MANNING

异步图书
www.epubit.com.cn

JavaScript 忍者秘籍（第2版）

[美] 约翰·莱西格 (John Resig)
[美] 拜尔·贝比奥特（Bear Bibeault）著
[美] 约瑟普·马瑞斯（Josip Maras）
一心一译前端小组 译

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

JavaScript 忍者秘籍（第2版）

[美] 约翰·莱西格 (John Resig)
[美] 拜尔·贝比奥特（Bear Bibeault）著
[美] 约瑟普·马瑞斯（Josip Maras）
一心一译前端小组 译

人民邮电出版社
北京

图书在版编目（CIP）数据

JavaScript忍者秘籍 / (美) 莱西格 (John Resig)，(美) 贝比奥特 (Bear Bibeault)，(美) 马瑞斯 (Josip Maras) 著；一心一译前端小组译. -- 2版. -- 北京 : 人民邮电出版社，2018.1
ISBN 978-7-115-47326-4

Ⅰ. ①J… Ⅱ. ①莱… ②贝… ③马… ④一… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2017)第289270号

版权声明

◆ 著　[美] 约翰·莱西格 (John Resig)
[美] 拜尔·贝比奥特（Bear Bibeault)
[美] 约瑟普·马瑞斯（Josip Maras）

译　一心一译前端小组

责任编辑　陈冀康

责任印制　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

三河市祥达印刷包装有限公司印刷

◆ 开本：800×1000　1/16

印张：28　插页：1

字数：605 千字　2018 年 1 月第 2 版

印数：3 001 – 6 000 册　2018 年 1 月河北第 1 次印刷

著作权合同登记号　图字：01-2016-7581 号

定价：99.00 元

读者服务热线：(010)81055410　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

ES6 速查表

模板字面量（template literal）是允许嵌入表达式的字符串字面量：'${ninja}'。

块级作用域变量：

- 使用新的 let 关键字创建块级作用域变量：let ninja = "Yoshi"。
- 使用新的 const 关键字创建块级作用域常量，常量在创建后不能被重新赋值：const ninja = "Yoshi"。

函数参数：

- **剩余参数**（rest parameter）可以将未命中形参的参数创建为一个不定数量的数组。

```
function multiMax(first,...remaining){/*...*/}multiMax(2,3,4,5);//first: 2;
    remaining: [3, 4, 5]
```

- **函数默认参数**（default parameter）允许在调用时没有值或 undefined 被传入时使用指定的默认参数值。

```
function do(ninja,action="skulk"){return ninja+" "+action;}
do("Fuma");//"Fuma skulk"
```

扩展语法（spread operator）允许一个表达式在期望多个参数（用于函数调用）或多个元素（用于数组字面量）或多个变量（用于解构赋值）的位置扩展：[...items,3,4,5]。

箭头函数（arrow function）可以创建语法更为简洁的函数。箭头函数不会创建自己的 this 参数，相反，它将继承使用执行上下文的 this 值：

```
const values = [0, 3, 2, 5, 7, 4, 8, 1];
values.sort((v1,v2)=> v1 - v2);/*OR*/ values.sort((v1,v2) => {return v1 - v2;});
value.forEach(value => console.log(value));
```

生成器（generator）函数能生成一组值的序列，但每个值的生成是基于每次请求，并不同于标准函数那样立即生成。每当生成器函数生成了一个值，它都会暂停执行但不会阻塞后续代码执行。使用 yield 来生成一个新的值：

```
function *IdGenerator(){
   let id = 0;
   while(true){ yield ++id; }
}
```

promise 对象是对我们现在尚未得到但将来会得到值的占位符。它是对我们最终能够得知异步计算结果的一种保证。promise 既可以成功也可以失败，并且一旦设定好了，就不能够有更多改变。
通过调用传入的 resolve 函数，一个 promise 就被成功兑现（resolve）（通过调用 reject 则 promise 被违背）。拒绝一个 promise 有两种方式：显式拒绝，即在一个 promise 的执行函数中调用传入的 reject 方法；隐式拒绝，如果正处理一个 promise 的过程中跑出了一个异常。

