TURING 图灵程序设计丛书 Web开发系列

Apress®

DOM Scripting

Web Design with JavaScript and the Document Object Model

Second Edition

JavaScript DOM 编程艺术（第2版）

[英] Jeremy Keith
[加] Jeffrey Sambells 著

杨涛 王建桥 杨晓云 等译
魏忠 审校

- Amazon超级畅销书最新
- 释放JavaScript和DOM编程的惊人潜力
- 涵盖HTML5及jQuery

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 Web开发系列

DOM Scripting

Web Design with JavaScript and the Document Object Model

Second Edition

JavaScript DOM 编程艺术（第2版）

[英] Jeremy Keith
[加] Jeffrey Sambells 著

杨涛 王建桥 杨晓云 等译
魏忠 审校

人民邮电出版社
北京

图书在版编目（CIP）数据

JavaScript DOM编程艺术 : 第2版 / (英) 基思(Keith,J.), (加) 桑布尔斯 (Sambells,J.) 著 ; 杨涛等译. — 北京 : 人民邮电出版社, 2011.4
(图灵程序设计丛书)
书名原文: DOM Scripting : Web Design with JavaScript and the Document Object Model,Second Edition
ISBN 978-7-115-24999-9

Ⅰ. ①J… Ⅱ. ①基… ②桑… ③杨… Ⅲ. ①JAVA语言—程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2011)第030849号

内 容 提 要

本书讲述了JavaScript、DOM和HTML5的基础知识，着重介绍DOM编程技术背后的思路和原则：平稳退化、渐进增强和以用户为中心等。这些概念对于任何前端Web开发工作都非常重要。本书将这些概念贯穿在书中的所有代码示例中，以便呈现用来创建图片库页面的脚本、用来创建动画效果的脚本和用来丰富页面元素呈现效果的脚本，最后结合所讲述的内容创建了一个实际的网站。

本书适合Web设计师和开发人员阅读。

图灵程序设计丛书

JavaScript DOM编程艺术（第2版）

◆ 著 [英] Jeremy Keith [加] Jeffrey Sambells
译 杨 涛 王建桥 杨晓云 等
审 校 魏 忠
责任编辑 傅志红
执行编辑 谢灵芝

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
中国铁道出版社印刷厂印刷

◆ 开本：800×1000 1/16
印张：18.75
字数：443千字 2011年4月第1版
印数：1－4 000册 2011年4月北京第1次印刷

著作权合同登记号 图字：01-2011-1367号

ISBN 978-7-115-24999-9

定价：49.00元

读者服务热线：(010)51095186 印装质量热线：(010)67129223

反盗版热线：(010)67171154

版 权 声 明

献给我的妻子和第一位读者 Jessica。

——Jeremy Keith

献给自始至终支持我的 Stephanie、Addison 和 Hayden。

——Jeffrey Sambells

上一版译者序

网上的生活越来越丰富多彩。从最初的(X)HTML 网页，到一度热炒的 DHTML 概念，再到近几年流行起来的 CSS，网站和网页的设计工作变得越来越简便，网上的内容越来越富于变化和色彩。但是，很多网页设计者和网民朋友都不太喜欢 JavaScript，这主要有以下几方面原因。第一，很多网页设计者认为 JavaScript 的可用性很差——早期的浏览器彼此很少兼容，如果想让自己编写出来的 JavaScript 脚本在多种浏览器环境里运行，就必须编写许多用来探测浏览器的具体品牌和具体版本的测试及分支代码（术语称之为“浏览器嗅探”代码）。这样的脚本往往到处是 `if...else` 语句，既不容易阅读，又不容易复查和纠错，更难以做到让同一个脚本适用于所有的浏览器。第二，对广大的网民来说，JavaScript 网页的可访问性很差——浏览器会时不时地弹出一个报错窗口甚至导致系统死机，让人乘兴而来、败兴而去。第三，JavaScript 被很多网站用来实现弹出广告窗口的功能，人们厌烦这样的广告，也就“恨”屋及乌地厌烦起 JavaScript 来了。第四，“JavaScript”这个名字里的“Java”往往让人们误以为其源于 Java 语言，而实际接触之后才发现它们根本没有任何联系。与 Java 语言相比，JavaScript 语言要简单得多。很多程序员宁肯钻研 Java，也不愿意去了解 JavaScript 的功能和用法。

