

JavaScript Frameworks
for Modern Web Dev



Apress®

JavaScript 开发框架权威指南

[美] Tim Ambler Nicholas Cloud 著
一心一译前端小组 译

• 现代Web开发不可或缺的18种JavaScript库和框架的详尽指南



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

JavaScript Frameworks
for Modern Web Dev

JavaScript 开发框架权威指南



[美] Tim Ambler Nicholas Cloud 著
一心一译前端小组 译

人民邮电出版社
北京

图书在版编目 (C I P) 数据

JavaScript开发框架权威指南 / (美) 安布勒 (Tim Ambler), (美) 克劳德 (Nicholas Cloud) 著; 一心一译前端小组译. — 北京: 人民邮电出版社, 2017. 5

ISBN 978-7-115-44719-7

I. ①J… II. ①安… ②克… ③一… III. ①JAVA语言—程序设计—指南 IV. ①TP312.8-62

中国版本图书馆CIP数据核字(2017)第048275号

版权声明

JavaScript Frameworks for Modern Web Development
by Tim Ambler and Nicholas Cloud ISBN: 978-1-4842-0662-1

Original English language edition published by Apress Media.

Copyright © 2015 by Apress Media.

Simplified Chinese-language edition copyright ©2017 by Post & Telecom Press

All rights reserved.

本书中文简体字版由 Apress L.P.授权人民邮电出版社独家出版。未经出版者书面许可,不得以任何方式复制或抄袭本书内容。

版权所有,侵权必究。

-
- ◆ 著 [美] Tim Ambler Nicholas Cloud
 - 译 一心一译前端小组
 - 责任编辑 陈冀康
 - 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>
 - 三河市海波印务有限公司印刷
 - ◆ 开本: 800×1000 1/16
 - 印张: 25.25
 - 字数: 605千字 2017年5月第1版
 - 印数: 1-2500册 2017年5月河北第1次印刷
 - 著作权合同登记号 图字: 01-2016-0777号

定价: 89.00元

读者服务热线: (010)81055410 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京东工商广字第8052号

内容提要

JavaScript 是现代 Web 开发必不可少的编程语言，但 JavaScript 的生态系统包括库、框架以及工具都在快速地发展且日益庞大。程序员学习的需求和面临的挑战也相应地增加。

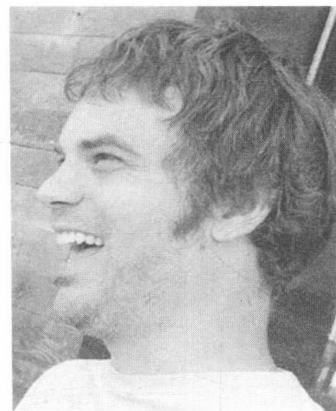
本书涵盖了在开发过程中常用的各种 JavaScript 工具，以帮助读者在大量流行的 JavaScript 工具中做选择。全书分为 16 章，从开发工具、模块加载器、客户端框架、服务端框架数据库交互、通信、管理控制流和其他有用框架等几个方面，涵盖了 Bower、Grunt、Yeoman、PM2、RequireJS、Browserify、Knockout、AngularJS、Kraken、Mach、Mongoose、Knex、Bookshelf、Faye、Q、Async.js、Underscore 和 Lodash 等框架和库。全书涵盖了客户端和服务端端的开发，通过细致的讲解、详细的代码示例，阐明了这些工具的用法。

本书主要面向熟悉 JavaScript 但对数量庞大的解决方案感到困惑的开发者，对于想要学习 JavaScript 并掌握各种相关工具的读者来说，具有很好的参考价值。

作者简介



Tim Ambler 是来自美国田纳西州那什维尔的一名软件工程师。他对编程的热情来自于父亲。在他小的时候，他的父亲就向他介绍了 Commodore 64 电脑。Tim 是几个流行的开源项目的作者，其中 whenLive 已经被 GitHub 员工采用。作为会议演说家和多产的作家，Tim 多次被在线出版物推荐，如 JavaScript Weekly 和 Node Weekly。Tim 目前与他的妻子和 2 只猫生活在南部。读者可以在 Twitter 上 @tkambler 关注他。



