Broadview
www.broadview.com.cn

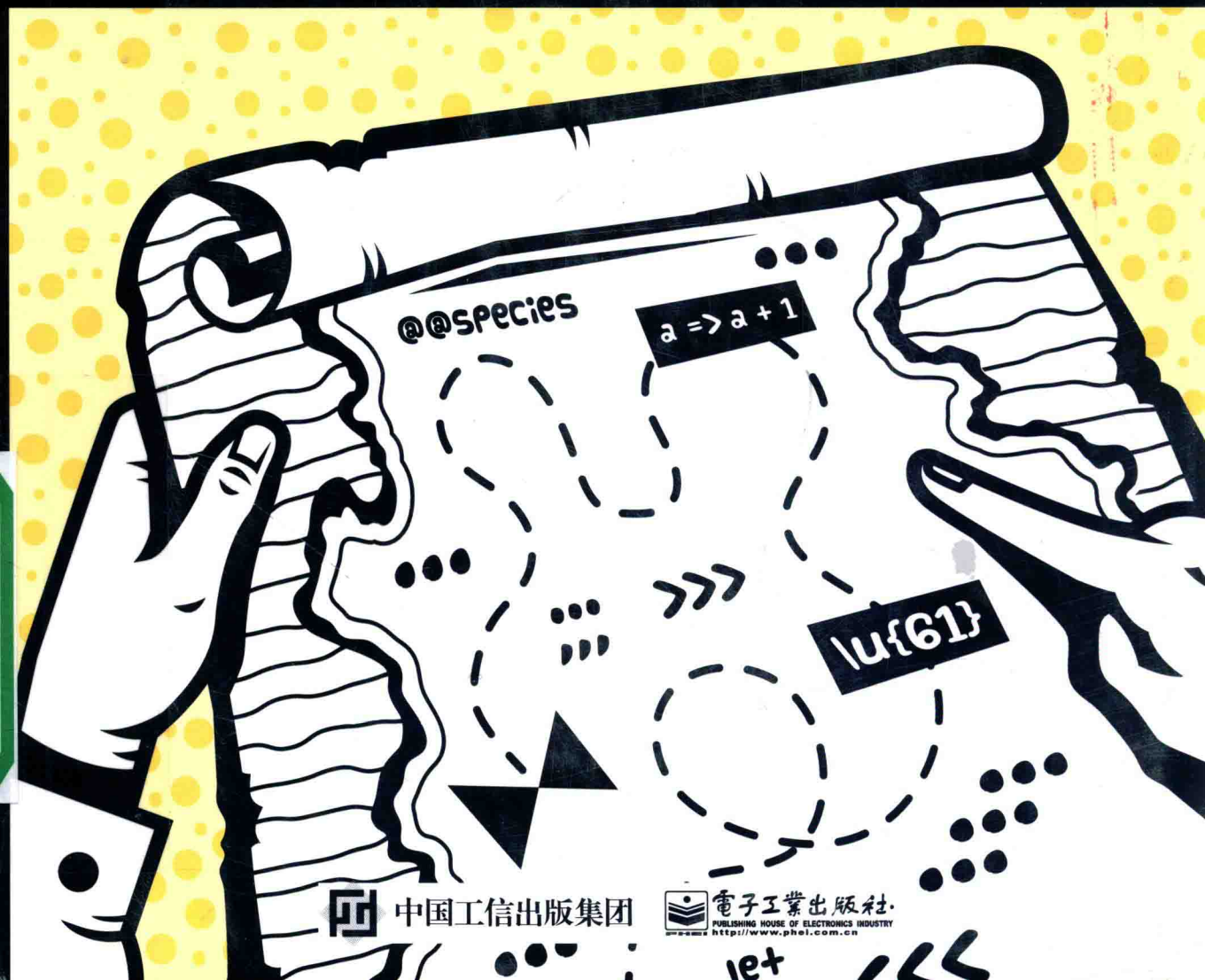
no starch press

UNDERSTANDING ECMASCRIPT 6

THE DEFINITIVE GUIDE FOR JAVASCRIPT DEVELOPERS

深入理解ES6

[美] NICHOLAS C. ZAKAS 著　刘振涛 译　贺师俊 张克军 李松峰 审校



中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

UNDERSTANDING ECMASCRIPT 6

THE DEFINITIVE GUIDE FOR JAVASCRIPT DEVELOPERS

深入理解ES6

[美] NICHOLAS C. ZAKAS 著

刘振涛 译

贺师俊 张克军 李松峰 审校

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

ES6 是 ECMAScript 标准十余年来变动最大的一个版本，其中添加了许多新的语法特性，既有大家耳熟能详的 Promise，也有闻所未闻的 Proxy 代理和 Reflection 反射；既有可以通过转译器（Transpiler）等方式在旧版本浏览器中实现兼容的 let、const、不定参数、展开运算符等功能，亦有无论如何都无法实现向前兼容的尾调用优化。深入理解 ES6 的特性对于所有 JavaScript 开发者而言至关重要，在可预见的未来，ES6 中引入的语言特性会成为 JavaScript 应用程序的主流特性，这也是本书的初衷。希望你通过阅读本书可以了解 ES6 的新特性，并在需要时能够随时使用。

版权贸易合同登记号　图字：01-2016-9347

图书在版编目（CIP）数据

深入理解 ES6 /（美）尼古拉斯·泽卡斯（Nicholas C. Zakas）著；刘振涛译. —北京：电子工业出版社，2017.7
书名原文: Understanding ECMAScript 6
ISBN 978-7-121-31798-9

Ⅰ. ①深… Ⅱ. ①尼… ②刘… Ⅲ. ①JAVA 语言－程序设计 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字（2017）第 129960 号

策划编辑：张春雨
责任编辑：徐津平
印　　刷：三河市良远印务有限公司
装　　订：三河市良远印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787 × 980　1/16　　印张：24.75　　字数：474 千字