- 使用 new Promise((resolve, reject) => });创建一个新的 promise 对象。
- 调用 resolve 函数来显式地兑现一个 promise。调用 reject 函数来显式地拒绝一个 promise。如果在处理的过程中发生异常则会隐式地拒绝该 promise。
- 一个 promise 对象拥有 then 方法，它接收两个回调函数（一个成功回调和一个失败回调）作为参数并返回一个 promise：

```
myPromise.then(val => console.log("Success"),err => console.log("Error"));
```

- 链式调用 catch 方法可以捕获 promise 的失败异常：myPromise. catch(e => alert(e));。

ES6 速查表（续）

类（Class）是 JavaScript 原型的语法糖：

```
class Person {
  constructor(name){this.name = name; }
  dance(){return true; }
}
class Ninja extends Person {
  constructor(name, level){
    super(name);
    this.level = level;
  }
  static compare(ninja1, ninja2){
    return ninja1.level - ninja2.level;
  }
}
```

代理（Proxy）可对对象的访问进行控制。当与对象交互时（当获取对象的属性或调用函数时），可以执行自定义操作。

```
const p = new Proxy(target, {
    get:(target, key) => { /*Called when property accessed through proxy*/ },
    set: (target, key, value) => { /*Called when property set through proxy*/ }
});
```

映射（Map）是键与值之间的映射关系：

- 通过 new Map()创建一个新的映射。
- 使用 set 方法添加新映射，get 方法获取映射，has 方法检测映射是否存在，delete 方法删除映射。

集合（Set）是一组非重复成员的集合：

- 通过 new Set()创建一个新的集合。
- 使用 add 方法添加成员，delete 方法删除成员，size 属性获取集合中成员的个数。

for...of 循环遍历集合或生成器。

对象与数组的解构（destructuring）：

- `const {name: ninjaName} = ninja;`
- `const [firstNinja] = ["Yoshi"];`

模块（Module）是更大的代码组织单元，可以将程序划分为若干个小片段：

```
export class Ninja{}; //导出 Ninja 类
export default class Ninja{} //使用默认导出
export {ninja};//导出存在的变量
export {ninja as samurai}; //导出时进行重命名

import Ninja from "Ninja.js"; //导入默认值
import {ninja} from "Ninja.js"; //导入单个导出
import * as Ninja from "Ninja.js"; //导入整个模块的内容
import {ninja as iNinja} from "Ninja.js"; //导入时重命名单个导出
```

内容提要

JavaScript 语言非常重要，相关的技术图书也很多，但至今市面没有一本对 JavaScript 语言的最重要部分（函数、闭包和原型）进行深入、全面介绍的图书，也没有一本讲述跨浏览器代码编写的图书。而本书弥补了这一空缺，是由 jQuery 库创始人编写的一本深入剖析 JavaScript 语言的书。

本书共分 4 个部分，从不同层次讲述了逐步成为 JavaScript 高手所需的知识。本书从 JavaScript 语言及最重要的特性谈起，由浅入深地探讨了函数、作用域、闭包、生成器函数、对象、数组、模块化、JavaScript 与 Web 页面的交互以及事件等主题，引导读者更加深入地了解 JavaScript 的方方面面，充分展示了 JavaScript 语言的各种特性。本书结合 ECMAScript 6 和 7 的相关概念，涵盖了流行的 JavaScript 框架所使用的技术。

本书适合具备一定 JavaScript 基础知识的读者阅读，也适合从事程序设计工作并想要深入探索 JavaScript 语言的读者阅读。

译者简介

一心一译前端小组是一群热爱前端和 JavaScript 的技术人员，他们因兴趣而走到一起，致力于为读者奉献高品质的中译版技术图书。

郭凯

美团点评酒旅事业群前端团队负责人，高级技术专家，资深互联网人，全栈工程师，工作狂，崇尚工匠精神，曾就职于音悦台、淘宝旅行。译作有《编写可维护的 JavaScript》《第三方 JavaScript 编程》《JavaScript 开发框架权威指南》，有 In、Juicer、jSQL、F2E.im、PM25 等开源项目，业余时间负责开源前端技术社区 F2E 的开发和维护。