Nicholas Cloud 是一名软件工程师，居住在非常潮湿的城市圣路易斯。过去十几年里，他利用自己的技能成就一番成功事业。通过 JavaScript、C#和 PHP，他开发了大量适用于多终端的 Web 应用、Web 服务、桌面应用。Nicholas 是开源软件的有力支持者，致力于用户级项目开发，并写了几个自己的开源库。在业余时间，他在不同的用户组发言、参加会议、写书、写技术文章、写博客。他的 Twitter 是 @nicholascloud。

技术评阅者简介



Robin Hawkes 致力于学习并热衷于结合设计与编码来解决问题。他是 *Foundation HTML 5 Canvas* (Apress, 2011) 一书的作者，这本书是关于使用 JavaScript 开发游戏的。他也是 ViziCities one-man 乐队的成员，ViziCities 是一个 WebGLS-3D 城市虚拟平台。过去 Robin 一直致力于 Mozilla 和 Pusher 的全球开发者关系。

译者简介

一心一译前端小组由一群热爱前端和 JavaScript 的技术人员组成，他们因兴趣而走到一起，致力于为读者奉献高品质的翻译技术图书。这个前端小组包括以下成员：

郭凯

美团点评酒旅事业群前端团队负责人，高级技术专家，资深互联网人，全栈工程师，工作狂，崇尚工匠精神，曾就职于音悦台、淘宝旅行。翻译作品有《编写可维护的 JavaScript》《第三方 JavaScript 编程》，有 In、Juicer、jQuery、F2E.im、PM25 等开源项目，业余时间负责开源前端技术社区 F2E 的开发和维护。

王思可

热爱前端开发，热爱技术。曾在阿里旅行工作，现在供职于灵雀云，希望能够更好地用技术服务于人。

朱佳慧

美团点评酒旅事业群前端开发工程师，2015 年加入美团。

赵荣娇

花名银翘，前端开发工程师，也是《响应式设计、改造与优化》一书的译者。著有《超实用的 CSS 代码段》《代码逆袭：超实用的 HTML 代码段》等图书。喜欢旅行，热爱前端开发。正在学习如何成为一名胜过好老师的妈妈。

马荃

美团点评酒旅事业群高级前端工程师，曾就职于创新工场和阿里巴巴，擅长前端工具自动化，关注前端体验和效率，乐于分享。

巩守强

猫眼基础交易负责人，美团前端高级技术专家，对于前端工程化、前端性能优化和客户端动态化感兴趣。

蔡平

曾在美团酒旅前端和 360 奇舞团工作，现任海致前端开发 Leader。喜欢前端技术，关注前端最新动态，追求高效的前端工程化解决方案。

康明轩

北京师范大学硕士，对世界充满好奇的孩子，迷恋于技术不能自拔的瘾士，掉在麻袋里的书虫。走一条从修地球到修电脑的非典型路线，现就职于北京指南针科技发展股份有限公司，从事网络应用开发，认为程序员也是一门可以长相厮守的手艺。

前 言

人们说我们生活在一个信息时代，但似乎没有一条是我所需要的或想了解的信息。事实上，（我越来越相信）这一切电子产品只会增加我们的困惑，发表内部独家新闻、裁决几乎还没开始的事件：这喋喋不休的洪流以光速移动，以至于几乎不可能听到重要的事。

——马修·弗拉曼 《The Kingdom of Ohio》

“技术发展迅速”是一条老生常谈的格言，而且有很好的理由：技术的确发展迅速。但此时，JavaScript 的确发展得特别迅速——就像马修·弗拉曼在俄亥俄王国所说的“喋喋不休的洪流以光速移动”。随着基于浏览器应用迅速增长的复杂度以及服务器端 JavaScript 的日益普及，这门语言正在经历被许多人所称谓的复兴之中。一切多亏了 Node.js。

