及时更新并做出注释，所以请确保每隔几个月就获取一次最新的代码。

如果你对本书中的示例代码有任何疑问，请在 github.com/marcwan/LearningNodeJS 上提交 issue。我们会看到这些 issue，并且会及时做出合理的解答。

致谢

我要感谢 PHPTR 各位朋友给予的支持和建议，是你们的帮助让本书和其他项目付诸实践。感谢本书文字编辑的杰出贡献。

非常感谢 Bill Glover 和 Idriss Juhoor，是你们出色的技术和样式审校让本书愈加精彩。

最后，感谢至爱 Tina，感谢一路有你。

目　录

第三部分 实战篇

第四部分 进阶篇

第一部分

基 础 篇