不管什么原因，JavaScript 曾经不受欢迎的确是一个事实。

现在，情况发生了极大的变化。因为几项新技术的出现，JavaScript 的春天似乎来了。首先，W3C（万维网联盟）推出的标准化 DOM（Document Object Model，文档对象模型）已经一统江湖，目前市场上常见的浏览器可以说没有不支持的。这对网页设计者来说意味着可以用简单的“对象检测”代码来取代那些繁复的浏览器嗅探代码，而按照 DOM 编写出来的 JavaScript 页面不像过去那样容易出问题，这对网民来说意味着浏览体验变得流畅了。其次，最近兴起的 Ajax 技术以 DOM 和 JavaScript 语言（以及 CSS 和 XHTML）为基本要素，基于 Ajax 技术的网站离不开 JavaScript 和 DOM 脚本。

其实，人们对 JavaScript 的恶劣印象在很大程度上来源于早期的程序员对这种语言的滥用。如果程序员在编写 JavaScript 脚本的时候能够把问题考虑得面面俱到，就可以避免许多问题，但可惜的是如此优秀的程序员太少了。事实上，即使是在 JavaScript 已经开始流行起来的今天，如果程序员在编写 JavaScript 脚本的时候不遵守相关的标准和编程准则，也仍会导致各种各样的问题。

在 2002 年前后，CSS 也是一种不太受人们欢迎的 Web 显示语言，除了用它来改变一下字体，几乎没有人用它来干其他的事情。但没过多久，人们对利用 CSS 设计网页布局的兴趣就一发而

不可收拾，整个潮流也从那时扭转了过来。现在，掌握 CSS 已经成为许多公司在招聘网站开发人员时的一项要求。

目前，DOM 编程技术的现状与 CSS 技术在 2002 年时的境况颇有几分相似。受 Google Maps 和 Flickr 等著名公司利用 DOM 编程技术推出的 Gmail、Google Suggest 等新型服务的影响和带动，DOM 编程人才的需求正日益增加。有越来越多的人开始迷上了脚本编程技术，并开始学习如何利用 DOM 技术去改善而不是妨碍网站的可用性和可访问性。

本书的作者 Jeremy Keith 是 Web 标准计划 DOM Scripting 任务组的负责人之一，他在这本书里通过大量示例证明了这样一个事实：只要运用得当，并且注意避开那些“经典的”JavaScript 陷阱，DOM 编程技术就可以成为 Web 开发工具箱里又一件功能强大甚至是不可或缺的好东西。

本书并不是一本参考大全类型的图书，作者只重点介绍了几种最有用的 DOM 方法和属性。本书的精华在于作者在书中提到的关于 JavaScript 和 DOM 脚本编程工作的基本原则、良好习惯和正确思路。如果读者能通过书中的几个案例真正领悟这些原则、习惯和思路，就一定能让自己的编程技术再上一个台阶。

这是一本非常实用的好书，是一本值得一读再读的好书。作为本书的译者，我们相信它会让每位读者、自建网站的设计者和来到自建网站的访问者都受益匪浅。

参加本书翻译的人员还有韩兰、李京山、胡晋平、高文雅。

序

第 2 版已经出版了。

首先，我要澄清一点：虽然我的名字印在了封面上，但我并没有参与这个版本的修订工作。这个新版本完全出自 Jeffrey Sambells 之手。出版社因为出新版的事找过我，但我的时间确实安排不开了。因此，看到自己的名字忝列其间，心中不禁顿生愧意。

我很高兴地向读者朋友们报告，新版本中所有的修订都非常符合我的期望——英文原书的封面除外。但不管怎么说，第二版的内容真的是太好了！在上一版的基础上，新版经过了扩展，涵盖了如下三个新领域：

- HTML5
- Ajax
- JavaScript 库（尤其是 jQuery）

相比之下，新版的内容又扩充了不少，但整本书仍然一直在强调最佳实践（特别是渐进增强），这正是让我喜出望外的地方。

新版本中的代码示例全部换成用 HTML5 标记来写了。有关 Ajax 的示例代码也精简得当，尽管简略，但上下文仍然能够传达出我在 Bulletproof Ajax[①] 中提出的观点：永远不要假设 Ajax（或 JavaScript，等等）一定可用。

