

畅销书《JavaScript高级程序设计》作者Nicholas C. Zakas最新力作
《JavaScript启示录》和《jQuery Cookbook》作者Cody Lindley作序推荐

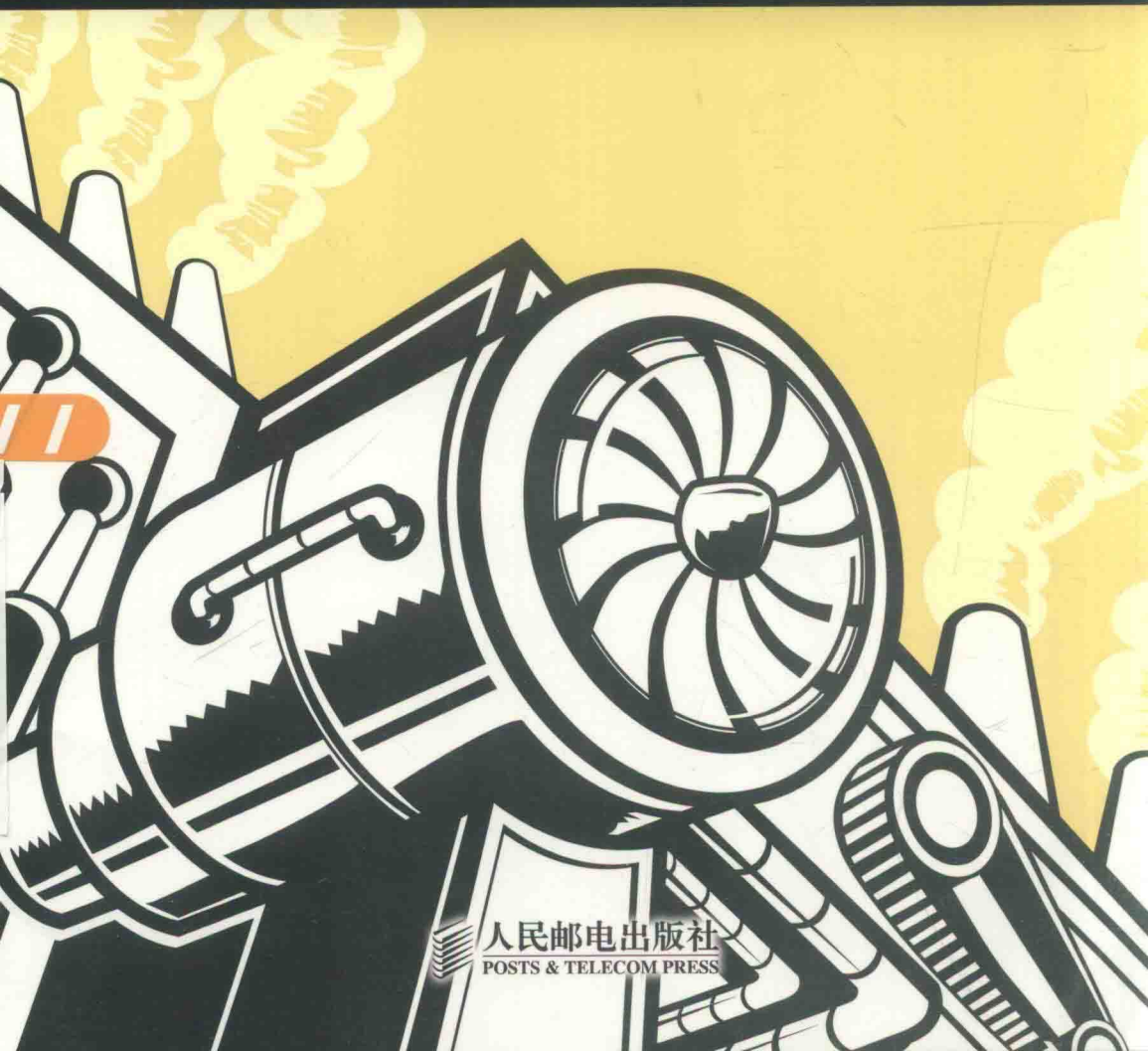


JavaScript

THE PRINCIPLES OF
OBJECT-ORIENTED
JAVASCRIPT

面向对象精要

[美] Nicholas C. Zakas 著 胡世杰 译



人民邮电出版社
POSTS & TELECOM PRESS

JavaScript

THE PRINCIPLES OF
OBJECT-ORIENTED
JAVASCRIPT

面向对象精要

[美] Nicholas C. Zakas 著 胡世杰 译



人民邮电出版社
北京

图书在版编目 (C I P) 数据

JavaScript面向对象对象精要 / (美) 尼古拉斯
(Nicholas, C. Z.) 著 ; 胡世杰译. — 北京 : 人民邮电
出版社, 2015. 4