版　　次：2017 年 7 月第 1 版
印　　次：2017 年 7 月第 2 次印刷
定　　价：99.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819　faq@phei.com.cn。

译者序

十年前谁也无法料到，彼时只能写小动画的玩具语言 JavaScript 竟会有如今之威力，这愈发显现出 Atwood 定律“凡是能用 JavaScript 写出来的应用，最终都会用 JavaScript 来写”的正确性。追本溯源，这与 ECMAScript 的发展功不可没。

然而，ECMAScript 的发展并非一帆风顺。

1999 年末，ECMA-262 第 3 版[1]正式定稿，在之后的五六年中，几乎看不到标准的任何新进展。直到 2005 年左右，随着 Google 在多个重交互的应用中普及 Ajax，开发者们逐渐接受这项新技术并逐步恢复对 JavaScript 的关注。于是，JavaScript 创始人 Brendan Eich 紧锣密鼓地筹划 ECMAScript 4 标准，直到 2007 年，耗时两年的 ECMAScript 4 标准扩充工作在 Jeff Dyer 看来已经达到 ECMAScript 3 的两倍[2]，Brendan 遂撰文[3] 进一步澄清与解释。

Douglas Crockford 认为这是一种过度复杂的税负[4]，并联合微软起草 ECMAScript 3.1 提案，同时，微软也在 TC-39 会议中正式反对 ES4 中的部分标准。冲突过后，占据舆论优势的 ECMAScript 3.1 于 2009 年作为 ES5 正式发布[5]。

ECMAScript 4 并未就此消亡。委员会全体成员将 ECMAScript 3.1 与 ECMAScript 4 中的精华保留，作为 ECMAScript Harmony（取和谐之意），它转而成为委员会的下一个目标 ECMAScript 6，并于 2015 年 6 月正式定稿，最终被命名为 ECMAScript 2015。委员会一改往日冗长的议程，约定每年必出一版，通常以当年年份命名。截至此书翻译完毕，ECMAScript 2016 也于 2016 年 6 月正式定稿[6]，最新标准尚在进程中[7]。

《Understanding ECMAScript 6》一书是作者 Nicholas C. Zakas 在 GitHub 开

源社区[8]撰写而成。作为标准的转述者，存在部分理解误区合情合理，本译作基于 No Starch Press 出版社于 2016 年 8 月出版的首印版，适当参考 GitHub 中的讨论集结而成。

