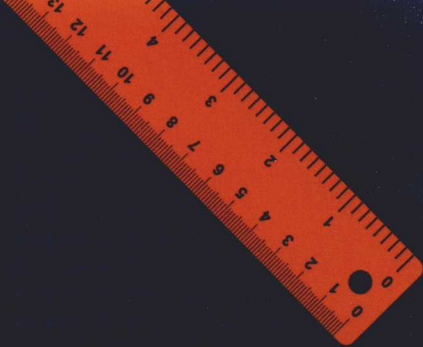




看透JavaScript

原理、方法与实践



JavaScript in Action
Principles, Methods and Practice

资深软件开发专家撰写，凝聚自己多年开发经验结晶，系统、深入、全面地阐释JavaScript，内容涉及流行的Web开发实践，结合实际案例进行讲解

韩路彪 著

清华大学出版社



看透JavaScript

原理、方法与实践

韩路彪 著



JavaScript in Action
Principles, Methods and Practice

清华大学出版社
北京

内 容 简 介

本书由资深软件开发专家撰写,凝聚其多年开发经验,系统、深入、全面地阐释 JavaScript,内容涉及流行的 Web 开发实践,结合实际案例进行讲解,授人以渔。本书内容并不局限于某个具体功能的使用方法,而是系统深入地讲解 JavaScript 的本质与结构,清晰阐释 ECMAScript、DOM 和 BOM 三部分内容的关系及重要性,以帮助开发人员全面深入了解前端开发技术。本书讲解的内容通俗易懂、深入浅出,特别是书中所用示例的设计,它们不仅可以让读者理解某个知识点的用法,更能让读者明白具体知识点所使用的场景,从而更深入地理解具体内容。

本书内容安排合理,架构清晰,注意理论与实践相结合,适合那些希望学习 Web 编程语言的初、中级程序员和希望精通 JavaScript 的程序员阅读。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

看透 JavaScript:原理、方法与实践 / 韩路彪著. —北京:清华大学出版社, 2017

ISBN 978-7-302-46794-6

I. ① 看… II. ① 韩… III. ① JAVA-语言-程序设计 IV. ① TP312.8

中国版本图书馆 CIP 数据核字(2017)第 052768 号

责任编辑:秦 健

封面设计:李召霞

责任校对:徐俊伟

责任印制:刘海龙

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印装者:三河市铭诚印务有限公司

经 销:全国新华书店

开 本:186mm×240mm 印 张:25.25 字 数:550 千字

版 次:2017 年 9 月第 1 版 印 次:2017 年 9 月第 1 次印刷

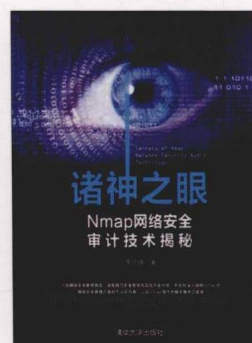
印 数:1 ~ 3000

定 价:79.00 元

产品编号:073287-01

本书内容并不局限于某个具体功能的使用方法，而是系统、深入地讲解JavaScript的本质与结构，清晰阐释ECMAScript、DOM和BOM三部分内容的关系及重要性，帮助开发人员全面深入了解前端开发技术。本书讲解的内容通俗易懂、深入浅出，特别是书中所用示例的设计，它们不仅可以让读者理解某个知识点的用法，更能让读者明白具体知识点所使用的场景，从而更深入地理解具体内容。

本书适合那些希望学习Web编程语言的初、中级程序员和希望精通JavaScript的程序员阅读。





献给

父亲韩志荣

父亲韩志荣

Preface 前言

第一次接触 JavaScript 时的场景现在已经记不清了，不过因为有其他语言的基础，我很快就上手了。当时感觉 JavaScript 就是一门不需要专门去学的语言，太简单了！

但是，随着使用的深入，才逐渐发现事实并非如此。JavaScript 既不像 C 语言这类面向过程的语言，也不像 C++、Java 这类面向对象的语言。JavaScript 中有些概念虽然在其他语言（这里主要指 C、C++ 和 Java）里也有，但是含义却不尽相同，而且 JavaScript 中还有很多其他语言所没有的用法和概念。例如，在函数体中还可以再定义另外一个函数，函数也是对象，而且有个特殊的 prototype 属性，另外也可以使用函数创建对象，以及由此引出的闭包、参数作用域链和琢磨不透的 this 指针等，后来感觉越用越不明白了。