ISBN 978-7-115-38384-6

I. ①J… II. ①尼… ②胡… III. ①JAVA语言—程序
设计 IV. ①TP312

中国版本图书馆CIP数据核字 (2015) 第026626号

版权声明

Simplified Chinese-language edition copyright©2015 by Posts and Telecom Press.
Copyright©2014 by Nicholas C.Zakas.Title of English-language original:The Principles of
Object-Oriented JavaScript, ISBN-13:978-1-59327-540-2, published by No Starch Press.
All rights reserved.

本书中文简体字版由美国 No Starch 出版社授权人民邮电出版社出版。未经出版者书面许可,
对本书任何部分不得以任何方式复制或抄袭。

版权所有, 侵权必究。

-
- ◆ 著 [美] Nicholas C.Zakas
译 胡世杰
责任编辑 陈冀康
责任印制 张佳莹 焦志炜
- ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市中晟雅豪印务有限公司印刷
- ◆ 开本: 720×960 1/16
印张: 7.25
字数: 117 千字 2014 年 4 月第 1 版
印数: 1—3 500 册 2014 年 4 月河北第 1 次印刷
著作权合同登记号 图字: 01-2013-7464 号
-

定价: 39.00 元

读者服务热线: (010)81055410 印装质量热线: (010)81055316
反盗版热线: (010)81055315

内 容 提 要

本书关注面向对象的原理和 ES5 对象新特性，目的是帮助那些已经熟知面向对象编程的读者掌握这些概念是如何在 JavaScript 中工作的。

本书内容简洁而精妙。全书共 6 章，分别深入探讨了原始类型和引用类型、函数、对象、构造函数和原型对象、继承和对象模式等主题和特性。在本书中，你将学到 JavaScript 独特的面向对象的编程方式：抛弃类的概念和基于类的继承，学习基于原型的继承和构造函数。你将学会如何创建对象、定义自己的类型、使用继承以及其他各种操作来充分使用对象。总而言之，你将学到用 JavaScript 语言进行专业编程所需熟知的一切。

本书适合熟悉面向对象编程的概念并希望将其应用于 JavaScript 的开发者阅读，也适合 JavaScript 新手学习参考。

译者序

JavaScript 是一门流行多年的语言，到现在依然有着旺盛的生命力。这与无数开发人员的使用和完善是分不开的。希望本书能够为这些开发者开阔视野，带来一些更深层次的领悟。

本书作者是一位非常资深的 JavaScript 开发者、作者以及演讲者。他能够深入浅出地介绍 JavaScript 的面向对象的特性，使得本书尤其适用于那些已经熟悉面向对象或 JavaScript 两者之一的开发者。

在你阅读这本书时，就好像听见作者将 JavaScript 中面向对象的特征性娓娓道来，看见一幅精准的设计蓝图在面前徐徐展现，让你能够自始至终都沉醉在优美的技术世界里无法自拔，感受到每一个简洁设计背后的意图，甚至不由自主地去思考为什么最终的设计是现在这个样子？如果是让自己来设计又会做成什么样子？会不会有更好的设计？

无论你的答案如何，作者的目的都达到了。作者不仅是要告诉读者 JavaScript 是如何实现面向对象的特性的，更重要的是引导每一位读者思考为什么如此实现。而这，才是最有价值的东西。

这是本人的第一本译作。感谢好友高博推荐我这个机会，让我能够进入技术书籍的翻译这样一个领域。感谢陈冀康编辑的辛勤工作，这本译作的问世离不开他的审阅和校对。感谢好友徐章宁在翻译本书过程中给予我的建议，最后感谢我挚爱的妻子黄静对我由于倾力翻译而疏于家事的体谅和支持。希望本书的出版能够给你们带来快乐。

胡世杰
于上海
2015 年 1 月

作者简介

Nicholas C. Zakas 是 Box 公司的一位软件工程师。他因 JavaScript 最佳实践的写作和演讲而知名。在此之前，他作为 Yahoo! 主页的资深前端工程师，在雅虎有 5 年的工作经验。他写了好几本书，包括 *Maintainable JavaScript* (O'Reilly Media, 2012) 和 *Professional JavaScript for Web Developers* (Wrox, 2012)。

