

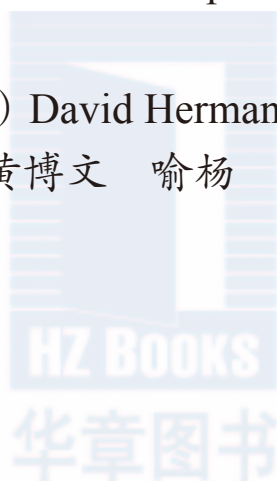
Effective 系列丛书

Effective JavaScript: 编写高质量 JavaScript 代码的 68 个有效方法

Effective JavaScript: 68 Specific Ways to Harness the Power of
JavaScript

(美) David Herman 著

黄博文 喻杨 译



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

Effective JavaScript: 编写高质量 JavaScript 代码的 68 个有效方法 / (美) 赫尔曼 (Herman, D.) 著; 黄博文, 喻杨译. —北京: 机械工业出版社, 2013.11

(Effective 系列丛书)

书名原文: Effective JavaScript: 68 Specific Ways to Harness the Power of JavaScript

ISBN 978-7-111-44623-1

I. E… II. ①赫… ②黄… ③喻… III. Java 语言 – 程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2013) 第 257668 号

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书版权登记号: 图字: 01-2013-0739

Authorized translation from the English language edition, entitled *Effective JavaScript: 68 Specific Ways to Harness the Power of JavaScript, 1E, 9780321812186* by David Herman, published by Pearson Education, Inc., Copyright © 2013.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Chinese Simplified language edition published by Pearson Education Asia Ltd., and China Machine Press Copyright © 2014.

本书中文简体字版由 Pearson Education (培生教育出版集团) 授权机械工业出版社在中华人民共和国境内 (不包括中国台湾地区和中国香港、澳门特别行政区) 独家出版发行。未经出版者书面许可, 不得以任何方式抄袭、复制或节录本书中的任何部分。

本书封底贴有 Pearson Education (培生教育出版集团) 激光防伪标签, 无标签者不得销售。

Effective 系列丛书经典著作, 亚马逊五星级畅销书, Ecma 的 JavaScript 标准化委员会著名专家撰写, JavaScript 语言之父、Mozilla CTO Brendan Eich 作序鼎力推荐! 作者凭借多年标准化委员会工作和实践经验, 深刻辨析 JavaScript 的内部运作机制、特性、陷阱和编程最佳实践, 将它们高度浓缩为极具实践指导意义的 68 条精华建议。

本书共分为 7 章, 分别涵盖 JavaScript 的不同主题。第 1 章主要讲述最基本的主题, 如版本、类型转换要点、运算符注意事项和分号局限等。第 2 章主要讲解变量作用域, 介绍此方面的一些基本概念, 以及一些最佳实践经验。第 3 章主要讲解函数的使用, 深刻解析函数、方法和类, 并教会读者在不同的环境下高效使用函数。第 4 章主要讲解原型和对象, 分析 JavaScript 的继承机制以及原型和对象使用的最佳实践和原则。第 5 章主要介绍数组和字典, 阐述将对象作为集合的用法以及使用数组和字典的一些陷阱。第 6 章介绍库和 API, 讲解如何设计良好的 API 的技巧, 以清楚、简洁和明确地表达程序, 并提高可重用率。第 7 章讲解并发, 在技术上讨论一些“约定成俗”的 JavaScript 用法。

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: 陈佳媛

印刷

2014 年 1 月第 1 版第 1 次印刷

186mm × 240mm · 11.25 印张

标准书号: ISBN 978-7-111-44623-1

定 价: 49.00 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

本书赞誉

“这是一本绝不辜负 Effective 软件开发系列期望的编程书籍。对于任何一个想要做到严谨编程的 JavaScript 开发者来说，这本书绝对不容错过。这本书阐述了 JavaScript 内部工作的一些细节，以期帮助读者更好地利用 JavaScript 语言优势。”