在本书翻译结束之际，感慨万千。首先感谢裕波，是他的引荐让我有机会翻译本书。特别感谢李松峰老师、Hax 老师与克军老师的不吝赐教，帮助我审校翻译内容。还要感谢博文视点的侠少（张春雨编辑），他高标准、严要求的专业态度时刻鞭策我前行。

感谢就职于腾讯的时光，带我入行的导师张坤、为我解答所有疑惑的 Leader 陈恕胜、共同学习成长的兄弟陈炜鑫及其他伙伴，你们一丝不苟的态度不断磨练我的心性。

最后，特别要感谢我的母亲杨虹女士，每当我不堪于兼顾工作与翻译的时候，总是您的鼓励点亮我前进的道路。

在本书的翻译过程中我力求还原作者本意，但限于时间与水平，翻译不当之处在所难免，还敬请各位读者不吝赐教，我也会及时与出版社同步以备再版时进行修正，或以勘误的形式公布。如您有任何想法与建议，欢迎写信至我的邮箱：*lenville@gmail.com*。

[1] https://www.ecma-international.org/publications/files/ECMA-ST-ARCH/ECMA-262, 3rd edition, December 1999.pdf

[2] https://mail.mozilla.org/pipermail/es-discuss/2007-October/001442.html

[3] https://brendaneich.com/2007/11/es4-news-and-opinion/

[4] https://mail.mozilla.org/pipermail/es-discuss/2008-March/002529.html

[5] http://www.ecma-international.org/publications/files/ECMA-ST-ARCH/ECMA-262 5th edition December 2009.pdf

[6] https://www.ecma-international.org/ecma-262/7.0/index.html

[7] https://tc39.github.io/ecma262/

[8] https://github.com/nzakas/understandinges6

关于作者

Nicholas C. Zakas 自 2000 年以来一直致力于 Web 应用程序的开发，重点关注前端开发，并以写作和讲述前沿最佳实践而闻名。他曾于雅虎主页任职 5 年有余，他也是多本书的作者，其中包括 *The Principles of Object-Oriented JavaScript*（No Starch Press 出版社）和 *Professional JavaScript for Web Developers*（Wrox 出版社）。

关于技术评审

Juriy Zaytsev（在网上以 kangax 著称）是纽约的一位前端网站开发人员。自 2007 年以来，他一直在探索 JavaScript 的怪异特性并撰写相关文章。Juriy 为多个开源项目做出过贡献，其中包括 Prototype.js 和其他的热门项目，如他自己的 Fabric.js。他是按需定制打印服务 printio.ru 的共同创始人，目前任职于 Facebook。

序

ECMAScript 6 如暴风雨般骤临世界，人们期待已久而它却突然出现，传播之快始料未及。每个人都与 ECMAScript 6 有着一段不同的故事，以下是我的故事。

2013 年，我还在一家创业公司工作，正在从 iOS 转向 Web 研发，之后我参加了 JavaScript 开源社区并共同创建了 Redux。当时我正在努力学习 Web 开发，而且我非常害怕，我的团队必须在短短几个月的时间内将我们的产品用 JavaScript 重构为 Web 版。

起初我认为用 JavaScript 编写大型软件的想法很可笑，但是一名团队成员说服了我，他说 JavaScript 不是一门玩具语言。我同意撇开成见试一试，于是打开 MDN 和 StackOverflow 首次深入学习 JavaScript。我对这门简约的语言着了迷，我的同事还教我如何使用工具，例如代码整理工具（linter）和代码合并工具（bundler）[1]。在这几个星期里我恍然大悟，原来我如此喜欢编写 JavaScript 代码。

但没有一门语言是完美的，由于使用过其他语言，我非常希望 JavaScript 也可以频繁更新，但在这 10 年间，ECMAScript 5 是唯一的重大更新，它只实现了一小部分特性，完全支持浏览器需要数 10 年的时间。彼时，即将到来的代号为 Harmony 的 ECMAScript 6（ES6）规范尚未完成，遥遥无期。“也许在 10 年内我能够写一些 ECMAScript 6 代码吧。”我想。