JavaScript 社区中正在迈开近乎狂热的创新步伐，虽然充满了无穷的魅力，但也提出了自己的独特的挑战。JavaScript 的生态系统包括库、框架以及工具，都在剧烈地成长。过去针对任何给定问题可能只有少量的解决方案，而今已经有许多解决方案可以选择，并且其数目日益增长。因此，开发人员要面对这样的艰难任务：在很多看起来不错的方案中选择合适工具。

如果你像我们一样好奇为什么最近 JavaScript 似乎吸引了如此多的注意力，那么值得停下来思考 JavaScript 的本质。这门语言由一个人在十天内创造，现在却作为许多我们知道 Web 站点的基础服务。一门原本设计用于解决相当简单问题的语言，现在却以最初没有预见到的创新方式所应用。更重要的是，JavaScript 是一门优美的表达语言，但它不是没有棱角和潜在的陷阱。尽管它灵活、高效以及无处不在，但是对于 JavaScript 的初学者来说，了解 JavaScript 的一些概念如事件循环、原型继承等，是特别具有挑战性的。

由于这些和很多其他原因，开发社区的很大一部分人都在探究如何最好地应用 JavaScript 的独特特性。毫无疑问，我们只是抓住了语言的表面和其背后社区提供的能力。对于那些对知识有着贪婪的需求和充满创造欲的人，现在是成为一名 JavaScript 开发者的最佳时机。

我们编写了本书来指导你在大量流行的 JavaScript 工具中做选择，这些工具解决开发技术栈的两端：浏览器和服务端。教程及本书中可下载的代码示例阐明了这些工具的用法，包括依赖管理、模块化代码模式、自动化重复任务构建、创建专业的服务、客户端应用的架构、灵活的水平扩展、执行事件日志记录和与不同的数据存储交互。

本书涵盖的库和框架包括 Bower、Grunt、Yeoman、PM2、RequireJS、Browserify、Knockout、AngularJS、Kraken、Mach、Mongoose、Knex、Bookshelf、Faye、Q、Async.js、Underscore 和 Lodash。

在写这本书时，我们的目标是对围绕 JavaScript 的“喋喋不休的洪流”做筛选。这样做能让我们相信，的确重要的事情需要被听到。我们希望这些文字能像对我们一样，也对您有所帮助。

本书读者对象

本书主要面向能胜任 JavaScript 的，但又对庞大数量、似乎能解决所有问题的方案感到困惑的开发者。本书帮你驱散迷雾，为你提供一个深入的指导，引导你学习知名组织现在正在使用并取得了很大成功的库和框架。本书话题涵盖客户端和服务端开发。如果你至少对浏览器文档对象模型 (DOM)、常用的客户端库 (类似 jQuery 和 Node.js) 有所熟悉，那么你会从本书中获益。

本书模式

本书涵盖了在整个开发过程中能用到的 JavaScript 工具，从项目的第一次代码提交到发布甚至除此以外的问题，为你提供了多种选择。为此，本书的章节可分为以下部分。

第 1 部分：开发工具

Bower

依赖管理已经不是一个新想法了——著名的例子包括 Node 的 npm、Python 的 pip 和 PHP 的 composer。然而有一种实践方案直到最近才被广泛采用，这个概念是管理 Web 应用前端资源——JavaScript 库、样式表、字体、图标和图像，以此作为构建现代 Web 应用的砖瓦。在这一章，你会学习几种管理方式。此领域的一个流行工具 Bower，可以为你提供一个机制来组织应用程序中的依赖项，改进你的开发过程。