最让我高兴的一点，就是新版本增加了主要介绍 jQuery 的章节。这一章把本书前面的典型代码示例，使用 jQuery 重写了一遍。这样一来，正好解释了人们对为什么使用库的种种疑问。它让你先理解了底层代码的工作原理，然后再告诉你使用库为什么能节省时间和精力。

总而言之，这本书新增的内容都十分精彩，对读者绝对有用。为了尽量多展示一些 jQuery 的方法，也限于篇幅，这一版以介绍库的附录代替了上一版介绍 DOM 方法的附录。这多少让我感到有一些遗憾，不过，我会争取在自己的博客上公布第 1 版的附录。

最后，我还是要给第 2 版再竖竖大姆指，另外再给读者一点建议。如果你买过本书第 1 版，恐怕找一些专门讲 HTML5、Ajax 或 jQuery 的书看会比较好。但如果你就是想知道怎么才能正确地使用 JavaScript，那这个经过扩展的新版本就是你的最佳选择。

Jeremy Keith

2011 年 1 月 3 日

---

① 中文版《Bulletproof Ajax 中文版》已经由人民邮电出版社出版。——编者注

前　言

这是一本讲述一种程序设计语言的书，但它不是专门写给程序员的，而主要是写给Web设计师的。具体地说，本书是为那些喜欢使用CSS和HTML并愿意遵守编程规范的Web设计师们编写的。

本书由代码和概念两大部分构成。不要被那些代码吓倒，我知道它们乍看起来很唬人，可只要抓住了代码背后的概念，就会发现你是在用一种新语言去阅读和编写代码。

学习一种新的程序设计语言看起来可能很难，但事实却并非如此。DOM 脚本看起来似乎比 CSS 更复杂，可只要领悟了它的语法，你就会发现自己又掌握了一样功能强大的 Web 开发工具。归根结底，代码都是思想和概念的体现。

我在这里要告诉大家一个秘密：其实没人能把一种程序设计语言的所有语法和关键字都记住。如果有拿不准的地方，查阅参考书就全解决了。但本书不是一本参考大全。本书只介绍最基本的 JavaScript 语法。

本书的真正目的是让大家理解DOM 脚本编程技术背后的思路和原则，或许你对其中一部分早就熟悉了。平稳退化、渐进增强、以用户为中心的设计对任何前端 Web 开发工作都非常重要。这些思路贯穿在本书的所有代码示例中。

你将会看到用来创建图片库页面的脚本、用来创建动画效果的脚本和用来丰富页面元素呈现效果的脚本。如果你愿意，完全可以把这些例子剪贴到自己的代码中，但更重要的是理解这些代码背后的“如何”和“为什么”。

如果你已经在使用 CSS 和 HTML 来把设计思路转化为活生生的网页，就应该知道 Web 标准有多么重要。还记得你是在何时发现自己只需修改一个 CSS 文件就可以改变整个网站的视觉效果吗？DOM 技术有着同样强大的威力。不过，能力越大，责任也就越大。因此，我不仅想让你看到用 DOM 脚本实现的超酷效果，更想让你看到怎样才能利用 DOM 脚本编程技术以一种既方便自己更体贴用户的方式去充实和完善网页。

如果需要本书所讨论的相关代码[①]，到 www.friendsofed.com 网站搜索本书的主页就可以查到。还可以在这个网站找到 friends of ED 出版社的所有其他好书，内容涉及 Web 标准、Flash、DreamWeaver 以及许多细分的计算机领域。

你对JavaScript的探索不应该在合上本书时就停止下来。我开设了http://domscripting.com/网站，在那里继续与大家共同探讨现代的、标准化的JavaScript。我希望你能到该网站看看。与此同时，我更希望本书能够对大家有所帮助。祝你们好运！