赵荣娇

前端开发工程师，也是《响应式设计、改造与优化》一书的译者。著有《超实用的 CSS 代码段》《代码逆袭：超实用的 HTML 代码段》等书籍。喜欢旅行，热爱前端开发。

王思可

北京大学在校学生，热爱 JavaScript，热爱技术。

巩守强

猫眼基础交易负责人，美团前端高级技术专家，对于前端工程化、前端性能优化和客户端动态化感兴趣。

康明轩

北京师范大学硕士，现就职于北京指南针科技发展股份有限公司，从事网络应用开发，认为编程也是一门可以长相厮守的手艺。

作者简介

John Resig 是可汗学院（Khan Academy）的一名资深工程师，是 jQuery JavaScript 库的创建者，也是《JavaScript 忍者秘籍（第 1 版）》和《精通 JavaScript》的作者。

John 开发了日本版画综合性数据库和图片搜索引擎：Ukiyo-e.org。他是美国和日本艺术学会的董事会成员，也是立命馆大学（Ritsumeikan University）的客座研究员，致力于研究浮世绘。

Bear Bibeault 编写软件已经超过 30 年，刚开始是通过 100 波特的电传打字机在控制数据网络超级计算机上编写井字程序。Bear 有电气工程双学位，本应从事设计天线之类的技术工作，但自从他在数字设备公司从事第一份工作起，他就更着迷于编程。Bear 还分别在 Dragon Systems、Works.com、Spredfast、Logitech、Caringo 等诸多公司工作过。Bear 目前是一名高级前端开发工程师，在一家对象存储软件的领先供应商工作，提供可伸缩性的海量存储和内容保护服务。

除了本书的第 1 版，Bear 也是其他一些 Manning 书籍的作者，包括《jQuery 实战》（第 1 版、第 2 版、第 3 版），《Ajax 实战》《原型脚本实战》。他也是 O'Reilly 出版社出版的“Head First”系列书籍的审稿人，例如《Head First Ajax》《Head Rush Ajax》和《Head First Servlets and JSP》。

除了日常工作外，他还经营着一家小型企业，致力于创建 Web 应用程序，为其他媒体提供服务，并作为“引领者”（超级管理员）管理 CodeRanch.com。

Josip Maras 是克罗地亚斯普利特大学电气工程学院、机械工程学院、造船建筑学院的博士后研究员。他获得软件工程博士学位，论文题目是“在 Web 应用程序开发中实现自动复用”，其中包括使用 JavaScript 实现的 JavaScript 解释器。在他的研究中，他已经出版了十多篇科学会议和期刊论文，主要是分析客户端 Web 应用程序的处理程序。

不做研究时，他从事教学工作，教授学生网站开发、系统分析和设计、Windows 开发等知识（过去 6 年培养了几百位学生）。他还拥有一个小型软件开发公司。

致谢

参与编写本书的人数之多会让你们感到吃惊。天才们的共同努力付出带来了此刻你捧在手里的这本书（或在屏幕上阅读的电子书）。

Manning 出版社的工作人员不知疲倦地帮助我们确保本书的质量，感谢他们的付出。没有他们，本书不可能完成。包括出版人 Marjan Bace、编辑 Dan Maharry，还有以下贡献者们：Ozren Harlovic、Gregor Zurowski、Kevin Sullivan、Janet Vail、Tiffany Taylor、Sharon Wilkey、Alyson Brener 和 Gordan Salinovic。

感谢本书的审阅者们，从简单细致的排版到术语的纠正以及代码中的错误修正，再到组织本书的章节，他们每一轮的审阅，对本书最终出版的质量都有很大的提高。非常感谢他们花时间审阅本书：Jacob Andresen、Tidjani Belmansour、Francesco Bianchi、Matthew Halverson、Becky Huett、Daniel Lamb、Michael Lund、Kariem Ali Elkoush、Elyse Kolker Gordon、Christopher Haupt、Mike Hatfield、Gerd Klevesaat、Alex Lucas、Arun Noronha、Adam Scheller、David Starkey 和 Gregor Zurowski。

特别感谢 Mathias Bynens 和 Jon Borgman，他们对本书提供技术校对。除了在多种环境中逐一校对书中的每一个示例代码，他们还对本书的技术准确性提供了非常重要的指导意见，他们找到初始版本已经遗失的信息，快速同步了浏览器上对 JavaScript 与 HTML5 的最新变化。