Grunt

Perl 语言的创造者拉里·沃尔，描述了伟大程序员的三种特质：懒惰，缺乏耐心，以及傲慢。这一章中，会集中讨论帮你加强这三种特质中“懒惰”的工具——Grunt。这个流行的任务处理器为开发者提供了创建命令行工具的框架。它能自动构建重复任务，例如运行测试脚本、拼接文件、编译 SASS/LESS 样式表、检查 JavaScript 代码错误等。阅读完这一章，你会了解如何使用几个流行的 Grunt 插件，以及如何创建和在社区中分享自己的插件。

Yeoman

Yeoman 为开发者提供了创建可重用模板 (“生成器”) 的机制，通过其模板描述项目文件模式 (项目初始化所需要的依赖，Grunt 任务等)，这样一来就能轻易做到反复重用。社区的广泛支持让你可以利用大量现存的模板。这一章中，会详细讨论安装 Yeoman 以及如何使用一些现存的生成器。随后，会讲解怎样创建和在社区中分享自己的模板。

PM2

本章中，会通过学习 PM2 来结束关于开发工具议题的讨论。PM2 是一个命令行工具，它简化了很多与运行 Node 应用相关的任务，包括监视应用状态以及为满足增长的需求而高效地伸缩应用。

第 2 部分：模块加载器

RequireJS 和 Browserify

JavaScript 对于在浏览器中加载外部依赖先天缺乏的特点，令开发者很沮丧。好在社区使用两种

不同和相互竞争的标准填补了这一空缺：异步模块加载（AMD）API 和 CommonJS。我们会深入探索两者的细节并看一看两者广泛使用的实现方式：RequireJS 和 Browserify。每一个都有其优点，我们都会详细探讨，但两者都对构建应用程序有深刻的积极影响。

第 3 部分：客户端框架

Knockout 和 AngularJS

近年来，Web 开发人员见证了所谓的“单页面应用程序的迅速流行”。具有这种行为的应用曾经只在桌面上可用，在浏览器中增加了代码的复杂度。这一节中，深入介绍两种广泛使用的前端框架：Knockout 和 AngularJS。它们提供了已经被证明的模式，能帮你解决频繁遇到的问题，减少代码的复杂性。Knockout 聚焦于解决视图和数据之间的关系，但另一方面也让开发者自由判定应用程序架构。AngularJS 则提供了一个更规范的方法，涵盖了视图、应用路由、数据传输和模块设计。

第 4 部分：服务端框架

Kraken 和 Mach

在没有服务端交互的情况下，客户端应用程序不会很有用。在本部分中，看一看两个流行的框架：Kraken 和 Mach。它们支持开发人员创建后端应用程序。Mach 不仅仅是一个简单的 Web 服务器，它是 Web 的 HTTP 层。Mach 既能够通过一个直观的、可扩展的 HTTP 栈来提供内容，又能获取内容。Mach 的接口包含了 Node.js 应用中 Web 页面请求的服务，也包括在浏览器中用 Mach 的 AJAX 请求获取 JSON 数据，甚至包括向其他网络堆栈请求的完全重写及代理。在许多方面，可以说 Mach 是 HTTP 层的瑞士军刀。

第 5 部分：管理数据库交互

Mongoose、Knex 和 Bookshelf

每个应用程序的核心都存在着任何开发栈中最重要的组成部分——用户搜索的数据。在本部分中，熟悉两个库，它们帮助你简化与流行的存储平台（如 MongoDB、MySQL、PostgreSQL 和 SQLite）交互的一些复杂性。读完这一节，你就能轻松地定义模式、关联、生命周期“钩子”等。

第 6 部分：通信

Faye

本部分中，介绍 Node.js 库 Faye。它为开发人员提供了一个健壮的、易于使用的平台，依靠服务器和所有主流浏览器之间的实时通信来构建产品。Faye 之所以流行，源于 Web 项目有着希望在各处运行的目标。Faye 通过提供无缝的回调支持多种通信协议来完成这个目标。

