O'REILLY®

第2版

JavaScript 经典实例

JavaScript Cookbook

Shelley Powers 著
李强 译

中国电力出版社

第2版

JavaScript经典实例

Shelley Powers 著

李强 译

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, Inc.授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

JavaScript 经典实例/（美）鲍尔斯（Powers, S.）著；李强译. —2版. —北京：中国电力出版社，2015.12

书名原文：JavaScript Cookbook, Second Edition

ISBN 978-7-5123-8188-9

I. ①J… II. ①鲍… ②李… III. ①JAVA语言－程序设计 IV. ①TP312

中国版本图书馆CIP数据核字（2015）第198997号

北京市版权局著作权合同登记

图字：01-2015-4221号

封面设计/ Ellie Volckhausen，张健

出版发行/ 中国电力出版社（http://www.cepp.sgcc.com.cn）

地　　址/ 北京市东城区北京站西街19号（邮政编码100005）

经　　销/ 全国新华书店

印　　刷/ 北京丰源印刷厂

开　　本/ 787毫米×980毫米　16开本　36.25印张　695千字

版　　次/ 2015年12月第一版　2015年12月第一次印刷

印　　数/ 0001－3000册

定　　价/ 98.00元（册）

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

目录

JavaScript的世界 1

第一部分 经典JavaScript

第1章 JavaScript不只是简单的构件块 13

1.1 JavaScript对象、基本类型和字面值之间的区别 13
1.2 从字符串提取一个列表 17
1.3 检查一个存在的、非空的字符串 19
1.4 插入特殊字符 24
1.5 使用新字符串替换模式 26
1.6 找到并突出显示一个模式的所有实例 28
1.7 使用捕获圆括号交换一个字符串中的单词 32
1.8 使用命名实体来替代HTML标签 34
1.9 把一个ISO 8601格式的日期转换为Date对象可接受的一种格式 34
1.10 使用带有定时器的函数闭包 37
1.11 记录消耗时间 40
1.12 把十进制数转换为一个十六进制值 40
1.13 把表中一列的所有数字加和 42
1.14 在角度和弧度之间转换 45
1.15 找到页面元素可容纳的一个圆的半径和圆心 45
1.16 计算圆弧的长度 47
1.17 使用ES6字符串新增方法而不会丢弃用户 48

第2章 JavaScript数组 51
2.1 在数组中搜索 51
2.2 用concat()和apply()将一个两维数组扁平化 53
2.3 删除或替换数组元素 54
2.4 提取一个数组的一部分 55
2.5 对每个数组元素应用一个函数 56
2.6 使用forEach() and call()遍历querySelectorAll()的结果 58
2.7 对数组中的每个元素执行一个函数并返回一个新数组 59
2.8 创建一个过滤后的数组 59
2.9 验证数组内容 60
2.10 使用一个关联数组来存储表单元素名和值 62
2.11 使用解构赋值简化代码 65
第3章 函数:JavaScript的构建块 68
3.1 放置函数并提升 68
3.2 把一个函数当做参数传递给另一个函数 70
3.3 实现递归算法 72
3.4 使用一个定时器和回调防止代码阻塞 75
3.5 创建能够记住其状态的函数 78
3.6 把函数参数转换到一个数组中 81
3.7 使用一个局部应用减少冗余性 83
3.8 使用缓存计算（Memoization）来提高应用程序性能 86
3.9 使用匿名函数包装全局变量 88
3.10 提供一个默认的参数 89
第4章 可扩展的JavaScript对象 91
4.1 保持对象成员私有 92
4.2 用原型扩展对象 93
4.3 继承一个对象的功能 96
4.4 通过定义一个新的属性来扩展对象 98
4.5 阻止对象可扩展性 100
4.6 阻止对对象的任何修改 101

4.7 为你的JavaScript对象提供命名空间 103
4.8 用Prototype.bind再次发现“this” 106
4.9 将对象方法链化 109

第5章 JavaScript和直接访问用户界面 111

5.1 访问一个给定的元素并找到其父元素和子元素 111
5.2 访问Web页面中所有的图像 114
5.3 使用Selectors API找出文章中的所有图像 119
5.4 设置元素的CSS样式属性 122
5.5 对无序列表应用条纹主题 125
5.6 找出共享同一属性的所有元素 126
5.7 插入一个新的段落 128
5.8 给新的段落添加文本 129
5.9 从HTML表格删除行 130
5.10 添加一个页面覆盖 133
5.11 创建可折叠的表单区段 136
5.12 隐藏页面区段 139
5.13 创建基于悬停的弹出信息窗口 140
5.14 显示一个带颜色的闪烁以表示一个动作 142

