

《JavaScript启示录》

图书基本信息

书名：《JavaScript启示录》

13位ISBN编号：9787115334947

10位ISBN编号：7115334943

出版时间：2014-3-1

作者：Cody Lindley

页数：126

译者：徐涛

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu111.com

《JavaScript启示录》

内容概要

javascript是web开发人员必须掌握的一门编程语言，但javascript语言及其相关技术正在变得越来越复杂。如何掌握javascript的基本概念和核心技术，往往让初学者和javascript新手感到束手无策。

《javascript启示录》力图在有限的篇幅内，通过考察原生javascript对象和所支持的细微差别，来给读者展现准确的javascript世界观，涉及对象、属性、复杂值、原始值、作用域、继承、this关键字、head对象等重要概念。本书帮助读者厘清这些概念，进而掌握应用它们的技术和技巧。

《javascript启示录》适合希望通过深入了解javascript对象来巩固对语言理解的高级初学者或中级javascript开发人员阅读，也适合准备研究javascript幕后知识的javascript库使用老手参考。

作者简介

cody lindley是一名客户端工程师（也称为前端开发人员）及flash开发者。他在html、css、javascript、flash、客户端性能技术方面有丰富的工作经验（11年以上），这些技术都与web开发有关。如果他不在编写客户端代码，就很可能是在研究界面/交互设计，或是在准备创作材料以及在各种会议上要用的演讲稿。当他不坐在电脑前时，他肯定正和他的妻子和孩子在爱达荷州的博伊西游玩，他们在博伊西可以进行三项全能运动训练、滑雪、骑山地自行车和公路自行车、登山、阅读、看电影或者讨论基督教世界观的理论依据。

徐涛（网名：汤姆大叔；微博：@tomxutao），微软最有价值专家（mvp）、项目经理、软件架构师，擅长大型互联网产品的架构与设计，崇尚敏捷开发模式，熟悉设计模式、前端技术、以及各种开源产品，曾获mcp、mcse、mcdba、mcts、mcsit、mcpd、pmp认证。《javascript编程精解》译者，博客地址：<http://www.cnblogs.com/tomxu>。

书籍目录

- 《javascript启示录》
- 第1章 javascript对象 1
 - 1.1 创建对象 1
 - 1.2 javascript构造函数构建并返回对象实例 6
 - 1.3 javascript原生/内置对象构造函数 7
 - 1.4 用户自定义/非原生对象构造函数 8
 - 1.5 使用new操作符实例化构造函数 10
 - 1.6 从构造函数创建字面量值 11
 - 1.7 原始值(或简单值) 13
 - 1.8 null、undefined、“string”、10、true和false等原始值不是对象 14
 - 1.9 如何存储和复制原始值 15
 - 1.10 原始值比较采用值比较 16
 - 1.11 原始值(string、number、boolean)在被用做对象时就像对象 17
 - 1.12 复杂值(或组合值) 18
 - 1.13 如何存储或复制复杂值 19
 - 1.14 复杂对象比较采用引用比较 20
 - 1.15 复杂对象具有动态属性 21
 - 1.16 typeof操作符 21
 - 1.17 动态属性支持易变对象 23
 - 1.18 构造函数实例都拥有指向其构造函数的constructor属性 24
 - 1.19 验证对象是否是特定构造函数的实例 26
 - 1.20 构造函数创建的实例可拥有自己独立的属性(实例属性) 27
 - 1.21 javascript对象和object()对象 28
- 第2章 对象与属性 29
 - 2.1 复杂对象可以将大多数javascript值作为属性 29
 - 2.2 封装复杂对象 30
 - 2.3 用点表示法或中括号表示法获取/设置/更新对象属性 31
 - 2.4 删除对象属性 34
 - 2.5 如何解决对象属性的引用 34
 - 2.6 使用hasownproperty验证对象属性不是来自原型链 37
 - 2.7 使用in操作符检查一个对象是否包含给定属性 37
 - 2.8 使用for in循环枚举(循环遍历)对象的属性 38
 - 2.9 宿主对象与原生对象 39
 - 2.10 使用underscore.js增强及扩展对象 40
- 第3章 object() 43
 - 3.1 object()对象概要 43
 - 3.2 object()参数 44
 - 3.3 object()属性和方法 45
 - 3.4 object()对象实例属性和方法 45
 - 3.5 使用对象字面量创建object()对象 46
 - 3.6 所有对象都继承自object.prototype 47
- 第4章 function() 49
 - 4.1 function()对象概要 49
 - 4.2 function()参数 50
 - 4.3 function()属性和方法 50
 - 4.4 function对象实例属性和方法 51
 - 4.5 函数总有返回值 51