第 7 部分：管理控制流

Q 和 Async.js

JavaScript 异步的特性给开发者很大的灵活度——与强迫开发者以线性方式执行代码相反，

JavaScript 允许开发者同时协调多个行为。不幸的是，伴随着这种灵活性的是额外相当程度的复杂性——也就是许多开发人员所说的“地狱回调”或“噩梦金字塔”。在本部分中，考察两种流行的库：Q 和 Async.js，它们将帮助你解决异步控制流的复杂性问题。

第 8 部分：其他有用的库

对一些其他非常有用的库，本书将忽略，不做涉及。但有些额外的库并不是不值一提。这一部分将介绍这些库。

Underscore 和 Lodash

Underscore（以及它的继承者 Lodash）是一个相当有用的函数集合，能简化很多频繁使用但实现起来冗长的模式。本章简要介绍了这些库，还提到了一些更流行的扩展，它们能用来进一步提高其效用。示例代码强调了这些库中使用最频繁的一部分。

源代码下载

本书的每个章节都包含很多示例，源代码的 zip 版本可以从 <http://www.apress.com/9781484206638> 或 www.epubit.com.cn 下载。zip 文件的子目录包含的源码对应每一章节中的示例。每章示例中源码的第一行注释都标识了源代码所在的文件路径。例如，如果你读到了第 10 章（Mach）的清单 10-1，实际的源码文件放在 `mach/example-000/no-such-file.js` 路径下，相对于提取后的 `mach.zip` 文件路径。

清单 10-1 一个假想的例子

```
// example-000/no-such-file.js
console.log('this is not a real example');
```

大部分示例在 Node.js 环境运行，Node.js 可以从 <https://nodejs.org> 下载。对于如何下载和安装每章中的示例，相应章节都会提前做相关知识解释。（例如，阐述 Mongoose 的第 11 章中必须运行 MongoDB）

任何关于运行示例代码的额外步骤（如执行 curl 请求）或与一个正在运行中的示例进行交互（如打开 Web 浏览器并导航到指定 URL）都会在清单旁做出解释。

致谢

本书的诞生和你们的鼓励与支持息息相关：

Nicholas Cloud，我的朋友及合作者。正是因为他，这本书才能够保持深度和广度。他的学识、经验以及坚定致力于项目的精神带来了不可度量的帮助。在此表示感谢。

Louise Corrigan、Kevin Walter、Christine Ricketts、Melissa Maldonado 和其他 Apress 的员工，在这本书的写作过程中给予我们支持。很感谢他们邀请我来创作并持续提供支持。

技术评审 Robin Hawkes，本书的示例和源代码得益于他们的洞察力和犀利的眼光。

Faye 的作者 James Coglan，谢谢你分享专业技术知识和反馈。

我的朋友及同事 Greg Jones、Jeff Crump、Seth Steele、Jon Zumbrun 和 Brian Hiatt，非常感谢你们的反馈和鼓励。

——Tim

感谢是很无力的东西，还有很多感激未能言表：

首先，感谢我的合作者 Tim，感谢他邀请我参与本书的创作。我们从未谋面，半年以来，我们通过远程合作办公，有足够的时间给彼此留下深刻的印象。非常感谢他对我的信任、鼓励和持续的努力。

其次，感谢 Apress 的员工：Kevin、Louise、Christine 和 Melissa，他们指导我整个出版的过程。毫不夸张地说，他们的耐心和细致引导，使我在整个写作过程中脚踏实地。他们都是顶尖的专业人士，期待有一天能与之共事。

再次，我感谢非常优秀的技术评审 Robin，比开发人员更加负责地阅读并执行代码。

最后，我不能偿还当研究 Mach、Knockout 这些主题知识时从 Michael Jackson (@mjackson) 和 Ryan Niemeyer (@RPNiemeyer) 那里所得到的，我只能希望读者把这份爱继续传递下去。