——**Erik Arvidsson**，高级软件工程师

“很少有像 David 这样的编程语言极客能如此舒适、友好地讲解编程语言。他带领我们领会 JavaScript 语法和语义，这个过程既令人陶醉又极其深刻。本书以舒适的节奏额外提供了一些“有问题”的现实案例。当你读完本书后，你会感觉自己获得了一种强大而全面的掌控能力。”

——**Paul Irish**，Google Chrome 开发主管

“在阅读本书之前，我以为它只是另一本关于如何更好地使用 JavaScript 编程的书籍。然而本书远不止如此，它还会使你更深入地理解 JavaScript 这门语言，这是至关重要的。如果没有这层对 JavaScript 的深入理解，那么你绝不会懂得语言本身的任何东西，只知道其他的程序员是如何编写代码的。”

“如果你想成为一名真正优秀的 JavaScript 开发者，那么请阅读本书。就我来说，我多么希望在第一次开始 JavaScript 编程时就已经阅读了它。”

——**Anton Kovalyov**，JSHint 开发者

“如果你正在寻找一本正式且极具可读性的并极具洞察力的 JavaScript 语言的书籍，那不用舍近求远了。JavaScript 开发者能够从其中找到珍贵的知识宝藏，甚至技术精湛的 JavaScript 程序员也一定能从中获益。对于有其他语言经验而想一头扎进 JavaScript 世界的从业人员来说，本书是迅速学习 JavaScript 的必读之物。然而，不管你的背景如何，但都不得不承认作者 Dave Herman 在探索 JavaScript 方面做得非常棒——JavaScript 的优美部分、不足部分或介于两者之间的所有内容都囊括于本书之中。”

——**Rebecca Murphey**，Bocoup 高级 JavaScript 开发者

“对于任何一位理解 JavaScript 并且想要完全掌握它的人员来说，本书是必不可少的读物。Dave Herman 带给了读者深刻的、具有研究和实践意义的 JavaScript 语言理解，通过一个接一个的例子指导并帮助读者达到与他同样的理解高度。这不是一本寻求捷径的书籍，恰恰相反，是一本难得的将经验提炼为指南的书籍。它是一本为数不多让我毫不犹豫推荐的关于 JavaScript 的书籍。”

——Alex Russell, TC39 成员, Google 软件工程师

“很少有人有机会同大师一起学习他们的手艺。这本书弥补了这种遗憾，其对 JavaScript 的研究就像随一位时间旅行哲学家回到公元前 5 世纪与柏拉图一同学习。”

——Rick Waldron, JavaScript 传教士, Bocoup



译者序

虽然 JavaScript 在诞生之初由于商业原因及缺乏规范，一直饱受诟病。但是随着时间的推移，人们已经逐渐走出了对这门语言的偏见和误解，开始领略它那强大的语言特性威力。当下 JavaScript 语言大红大紫，研究讨论 JavaScript 的相关书籍早已汗牛充栋，但是这本书作为 Effective 软件开发系列中的一员，却是不可或缺的。

学会编写 JavaScript 程序容易，但要成为专家却实属不易。一方面是由于 JavaScript 语言的设计思想与 Java、C# 等大众语言区别很大，另一方面是由于其设计时的仓促性导致 JavaScript 语言本身精华与糟粕并存。本书的作者 David Herman 作为 JavaScript 标准化的参与者，在书中自然对 JavaScript 的精华和糟粕都进行了深入阐述，并且给出了很多实用的建议。这些建议都来自于第一线的实践经验，无论是初学者还是高级程序员，都可以从中吸收养分，进而快速成长。

本书深入阐述了 JavaScript 语言，通过它可以了解到如何有效地编写出高移植性、健壮的程序和库。本书传承了 Effective 软件开发系列的简明场景驱动风格，通过提示、技术及实用的示例代码解释 JavaScript 中的重要概念。