技术评审者简介

来自英国的 Angus Croll 现在是位于旧金山的推特的网页框架团队的一员，同时也是推特的开源框架 Flight 的合作开发者和主要维护者。他对 JavaScript 和文学有同等的兴趣爱好，并热情拥护艺术家和有创造力的思想家们更多地参与软件开发。Angus 经常在世界各地的研讨会上发言，现正在为 No Starch Press 编写两本书。

译者简介

胡世杰，1981 年生，硕士毕业于上海交通大学。目前在 EMC 中国卓越研发集团任高级开发工程师，在软件开发领域有超过 10 年的经验。对各种主流开发语言均有涉猎。平时热爱看书和游戏，是婚姻生活中的情感专家和沟通大师。

序

Nicholas Zakas 这个名字就是 JavaScript 开发的代名词。我可以用数页的篇幅漫谈他获得的专业荣誉，但我不准备这么做。Nicholas 是一位非常著名的顶级 JavaScript 开发者及作者，他不需要介绍。不过，在赞扬这本书之前，我打算先谈一些个人感受。

我和 Nicholas 的关系来自多年来以一个 JavaScript 小學生的身份研究他的书，读他的博客，听他的演讲，关注他的推特更新。我们的第一次会面是在几年前，当时我请他在一个 jQuery 研讨会上发言。结果他做了一场高质量的演讲。此后，我们一直在网络上保持着密切的联系。从那时起，我就越来越崇拜他，不仅仅因为他是 JavaScript 委员会的一位领导者和开发者。他的语言是如此优雅和使人深思，他的行为是如此亲切。他永远是一位乐于助人的开发者、诲人不倦的演讲者和超越自我的作者。当他演讲时，你应该倾听，不仅因为他是一位 JavaScript 专家，更因为他高尚的人品还在他的专业地位之上。

本书的标题和简介清楚地展现了 Nicholas 的意图：他写这本书是为了帮助那些对类（也就是 C++ 或 Java）习以为常的程序员转换到一个没有类的语言上去。他在本书中解释了如何在 JavaScript 中实现封装、聚合、继承以及多态。有基础的程序员想要转投面向对

象的 JavaScript 开发，这是一本理想的教材。如果你是一位其他语言的开发者，你会发现这是一本语言简洁而精妙的 JavaScript 方面的书。

不过，本书同时也为 JavaScript 程序员服务。很多 JavaScript 开发者对对象的理解仍停留在 ECMAScript 3 (ES3) 的版本，他们需要适当了解 ECMAScript 5 (ES5) 版本的对象功能。本书就可以满足他们的需求，帮他们弥补 ES3 对象和 ES5 对象之间的知识空白。

现在你可能在想，“没啥了不起的。介绍 ES5 的 JavaScript 书已经有好几本了。”好吧，确实如此。不过我相信本书是目前唯一一本在整个叙述过程中都将 ES5 对象视为一等公民并时刻深入对象本质的著作。本书不仅对 ES5 对象进行介绍，还介绍了在你学习 ES5 新特性的时候所需要了解的 ES3 和 ES5 之间的差异。

这是一本关注面向对象的原理和 ES5 对象新特性的书。作为一名作者，我坚信本书是我们在等待脚本环境更新至 ES6 之前的必读之作。

Cody Lindley (www.codylindley.com)

*JavaScript Enlightenment*¹, *DOM Enlightenment* 和
jQuery Enlightenment 的作者

于博伊西，爱达荷

2013 年 12 月 16 日

¹ 编者注：中文版《Java Script 启示录》由人民邮电出版社于 2014 年 3 月出版 (ISBN 978-115-33494-7, 定价 35 元)。

致 谢

我要感谢 Kate Matsudaira 让我相信自助出版一本 Ebook 是传递信息最好的方式。没有她的建议，我可能还在想我该如何处理本书中包含的信息。

感谢 Rob Friesel 再次对本书的初稿提供的精彩反馈，以及 Cody Lindley 提供的建议。感谢 Angus Croll 对本书的定稿提供的技术评审——他的精益求精为本书增色不少。

感谢我在一次研讨会上遇到的 Bill Pollock，是他推动了这本书的出版。

前言

大多数开发者将面向对象的编程联想为那些在学校中学到的基于类的语言，比如 C++ 和 Java。在用这些语言完成任务之前，必须先创建类，哪怕只是写一个简单的命令程序。目前业界常用的设计模式也强调了基于类的概念。JavaScript 不使用类，这也是人们在学了 C++ 或 Java 之后再学习 JavaScript 时感到困惑的原因。