随着近几年 B/S 结构的快速发展，JavaScript 已经成了一门不可或缺的语言。GitHub 在 2015 年 8 月 20 日发布的统计数据显示，JavaScript 是现在最受欢迎的语言。另一篇开源社区的文章更是指出，在所有 stars 超过 10 000 个的仓库中，基于 JavaScript 开发的仓库占到 55.7%！

既然离不开那就将其弄明白。经过一段时间的努力终于将其弄清楚了，而且也明白了其背后的实现原理。既然花费了不少精力，那么何不让更多的人受益，以使自己的付出更有意义呢。于是就有了大家手中的这本书。

本书的目标是帮助读者理解 JavaScript 的各种语法及底层的实现原理，进而灵活使用其所提供的内在对象，所以本书并不是针对零基础的读者。如果您要阅读本书，至少需要使用过 JavaScript，如果同时还使用过 C/C++ 或者 Java 就更好了。

本书特点

- **系统：**全书采用了总分总的结构。首先整体介绍了 JavaScript、ECMAScript、DOM、BOM 和 HTML5 之间的关系，然后依次对每一项内容进行讲解，最后进行总结。全书系统介绍了 JavaScript 的内部结构；提出了将 JavaScript 分为两种对象的思维方式；深入分析 JavaScript 中各种属性和变量的关系；系统阐述 DOM 及其节点的结构；

深入分析了 HTML5 中 6 种新增内容的使用方法。

- **全面**：全面包含整体和细节两个方面。从整体来说，包含 JavaScript 基础、ECMAScript 2015、DOM、BOM 和 HTML5 这 5 个部分，几乎涵盖了 JavaScript 的所有相关知识。细节指每一部分内容的全面性，例如，JavaScript 基础里涵盖了 this 指针、变量作用域、prototype 继承、闭包、对象与对象之间的关系等比较容易出错的内容，ECMAScript 2015（ECMAScript 6）中介绍了绝大部分新增的内容，而 HTML5 部分几乎对所介绍的每种功能都讲解了其所包含的全部方法和属性。
- **深入**：本书并不仅仅给大家介绍 JavaScript 中的各种语法，还说明了底层的实现原理，以及使用中需要注意的地方。另外，对于读者可能不熟悉的内容（例如，数据库、多线程等）会先介绍相关的概念，然后介绍 JavaScript 中的操作方法。
- **通俗易懂**：本书尽量使用通俗易懂的文字给大家介绍相关知识，避免因为一些专有名词而给大家造成理解上的困难，对于不容易理解的地方还会通过比喻或举例来帮助大家理解。

本书结构

本书一共分为 5 篇。

第一篇整体介绍 JavaScript 的结构，以及与 ECMAScript、DOM、BOM、HTML5 的关系，另外还对 ECMAScript 的语法及背后的原理进行了系统讲解。

第二篇系统介绍 ECMAScript 2015 中新增的内容。

第三篇介绍 DOM 的结构及其所包含的各种子标准。

第四篇介绍 BOM 中的 4 个对象。

第五篇介绍 HTML5 中的 6 种实用功能，并对全书进行总结。

致谢

本书的整个编写过程都离不开父亲韩志荣的支持和在背后的默默付出，这种感谢是无法言表的。另外，还要感谢清华大学出版社的秦健编辑以及他所在的团队，如果没有他们的支持和付出，本书也不可能跟大家顺利见面。