全书共涉及 68 条关于 JavaScript 程序设计的建议。第 1 章可以让初学者快速熟悉 JavaScript，了解 JavaScript 中的原始类型、隐式强制转换、编码类型等基本概念；第 2 章着重讲解了有关 JavaScript 的变量作用域的建议，不仅介绍了怎么做，还介绍了操作背后的原因，帮助读者加深理解；第 3 章和第 4 章的主题涵盖函数、对象及原型三大方面，这可是 JavaScript 区别于其他语言的核心，读者也不必紧张，在 David Herman 大叔的指引下，你可以轻松掌握这些核心内容，了解到业界最佳实践；第 5 章则阐述了数组和字典这两种容易混淆的常用类型及具体使用时的建议，避免陷入一些陷阱；第 6 章讲述了库和 API 设计；第 7 章讲述了并行编程，这是晋升为 JavaScript 专家的必经之路。

想要深入了解 JavaScript 并获取一线专家的宝贵经验吗？那么，这本书正好适合你。

我和同事喻杨在翻译这本书的过程中投入了不少精力，生怕给这本经典之作留下一些遗憾。感谢华章公司的编辑们对我们的支持。

最后，希望本书能给大家带来一次超凡的阅读体验。

黄博文

序

众所周知，我在 1995 年 5 月用了 10 天时间创建了 JavaScript 语言。迫于现实的压力和冲突管理的势在必行，我将 JavaScript 设计为“看起来像 Java”、“方便初学者”、“在网景浏览器中几乎能控制它的一切”的编程语言。

鉴于极具挑战性的要求和非常短的时间表，我的解决方案除了正确获得了两大特性之外（第一类函数、对象原型），还将 JavaScript 设计为一开始就极具延展性。我知道一些开发者不得不对最开始的几个版本“打补丁”来修正错误，这些先驱的方法比我使用内置库胡乱拼凑的方法更好。举例来说，虽然许多编程语言都限制了可变性，在运行时不能修改或扩展内置对象，或者标准库的名称绑定不能通过赋值被覆盖，但是 JavaScript 几乎允许完全改变每个对象。

总的来说，我相信这是一个很好的设计决定。它明确了某些领域的挑战（如在浏览器的安全边界内，安全地混用可信和不可信的代码）。对于 JavaScript 来说，支持所谓的猴子补丁是很重要的。凭借它，开发者可以编辑标准对象来解决 Bug 和仿效“未来”的功能改造一些旧的浏览器（称为 polyfill 库的 shim，在美式英语中称为 spackle）。

除了这些平凡的应用之外，JavaScript 的可塑性鼓励用户沿着几个更富有创造性的方向形成和发展创新网络。这使得用户仿效其他编程语言，创建了许多工具包和框架库：基于 Ruby 的 Prototype、基于 Python 的 MochiKit、基于 Java 的 Dojo 以及基于 Smalltalk 的 TIBET。随后是 jQuery 库（“JavaScript 的新浪潮”），2007 年我第一次看到它，对我来说，我似乎是一个后来者。它暴风雨般地引领着 JavaScript 的世界，异于其他编程语言，而从旧的 JavaScript 库中学习，开辟了浏览器的“查询和做”（query and do）模型，从根本上简化了浏览器。

这引领着 JavaScript 用户群及其创新网络，促成了一种 JavaScript “自家风格”。这种风格仍在仿效和简化其他的程序库，也促成了现代 Web 标准化的工作。

在这一演变的过程中，JavaScript 保持了向后兼容，当然默认情况下，它是可变的。另外，在 ECMAScript 标准最新的版本中新增了一些方法来冻结对象和封闭对象的属性，以防止对象扩展和属性被覆盖。JavaScript 的演进之旅远未结束。这就像生活的语言和生物系统一样，变化始终存在。在此之前，我无法预见一个单一的横扫其他程序库的“标准库”或编码风格。

没有语言不是怪癖的，或者所有语言几乎都是受限地执行常见的最佳实践。**JavaScript** 是一门怪癖的或者充满限制主义色彩的编程语言。因此，与大多数其他的编程语言相比，想要高效地使用 **JavaScript** 编程，开发人员必须学习和追求优秀的风格、正确的用法和最佳的实践。当考虑如何做到最高效，我相信避免过度编程和构建刚性或教条式的风格指南是至关重要的。