John Resig

感谢我的父母多年来对我的支持和鼓励。他们为我提供了很多有用的资源和工具，激发我对编程的兴趣——他们一直都鼓励着我。

Bear Bibeault

首先要特别感谢 coderanch.com（JavaRanch）的工作人员。如果不加入 CodeRanch，我就不会有机会开始参与写作本书，我衷心感谢 Paul Wheaton 和 Kathy Sierra 让我开

始这项工作，以及以下同事对我的鼓励和支持，包括（但不限于）：Eric Pascarello、Ernest Friedman-Hill、Andrew Monkhouse、Jeanne Boyarsky、Bert Bates 和 Max Habibi。

感谢我的伴侣 Jay，感谢他忍受我参与这项工作时，只专注于浏览器和缺乏打字技术的胖手指。除了抱怨一下糟糕的 Word、某个浏览器或者任何使我愤怒的事之外，我几乎很少从键盘上抬起头来。

最后，感谢我的合著者 John Resig 和 Josip Maras，没有他们就没有这本书。

Josip Maras

最大的感谢献给我的妻子——Josipa，感谢她忍受我将大量时间投入到本书的写作中。

还要感谢 Maja Stula、Darko Stipanicev、Ivica Crnkovic、Jan Carlson 和 Bert Bates：感谢他们的指导和有用的建议，以及在本书截止日期之前对我的包容。

最后，感谢我的家人——Marijas、Vitomir 和 Zdenka——感谢你们的陪伴。

序

自 2008 年我编写《JavaScript 忍者秘籍》起，到现在 JavaScript 的世界发生了翻天覆地的变化。我们现在编写的 JavaScript，虽然大部分仍然是基于浏览器，但是几乎快认不出来了。

由于功能全面、跨平台的特性，JavaScript 的流行度呈爆发式增长。Node.js 是一个强大的平台，已基于 Node.js 开发了无数的生产应用程序。开发人员实际上是在使用一种语言——JavaScript 编写应用程序以及同时可运行于浏览器端、服务器端甚至移动设备上的本地应用。

现在所需的 JavaScript 知识，比以前任何时候都更为重要。对 JavaScript 这门语言有一个基本的理解，并且了解最佳编程方式，有助于创建几乎可在任何平台运行的应用程序，几乎没有其他语言可以做到这一点。

与以往 JavaScript 增长的时代不同，平台不兼容的情况没有得到改善。你常常会垂涎于使用最基本的浏览器新特性，但是过时的浏览器却占领了太多的市场份额。我们已经进入了一个和谐的时间，大多数用户都在快速更新最符合标准的平台。浏览器厂商甚至推出专门针对开发人员的特性，希望这能使他们的生活更加简单。

浏览器目前提供给我们的工具以及开源社区，是过去实践之后的光明。我们现在有大量的可供选择的测试框架，有持续集成测试的能力，可以生成代码覆盖报告，在真正的全球移动设备上做性能测试，甚至可在任何平台上自动加载虚拟浏览器进行测试。

这本书的第 1 版极大地受益于 Bear Bibeault 的开发洞察力。这个版本得到 Josip Maras 大量的帮助，他研究 ECMAScript 6 和 7 的相关概念，深入了解测试最佳实践，了解流行的 JavaScript 框架所使用的技术。

我们编写 JavaScript 的方式发生了巨大的变化，这很难阐述。好在这本书可以帮助你了解当前的最佳实践。不仅如此，本书还会帮助你改善思维方式，如何将开发实践作为一个整体，以确保为未来编写 JavaScript。

John Resig

前言

JavaScript 非常重要。过去并非如此，但现在的确如此。如今 JavaScript 已经成为最重要的、使用最广泛的编程语言。

Web 应用程序为用户带来了丰富的用户界面体验，如果没有 JavaScript，可能只能显示小照片。Web 开发人员比以往任何时候都更需要熟练掌握 JavaScript 语言，JavaScript 为 Web 应用程序注入了生命。