---

① 本书代码示例也可从图灵网站 www.turingbook.com 本书网页免费注册下载。——编者注

致　谢

没有我的朋友和同事 Andy Budd[①]（http:www.andybudd.com）与 Richard Rutter（http:www.Clagnut.com）的帮助，本书的面世就无从谈起。Andy 在我们的家乡 Brighton 开设了一个名为 Skillswap（http:www.skillswap.org）的免费培训网站。在 2004 年 7 月，Richard 和我在那里做了一次关于 JavaScript 和 DOM 的联合演讲。演讲结束后，我们来到附近的一家小酒馆，在那里，Andy 建议我把演讲的内容扩展成一本书，也就是本书的第 1 版。

如果没有两方面的帮助，我大概永远也学不会编写 JavaScript 代码。一方面是几乎每个 Web 浏览器里都有“view source”（查看源代码）选项。谢谢你，“view source”。另一方面是那些多年来一直在编写让人叹为观止的代码并解说重要思路的 JavaScript 大师们。Scott Andrew、Aaron Boodman、Steve Champeon、Peter-Paul Koch、Stuart Langridge 和 Simon Willison 只是我现在能想到的几位。感谢你们所有人的分享精神。

感谢 Molly Holzschlag 与我分享她的经验和忠告，感谢她对本书初稿给予反馈意见。感谢 Derek Featherstone 与我多次愉快地讨论 JavaScript 问题，我喜欢他思考和分析问题的方法。

我还要特别感谢 Aaron Gustafson，他在我写作本书期间向我提供了许多宝贵的反馈和灵感。

在写作第 1 版期间，我有幸参加了两次非常棒的盛会：在得克萨斯州 Austin 举办的“South by Southwest”和在伦敦举办的@media。我要感谢这两次盛会的组织者 Hugh Forrest 和 Patrick Griffiths，是他们让我有机会结识那么多最友善的人——我从没想过我能有机会与他们成为朋友和同事。

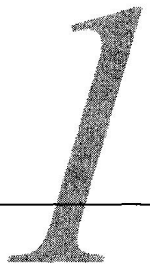
最后，我要感谢我的妻子 Jessica Spengler，不仅因为她永远不变的支持，还因为她在本书初稿的校对工作中做出的专业帮助。谢谢你，我的人生伴侣。

——Jeremy Keith

① Andy Budd 是超级畅销书《精通 CSS：高级 Web 标准解决方案（第 2 版）》的作者，该书已由人民邮电出版社出版。——编者注

目　录

第 1 章 JavaScript 简史



本章内容

- JavaScript 的起源
- 浏览器战争
- DOM 的演变史

本书第 1 版面世的时候，做一名 Web 设计师是件很让人很兴奋的事。5 个年头过去了，这个职业依然保持着强大的吸引力。特别是 JavaScript，经历了从被人误解到万众瞩目的巨大转变。Web 开发呢，也已从混乱无序的状态，发展成一门需要严格训练才能从事的正规职业。无论设计师还是开发人员，在创建网站的过程中都积极地采用标准技术，Web 标准已经深入人心。

当网页设计人员谈论起与 Web 标准有关的话题时，HTML（超文本标记语言）和 CSS（层叠样式表）通常占据着核心地位。不过，W3C（万维网联盟）已批准另一项技术，所有与标准相兼容的 Web 浏览器都支持它，这就是 DOM（文档对象模型）。我们可以利用 DOM 给文档增加交互能力，就像利用 CSS 给文档添加各种样式一样。

在开始学习 DOM 之前，我们先检视一下使网页具备交互能力的程序设计语言。这种语言就是 JavaScript，它已经诞生相当长的时间了。

1.1 JavaScript 的起源

JavaScript 是 Netscape 公司与 Sun 公司合作开发的。在 JavaScript 出现之前，Web 浏览器不过是一种能够显示超文本文档的简单的软件。而在 JavaScript 出现之后，网页的内容不再局限于枯燥的文本，它们的可交互性得到了显著的改善。JavaScript 的第一个版本，即 JavaScript 1.0 版本，出现在 1995 年推出的 Netscape Navigator 2 浏览器中。