一些实验性的“转译器（Transpiler）”，如谷歌的 Traceur，可以将代码从

---

[1] 译者注：代码压缩工具（minifier）对于生产力和性能来说也至关重要。

ECMAScript 6 转换成 ECMAScript 5。它们大多功能非常有限，或难以插入现有的 JavaScript 构建管道。但是，随后出现的新型转译器 6to5 改变了一切。它易于安装，可以很好地集成在现有的工具中，生成的代码可读，于是其像野火般蔓延开来。6to5 现在被称作 Babel，在标准定稿前就开始为主流受众提供 ECMAScript 6 的特性。几个月以来，ECMAScript 6 无处不在。

出于各种原因，ECMAScript 6 已经把社区割裂开来。正如本书所讲，在许多主流浏览器中 ECMAScript 6 仍未完全实现。当你学习这门语言时，不得不进行的构建步骤足以使人退缩。一些库的文档和示例中有 ECMAScript 6 的代码，你可能想知道这些库是否可以在 ECMAScript 5 环境中使用。这令人感到困惑，由于这门语言之前几乎从未改变过，因此许多人对于新特性的加入并没有十分期待，而有一部分人在焦急地等待新功能的到来，并希望所有的这些新功能能放在一起使用——在某些情况下，甚至为了使用而使用，不管是否必要。

正当我对 JavaScript 的使用逐渐熟练时，我感觉再往前走很困难，我不得不学习一门新的语言。那几个月的时间里我感到很糟糕。最后在圣诞节前夕，我开始阅读本书的草稿，我简直爱不释手，在凌晨 3 点，当参加聚会的每一位成员都已熟睡，而我却理解了 ECMAScript 6！

Nicholas 是一位非常有天赋的老师。他以直截了当的方式传达深刻的细节，让你能够理解所有这些知识。除了本书之外，他也因创建 ESLint 而出名，这是一个被下载了数百万次的 JavaScript 代码分析器。

Nicholas 对 JavaScript 的了解程度很少有人能够企及，所以不要错过吸取新知识的机会。阅读本书，你将对掌握 ECMAScript 6 充满信心。

Dan Abramov

React 核心团队成员及 Redux 的创造者

鸣谢

感谢 Jennifer Griffith-Delgado、Alison Law 及 No Starch Press 的每位同仁，感谢你们对本书的支持与帮助，即使在我病重时生产力降低，你们依然保持理解与耐心，我永远难忘。

非常感谢技术编辑 Juriy Zaytsev，非常感谢 Axel Rauschmayer 博士对于本书的反馈，在几次对话中你澄清的一些概念对我非常有帮助。

感谢所有对托管在 Github 上的本书当前版本提交修订的每一位同仁：404、alexyans、Ahmad Ali、Raj Anand、Arjunkumar、Pahlevi Fikri Auliya、Mohsen Azimi、Peter Bakondy、Sarbbottam Bandyopadhyay、blacktail、Philip Borisov、Nick Bottomley、Ethan Brown、Jeremy Caney、Jake Champion、David Chang、Carlo Costantini、Aaron Dandy、Niels Dequeker、Aleksandar Djindjic、Joe Eames、Lewis Ellis、Ronen Elster、Jamund Ferguson、Steven Foote、Ross Gerbasi、Shaun Hickson、Darren Huskie、jakub-g、kavun、Navaneeth Kesavan、Dan Kielp、Roy Ling、Roman Lo、Lonniebiz、Kevin Lozandier、Josh Lubaway、Mallory、Jakub Narębski、Robin Pokorný、Kyle Pollock、Francesco Pongiluppi、Nikolas Poniros、AbdulFattah Popoola、Ben Regenspan、Adam Richeimer、robertd、Marián Rusnák、Paul Salaets、Shidhin、ShMcK、Kyle Simpson、Igor Skuhar、Yang Su、Erik Sundahl、Dmitri Suvorov、Kevin Sweeney、Prayag Verma、Rick Waldron、Kale Worsley、Juriy Zaytsev 还有 Eugene Zubarev。

此外，感谢 Casey Visco 在 Patreon 上支持本书。

前言

JavaScript 核心的语言特性是在标准 ECMA-262 中被定义的。该标准中定义的语言被称作 ECMAScript，它是 JavaScript 的子集。在浏览器与 Node.js 环境中通过附加的对象和方法可添加更多新功能，而 JavaScript 的核心依然保持 ECMAScript 的定义。总的来说，ECMA-262 标准的持续发展对于 JavaScript 的成功功不可没。ECMAScript 6 是 JavaScript 最新的重大更新，本书将为你讲解其中的改动。