——Nicholas

目 录

第 1 章 Bower	1	2.4.6 多任务选项	17
1.1 准备工作	1	2.4.7 模板配置	18
1.2 配置 Bower	2	2.4.8 命令行选项	19
1.3 清单文件 (Manifest)	2	2.4.9 提供反馈	19
1.4 查找、添加和删除 Bower 包	3	2.4.10 错误处理	20
1.4.1 查找包	3	2.5 操作文件系统	20
1.4.2 添加包	3	2.5.1 源-目标映射	20
1.4.3 删除包	5	2.5.2 监视文件变化	22
1.5 语义化版本控制	5	2.6 创建 Grunt 插件	25
1.6 维护依赖链	6	2.6.1 开始	25
1.7 创建 Bower 包	7	2.6.2 创建任务	26
1.7.1 选择有效的包名	7	2.6.3 将任务发布到 npm	28
1.7.2 在 Git 标签中使用语义化版本号 (Semver)	7	2.7 小结	28
1.7.3 将软件包发布到注册中心	7	2.8 相关资源	29
1.8 小结	8	第 3 章 Yeoman	30
第 2 章 Grunt	9	3.1 安装 Yeoman	30
2.1 安装 Grunt	10	3.2 创建第一个项目	30
2.2 Grunt 是如何工作的	10	3.3 创建你的第一个脚手架	34
2.2.1 Gruntfile.js	10	3.3.1 Yeoman 脚手架是一个 Node 模块	34
2.2.2 任务 (Tasks)	11	3.3.2 子脚手架	35
2.2.3 插件 (Plugins)	11	3.3.3 定义二级命令	39
2.2.4 配置	12	3.3.4 可组合性	41
2.3 将 Grunt 添加到项目中	12	3.4 小结	41
2.4 处理任务	14	3.5 相关资源	42
2.4.1 配置管理	14	第 4 章 PM2	43
2.4.2 任务描述	15	4.1 安装	43
2.4.3 异步任务	15	4.2 与进程一起工作	43
2.4.4 任务依赖	16	4.2.1 从错误中恢复	46
2.4.5 多任务	16	4.2.2 监控文件变化	47

4.3	监控日志	48	6.9	用 Transforms 扩展 Browserify	97
4.4	监控资源占用	49	6.9.1	brfs	97
4.4.1	监控本地资源	49	6.9.2	folderify	98
4.4.2	监控远程资源	49	6.9.3	bulkify	98
4.5	进程的高级管理	52	6.9.4	Browserify-Shim	99
4.6	多核处理器的负载均衡	55	6.10	小结	100
4.7	小结	59	6.11	相关资源	100
4.8	相关资源	59			
第 5 章	RequireJS	60	第 7 章	Knockout	101
5.1	运行示例	61	7.1	View、Model 与 View Model	102
5.2	使用 RequireJS	61	7.1.1	菜谱列表	103
5.2.1	安装	62	7.1.2	菜谱详情	106
5.2.2	配置	62	7.2	将视图绑定到 DOM	108
5.2.3	应用模块和依赖	64	7.3	视图模型与表单	109
5.2.4	路径和别名	66	7.3.1	切换到“编辑”模式	109
5.2.5	Shims	69	7.3.2	更改菜谱的标题	112
5.2.6	加载器插件	73	7.3.3	更改菜谱的份量与 烹饪时间	112
5.2.7	缓存清除	78	7.3.4	添加与删除食材	114
5.3	RequireJS 优化	80	7.3.5	操作步骤	118
5.3.1	配置 r.js	80	7.3.6	引文	119
5.3.2	运行 r.js 命令	81	7.4	自定义组件	120
5.4	小结	82	7.4.1	input-list 组件的视图模型	120
第 6 章	Browserify	84	7.4.2	input-list 模板	121
6.1	AMD API 与 CommonJS 对比	84	7.4.3	注册 input-list 组件	123
6.2	安装 Browserify	85	7.5	Subscribable: 简单的消息传递	124
6.3	创建你的第一个 Bundle	85	7.6	小结	126
6.4	可视化依赖树	87	7.7	相关资源	127
6.5	发生变化时重新打包文件	88	第 8 章	AngularJS	128
6.5.1	通过 Grunt 监听文件变化	88	8.1	声明式 Web 编程	128
6.5.2	通过 Watchify 监听文件 变化	88	8.1.1	命令式编程	128
6.6	使用多个打包文件	90	8.1.2	声明式编程	129
6.7	Node 方式	92	8.2	模块: 构建松散耦合程序的基石	130
6.7.1	模块解析方案和 NODE_PATH 环境变量	93	8.3	指令 (Directive): DOM 的抽 象层	132
6.7.2	依赖管理	95	8.4	加入逻辑	134
6.8	定义浏览器指定模块	96	8.4.1	作用域与原型继承	134
			8.4.2	用控制器操作作用域	135