在 JavaScript 1.0 发布时，Netscape Navigator 主宰着浏览器市场，微软的 IE 浏览器则扮演着追赶者的角色。微软在推出 IE 3 的时候发布了自己的 VBScript 语言，同时以 JScript 为名发布了 JavaScript 的一个版本，以此很快跟上了 Netscape 的步伐。面对微软公司的竞争，Netscape 和 Sun 公司联合 ECMA（欧洲计算机制造商协会）对 JavaScript 语言进行了标准化。于是出现了 ECMAScript 语言，这是同一种语言的另一个名字。虽说 ECMAScript 这个名字没有流行开来，

但人们现在谈论的 JavaScript 实际上就是 ECMAScript。

到了 1996 年，JavaScript、ECMAScript、JScript——随便你们怎么称呼它——已经站稳了脚跟。Netscape 和微软公司在各自的第 3 版浏览器中都不同程度地支持 JavaScript 1.1 语言。

注意　JavaScript 与 Sun 公司开发的 Java 程序语言没有任何联系。JavaScript 最开始的名字是 LiveScript，后来选择“JavaScript”作为其正式名称的原因，大概是想让它听起来有系出名门的感觉。但令人遗憾的是，这一选择容易让人们把这两种语言混为一谈，而这种混淆又因为各种 Web 浏览器确实具备这样或那样的 Java 客户端支持功能而进一步加剧。事实上，Java 在理论上几乎可以部署在任何环境，但 JavaScript 却倾向于只应用在 Web 浏览器。

JavaScript 是一种脚本语言，通常只能通过 Web 浏览器去完成一些操作而不能像普通意义上的程序那样独立运行。因为需要由 Web 浏览器进行解释和执行，所以 JavaScript 脚本不像 Java 和 C++等编译型程序设计语言那样用途广泛。不过，这种相对的简单性也正是 JavaScript 的长处：比较容易学习和掌握，所以那些本身不是程序员，但希望通过简单的剪贴操作把脚本嵌入现有网页的普通用户很快就接受了 JavaScript。

JavaScript 还向程序员提供了一些操控 Web 浏览器的手段。例如，JavaScript 语言可以用来调整 Web 浏览器窗口的高度、宽度和位置等属性。这种设定浏览器属性的办法可以看做是 BOM（浏览器对象模型）。JavaScript 的早期版本还提供了一种初级的 DOM。

1.2　DOM

什么是 DOM？简单地说，DOM 是一套对文档的内容进行抽象和概念化的方法。

在现实世界里，人们对所谓的“世界对象模型”都不会陌生。例如，当用“汽车”、“房子”和“树”等名词来称呼日常生活环境里的事物时，我们可以百分之百地肯定对方知道我们说的是什么，这是因为人们对这些名词所代表的东西有着同样的认识。于是，当对别人说“汽车停在了车库里”时，可以断定他们不会理解为“小鸟关在了壁橱里”。

我们的“世界对象模型”不仅可以用来描述客观存在的事物，还可以用来描述抽象概念。例如，假设有个人向我问路，而我给出的答案是“左边第三栋房子”。这个答案有没有意义将取决于那个人能否理解“第三”和“左边”的含义。如果他不会数数或者分不清左右，则不管他是否理解这几个概念，我的回答对他都不会有任何帮助。在现实世界里，正是因为大家对抽象的世界对象模型有着基本的共识，人们才能用非常简单的话表达出复杂的含义并得到对方的理解。具体到这里的例子，你可以相当有把握地断定，其他人对“第三”和“左边”的理解和我完全一样。

这个道理对网页也同样适用。JavaScript 的早期版本向程序员提供了查询和操控 Web 文档某些实际内容（主要是图像和表单）的手段。因为 JavaScript 预先定义了“images”和“forms”等术语，我们才能像下面这样在 JavaScript 脚本里引用“文档中的第三个图像”或“文档中名为

'details' 的表单"：

```
document.images[2]
document.forms['details']
```

现在的人们通常把这种试验性质的初级 DOM 称为"第 0 级 DOM"（DOM Level 0）。在还未形成统一标准的初期阶段，"第 0 级 DOM"的常见用途是翻转图片和验证表单数据。Netscape 和微软公司各自推出第四代浏览器产品以后，DOM 开始遇到麻烦，陷入困境。

1.3 浏览器战争

Netscape Navigator 4 发布于 1997 年 6 月，IE 4 发布于同年 10 月。这两种浏览器都对它们的早期版本进行了许多改进，大幅扩展了 DOM，使能够通过 JavaScript 完成的功能大大增加。而网页设计人员也开始接触到一个新名词：DHTML。