写字并不像看书那么简单，虽然笔者已经尽力了，但是由于精力和能力的原因难免还会存在表述不准确甚至不正确的地方，还请大家不吝批评指正。

Connect 目 录

第一篇 JavaScript 基础知识

第 1 章 JavaScript 怎么学 2

1.1 三种学习方法 2

1.2 JavaScript 的学习方法 3

第 2 章 JavaScript 简介 6

2.1 ECMAScript 概述 7

2.2 DOM 概述 8

2.3 BOM 概述 8

2.4 HTML5 概述 9

第 3 章 JavaScript 的本质 10

3.1 工具介绍 10

3.2 JavaScript 的结构 15

3.3 JavaScript 的内存模型 17

第 4 章 function 类型对象 20

4.1 创建 function 20

4.2 用作对象 23

4.3 处理业务 23

4.4 创建对象 49

4.5 三种子类型 57

第 5 章 object 类型对象 62

5.1 创建 object 类型对象的三种方式 62

5.2 对象的属性 66

5.3 属性的描述 74

第 6 章 直接量及其相关对象 79

6.1 直接量的保存方式 79

6.2 直接量的封包与解包 80

6.3 直接量的包装对象 81

第 7 章 点运算符与 this 关键字 90

7.1 点运算符 90

7.2 this 的含义 90

7.3 关联方法后的 this 91

7.4 内部函数中的 this 91

7.5 对象的属性不可以继承 93

第 8 章 Global 与 Window 对象 94

8.1 Global 对象 94

8.2 Window 对象的特殊性 94

第二篇 ECMAScript 2015 中的新特性

第 9 章 类.....98

9.1 新类型 class98

9.2 class 的用法98

第 10 章 模块.....100

10.1 模块的基本用法100

10.2 导入后重命名101

10.3 默认导出102

10.4 导入为命名空间103

10.5 打包导出103

第 11 章 新增语法105

11.1 let 和 const105

11.2 字符串模板106

11.3 函数参数默认值106

11.4 参数扩展107

11.5 箭头函数107

11.6 for-of 遍历108

第 12 章 新增对象111

12.1 Symbol111

12.2 容器对象112

12.3 缓存对象121

12.4 异步处理128

12.5 GeneratorFunction 和 Generator132

12.6 反射与代理134

第 13 章 原有对象的新增属性140

13.1 Object140

13.2 String143

13.3 Number150

13.4 Array150

13.5 Math154

13.6 RegExp155

第三篇 DOM

第 14 章 DOM 概述162

14.1 DOM 标准的结构162

14.2 DOM 标准的特点163

第 15 章 DOM 核心164

15.1 文档对象164

15.2 异常对象177

15.3 辅助对象178

第 16 章 DOM 中的 HTML 标准182

16.1 HTMLCollection182

16.2 HTMLOptionsCollection184

16.3 HTMLDocument184

16.4 HTMLElement186

第 17 章 DOM 事件187

17.1 事件流187

17.2 EventTarget189

17.3 EventListener190

17.4 Event190

17.5 DOM0 级事件199

第 18 章 DOM 样式和视图201

18.1 Views201

18.2	StyleSheets	202
18.3	CSS	204
第 19 章 DOM 遍历和范围		
19.1	遍历	215
19.2	范围	219

第四篇 BOM

第 20 章 window 对象		
20.1	窗口、框架相关属性	228
20.2	窗口操作相关属性	230
20.3	弹出窗口相关属性	231
20.4	浏览器对象相关属性	231
20.5	定时器相关属性	231
20.6	其他属性	234
第 21 章 location 对象		
21.1	8 个读写属性	235
21.2	三个方法属性	237
第 22 章 其他对象		
22.1	history 对象	238
22.2	navigator 对象	238

第五篇 HTML5

第 23 章 本地存储		
23.1	本地存储的分类	242
23.2	Storage 存储	243
23.3	StorageEvent	245
23.4	IndexedDB 数据库	248

第 24 章 canvas 作图		
24.1	绘制矩形	272
24.2	绘制路径	274
24.3	组合与剪切	282
24.4	坐标检测	285
24.5	修改颜色和样式	286
24.6	插入文本	290
24.7	插入图片	292
24.8	环境的保存和恢复	292
24.9	移动坐标原点	293
24.10	旋转坐标系	294
24.11	缩放	295
24.12	自由转换	297
24.13	阴影	298
24.14	动画	299
24.15	游戏	300

第 25 章 WebSocket		
25.1	网络传输的原理和底层协议	306
25.2	WebSocket 简介	307
25.3	WebSocket 的优势	308
25.4	基本应用	308
25.5	RESTful 应用	312
25.6	在线聊天系统	313
25.7	传输对象和二进制数据	318

