D.1.3　JSPM

JSPM 是使用 SystemJS 构建的包管理器，用动态模块加载。这个包管理器本身与 npm 类似，但其包仓库与注册无关。在 npm、GitHub 或自定义仓库中注册包，都可以使用 JSPM 的 CLI 安装。JSPM 不会在服务器上构建和预编译资源，而是通过 SystemJS 按需将包交付给客户端。与 Bower 类似，JSPM 也使用打平的依赖结构。

D.1.4　Yarn

Yarn 是 Facebook 开发的定制包管理器，从很多方面看是 npm 的升级版。Yarn 可以通过自己的注册表访问相同的 npm 包，并且安装方式与 npm 也相同。Yarn 和 npm 的主要区别是提供了加速安装、包缓存、锁文件等功能，且提供了改进了包安全功能。

D.2　模块加载器

模块加载器可以让项目按需从服务器获取模块，而不是一次性加载所有模块或包含所有模块的 JS 文件。ECMAScript 6 模块规范定义了浏览器原生支持动态模块加载的最终目标。但现在，仍有很多浏览器不支持 ES6 模块加载。因此，模块加载器作为某种腻子脚本，可以让客户端实现动态模块加载。

D.2.1　SystemJS

SystemJS 模块加载器可以在服务器上使用，也可以在客户端使用。它支持所有模块格式，包括 AMD、CommonJS、UMD 和 ES6；也支持浏览器内转译（考虑到性能，不推荐在大型项目中使用）。

D.2.2　RequireJS

RequireJS 构建于 AMD 模块规范之上，支持特别旧的浏览器。虽然 RequireJS 经实践证明很不错，但 JavaScript 社区整体上还是会抛弃 AMD 模块格式。因此不推荐在大型项目中使用 RequireJS。

D.3　模块打包器

模块打包器可以将任意格式、任意数量的模块合并为一个或多个文件，供客户端加载。模块打包器会分析应用程序的依赖图并按需排序模块。一般来说，应用程序最终只需要一个打包后的文件，但多个结果文件也是可以配置生成的。模块打包器有时候也支持打包原始或编译的 CSS 资源。最终生成的文件可以自执行，也可以多个资源拼接在一起按需执行。

D.3.1　Webpack

Webpack 拥有强大的功能和可扩展能力，是今天非常流行的打包工具。Webpack 可以绑定不同的模块类型，支持多种插件，且完全兼容大多数模板和转译库。

D.3.2　JSPM

JSPM 是构建在 SystemJS 和 ES6 模块加载器之上的包管理器。JSPM 建议的一个工作流是把所有模

块打包到一个文件，然后通过 SystemJS 加载。可以通过 JSPM CLI 使用这个功能。

D.3.3　Browserify

Browserify 是稍微有点历史但久经考验的模块打包器，支持 Node.js 的 CommonJS `require()`依赖语法。

D.3.4　Rollup

Rollup 在模块打包能力方面与 Browserify 类似，但内置了摇树优化功能。Rollup 可以解析应用程序的依赖图，排除没有实际使用的模块。

D.4　编译/转译工具及静态类型系统

在代码编辑器中写的 Web 应用程序代码通常不是实际发送给浏览器的代码。开发者通常希望使用很新的 ECMAScript 特性，而这些特性未必所有浏览器都支持。此外，开发者也经常希望使用静态类型系统或特性在 ECMAScript 规范之外强化自己的代码。有很多工具可以满足上述需求。

D.4.1　Babel

Babel 是将最新 ECMAScript 规范代码编译为兼容 ECMA 版本的一个常用工具。Babel 也支持 React 的 JSX，支持各种插件，与所有主流构建工具兼容。

D.4.2　Google Closure Compiler

Google Closure Compiler 是强大的 JavaScript 编译器，能够执行各种级别的编译优化，同时也是稳健的静态类型检查系统。其类型注解要求以 JSDoc 风格编写。

D.4.3　CoffeeScript

CoffeeScript 是 ECMAScript 语法的增强版，可以直接编译为常规 JavaScript。CoffeeScript 中绝大部分是表达式，这是受到了 Ruby、Python 和 Haskell 的启发。