本书以一种平衡的方式讲解 **JavaScript** 编程。它基于一些具体的实证和经验，而不迂回于刚性或过度的方法。我认为对于许多寻找以不牺牲表现力，并希望自由采用新思想和编程范式来编写高效 **JavaScript** 的人来说，它是一位重要的助手和可信赖的向导。它也是一本备受关注、充满乐趣和拥有许多令人叹为观止的示例的读物。

最后，自 2006 年，我有幸结识 **David Herman**。那是我第一次以 **Mozilla** 代表的身份邀请他作为特邀专家在 **Ecma** 标准化机构工作。**Dave** 深刻、谦逊的专家意见以及他的热情让 **JavaScript** 在书中的每一页大放异彩。本书绝无仅有！

Brendan Eich



前 言

学习一门编程语言，需要熟悉它的语法、形式和结构，这样我们才能编写合法的、符合语义的、具有意义和行为正确的程序。但除此之外，掌握一门语言需要理解其语用，即使用语言特性构建高效程序的方法。后一种范畴是特别微妙的，尤其是对 JavaScript 这样一种灵活而富有表现力的编程语言来说。

这是一本关于 JavaScript 语用学的书。这不是一本入门书籍，我假设你在一定程度上熟悉了 JavaScript 和通常的编程。很多优秀的 JavaScript 入门书籍可供参考，例如，Douglas Crockford 的《JavaScript: The Good Parts》和 Marijn Haverbeke 的《Eloquent JavaScript》。本书的目的是帮助你加深理解如何有效地使用 JavaScript 构建更可预测、可靠和可维护的 JavaScript 应用程序和库。

JavaScript 与 ECMAScript

在深入本书之前澄清一些术语是有必要的。这是一本关于举世皆知的 JavaScript 编程语言的书籍。然而，官方标准定义的规范性的描述是一门称该语言为 ECMAScript。历史很令人费解，但这可以归结为版权问题：出于法律原因，Ecma 国际标准化组织不能使用“JavaScript”作为其标准名称。（更糟的是，标准化组织将其原来的名称 ECMA（欧洲计算机制造商协会的英文首字缩写）改为不是全大写的 Ecma 国际标准化组织。彼时，ECMAScript 这个名字大约也是早已注定。）

正式来说，当人们提到 ECMAScript 时，通常是指由 Ecma 国际标准化组织制定的“理想”语言。与此同时，JavaScript 这个名字意味着来自语言本身的所有事物，例如某个供应商特定的 JavaScript 引擎。通常情况下，人们经常交替使用这两个术语。为了保持清晰度和一致性，在本书中，我将只使用 ECMAScript 来谈论官方标准，其他情况，我会使用 JavaScript 指代语言。我还会使用常见的缩写 ES5 来指代第 5 版的 ECMAScript 标准。

关于 Web

避开 Web 来谈 JavaScript 是很难的。到目前为止，JavaScript 是唯一为用于客户端应用程

序脚本的所有主流浏览器提供内置支持的编程语言。此外，近年来，随着 Node.js 平台的问世，JavaScript 已经成为一个实现服务器端应用程序的流行编程语言。

不过，本书是关于 JavaScript 而非 Web 的编程。有时，谈论一些 Web 相关的例子和应用程序的概念是帮助读者理解。但是，这本书的重点是 JavaScript 语言的语法、语义和语用，而不是 Web 平台的 API 和技术。

关于并发

JavaScript 一个新奇的方面是在并发环境中其行为是完全不明朗的。ECMAScript 标准（包括第 5 版）关于 JavaScript 程序在交互式或并发环境中的行为只字未提。第 7 章涉及并发，因此，我只是从技术角度介绍一些非官方的 JavaScript 特性。但实际上，所有主流的 JavaScript 引擎都有一个共同的并发模型。尽管在标准中未提及并发，但是致力于并发和交互式的程序是 JavaScript 编程的一个核心概念。事实上，未来版本的 ECMAScript 标准可能会正式地标准化这些 JavaScript 并发模型的共享方面。