第 26 章 多线程处理		
26.1	单线程与多线程的概念	335
26.2	线程与进程的概念	337
26.3	JavaScript 中的多线程	338

第 27 章 获取位置信息 347

- 27.1 geolocation 对象 347
- 27.2 Position 接口 348
- 27.3 PositionError 接口 349
- 27.4 PositionOptions 接口 350
- 27.5 显示到地图上 353

第 28 章 富文本编辑器与公式编辑器 356

- 28.1 富文本编辑器的原理 356

28.2 公式编辑器 364**第 29 章 总结** 380

- 29.1 ECMAScript 380
- 29.2 DOM 和 BOM 381
- 29.3 HTML5 382

附录 ExcelibIDB 源代码 384

第一篇

JavaScript 基础知识

第1章介绍学习 JavaScript 的方法。磨刀不误砍柴工，好的学习方法可以让学习更加轻松。

第2章对 JavaScript 做了整体介绍，包括 JavaScript 的历史，ES、DOM、BOM 的概念等内容。这一章大致了解一下即可。

第3章介绍了所用到的工具以及 JavaScript 的本质和结构。这一章的内容非常重要，请读者认真学习，仔细琢磨。纲举目张，这一章就是 JavaScript 的纲。

第4~6章分别介绍了JavaScript中的三类对象属性,包含prototype属性对象、函数执行的原理、参数的作用域链、闭包、对象的不同类型属性等很多重要内容。

第7章介绍点运算符和 this 关键字。这是很多 JavaScript 开发者，特别是新手容易困惑的地方。

第 8 章介绍 Global 和 Window 对象。这一章对于理解 JavaScript 整体的对象结构非常重要。

第 1 章 JavaScript 怎么学

学习方法对于学习，就像图纸对于生产加工一样重要。如果没有好的学习方法，就像加工零件没有图纸，学习的效率也不会很高。对于学习 JavaScript 来说更是如此。

1.1 三种学习方法

在介绍学习方法之前请大家先看一个示例。记得我在小的时候经常看见大人查看地图，于是我自己也装模作样地看。那时候我是在地图上随便找个地方就看，看那一片都有哪些名字，有几条线，每条线都是什么走向。不过这样看过之后觉得跟没看差不多，而且看的时间越长反而记住的越少。

稍长后我明白了地图是用来查的，去什么地方之前拿出来查一查就知道怎么走了。不过这么看的时候脑子里记住的还是各种线，路线长的时候就记不住了，而且容易搞混。

再后来才知道看地图还有更好的方法。首先要弄明白为什么要看地图。看地图的目的一般来说可以分为两大类：找位置和找路线。而要想完成这两个目的，最好先在脑子里面建立一套完整的框架，这样找起来就容易了。对于位置来说，应该先看整张图可以分为几块。例如，对于我的家乡长子县的地图来说，就应该看划分为多少个乡镇，以及每个乡镇的大概形状和位置关系，然后对每个乡镇仔细查看和学习，这样各个乡镇的位置之间就有了层级关系，再找某个具体位置的时候就容易多了。另外，还可以看长子县在市级行政区域中所处的位置。例如，在长子县标准地图的左上角可以看出长子县位于晋东南地区，这样在地图上查找位置就容易多了。对于路线也是一样的道理，首先要看都有哪些主干线，每条线的大致走向、通过哪些乡镇，然后再看支线、支线的支线……这样看起来就比较系统了，脑子里存储的不是一条一条线而是一个有层次结构的树，这时候即使没走过的路线也能“猜”得八九不离十。

上面所说的只是单纯看地图，如果想真正看透还需要查阅更多资料，甚至亲自走一趟，有些东西只通过地图是看不出来的。例如，长子县的标准地图中有个“常张乡”，其中的第二个字如果从文字表面来看应该读“zhāng”，不过本地人实际上将其读作“cháng”。再例如，

“长子县”的第一个字应该读“cháng”还是“zhǎng”呢？实际应该读“zhǎng”，是大儿子的意思，据说是尧王长子丹朱的封地，这些通过地图是看不出来的。一张小县城的地图都这么复杂，更何况市地图、省地图、国家地图甚至世界地图了。虽然地图不一样，但是看地图的方法都是相通的。