第6章 基本测试和可访问性 145

6.1 使用JSHint清理代码 145
6.2 使用QUnit测试代码 147
6.3 在各种环境中测试应用程序 150
6.4 不同编码技术的性能测试 153
6.5 突出显示错误的可访问性 156
6.6 创建一个可访问性自动更新区域 162

第7章 创建和使用JavaScript库 165

7.1 找到完美的库 165
7.2 使用Modernizr.load测试功能 166
7.3 超越Math对象的能力 167
7.4 求出两个日期之间相差的天数 170

7.5 使用一个外部库：构建于jQuery框架之上 171
7.6 使用一个jQuery插件 173
7.7 使用Mousetrap处理键盘快捷键 176
7.8 使用工具库Underscore 179
7.9 打包你的代码 181
7.10 添加对私有数据成员的支持 183
7.11 简化你的库 186
7.12 寄存库 187
7.13 通过CDN服务代码 190
7.14 把库转换为一个jQuery插件 191
7.15 安全地把几个库组合到你的应用程序中 193

第8章 简化的客户端-服务器通信和数据 196

8.1 处理从Ajax调用返回的一个XML文档 196
8.2 从一个XML树提取相关信息 198
8.3 解析一个JSON格式化字符串 202
8.4 使用JSON把一个对象转换为过滤的/转换的字符串 204
8.5（使用JSONP）对另一个域进行Ajax请求 206
8.6 处理来自一个Ajax请求的JSON 208
8.7 从服务器填充一个选项列表 210
8.8 使用定时器以新数据自动更新页面 214

第9章 创建富媒体和交互Web效果 217

9.1 在画布中创建一个动态的线条图表 217
9.2 向一个SVG文件添加JavaScript 222
9.3 从Web页面脚本访问SVG 226
9.4 在HTML中加入SVG和画布元素 228
9.5 当一个音频文件开始播放的时候运行一个例程 230
9.6 用JavaScript和video元素控制视频 232
9.7 通过画布为视频添加过滤效果 235

第二部分 JavaScript全面兴起

第10章 新的ECMAScript标准对象 243

10.1 在浏览器应用程序中使用let 244
10.2 创建非重复值的一个集合 247
10.3 用唯一的不同的键创建唯一的键/值对 249
10.4 创建绝对唯一的对象属性键 252
10.5 使得遍历任务变简单 254
10.6 创建优雅地结束的函数 255
10.7 使用Proxy实现即时对象行为修改 257
10.8 创建一个真正的类并扩展它（略微借助于Traceur） 260
10.9 使用Promise实现高效异步处理 263

第11章 Node:服务器上的JavaScript 266

11.1 响应一个简单浏览器请求 266
11.2 提供格式化的数据 269
11.3 读取和写入文件数据 271
11.4 在Node中使用let和其他的ES6添加 275
11.5 使用REPL交互式地尝试Node代码段 277
11.6 从终端获取输入 279
11.7 使用Node定时器并理解Node事件循环 281
11.8 管理回调地狱 285
11.9 用一个Node应用程序访问命令行功能 289
11.10 在同一端口上运行Node和Apache 292
11.11 保持一个Node实例启动并运行 294
11.12 监控应用程序修改和重启 296
11.13 用Request进行屏幕抓取 297
11.14 在Commander的帮助下创建一个命令行工具 299

第12章 模块化和管理JavaScript 302

12.1 使用脚本加载器来加载脚本 303

12.2 以HTML5的方式异步加载脚本 306
12.3 将JavaScript转换为AMD和RequireJS 307
12.4 将RequireJS和jQuery或其他的库一起使用 310
12.5 加载和使用Dojo模块 314
12.6 使用npm安装和维护Node模块 315
12.7 通过npm搜索一个具体的Node模块 317
12.8 将你的库转换为Node模块 318
12.9 将自己的代码应用到所有环境中 320
12.10 创建一个可安装的Node模块 324
12.11 使用Bower打包和管理客户端依赖性 328
12.12 用Browserify编译Node.js模块以便在浏览器中使用 330
12.13 对你的Node模块进行单元测试 332
12.14 用Grunt运行任务 335

第13章 API的乐趣 340

13.1 通过一个RESTful API访问JSON格式的数据 340
13.2 使用Restify创建一个RESTful API 344
13.3 在桌面浏览器中支持类似移动平台的通知 349
13.4 在浏览器中本地加载文件 352
13.5 使用Web Worker和File API创建一个Mini的E-Pub阅读器 354
13.6 探索Google Maps和其他的API 359
13.7 从一个Node应用程序访问Twitter API 366