1.3.1 DHTML

DHTML 是"Dynamic HTML"（动态 HTML）的简称。DHTML 并不是一项新技术，而是描述 HTML、CSS 和 JavaScript 技术组合的术语。DHTML 背后的含义是：

- 利用 HTML 把网页标记为各种元素；
- 利用 CSS 设置元素样式和它们的显示位置；
- 利用 JavaScript 实时地操控页面和改变样式。

利用 DHTML，复杂的动画效果一下子变得非常容易实现。例如，用 HTML 标记一个页面元素：

```
<div id="myelement">This is my element</div>
```

然后用 CSS 为这个页面元素定义如下位置样式：

```
#myelement {
  position: absolute;
  left: 50px;
  top: 100px;
}
```

接下来，只需利用 JavaScript 改变 myelement 元素的 left 和 top 样式，就可以让它在页面上随意移动。不过，这只是理论而已。

不幸的是，NN 4 和 IE 4 浏览器使用的是两种不兼容的 DOM。换句话说，虽然浏览器制造商的目标一样，但他们在解决 DOM 问题时采用的办法却完全不同。

1.3.2 浏览器之间的冲突

Netscape 公司的 DOM 使用了专有元素，这些元素称为*层*（layer）。层有唯一的 ID，JavaScript 代码需要像下面这样引用它们：

```
document.layers['myelement']
```

而在微软公司的 DOM 中这个元素必须像下面这样引用：

```
document.all['myelement']
```

这两种 DOM 的差异并不止这一点。假设你想找出 myelement 元素的 left 位置并把它赋值给变量 xpos，那么在 Netscape Navigator 4 浏览器里必须这样做：

```
var xpos = document.layers['myelement'].left;
```

而在 IE 4 浏览器中，需要使用如下所示的语句才能完成同样的工作：

```
var xpos = document.all['myelement'].leftpos;
```

这就导致了一种很可笑的局面：程序员在编写 DOM 脚本代码时必须知道它们将运行在哪种浏览器环境里，所以在实际工作中，许多脚本都不得不编写两次，一次为 Netscape Navigator，另一次为 IE。同时，为了确保能够正确地向不同的浏览器提供与之相应的脚本，程序员还必须编写一些代码去探查在客户端运行的浏览器到底是哪一种。

DHTML 打开了一个充满机会的新世界，但想要进入其中的人们却发现这是个充满苦难的世界。因此，没多久，DHTML 就从一个大热门变成了一个人们不愿提起的名词，而对这种技术的评价也很快地变成了“宣传噱头”和“难以实现”。

1.4　制定标准

就在浏览器制造商以 DOM 为武器展开营销大战的同时，W3C 不事声张地结合大家的优点推出了一个标准化的 DOM。令人欣慰的是，Netscape、微软和其他一些浏览器制造商们还能抛开彼此的敌意而与 W3C 携手制定新的标准，并于 1998 年 10 月完成了“第 1 级 DOM”(DOM Level 1)。

回到刚才的例子，我们已经用 标签定义了一个 ID 为 myelement 的页面元素，现在需要找出它的 left 位置并把这个值保存到变量 xpos 中。下面是使用新的标准化 DOM 时需要用到的语法：

```
var xpos = document.getElementById('myelement').style.left
```

乍看起来，这与刚才那两种非标准化的专有 DOM 相比并没有明显的改进。但事实上，标准化的 DOM 有着非常远大的抱负。

浏览器制造商们感兴趣的只不过是通过 JavaScript 操控网页的具体办法，但 W3C 推出的标准化 DOM 却可以让*任何一种*程序设计语言对*使用任何一种*标记语言编写出来的*任何一份*文档进行操控。

1.4.1　浏览器以外的考虑

DOM 是一种 API（应用编程接口）。简单地说，API 就是一组已经得到有关各方共同认可的基本约定。在现实世界中，相当于 API 的例子包括（但不限于）摩尔斯码、国际时区、化学元素周期表。以上这些都是不同学科领域中的标准，它们使得人们能够更方便地交流与合作。如果没