8.5 通过服务与依赖注入 实现松散耦合.....	138	10.2 安装.....	203
8.5.1 依赖注入 (Dependency Injection)	138	10.3 Mach Web 服务	203
8.5.2 简单的控制器与复杂的 服务	139	10.3.1 HTTP 路由	205
8.6 创建路由.....	142	10.3.2 建立连接.....	210
8.6.1 路由参数.....	143	10.3.3 公共的中间件.....	212
8.6.2 路由的 Resolve.....	144	10.3.4 路由重写.....	226
8.7 创建复杂表单.....	145	10.3.5 主机映射.....	228
8.7.1 表单验证.....	146	10.3.6 自定义中间件.....	232
8.7.2 条件逻辑.....	150	10.4 Mach HTTP 客户端.....	234
8.7.3 列表.....	151	10.5 Mach HTTP 代理.....	236
8.8 小结.....	153	10.6 小结.....	239
8.9 相关资源.....	154	第 11 章 Mongoose	240
第 9 章 Kraken	155	11.1 MongoDB 的基本概念.....	240
9.1 环境感知的配置.....	156	11.2 Mongoose 的一个简单示例.....	243
9.2 注册基于配置的中间件.....	162	11.2.1 针对 JSON 数据创建一个 Mongoose 模式.....	243
9.3 结构化路由注册.....	165	11.2.2 使用 Mongoose 导入 数据.....	244
9.3.1 索引配置.....	165	11.2.3 通过 Mongoose 查询 数据.....	247
9.3.2 目录配置.....	166	11.3 使用模式 (Schemas)	248
9.3.3 路由配置.....	167	11.3.1 数据类型.....	248
9.4 Dust 模板.....	169	11.3.2 嵌套模式.....	250
9.4.1 上下文及引用.....	169	11.3.3 默认属性值.....	250
9.4.2 片段.....	171	11.3.4 必要属性.....	251
9.4.3 迭代.....	172	11.3.5 第二索引.....	251
9.4.4 条件句.....	173	11.3.6 模式校验.....	252
9.4.5 局部模板.....	173	11.3.7 模式引用.....	255
9.4.6 块.....	174	11.3.8 模式中间件.....	258
9.4.7 过滤器.....	175	11.4 使用模型和文档.....	259
9.4.8 上下文辅助器.....	176	11.4.1 文档实例方法.....	262
9.4.9 Dust 辅助器.....	182	11.4.2 文档虚拟属性.....	263
9.4.10 使用 Kraken.....	186	11.4.3 静态模型方法.....	265
9.5 小结.....	200	11.5 使用查询.....	266
9.6 相关资源.....	200	11.5.1 Model.find().....	266
第 10 章 Mach	202	11.5.2 使用查询运算符 查找文档.....	272
10.1 章节例子.....	202	11.6 小结.....	278