面向对象的语言有如下几种特性。

封装 数据可以和操作数据的功能组织在一起。这就是对象的定义，十分简单。

聚合 一个对象能够引用另一个对象。

继承 一个新创建的对象和另一个对象拥有同样的特性，而无需显式复制其功能。

多态 一个接口可被多个对象实现。

JavaScript 拥有上述全部特性，因为语言本身没有类的概念，所以某些特性可能不是以你所期望的方式实现的。乍一看，一个 JavaScript 程序可能像是一个用 C 来编写的面向过程的程序。如果写一个函数并传递一些参数，就有了一个看上去没有对象也可工作的脚本。但是仔细观察，你就会在点号的使用上发现对象的

存在。

很多面向对象的语言使用点号来访问对象的属性和方法，JavaScript 也不例外。但是你永远不需要在 JavaScript 中写一个类定义，导入一个包或包含一个头文件。你只是用你需要的数据类型开始编写代码，然后有无数种方法可以把它们组织在一起。可以用面向过程的方式编写 JavaScript，但它真正的威力是在你利用其面向对象的特性时才能展现的。而这就是本书要告诉你的。

不要搞错：你在传统的面向对象的语言中学到的很多概念都不一定适用于 JavaScript。初学者往往对此迷惑不已。你在阅读的过程中会迅速发现 JavaScript 的弱类型特性允许你用比其他语言更少的代码完成同样的任务。你无需预先设计好类就可以开始编写代码。需要一个具有某个字段的对象了？随时随地都可创建。忘了给那个对象添加一个方法？没关系——以后补上。

在本书中，你将学到 JavaScript 独特的面向对象的编程方式。抛弃类的概念和基于类的继承，学习基于原型的继承和功能类似的构造函数。你将学会如何创建对象，定义自己的类型，使用继承以及其他各种操作来充分使用对象。一句话，你将从专业级别理解和使用 JavaScript 程序所需知道的一切。享受它吧！

本书目标读者

本书的目的是帮助那些已经熟知面向对象编程的读者掌握这些概念是如何在 JavaScript 中工作的。只要你熟悉 Java、C#或其他面向对象的编程语言，那么本书就是为你而著。尤其是以下 3 种人群。

- 熟悉面向对象编程的概念并希望将其应用于 JavaScript 的开发者。
- 希望更有效地组织代码的网页应用程序开发者和 Node.js 开发者。
- 想深入了解 JavaScript 的开发新手。

本书不是写给那些从未编写过 JavaScript 的新手的。你需要对

如何编写和执行 JavaScript 代码有一个清楚的认识才能跟得上。

概览

第 1 章“原始类型和引用类型”介绍了 JavaScript 中的两种数据类型：原始类型和引用类型。你会学到它们之间的区别，知道为什么理解它们之间的区别对于理解整个 JavaScript 非常重要。

第 2 章“函数”解释了 JavaScript 函数的输入和输出。函数是 JavaScript 的一等公民，它们使得 JavaScript 成为一门有趣的语言。

第 3 章“理解对象”深度探索 JavaScript 对象的组成。JavaScript 对象的行为和其他语言的对象不同，深入了解对象的工作原理是掌握 JavaScript 语言的关键。

第 4 章“构造函数和原型对象”将目光聚焦于构造函数，拓展了之前对函数的讨论。所有的构造函数都是函数，但它们在使用上有一点区别。本章在探索这些区别之外还讨论了如何创建自定义类型。

第 5 章“继承”解释了 JavaScript 中的继承是如何实现的。JavaScript 里没有类，却不代表 JavaScript 里不能继承。在本章，你将学到原型继承以及它和类继承的区别。

第 6 章“对象模式”带你浏览常用的对象模式。JavaScript 拥有很多不同的方式来创建和组合对象，本章为你介绍其中最流行的几种模式。

帮助与支持

如果你对本书有任何疑问、评论或其他反馈，请访问 <http://groups.google.com/group/zakasbooks> 上的邮件列表。

目 录

第 1 章	原始类型和引用类型	1
1.1	什么是类型	2
1.2	原始类型	3
1.2.1	鉴别原始类型	4
1.2.2	原始方法	6
1.3	引用类型	6
1.3.1	创建对象	6
1.3.2	对象引用解除	7
1.3.3	添加删除属性	8
1.4	内建类型实例化	8
1.4.1	字面形式	9
1.4.2	对象和数组字面形式	9
1.4.3	函数字面形式	10
1.4.4	正则表达式字面形式	11
1.5	访问属性	11
1.6	鉴别引用类型	12
1.7	鉴别数组	13
1.8	原始封装类型	14