- 4.6 函数是“一等公民”(不仅语法, 还有值) 52
- 4.7 函数的参数传递 53
- 4.8 this和arguments适用于所有函数 53
- 4.9 arguments.callee属性 54
- 4.10 函数实例的length属性和arguments.length 55
- 4.11 重定义函数参数 55
- 4.12 代码执行完成前取消函数执行 56
- 4.13 定义函数(语句、表达式或构造函数) 57
- 4.14 调用函数[函数、方法、构造函数或call()和apply()] 57
- 4.15 匿名函数 59
- 4.16 自调用的函数表达式 59
- 4.17 自调用的匿名函数语句 59
- 4.18 函数可以嵌套 60
- 4.19 给函数传递函数, 从函数返回函数 61
- 4.20 函数定义之前调用(函数提升) 61
- 4.21 函数可以调用自身(递归) 62
- 第5章 head/全局对象 64
 - 5.1 head/全局对象概要 64
 - 5.2 head对象内的全局函数 65
 - 5.3 head对象与全局属性、全局变量 65
 - 5.4 引用head对象 67
 - 5.5 head对象是隐式的, 通常不显式引用 67
- 第6章 this关键字 69
 - 6.1 this概要及this如何引用对象 69
 - 6.2 如何确定this值 70
 - 6.3 在嵌套函数中用this关键字引用head对象 71
 - 6.4 充分利用作用域链研究嵌套函数问题 73
 - 6.5 使用call()或apply()控制this值 73
 - 6.6 在用户自定义构造函数内部使用this关键字 75
 - 6.7 原型方法内的this关键字引用构造函数实例 75
- 第7章 作用域和闭包 77
 - 7.1 javascript作用域概要 77
 - 7.2 javascript没有块作用域 78
 - 7.3 在函数中用var声明变量, 避免作用域陷阱 78
 - 7.4 作用域链(词法作用域) 79
 - 7.5 作用域链查找返回第一轮值 81
 - 7.6 函数定义时确定作用域, 而非调用时确定 81
 - 7.7 闭包是由作用域链引起的 82
- 第8章 函数原型属性 84
 - 8.1 原型链概要 84
 - 8.2 为何要关注prototype属性 85
 - 8.3 原型在所有function()实例上都是标准的 85
 - 8.4 默认的prototype属性是object()对象 86
 - 8.5 将构造函数创建的实例链接至构造函数的prototype属性 87
 - 8.6 原型链的最后是object.prototype 88
 - 8.7 原型链返回在链中找到的第一个匹配结果 88
 - 8.8 用新对象替换prototype属性会删除默认构造函数属性 89
 - 8.9 继承原型属性的实例总是能够获得最新值 90
 - 8.10 用新对象替换prototype属性不会更新以前的实例 91

- 8.11 用户自定义构造函数像原生构造函数一样原型继承 92
- 8.12 创建继承链 94
- 第9章 array() 95
 - 9.1 array()对象概要 95
 - 9.2 array()参数 96
 - 9.3 array()属性和方法 96
 - 9.4 数组对象实例属性和方法 96
 - 9.5 创建数组 97
 - 9.6 数组添加及更新 98
 - 9.7 长度与索引 99
 - 9.8 定义预定义长度的数组 100
 - 9.9 可以通过设置数组长度添加或删除值 100
 - 9.10 数组包含数组(多维数组) 101
 - 9.11 遍历数组 101
- 第10章 string() 103
 - 10.1 string()对象概要 103
 - 10.2 string()参数 104
 - 10.3 string()属性和方法 104
 - 10.4 字符串对象实例属性和方法 104
- 第11章 number() 106
 - 11.1 number()对象概要 106
 - 11.2 整数和浮点数 106
 - 11.3 number()参数 107
 - 11.4 number()属性 108
 - 11.5 数字对象实例属性和方法 108
- 第12章 boolean() 109
 - 12.1 boolean()对象概要 109
 - 12.2 boolean()参数 109
 - 12.3 boolean()属性和方法 110
 - 12.4 布尔对象实例属性和方法 110
 - 12.5 非原始false布尔对象转换为true 111
 - 12.6 某些值是false，其他都是true 111
- 第13章 使用原始值：字符串、数字和布尔值 113
 - 13.1 访问属性时原始值/字面量值被转换为对象 113
 - 13.2 通常应使用原始字符串、数字和布尔值 115
- 第14章 null 116
 - 14.1 null值概要 116
 - 14.2 typeof(null)的返回值为“object” 116
- 第15章 undefined 118
 - 15.1 undefined值概要 118
 - 15.2 在全局作用域中定义undefined变量 119
- 第16章 math函数 120
 - 16.1 内置math对象概要 120
 - 16.2 math属性和方法 120
 - 16.3 math不是构造函数 122
 - 16.4 math常数无法增大/改变 122
- 附录a 回顾 123
- 附录b 总结 126