JavaScript 不再只应用于浏览器了。JavaScript 打破了浏览器的界限，可应用于服务端的 Node.js，可应用于桌面设备和移动设备如 Apache Cordova，甚至可内置在设备中如 Espruino 和 Tessel。虽然本书主要集中介绍在浏览器端执行 JavaScript，但本书介绍的 JavaScript 基础适用范围非常广泛。深刻理解概念、了解多种技巧有助于你成为全栈 JavaScript 工程师。

随着使用 JavaScript 的开发人员逐渐增多，熟练掌握 JavaScript 基础比以往任何时候都更加重要，这样才能成为真正的 JavaScript “忍者”。

目标读者

如果你不熟悉 JavaScript，那么这本书并不适合作为你的第一本 JavaScript 图书。但也别太担心，我们试图介绍基本的 JavaScript 概念，对于初学者也相对容易理解。但是，本书更适合于至少掌握 JavaScript 基础、了解 JavaScript 代码执行的浏览器环境，并且希望深入理解 JavaScript 语言的 Web 开发人员。

路线图

本书通过 4 个部分，让你从“学徒”晋升为“忍者”。

第 1 部分介绍我们后续学习的主题和所需要的工具。

- 第 1 章介绍 JavaScript 语言及最重要的特性，推荐目前我们开发应用时需要遵循的最佳实践，包括测试和性能分析。

- 因为我们对 JavaScript 的研究是基于浏览器上下文，因此在第 2 章中，我们介绍客户端 Web 应用的生命周期，这有助于我们理解在开发 Web 应用程序时 JavaScript 所扮演的角色。

第 2 部分重点关注 JavaScript 的核心支柱之一——函数。我们将研究为什么函数如此重要，函数之间的区别，以及定义和调用函数的细节内容。我们还将特别关注一个新的函数类型——生成器函数，它在处理异步代码时尤为有效。

- 第 3 章从彻底检查 JavaScript 函数的定义开始涉足基础语言，也许你会感到吃惊。预期中可能是把对象作为重点，但是，让我们充分理解函数、JavaScript 函数式语言，从普通的 JavaScript 程序员升级为 JavaScript“忍者”！
- 在第 4 章中，我们继续研究函数，深入研究函数调用的机制，以及隐式函数参数的来龙去脉。
- 关于函数的内容还没有结束，在第 5 章我们把讨论推向更高的一个层级，研究两个密切相关的概念——作用域和闭包。闭包是函数式编程中的关键概念，闭包允许更细粒度地控制程序中声明和创建的对象作用域范围。控制对象的作用域范围是“忍者”编写代码的关键因素。即使不阅读后续的章节（但我们希望大家不要停下来），编程水平也会比刚开始学习时提高很多。
- 在第 6 章中，我们通过一种全新的函数类型（生成器函数）和一个新的对象类型（promise）帮助我们处理异步代码，最后结束对函数的研究。我们还展示了如何结合 generator 与 promise，优雅地处理异步代码。

第 3 部分研究 JavaScript 的第二支柱——对象。我们将彻底地探索 JavaScript 中的面向对象，研究如何保护对对象的访问，如何处理集合和正则表达式。

- 第 7 章阐述对象，彻底了解 JavaScript 中面向对象是如何工作的。此外，我们还将引入一个新的 JavaScript 关键字：class。其背后概念可能与你所期望的有所不同。
- 第 8 章继续探索对象，我们将学习使用多种不同的技术保护对对象的访问。
- 在第 9 章中，我们将特别关注 JavaScript 中几种不同类型的集合。数组，从 JavaScript 诞生起就是 JavaScript 的一部分，map 和 set 是最近新加入 JavaScript 的集合类型。
- 第 10 章着重介绍正则表达式，正则表达式是经常被忽略的一项语言特性，但正确使用正则表达式，可以减少很多代码量。我们将学习如何构建和使用正则表达式，以及如何使用正则表达式及其相关方法，优雅地解决一些重复出现的问题。
- 在第 11 章中，我们将学习使用不同技术实现代码模块化：更小、相对松耦合的代码片段，以及改善代码的机构和组织方式。