第 12 章 Knex 和 Bookshelf	279	第 15 章 Async.js	344
12.1 Knex	279	15.1 顺序流	345
12.1.1 安装命令行工具	280	15.2 并行流	346
12.1.2 把 Knex 添加到你的 项目	280	15.3 管线流	348
12.1.3 配置 Knex	280	15.4 循环流	352
12.1.4 SQL 查询构建器	281	15.4.1 为真则循环执行	352
12.1.5 迁移脚本	287	15.4.2 为假则循环执行	354
12.1.6 种子脚本	291	15.4.3 循环重试	355
12.2 Bookshelf	291	15.4.4 无限循环	357
12.2.1 什么是对象映射关系	292	15.5 批处理流	358
12.2.2 创建 Bookshelf 模型	292	15.5.1 异步队列	358
12.2.3 关系	299	15.5.2 异步负载	359
12.3 小结	306	15.6 小结	361
12.4 相关资源	307	第 16 章 Underscore 和 Lodash	362
第 13 章 Faye	308	16.1 安装及用法	363
13.1 HTTP、Bayeux 和 WebSocket	308	16.2 聚合和索引	364
13.1.1 WebSocket	310	16.2.1 countBy()	364
13.1.2 Bayeux 协议	310	16.2.2 groupBy()	365
13.2 开始使用 Faye	312	16.2.3 indexBy()	366
13.3 发布/订阅消息系统	313	16.3 选择	367
13.4 小结	318	16.3.1 从集合中选择数据	367
13.5 相关资源	318	16.3.2 从对象中选择数据	369
第 14 章 Q	319	16.4 链式调用	373
14.1 时间就是一切	319	16.5 函数计时	375
14.2 Promise 对比回调函数	322	16.5.1 defer()	375
14.3 Q 的 Promise	324	16.5.2 debounce()	377
14.3.1 Deferreds 和 Promises	324	16.5.3 throttle()	378
14.3.2 值和错误	328	16.6 模板	380
14.3.3 报告进度	333	16.6.1 模板内的循环及其他 JavaScript 代码	381
14.3.4 终点	336	16.6.2 书写不加鳄鱼标记的 代码	382
14.4 控制流	338	16.6.3 从模板中获取数据对象	383
14.4.1 顺序流	338	16.6.4 默认模板数据	384
14.4.2 平行流	339	16.7 小结	385
14.4.3 管道流	341	16.8 相关资源	386
14.5 小结	342		
14.6 相关资源	343		

Bower

九层之台，起于累土。

—— 文森特·梵高

包管理（Package Management），又作**依赖关系管理**（Dependency Management），并不是什么新奇的概念。此类工具为开发者提供了一种机制，以管理软件项目所依赖的各种第三方库。一些得到广泛应用的例子有：

- npm: Node.js 的包管理工具；
- Composer: 一种 PHP 依赖关系管理工具；
- pip: PyPA 的推荐工具，用于安装 Python 包；
- NuGet: 包括 .NET 在内的微软开发平台的包管理工具。

尽管包管理并不是新概念，但是将其广泛应用于前端资源的管理却是最近才有的事情。这些资源包括 JavaScript 库、样式表、字体、图标（icon）以及图像等，它们是现代网络应用的基本构件。随着现代网络应用的构建基础越来越复杂，对包管理工具的需求日益凸显。那些曾经以某些万金油式的第三方库（如 jQuery）为基础开发出的网络应用，也开始逐渐转向一些功能专一且体积小巧的库。这样做使得软件模块更加小巧，从而易于测试，同时应用的灵活性也得到了加强，可以方便地通过第三方库进行扩展，并在必要的时候进行替换。

本章旨在让你快速上手 Bower，这是一款源自 Twitter 开源项目的前端包管理工具。本章涵盖的主题有：

- Bower 的安装和配置；
- 在项目中添加 Bower；
- 查找、添加和删除包；
- 语义化版本控制；
- 管理依赖链；
- 创建 Bower 包。

1.1 准备工作

用户与 Bower 之间的所有交互都通过命令行工具来完成，该工具可以通过 npm 安装。如果你还没有安装 Bower，那么请在继续阅读之前，按照清单 1-1 进行安装。