ECMAScript 6 之路

2007 年，JavaScript 走向了发展中的转折点，逐渐兴起的 Ajax 开创了动态 Web 应用的新时代，而自 1999 年第三版 ECMA-262 发布以来，JavaScript 却没有丝毫改变。当时，负责推动 ECMAScript 语言发展的 TC-39 委员会将大量规范草案整合在了 ECMAScript 4 中，新增的语言特性涉足甚广，包括：模块、类、类继承、私有对象成员、可选类型注释及众多其他的特性。

然而，TC-39 组织内部对 ECMAScript 4 的动议草案产生了巨大分歧，部分成员认为不应该一次性在第四版标准中加入过多的新功能，而来自雅虎、谷歌和微软的技术负责人则共同商讨并提交了一份“ECMAScript 3.1”草案作为下一代 ECMAScript 的可选方案，此处的“3.1”意在表明只是对现有标准进行小幅的增量修改。

ECMAScript 3.1 引入的语法变化极少，这一版标准相对而言更专注于优化

属性特性，支持原生 JSON，以及为已有对象增添新的方法。委员会曾经尝试融合 ECMAScript 3.1 与 ECMAScript 4，但由于对峙双方对语言未来的发展方向分歧过大，最后以失败告终。

到了 2008 年，JavaScript 创始人 Brendan Eich 宣布 TC-39 委员会将合力推进 ECMAScript 3.1 的标准化工作。他们选择将 ECMAScript 4 中提出的大部分针对语法及特性的改动暂时搁置，到下一个版本 ECMAScript 的标准化工作完成之后，委员会全体成员再努力融合 ECMAScript 3.1 和 4 中的精华，他们还给这个版本起了一个昵称——ECMAScript Harmony（取和谐之意）。

经过标准化的 ECMAScript 3.1 最终作为 ECMA-262 第五版正式发布，它同时也被称为 ECMAScript 5。委员会表示他们永不发布第四版，以避免与从未面世的“ECMAScript 4”产生命名冲突。基于 ECMAScript Harmony 的工作随后陆续展开，继承了精华的 ECMAScript 6 将成为继 ECMAScript 5 之后发布的首个新标准。

ECMAScript 6 标准的特性已于 2015 年全部完成，并被正式命名为“ECMAScript 2015”（由于开发者们对 ECMAScript 6 更为熟悉，因此本书将继续沿用此称谓）。新标准的变化俯拾即是，大到全新的对象和模式、大幅的语法改动，小到为已有对象扩充新的方法。更令人激动的是，ECMAScript 6 中点滴的变化全都致力于解决开发者实际工作中遇到的问题。

关于本书

深入理解 ECMAScript 6 的特性对于所有 JavaScript 开发人员来说至关重要，在可预见的未来，ECMAScript 6 中引入的语言特性将构成构建 JavaScript 应用程序的基础。这也是本书的初衷，笔者希望你通过阅读本书来了解 ECMAScript 6 的新特性，并在需要时随时能够予以使用。

浏览器与 Node.js 中的兼容性

开发者们正积极地为 Web 浏览器及 Node.js 这些 JavaScript 的宿主环境添加 ECMAScript 6 的新功能。本书只关注规范中定义的正确行为，不会对比每种实现间的差异。如此一来，读者所使用的 JavaScript 环境有可能与本书中描述的不一致。

本书的目标读者

本书是专门为熟悉 JavaScript 和 ECMAScript 5 的读者准备的指南，帮助大家理解 ECMAScript 5 和 6 之间的差异。对 ECMAScript 6 早已熟稔于心的读者不必继续阅读下去。本书特别适合想了解语言未来特性的 JavaScript 中高级开发者，无论你的工作环境是 Node.js 还是 Web 浏览器，本书都非常适合你。