第14章 JavaScript框架 373

14.1 使用Express-Generator生成一个Express站点 374
14.2 将一个生成的Express站点转换为一个基本的MVC app 381
14.3 选择一个SPA框架：解构TodoMVC 395
14.4 使用OAuth框架 406
14.5 用Web组件扩展可能性 419

第15章 高级客户端－服务器通信和流 431

15.1 允许跨域请求 431

15.2 在Ajax中实现一个PUT请求 434
15.3 通过Ajax发送二进制数据并加载到图像中 437
15.4 跨域共享HTTP Cookies 439
15.5 在客户端和服务器之间建立双向通信 441
15.6 使用变换流上传和压缩文件 446
15.7 测试WebSockets应用程序的性能和功能 448

第16章 数据可视化和客户端/服务器图形 451

16.1 使用D3创建一个SVG柱状图 452
16.2 用雷达图映射数据点变化 457
16.3 通过WebSocket形成一个滚动的时间线 460
16.4 产生所生成的Web页面内容的屏幕截图（PhantomJS） 464
16.5 将图形转换为文本（Ocrad.js） 469
16.6 裁剪（或者叫做修改）上传的图像 472

第17章 数据和持久性 476

17.1 验证表单数据 476
17.2 使用HTML5持久化信息 482
17.3 针对客户端存储使用sessionStorage 485
17.4 创建一个localStorage客户端数据存储项 493
17.5 使用Squel.js查询一个MySQL数据库 496
17.6 使用IndexedDB在客户端持久化较大的数据块 499
17.7 使用Dropbox数据存储访问云中的数据 502

第18章 JavaScript迈上移动之路 512

18.1 创建一款可安装的、寄存的Web App 512
18.2 为Amazon Appstore打包Web App 518
18.3 使用Cordova(PhoneGap)构建一款基本的Android App 520
18.4 将Where Am I移植到Android 527
18.5 创建一个Geolocation Firefox OS App 535
18.6 将Geolocation App移植到一个Google Chrome App 544

18.7 在Kindle Fire OS环境中发布Geolocation App 551
18.8 调试Android或Amazon Fire OS App 555
18.9 获取有关设备的信息 557

附录A 认识jsBin和jsFiddle 563

JavaScript的世界

1996年，我编写了自己的第一本JavaScript图书。那时候必须匆忙地寻找足够的资料来充实一本图书。那时候，还没有DHTM，没有ECMAScript，没有移动开发，并且也没有Node.js。表单验证和弹出式警告框都很新鲜。在编写本书的第2版的时候，我遇到了完全相反的问题：JavaScript的世界那么大，我想去看看，用一本书的篇幅也介绍不完它。但是，我在本书中已经将自己的知识倾囊相授了。

JavaScript的世界是本书的关键。JavaScript的应用从浏览器扩展到了服务器，扩展到了移动环境，扩展到了云。我们已经超越了简单的库，到了复杂的模块系统；从基本的动画到丰富的数据可视化，还加入了一些音频和视频以增添乐趣和情怀。得益于高级的框架，整个应用程序通过一个HTML页面就可以提供，并且MEAN也不再是适用于糟糕的人的一个形容词了。

Ajax仍然在发展，并且一直在应用，但是现在，它加入了直接和间接的双工通信，不再需要伪造服务器—客户端通信了，因为我们已经真正地有了这种通信。我们可以连接到Twitter和Dropbox，为Android设备创建App，并且在浏览器中直接打开ePub文件进行阅读。客户端和服务器上可用的库和模块，负责如此之多的、复杂的、繁琐的细节，我们可以专注于创建应用程序所独有的东西。十年之前，我能找到一个满足自己的需求的库就会惊讶不已了。现在，要是找不到一个满足我的需求的库，我才会感到惊奇。

我们拥有了所有这些，但还是用JavaScript作为语言。还是使用String和Number，Array和Function，并且还是使用最基本的语句：

```
var someVar = 'Hello, World?';
```

然而，如今的JavaScript和我1996年编写第一本书时的那种语言已经不同了。它发展并壮大了，有了ECMAScript 5，现在是ECMAScript 6，甚至还会通过最新的增加成为ECMAScript 7。似乎这门语言每个月都会有新增加的内容。我的意思是，几乎每个月都有一个新的版本。

使得JavaScript更加有趣和丰富的是，标准组织和服务提供商所提供的API与日俱增。

还没有一个让JavaScript程序员更为激动的时刻。但是，已经有点令人应接不暇了，而这也正是本书关注的焦点：掌控这个精彩的JavaScript的世界。

本书的读者

为了包含如今使用中的JavaScript相关的众多领域和主题，我必须先给出一个承诺：这不是一本针对JavaScript初学者的图书。对于那些新手，有如此之多的很好的图书和教程，因此我们坦然地将本书的门槛设置得比本书第一版高一些。