1.2 JavaScript 的学习方法

“治大国若烹小鲜”，按照中国的传统文化来讲应该是一通百通。虽然上文介绍的是看地图的方法，但是同样适用于 JavaScript 以及其他知识和技术的学习。

对于学习 JavaScript 来说，大致也有跟看地图类似的三种学习方法。第一种是先找到像字典一样的 JavaScript 资料，然后从头到尾一点一点看；第二种是在遇到问题后直接寻找解决方案；第三种是先整体思考 JavaScript 是什么、怎么实现的及学习 JavaScript 的目的，然后再有目标地进行系统学习，最后在脑子里建立起一个整体的框架，并且弄明白每个地方的实现原理，这样就差不多了，当然更重要的是多实践多总结。

对于前两种方法，本书就不多说了。第三种方法虽然看起来很复杂，但是只要大家跟着本书来学习，应该会觉得很轻松，而且在学习完之后对 JavaScript 的认识应该也会有质的提升，再使用 JavaScript 的时候会得心应手。

对于第三种学习方法来说，首先要明白为什么需要 JavaScript（或者说 JavaScript 是怎么提出来的）、JavaScript 是什么、有什么用，以及它的结构是什么样的？对于这些问题本书将在接下来的两章中给大家介绍。这里先给大家介绍 JavaScript 的三点特性，让大家对 JavaScript 有一个整体的认识。

1.2.1 JavaScript 是一种面向对象的语言

要理解面向对象首先要明白什么是对象。网页中的一段文字、一个文本框、一张图片、一个样式表规则以及浏览器的导航器等都是对象。另外，JavaScript 中也有为了方便操作自身提供的对象，而且可以自定义对象，JavaScript 中的函数其实也是一种对象。JavaScript 的目的就是要操作这些对象，例如，“把某个文本框的内容清空”，这就是对这个文本框对象进行操作，再例如“返回上一页”就是对导航器这个对象进行操作。只不过 JavaScript 不是中国人开发的语言，所以它并不认识上面的语句，要想操作某个对象，就必须使用 JavaScript 所规定的语句才行。从这里可以看出学习 JavaScript 主要包含以下三方面内容。

□ JavaScript 怎么操作对象。

□ JavaScript 中都包含哪些对象，每一类对象都有些什么功能。

□ 不同对象之间是什么关系。

1.2.2 JavaScript 是一种脚本语言

脚本语言的功能是修改或者称为“操作”，而不是创造，所以，JavaScript 并不能从无到有创建出来一个页面，而只能是对页面进行修改（有的读者可能会觉得 JavaScript 可以在一个空白页面上随意写内容，但那只是对页面进行修改，页面本身是由浏览器创建的，打开新页面也是类似的道理，如果没有浏览器，JavaScript 自己是创建不出来页面的）。因此，JavaScript 并没有类似于其他语言的入口函数 main，真正的入口函数 main 在浏览器程序中。

脚本语言是一堆命令的集合，一般来说会有一个解释器，由其负责从头到尾一条一条语句进行解释，然后根据解释后的语句含义进行操作。例如“给页面内所有文本框添加一个内容变化监听函数 XXX”，如果解释器可以理解这句话，那么它会找到所有文本框，然后添加监听函数，这就叫解释执行。它是由解释器将内容翻译后根据其含义进行相关操作的，即操作最终是由解释器来完成的，而不是将脚本编译为机器码来执行的。

JavaScript 是一种比较复杂的脚本语言，它跟编译型语言一样也有自己的变量、函数，其执行过程跟编译器一样首先生成语法树，然后解释器生成一条一条的中间码（类似于“给页面内所有文本框添加一个内容变化监听函数 XXX”这样的语句，当然不会是中文，甚至不一定使用文字，只要描述出来含义就可以），最后一条一条执行。