致谢

这本书在很大程度上要归功于 JavaScript 的发明者——Brendan Eich。我深深感谢 Brendan 邀请我参与 JavaScript 标准化工作，以及他对我在 Mozilla 的职业生涯中给予的指导和支持。

本书中的大部分材料是受优秀的博客文章和在线论文的启发。我从 Ben “cowboy” Alman、Erik Arvidsson、Mathias Bynens、Tim “creationix” Caswell、Michaeljohn “inimino” Clement、Angus Croll、Andrew Dupont、Ariya Hidayat、Steven Levithan、Pan Thomakos、Jeff Walden，以及 Juriy “kangax” Zaytsev 的博客中学到很多东西。当然，本书的最终资源来自 ECMAScript 规范。ECMAScript 规范自第 5 版以来由 Allen Wirfs-Brock 不知疲倦地编辑和更新。Mozilla 开发者网络仍然是 JavaScript API 和特性最令人印象深刻的、高品质在线资源之一。

在策划和写作这本书的过程中，我有许多顾问。在我开始写作之前，John Resig 就以作者的角度给了我很多有用的建议。Blake Kaplan 和 Patrick Walton 帮我在早期阶段整理我的想法和规划出这本书的组织结构。在写作的过程中，我从 Brian Anderson、Norbert Lindenberg、Sam Tobin-Hochstadt、Rick Waldron 和 Patrick Walton 那里得到了很好的建议。

很高兴能够和 Pearson 的工作人员共事。Olivia Basegio、Audrey Doyle、Trina MacDonald、Scott Meyers 和 Chris Zahn 一直关注我提出的问题，对我的拖延报以耐心，并通融我的请求。我无法想象还能有一个更愉快的写作经历。我对能为 Effective 系列写一本书感到非常荣幸。因为很久

以前我就是《Effective C++》的粉丝，我曾经怀疑我是否有亲自书写一本 Effective 系列书籍的荣幸。

我也简直不敢相信自己有这样的好运气，能够找到梦之队一样的技术编辑。我很荣幸 Erik Arvidsson、Rebecca Murphey、Rick Waldron 和 Richard Worth 同意编辑这本书，他们为我提供了许多宝贵的批评和建议。他们多次纠正了书中一些真正令人尴尬的错误。

写一本书比我预想的要难得多。如果不是朋友和同事的支持，我可能已经失去了勇气。在我怀疑自己的时候，Andy Denmark、Rick Waldron 和 Travis Winfrey 总是给予我鼓励。

我绝大部分时候是在旧金山柏丽附近的神话般的 Java Beach 咖啡厅里写作这本书的。那里的工作人员都知道我的名字，并且我还没点餐之前，他们就知道我想要点什么。我很感谢他们提供了一个舒适的工作场所，并给我提供食物和咖啡。

我的毛茸茸的猫科小友 Schmoopy 为本书做出了它的最大贡献。至少，它不停地跳上我的膝盖，坐在屏幕前（有可能是笔记本电脑比较温暖）。Schmoopy 自 2006 年以来一直是我的忠实伙伴，我不能想象我的生活能离得开这个小毛球。

我的整个家庭对这个项目从开始到结束一直都很支持和激动。遗憾的是，我无法在我的爷爷和奶奶（Frank 和 Miriam Slamar）去世之前和他们分享这本书的成品。但他们会为我感到激动和自豪，而且本书中有一小段我儿时与爷爷 Frank 编写 BASIC 程序的经历。