最后，第 4 部分研究 JavaScript 与 Web 页面的交互以及浏览器如何处理事件，最后结束本书。在结束之前的最后一个重要话题是跨浏览器开发。

- 第 12 章研究如何通过 DOM API 动态修改页面，如何处理元素属性、样式，以及一些重要的性能注意事项。

- 第 13 章讨论 JavaScript 的单线程执行模型的重要性，以及单线程执行模型对事件循环的影响。我们还将学习间隔定时器的工作原理，以及如何使用它们提高 Web 应用程序的性能。
- 第 14 章检查开发时主要关心的 5 项跨浏览器问题：浏览器缺陷、缺陷修复、外部代码、功能缺失和回归。讨论诸如特性模拟和对象检测等方法，有助于跨浏览器开发的挑战。

代码规范

代码清单或文本中的所有源代码都采用等宽字体，与普通文本进行区分。

在某些情况下，为了适应页面，会对源代码进行格式化。一般来说，编写源代码时需要考虑页面宽度限制，但有时你会发现本书中的代码和下载的代码之间的格式有所不同。在极少数情况下，为了不改变代码的含义，代码过长无法被格式化，本书的代码清单中使用行连续符号进行标记。代码注释和许多列表用于突出重要概念。

源代码下载

本书中示例的源码清单（以及一些未在文本出现的其他代码）可以在本书的网页 https://manning.com/books/secrets-of-the-javascript-ninja-second-edition 或异步社区（www.epubit.com.cn）下载。

本书的示例代码按章节分类，每一章为一个文件夹。文件夹排列顺序由本地 Web 服务器完成，如 Apache HTTP 服务器。将下载的代码解压缩到所选择的文件夹，并将该文件夹设置为应用程序的根目录。

除了少数示例外，大多数示例都不需要 Web 服务器，如果需要，可以直接加载到浏览器中执行。

在线交流

本书作者和 Manning 出版社邀请读者访问本书的论坛，该论坛由 Manning 出版社直接运营，在论坛上你可以评论本书、询问技术问题，并获得作者和其他读者的帮助。在浏览器上登录 https://manning.com/books/secrets-of-the-javascript-ninja-second-edition，访问并订阅论坛，然后单击作者在线链接。作者在线页面提供关于如何注册并登录论坛，可以获得哪些帮助，以及论坛行为规则等相关信息。

Manning 承诺为读者提供一个读者与作者能够进行有意义交流的场所。Manning 不强制要求作者的参与次数，对本书论坛的贡献仍然是自愿的（无报酬）。我们建议读者尝试询问一些有挑战性的问题，以免作者丧失兴趣！

本书在售期间，在线交流论坛和先前发布的讨论帖都可以在出版商的网站上访问。

封面插图简介

《JavaScript 忍者秘籍》（第 2 版）封面上的图像是“能乐剧（Noh）演员，武士”，是 19 世纪中期一位不知名的日本艺术家制作的木刻版画。能乐剧（Noh）衍生自日语**天赋**和**技能**，是从 14 世纪开始出现的一种经典音乐剧。许多人物都戴着面具，男性同时扮演男性和女性的角色。作为日本数百年来的英雄人物，武士经常在表演中出现。在本书封面中，精致的服装和威武的雄姿展示了这位艺术家的精湛技艺。

武士和“忍者”都是日本艺术作品中善于打斗的勇士，他们都非常勇敢和精明。武士是精英士兵，受过良好教育，文武双全。战争时，武士们穿着精致的盔甲和彩色的服装，震慑对方。“忍者”则是通过武术技能筛选，而不是依仗社会地位或受教育水平。“忍者”们身着黑色服装，蒙面，单独行动或小组出动，以诡计或隐身攻击敌人，千方百计地完成任务。他们的编码唯一，并且保密。

封面插图是 Manning 出版社的一位编辑多年来收集到的三个日本人物版画中的一个。当我们在为本书寻找“忍者”封面时，这幅引人注目的武士版画吸引了我们，该版画细节精致、色彩鲜明，生动地描绘出一位威武的武士志在必得的决心。

有时很难区分不同的计算机图书。Manning 非常有创意，使用 200 年前的版画作为计算机图书的封面，这些插画描绘世界各地丰富多彩的传统服装，将其印刷在封面上，带来新的活力。

目录

第 1 部分　热身

第 2 部分　理解函数