本书不适合从未写过 JavaScript 代码的初学者，读者们需要对这门语言的基础知识有一定的理解，这样才能发挥本书的最大效用。

本书概览

本书中的每一个章节与附录都涵盖有 ECMAScript 6 的不同方面，许多章节一开始都会讨论 ECMAScript 6 中新变化的来龙去脉，以及这些改动试图解决的问题。所有章节都包含代码示例来帮助你学习新的语法及概念。

- **第 1 章　块级作用域绑定**　讨论 var 在块级作用域中的替代方案——let 和 const。
- **第 2 章　字符串和正则表达式**　详尽介绍字符串模板，以及新增的操作与检查字符串的功能。
- **第 3 章　函数**　讨论函数的多处改动，包括箭头函数（Arrow Function）、默认参数（Default Parameters）、不定参数（Rest Parameters）等。
- **第 4 章　扩展对象的功能性**　解读对象创建、修改及使用方面的改动，包括对象字面量语法的变化、新的反射方法等。
- **第 5 章　解构：使数据访问更便捷**　介绍一种通过简明的语法分解对象和数组的方法——对象和数组解构。
- **第 6 章　Symbol 和 Symbol 属性**　介绍定义属性的新途径——Symbol。Symbol 是一种新的原始类型，可用于创建外部无法直接访问的对象属性和方法。
- **第 7 章　Set 集合与 Map 集合**　详述四种新的集合类型：Set、WeakSet、Map 及 WeakMap。这些类型为数组增添了新的语义、去重机制，以及专门为 JavaScript 设计的内存管理机制，极大地扩展了数组的实用性。
- **第 8 章　迭代器（Iterator）和生成器（Generator）**　这两个全新的功能可以协助你更有效地处理集合数据，在早期版本的 JavaScript 中无法实现这样的功能。
- **第 9 章　JavaScript 中的类**　介绍 JavaScript 中首次正式加入的类概念。接触过其他语言的开发者通常会对 JavaScript 的语法感到困惑，新增的

类语法使 JavaScript 变得更易上手，而且对热衷于 JavaScript 的开发者来说新的语法变得更加简洁。

- **第 10 章　改进数组的功能**　详述针对原生数组进行的改动，以及这些有趣的变化为开发者所带来的新体验。
- **第 11 章　Promise 与异步编程**　介绍语言的新成员——Promise。它是草根群体不断努力的结晶，由于各大 JavaScript 库的鼎立支持，这一功能逐渐被广大开发者所接受。ECMAScript 6 正式将 Promise 纳入标准并为其提供可用的 Polyfill。
- **第 12 章　代理（Proxy）和反射（Reflection）API**　介绍正式加入 JavaScript 的反射 API 和新的代理对象，开发者可以通过代理对象拦截每一个在对象中执行的操作，代理也赋予了开发者空前的对象控制权，同样也为定义新的交互模式带来无限可能。
- **第 13 章　用模块封装代码**　详述 JavaScript 的官方模块风格。加入这一定义旨在代替过去几年中出现过的许多非正式的模块定义风格。
- **附录 A　ECMAScript 6 中较小的改动**　涵盖了 ECMAScript 6 中实现的其他改动，它们与每一章所涉及的主题关系不大，一般很少使用这些功能。
- **附录 B　了解 ECMAScript 7（2016）**　描述了在 ECMAScript 7 中实现的三个附加功能，它们在近期的影响力不会像 ECMAScript 6 一样大。

排版约定

本书使用以下的排版约定：

等宽字体代码块表示较长的代码示例，如下所示：

```
function doSomething() {
    // empty
}
```

在代码块中，`console.log()`语句右侧的注释表示在浏览器或 Node.js 控制台中显示的代码执行结果，例如：

```
console.log("Hi");        // "Hi"
```

如果代码块中的某行代码引发错误，也会在代码的右侧指示：

```
doSomething();            // 抛出错误
```

帮助与支持

如果你在阅读本书时有任何疑问，请发送邮件至我的邮件列表，地址为 *http://groups.google.com/group/zakasbooks*。

轻松注册成为博文视点社区用户（www.broadview.com.cn），扫码直达本书页面。

- **下载资源：**本书如提供示例代码及资源文件，均可在下载资源处下载。
- **提交勘误：**您对书中内容的修改意见可在提交勘误处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **交流互动：**在页面下方读者评论处留下您的疑问或观点，与我们和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/31798

目录