最后，我要感谢我一生的挚爱 Lisa Silveria，我对她的付出无以为报。

目 录

本书赞誉	
译者序	
序	
前言	
第 1 章 让自己习惯 JavaScript	1
第 1 条：了解你使用的 JavaScript 版本	1
第 2 条：理解 JavaScript 的浮点数	6
第 3 条：当心隐式的强制转换	8
第 4 条：原始类型优于封装对象	13
第 5 条：避免对混合类型使用 == 运算符	14
第 6 条：了解分号插入的局限	16
第 7 条：视字符串为 16 位的代码单元序列	21
第 2 章 变量作用域	25
第 8 条：尽量少用全局对象	25
第 9 条：始终声明局部变量	27
第 10 条：避免使用 with	28
第 11 条：熟练掌握闭包	31
第 12 条：理解变量声明提升	34
第 13 条：使用立即调用的函数表达式创建局部作用域	36
第 14 条：当心命名函数表达式笨拙的作用域	38
第 15 条：当心局部块函数声明笨拙的作用域	41
第 16 条：避免使用 eval 创建局部变量	43
第 17 条：间接调用 eval 函数优于直接调用	44
第 3 章 使用函数	46
第 18 条：理解函数调用、方法调用及构造函数调用之间的不同 ..	46
第 19 条：熟练掌握高阶函数	48
第 20 条：使用 call 方法自定义接收者来调用方法	51
第 21 条：使用 apply 方法通过不同数量的参数调用函数	53
第 22 条：使用 arguments 创建可变参数的函数	54
第 23 条：永远不要修改 arguments 对象	56
第 24 条：使用变量保存 arguments 的引用	58
第 25 条：使用 bind 方法提取具有确定接收者的方法	59

第 26 条: 使用 bind 方法实现 函数柯里化	61	第 5 章 数组和字典	91
第 27 条: 使用闭包而不是字符串 来封装代码	62	第 43 条: 使用 Object 的直接实例 构造轻量级的字典	91
第 28 条: 不要信赖函数对象的 toString 方法	63	第 44 条: 使用 null 原型以防止 原型污染	94
第 29 条: 避免使用非标准的栈 检查属性	65	第 45 条: 使用 hasOwnProperty 方法 以避免原型污染	95
第 4 章 对象和原型	67	第 46 条: 使用数组而不要使用 字典来存储有序集合	99
第 30 条: 理解 prototype、getPrototype Of 和 __proto__ 之间的 不同	67	第 47 条: 绝不要在 Object.prototype 中增加可枚举的属性	102
第 31 条: 使用 Object.getPrototypeOf 函数而不要使用 __proto__ 属性	69	第 48 条: 避免在枚举期间修改 对象	103
第 32 条: 始终不要修改 __proto__ 属性	70	第 49 条: 数组迭代要优先使用 for 循环而不是 for...in 循环	108
第 33 条: 使构造函数与 new 操作 符无关	71	第 50 条: 迭代方法优于循环	109
第 34 条: 在原型中存储方法	73	第 51 条: 在类数组对象上复用 通用的数组方法	113
第 35 条: 使用闭包存储私有数据	75	第 52 条: 数组字面量优于数组 构造函数	114
第 36 条: 只将实例状态存储在 实例对象中	76	第 6 章 库和 API 设计	116
第 37 条: 认识到 this 变量的隐式 绑定问题	78	第 53 条: 保持一致的约定	116
第 38 条: 在子类的构造函数中 调用父类的构造函数	81	第 54 条: 将 undefined 看做 “没有值”	117
第 39 条: 不要重用父类的属性名	84	第 55 条: 接收关键字参数的 选项对象	121
第 40 条: 避免继承标准类	86	第 56 条: 避免不必要的状态	125
第 41 条: 将原型视为实现细节	88	第 57 条: 使用结构类型设计 灵活的接口	127
第 42 条: 避免使用轻率的猴子补丁	88	第 58 条: 区分数组对象和类数组 对象	130

第 59 条：避免过度的强制转换 ····· 134	第 65 条：不要在计算时阻塞事件队列 ····· 153
第 60 条：支持方法链 ····· 137	第 66 条：使用计数器来执行并行操作 ····· 156
第 7 章 并发 ····· 140	第 67 条：绝不要同步地调用异步的回调函数 ····· 160
第 61 条：不要阻塞 I/O 事件队列 ····· 140	第 68 条：使用 promise 模式清洁异步逻辑 ····· 162
第 62 条：在异步序列中使用嵌套或命名的回调函数 ····· 143	
第 63 条：当心丢弃错误 ····· 147	
第 64 条：对异步循环使用递归 ····· 150	