如果你已经使用JavaScript几个月了，可能要动手尝试一下Node或Ajax开发了，那么你应该熟悉本书的内容。一些主题可能有挑战性，不过其挑战的方式不错。

本书组织结构及其原因

我最初想要用一张很大的图来表示JavaScript的世界，我可能会把这张图分段，然后分别介绍每一章。但很快我就意识到，JavaScript中没有哪一部分是孤立于其他的部分而单独存在的。如果有的话，JavaScript是一幅很大的、带有众多的交集的文氏图，更像是螺旋图，而不是清晰的、相连的泡泡图。这从视觉上看上去太宏大了。相反，我将本书划分为18个松散定义的章，内容重叠的地方由交叉引用的方式加以处理。

本书分为两部分，分别是“经典JavaScript”和“JavaScript全面兴起”。

“经典JavaScript”部分是该语言的牢固基础，我们在过去的十多年里就拥有了这部分，并且也没有放弃。但是这部分也不是静止不发展的。我们有了好朋友String、Number、Boolean、Array、Function和Object，但是，得益于ECMAScript 5和6，我们可以用这些对象做的事情还很多。此外，在进入到较为先进、复杂的JavaScript用法之前，我们仍然需要理解如何使用Ajax、操作JSON、创建和使用库，以及将较为流行的库之一（jQuery）加入到我们的应用程序中。我们还需要理解如何在浏览器中工作，这仍然是大多数JavaScript开发的工作环境，并且要在此环境中测试我们所创建的内容以确保其可访问性。

现在，在所有的现代浏览器中，视频和音频以及Canvas和SVG都得到了支持，对这些富媒体元素的基本理解也是基础。

“JavaScript全面兴起”部分，基本上就是所有其他的内容了。这包括ECMAScript 6中引入的新的对象、服务器上的JavaScript（Node）、复杂的框架（在服务器和客户端），以及模块化的JavaScript。这还包括移动设备中的JavaScript、数据可视化、服务器中可用的图形化工具、双向的客户端-服务器通信，以及可用的API、库和模块组成的丰富多彩的世界。

似乎经常会令人困惑，但是，你在不同的环境中尝试的越多，就越能够意识到JavaScript是让所有功能走到一起的关键。下面分别介绍每一章的内容。

第一部分，经典JavaScript

第一部分关注JavaScript的传统用法，因为它们在过去的几年里已经实践过，但是有了一些更新，包括加入了新的思路、修改，并且改进了功能。

第1章 JavaScript不只是简单的构件块

介绍了一些熟悉的老朋友的用法，包括String、Number、Boolean、RegExp、Math和Date。这些介绍超出了基础知识的范围，并且已经涉及ECMAScript 5和6给我们带来的一些新的扩展。

第2章 JavaScript数组

可能JavaScript中没有哪一部分比简单的、基础的Array改变得更多了。本章超出了基本的Array的用法，介绍了新的功能。

第3章 函数:JavaScript的构建块

无处不在的函数，没有它的话，我们还能做什么？在JavaScript中，没有函数的话，我们能做的事情微不足道。本章介绍了一些较为高级的函数用法，并且介绍了较为现代的函数用法。我们介绍了3种基本的函数构造类型，以及极为有用的IIFE()。

第4章 可扩展的JavaScript对象

Array正在经受着变化，在概念和用法上亦步亦趋的是JavaScript Object，如果它不是可分配的话，就什么也干不了，由此，它成为第5章的主题。本章的大部分都关注其可分配性，包括好的用法和不好的用法。我还简单介绍了函数式编程相对于面向对象编程逐渐变得流行起来。

第5章 JavaScript和直接访问用户界面

你逃不出DOM的手掌心，它对一切了如指掌（除非第14章所介绍的Shadow DOM）。如今，操作DOM充满了很多乐趣，得益于新的查询能力。尽管大多数人使用jQuery，理解在这个库及其他的流行库的表面之下发生了什么，还是很重要的。

第6章 基本测试和可访问性

不管JavaScript有多么新，仍然有JavaScript的最佳实践需要遵守，例如，保持代码整齐、可测试，并且确保可访问性。我们现在有了新的工具可以让这些必须的任务变得更为容易，并且更加有趣。