精彩短评

- 1、 javascript知识巩固 阅读前需要有一定的javascript基础
- 2、 书很薄，很快就翻完了，感觉没啥营养，尤其是在我看完the good parts之后。
- 3、 主题就是 JavaScript 中的对象，简述对象的特点、作用域链、原型链等概念。
- 4、 一本基础的书
- 5、 因为大叔译的，果断买来读一读~其实作用并不大，还没有大叔自己博客的内容质量高
- 6、 全是干货
- 7、 经典书籍，需要有一点基础，但是翻译是渣渣
- 8、 中文看得很吃力，减一星。
- 9、 去年读的，很快能读完，当时觉得特别好，对于刚接触js面向对象的，可以很快理解，着眼于ES3的细节。。。现在看依然不过时
- 10、 我不知道是不是原书写得就不通顺，还是翻译的有问题，总之读起来就让你觉得别扭。要么直接出影印，要么多给译者点报酬，也好让人家用点心。PS. 这本书译者的其它译作反映也不好，以后留神吧。
- 11、 书很简洁，花了一个半天看完，其中部分章节把我感到困惑的一些问题都阐述得很清楚，感觉很受用。
- 12、 内容相对来说比较初级。
- 13、 对象角度下的js，内容正好是想要了解的。
- 14、 作者非常厉害，用最简单的方式解释了看起来很复杂的prototype。让我这种JS初学者都能一下看懂。
- 15、 js博大精深。
- 16、 前面几张还行，本身就不是很全面详细的js书籍，有点像备忘录，看过全书高程和精粹再看忍者秘籍，看这个基本上没有新的知识点了，就当是对之前看的书知识点的梳理，总得来说还是比较短小精悍的，要是能把后面三四章省略更能体现精炼特色，本身后面三四章也没什么内容。
- 17、 言简意赅，cheatsheet类型的书
- 18、 对于有一点js基础的人来说，选择这本来进阶是一本好书。
- 19、 又一本被 **徐涛** @TomXuTao 糟蹋的好书，另一本是 *Secrets of the JavaScript Ninja* 我想说作为译者如果没时间去把书的知识先搞懂，那就不要翻译了，查单词用词典就足够了另外排版也是好多错乱的（空格、表格、代码的排版）
- 20、 偏简单，都是最基本的知识，如果了解js，建议不要看了。
- 21、 入门级读物
- 22、 前几天刚被坑，怒刚一波js。
- 23、 都是知识点的罗列，看了一遍。但是我对js的语法还不是很熟悉，写一段时间，还要再回来翻一翻。
- 24、 书很薄，this、作用域这些讲得清晰明了，后面几章没什么内容，全是方法、属性的罗列
- 25、 半天就可以看完,纯干货,虽然只有几十页,但对巩固基础很有帮助,值这个价
- 26、 讲解到位，并且点到即止，不啰嗦的好书，在我看来可以和Effective Javascript, JavaScript the good parts这样的书归为一类，更适合已经入门了的JS学习者。如果只粗读一遍有点浪费，值得反复翻。
- 27、 Javascript，类Java的脚本语言，对象为王
- 28、 2016.18就是大量js知识点的罗列 不深入 推荐翻翻吧 还不错
- 29、 罗列的知识点，很多，但都是些细节，都很基础
- 30、 没一会儿就读完了
- 31、 洗脑之作
- 32、 不错的书呀 很薄但都是值得细细读一下
- 33、 快速过了一下JavaScript“基础”，扫清了过去两年的部分疑惑，印证了曾经总结出来的经验。
- 34、 没有什么惊艳之处（读过《JS模式》之后。。真是一本看尽长安花
- 35、 基本