当然，随着前端的功能越来越复杂，如果纯粹按照上述方法来解释执行就会严重影响效率，所以后来有的解释器（这里严格来说已经不能称为解释器了，现在通用的叫法为 JavaScript 引擎），例如 JavaScriptCore、SpiderMonkey 会将使用得比较频繁的代码直接编译为 CPU 所执行的机器码，有的引擎（例如 V8）甚至会将所有 JavaScript 代码全部编译为机器码，这样就类似于编译型语言了。不过，JavaScript 引擎还是会在加载网页的时候直接下载源代码，然后编译，最后执行，这对编译速度的要求就更高了。对于我们来说，并不需要过分关注其底层是怎么实现的，只需要明白 JavaScript 脚本是用于表述对各种对象怎么操作的一种描述就可以了。

1.2.3 JavaScript 是一种事件驱动的语言

事件驱动是指 JavaScript 引擎并不是在看到代码之后就会立即执行，而是会在合适的时间才去执行。这个合适的时间是指当某个事件发生之后（例如一个输入框的内容发生了变化，这就是一个事件）。只有当相应的事件发生了之后，相应的操作才会执行，这就是事件驱动。事件驱动在我们的日常生活中也是比较常见的一种模式。例如，银行的营业员处理业务就是一种事件驱动的模式，只有客户到来的时候才需要提供服务，客户到来就是一个事件。

事件驱动包含三个关键内容：事件、事主和处理方法。对于上述银行的例子来说，客户要办理的业务是事件，营业员是事主，而具体每项业务怎么办理就是处理方法。例如，一个

客户找到营业员甲办理存款业务，那么事件是存款，事主是营业员甲，处理方法是营业员具体办理存款的操作，而客户只是触发了这一事件。对于每个营业员来说，在上岗前都需要进行培训，培训在事件模型里就是绑定事件。例如，营业员甲负责存取款业务，营业员乙负责开户办卡业务，这就相当于营业员甲绑定了存取款事件，营业员乙绑定了开户办卡事件，对于没有绑定的事件是无法处理的，就像找营业员甲开户是不可以的。另外，对于同一事件不同的营业员的处理方式也可能不一样，就像同样是开户，但是不同银行的营业员的开户方式可能就不一样。

在事件模型中，我们所要做的就是给需要处理事件的事主绑定处理方法，就像给营业员进行业务培训一样，绑定完事件之后，其他的事情就不需要我们参与了。虽然底层的实现不需要我们来参与，但是明白了底层的实现原理，可以让我们对事件驱动理解得更加深刻。

事件驱动模型在底层一般都是通过队列来实现的，这与在银行窗口前排队差不多。当发生一个事件时就会将其排入队列中，其中写清楚事件是什么、事主是谁，然后事件管理器定时查看队列，如果队列中有事件，那么事件管理器就会找跟事主相关的此类事件的处理方法，如果找不到，该事件就会被丢弃，如果找到就会执行相应的处理方法。这跟银行的排队稍微有点区别，银行的排队是直接排在各自事主（营业员）前面，而事件驱动模型中所有的事件都排在一个队列里，这就像到银行办业务的人都排成一个长队（类似于取号排队，但是不分业务，全部排在一起的那种），然后大堂经理负责安排业务，排队的人需要告诉大堂经理准备到哪个窗口、办理什么业务。当然，直接按照银行的排队模型来实现从技术上来说并非不可以，而且在有的地方确实就是这么实现的，这只是业务和设计的问题。另外，这里所说的只是原理，具体怎么实现还要看开发者是如何设计的，例如，也可以直接将事件处理方法的地址保存到队列中。

理解了事件模型，我们就能明白 JavaScript 中的代码虽然是用于描述怎么操作对象的，但是并不一定要立即操作对象。

现在从宏观上明白这三点就可以了，至于 JavaScript 具体是怎么描述对对象的操作、怎么创建对象、自身包含哪些内置对象、对象之间有些什么关系等内容，随后整本书将进行详细介绍。大家带着这些问题去学习接下来的内容，这样学起来会更加轻松，而且也能够理解得更加透彻。

当然，这里介绍的学习方法，接下来还需要大家自己去使用、实践，只有这样才能真正受益，否则仅是学了一种方法而已。例如，虽然笔者知道地图怎么看，但是因为没在这上面花太多精力，所以对很多具体的地图并不熟悉。