第7章 创建和使用JavaScript库

本章我们介绍库创建的基本知识，包括压缩库，将其寄存在GitHub或CDN，使用外部库（jQuery 和Underscore）及将库转换为一个jQuery插件。我们还将仔细看看jQuery，但不是所有的库做所有的事情，我们还将看看那些专注于单个类型的任务的库。一旦准备好了基本的库，我们可以继续模块化代码，参见第12章。

第8章 简化的客户端-服务器通信和数据

如果不能很好地理解Ajax及如何操作JSON和XML的话，你无法使用新的通信技术（例如，WebSocket）。是的，XML仍然存在。在使用第14章介绍的新的客户端-服务器通信技术之前，理解本章所介绍的技术是很有必要的。

第9章 创建富媒体和交互Web效果

本章介绍了Canvas元素、2D图形、SVG，以及音频和视频元素的基本用法。本章还涉及这些媒体类型的组合（整合SVG和Canvas），以及在视频运行的时候修改它。第16章将介绍数据可视化、较为难以理解的图形化工具，以及服务器端的图形。

第二部分 JavaScript全面兴起

第二部分之所以起这个标题，是因为在过去几年里，JavaScript开发者绝对想象不到我们今天所能做的这些事情。这一部分介绍的并不都是全新的技术，但都是最前沿的技术，例如高级的客户端-服务器通信、数据可视化、OAuth以及移动开发。

第10章 新的ECMAScript标准对象

ECMAScript 6引入了几个新的对象。我介绍了所有这些，或者说至少，在编写本书的时候已经知道的所有对象（是的，JavaScript在变化中，有时候，变化的有点太快了令人难以跟上）。我所喜爱的新的添加是什么？函数生成器和迭代器。

第11章 Node:服务器上的JavaScript

本章快速介绍了Node.js的用法。你不需要有Node的经验，就可以使用本章中的示例，但是，本章确实介绍的有点快。它回答了最常问到的Node的问题：如何保持一个Node服务器运行，如何在和Apache相同的服务器上运行Node，当应用程序变化的时候或者由于某些原因它崩溃的时候，如何自动重启Node。本章还探讨了如何使用Node模块，如何创建基于Node的独立应用。

第12章 模块化和管理JavaScript

本章完全介绍JavaScript模块的新世界。本章继续介绍第7章所涉及的旧的库但仍然进行必要的代码复用，还加入了一些新的工具和方法，以创建、发布和使用模块化的代码。本章介绍了模块化的不同方法（AMD和Common JS），创建Node模块，使用RequireJS、Bower、Browserify及任务管理器Grunt。如果我们的应用名称没有创新的话，我们用JavaScript也没什么价值。

第13章 API的乐趣

API是程序员的协议。它是接入到浏览器内部工作的一个接口，但是，它也是访问来自一个媒体服务器或远程资源的服务或数据的一种方法。API还会在图形化、数据和移动应用中使用，这些将在第16章到第18章介绍。本章介绍了3种不同的API：构建于REST的原理之上的远程API、在浏览器中引入了新功能的W3C API，以及在本地作用的远程API。

第14章 JavaScript框架

有些较为复杂的组件是框架，而它们要么位于服务器上（ExpressJS），或者在客户端中（AngularJS、Backbone或Ember）。框架还包含了关于内容问题或业务用法的复杂策略（例如用于验证的OAuth），以及非常新的Web组件。

在进入本章之前，略微谈了点其他的话题：我想要在本书中展示至少一种流行的客户端MV*框架工具，但是没法决定用哪一种。它们的区别很大，详细介绍一种框架的话，会导致这一节对于不使用这一特定框架的读者来说毫无用处。因此，我决定分析ToDoMVC Web应用程序，并且深入了解如何使用3种较为流行的框架工具Angular、Ember.js和Backbone实现它。好在，我所使用的过程可以用于其他的框架。

我决定更加深入地介绍OAuth框架，因为在我们所想要使用的众多API中，越来越多地用到对数据和服务的授权。第13章介绍了OAuth还可以和Twitter API一起使用，此外，第17章介绍了Dropbox Datastores。

我还介绍了Web组件，但是并不依赖于一个polyfill（例如，Google的Polymer），因为在学习一些新的内容的时候，我很担心依赖于专有性的技术。

第15章 高级客户端-服务器通信和流

客户端-服务器通信要比当前所有现代浏览器中所存在的WebSockets和CORS（跨源资源共享）好很多。实时的、双向的通信可以大大地简化我们的生活。因为这种类型的通信是一种流，本章将介绍Node的新的传输流。

第16章 数据可视化和客户端/服务器图形

Canvas元素和SVG首次接受广泛的支持，而当它们让位于更为正式的数据可视化的