2 目 录

1.9 总结	16
第 2 章 函数	17
2.1 声明还是表达式	18
2.2 函数就是值	19
2.3 参数	21
2.4 重载	23
2.5 对象方法	24
2.5.1 this 对象	25
2.5.2 改变 this	26
2.6 总结	29
第 3 章 理解对象	31
3.1 定义属性	32
3.2 属性探测	33
3.3 删除属性	35
3.4 属性枚举	36
3.5 属性类型	37
3.6 属性特征	38
3.6.1 通用特征	39
3.6.2 数据属性特征	40
3.6.3 访问器属性特征	41
3.6.4 定义多重属性	43
3.6.5 获取属性特征	44
3.7 禁止修改对象	45
3.7.1 禁止扩展	45
3.7.2 对象封印	45
3.7.3 对象冻结	47
3.8 总结	48
第 4 章 构造函数和原型对象	49
4.1 构造函数	49

4.2	原型对象	53
4.2.1	[[Prototype]]属性	54
4.2.2	在构造函数中使用原型对象	57
4.2.3	改变原型对象	60
4.2.4	内建对象的原型对象	62
4.3	总结	63
第 5 章	继承	65
5.1	原型对象链和 Object.prototype	65
5.1.1	继承自 Object.prototype 的方法	66
5.1.2	修改 Object.prototype	68
5.2	对象继承	69
5.3	构造函数继承	72
5.4	构造函数窃取	75
5.5	访问父类方法	77
5.6	总结	78
第 6 章	对象模式	79
6.1	私有成员和特权成员	80
6.1.1	模块模式	80
6.1.2	构造函数的私有成员	82
6.2	混入	84
6.3	作用域安全的构造函数	90
6.4	总结	92
索引		93

第 1 章

原始类型和引用类型

大多数开发者在使用 Java 或 C#等基于类的语言的过程中学会了面向对象编程。由于 JavaScript 没有对类的正式支持，这些开发者在学习 JavaScript 时往往会迷失方向。JavaScript 不需要在开头就定义好各种类，你可以在写代码的过程中根据需要创建数据结构。由于 JavaScript 缺少类，也就缺少用于对类进行分组的包。在 Java 中，包和类的名字不仅定义了对象的类型，也在工程中列出文件和目录的层次结构，JavaScript 编程就好像从一块空白石板开始：你可以在上面组织任何你想要的东西。些开发者选择模仿其他语言的结构，也有一些人则利用 JavaScript 的灵活性来创建一些全新的东西。对没有掌握 JavaScript 的人来说，这种选择的自由令人崩溃，然而一旦你熟悉了它，你会发现 JavaScript 是一种无比灵活的语言，可以很轻松地适应你的编程习惯。

为了便于开发者从传统的面向对象语言过渡, JavaScript 把对象作为语言的中心。几乎所有 JavaScript 的数据要么是一个对象要么从对象中获取。其实就连函数在 JavaScript 中也被视为对象, 这使得它们成为 JavaScript 的一等公民。

使用和理解对象是理解整个 JavaScript 的关键。你可以在任何时候创建对象, 在任何时候给对象添加、删除属性。JavaScript 对象是如此灵活, 可以创造出其他语言不具有的独特而有趣的模式。

本章致力于鉴别和使用两种 JavaScript 基本数据类型: 原始类型和引用类型。虽然两者都通过对象进行访问, 但是理解它们行为之间的区别是非常重要的。

1.1 什么是类型

JavaScript 虽然没有类的概念, 但依然存在两种类型: 原始类型和引用类型。原始类型保存为简单数据值。引用类型则保存为对象, 其本质是指向内存位置的引用。

为了让开发者能够把原始类型和引用类型按相同方式处理, JavaScript 花费了很大努力来保证语言的一致性。

其他编程语言用栈储存原始类型, 用堆储存引用类型, JavaScript 则完全不同: 它使用一个变量对象追踪变量的生存期。原始值被直接保存在变量对象内, 而引用值则作为一个指针保存在变量对象内, 该指针指向实际对象在内存中的存储位置。虽然看上去原始值和引用值一样, 但是它们还是有区别的, 本章稍后会介绍。

当然, 原始类型和引用类型还有其他区别。