36、巩固知识

精彩书评

- 1、相当给力，把各种小知识点很好梳理了一边。相当给力，把各种小知识点很好梳理了一边。相当给力，把各种小知识点很好梳理了一边。相当给力，把各种小知识点很好梳理了一边。相当给力，把各种小知识点很好梳理了一边。
- 2、1.书中都是精华，很喜欢这种很薄的书2.列举大量知识点，都是些应该知道的知识点3.但都比较简单，看过犀牛书应该就不用买这本了4.看代码就可以了，几乎不需要看内容5.看了3天终于看完了，推荐大家阅读
- 3、这本书主要讲了对象的特点、对象与原始值之间的关系、作用域、原型、this关键字.....总之就是围绕着「对象」这一概念展开的。对我来说这本书的内容基本没有什么干货，说的那些90%以上都已经知道并掌握。不过，正像封底所说的——「本书将会帮助你巩固对JavaScript的理解」。在看书的同时，对已掌握的知识进行了梳理，现在更加系统化、体系化了。该译本有很多翻译的错误，还有一些漏掉、不通顺的地方，容易给基础不那么扎实的人造成误解、困扰！
- 4、JS对象JS几乎所有东西都是对象或用起来像对象摘要自定义构造函数，保持名称第一个字母大写字面量表达式创建对象{};[]创建的是对象1,'string',true,创建的是原始值new关键字调用构造函数的语法来创建对象new Number()new String()new Boolean() 创建的是对象Number()String()Boolean() 没有new关键字的，则创建的是原始值null undefined是原始值原始值 按值复制；对象 按引用存储，===比较是否指向同一个对象构造函数实例都拥有指向其构造函数的constructor属性，instanceof只适用于构造函数创建返回的复杂对象和实例相较于点，中括号可用来访问特殊名的属性in可坚持一个对象的属性，包括原型链；hasOwnProperty()可检查来自非原型链属性的对象Chapter4 函数函数返回值默认undefined，构造函数返回值为对象实例函数是对象，因为可有属性，是个值（最终返回值）this arguments适用于所有函数，argums.callee函数定义有三种函数构造函数new Function()函数语句function a(){}函数表达式var a = function(){}调用函数，call(),apply()区别是参数传递的不同，前者传递多个分开的参数，后者传递多个参数组成的数组自调用的函数表达式，若要立即调用函数，需在函数外面的圆括号，或任何将函数转换为表达式的符号。(function (){}()),(function (){}())!,function (){}(), var a = function (){}()function (){}()不会立即执行函数表达式没有被提升，只有函数语句被提升，即函数定义前可被调用Chapter5 head/全局对象head对象，包含所有对象的对象浏览器是window，this全局对象，全局属性是直接包含在head对象内不的值显示引用head，性能代价高，若只依靠作用域链，避免显示引用head会更快Chapter6 thisthis在函数内部使用，引用包含函数的对象自定义构造函数：new，this指对象实例；若没有new，this是上下文call apply，控制this值this的宿主函数北封装在另一个函数内或另一个函数的上下文中被调用，this永远是对head对象的引用；可在父函数that保留this引用构造函数的prototype属性的函数中使用this，this引用调用方法的实例Chapter7 作用域&闭包作用域是执行代码的上下文，js作用域包括全局作用域，函数作用域，eval作用域作用域链（词法作用域）：包含函数的函数，会创建堆栈执行作用域，这些链接在一起的栈称为作用域链；var vs 全局作用域函数定义时确定作用域，而非调用时确定，所以又叫词法作用域闭包：让函数向全局作用域返回一个嵌套函数，但该函数仍能通过作用域链访问其父函数的作用域；闭包是由作用域链引起的作用域链式基于代码的编写方式创建的，而不是基于调用函数所在的上下文，这使得函数即使从一个不同的上下文调用函数，也能够访问最初编写代码时所在的作用域，这称为闭包。Chapter8 函数原型属性原型链 最后是Object.prototype默认的prototype属性是Object()对象myArray.proto或myArray.constructor.prototype引用Array.prototype用新对象替换prototype属性回删除默认构造函数属性， constructorArray.prototype={},myArray.constructor===Object();所以要Array.prototype={constructor:Array};用新对象替换prototype属性不会更新以前的实例：Array.prototype={a:1}不会更新；Array.prototype.a=2会更新原型继承链，实例化想要继承的对象，将该对象实例作为要继承该对象实例的函数的prototype属性值，eg:var Person = function () {} var Chef = function () {} Chef.prototype = new Person(); var cody = new Chef()Chapter12 Boolean()Boolean(0 -0 null false NAN undefined “ ”)都是false非原始false布尔对象转换为true，eg:Boolean('false')//true var falseValue=new Boolean(false); if(falseValue){console.log('falseValue is truthy');}Others直接在原始数字上访问属性时，该值被当作对象之前必须进行评估，1..toString(), 第一个点被认为是数字十进制的小数点；临时包装器对象null可表明属性拥有一个空值，等待赋值；typeof null返回object,===null 来判断，==无法区分null

《JavaScript启示录》

和undefinedundefined告诉你有东西丢失了；永远不要将一个值设置为undefined，若制定一个属性或变量值不可用，应为null声明的变量未指定值试图访问的对象属性未定义，并且不存在于原型链undefined in this全局作用域定义undefined

5、书不错，不是入门的书，需要至少知道一些JS的东西，薄薄的一本把很多东西讲的非常明白，比如this关键字、闭包、作用域但是翻译的不好，有能力的还是推荐去看原版吧。紫薯布丁紫薯布丁紫薯布丁紫薯布丁

《JavaScript启示录》

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:www.tushu111.com