D.4.4　TypeScript

微软的 TypeScript 是 JavaScript 支持类型的超集，增加了稳健的静态类型检查和主要语法增强。因为它是 JavaScript 严格的超集，所以常规 JavaScript 代码也是有效的 TypeScript 代码。TypeScript 也可以使用类型定义文件指定已有 JavaScript 库的类型信息。

D.4.5　Flow

Flow 是 Facebook 推出的简单的 JavaScript 类型注解系统，其类型语法与 TypeScript 非常相似，但除了类型声明没有增加其他语言特性。

D.5　高性能脚本工具

关于 JavaScript 的一个常见批评是运行速度慢，不适合要求很高的计算。无论这里所说的“慢”是否符合实际，毋庸置疑的是这门语言从一开始就没有考虑支持敏捷的计算。为解决性能问题，有很多项目致力于改造浏览器执行代码的方式，以便让 JavaScript 代码的速度可以接近原生代码速度，同时利用硬件优化。

D.5.1　WebAssembly

WebAssembly 项目（简称 Wasm）正在实现一门语言，该语言可以在多处执行（可移植）并以二进制语言形式存在，可以作为多种低级语言（如 C++和 Rust）的编译目标。WebAssembly 代码在浏览器的一个与 JavaScript 完全独立的虚拟机中运行，与各种浏览器 API 交互的能力极为有限。它可以与 JavaScript 和 DOM 以间接、受限的方式交互，但其更大的目标是创造一门可以在 Web 浏览器中（以及在任何地方）运行的速度极快的语言，并提供接近原生的性能和硬件加速。WebAssembly 系列规范在 2019 年 12 月 5 日已成为 W3C 的正式推荐标准，是浏览器技术中非常值得期待的领域。

D.5.2　asm.js

asm.js 的理论基础是 JavaScript 编译后比硬编码 JavaScript 运行得更快。asm.js 是 JavaScript 的子集，可以作为低级语言的编译目标，并在常规浏览器或 Node.js 引擎中执行。现代 JavaScript 引擎在运行时推断类型，而 asm.js 代码通过使用词法提示将这些类型推断（及其相关操作）的计算大大降低。asm.js 广泛使用了定型数组（`TypedArray`），相比常规的 JavaScript 数组能够显著提升性能。asm.js 没有 WebAssembly 快，但通过编译显著提升了性能。

D.5.3　Emscripten 与 LLVM

虽然 Emscripten 从未在浏览器中执行，但它是重要的工具包，可以将低级代码编译为 WebAssembly 和 asm.js。Emscripten 使用 LLVM 编译器将 C、C++和 Rust 代码编译为可以直接在浏览器中运行的代码（asm.js），或者可以在浏览器虚拟机中执行的代码（WebAssembly）。

D.6　编辑器

VIM、Emacs 及其同类的文本编辑器非常优秀，但随着构建环境和项目规模逐渐复杂，编辑器最好能够自动化常见任务，如代码自动完成、文件自动格式化、自动检查代码错误、自动补足项目目录。目前有很多编辑器和 IDE 支持这些功能，既有免费的也有收费的。

D.6.1　Sublime Text

Sublime Text 是比较流行的闭源文本编辑器。它可用于开发各种语言，还提供了大量可扩展的插件，由社区来维护。Sublime Text 的性能非常突出。

- **类型**：收费

D.6.2　Atom

Atom 是 GitHub 的开源编辑器，与 Sublime Text 有很多相同的特性，如社区在蓬勃发展且拥有第三方扩展包。Atom 的性能稍差，但它在不断地提升。

- **类型**：免费

D.6.3　Brackets

Brackets 是 Adobe 的开源编辑器，与 Atom 类似。但 Brackets 是专门为 Web 开发者设计的，提供了许多非常令人印象深刻的、面向前端编码的独特功能。该编辑器还有丰富的插件。

- **类型**：免费

D.6.4　Visual Studio Code

微软的 Visual Studio Code 是基于 Electron 框架的开源代码编辑器。与其他主流编辑器一样，Visual Studio Code 是高度可扩展的。

- **类型**：免费

D.6.5　WebStorm

WebStorm 是 JetBrains 的高性能 IDE，号称终极项目开发工具包，集成了前沿的前端框架，也集成了大多数构建工具和版本控制系统。

- **类型**：免费试用；之后收费。

D.7　构建工具、自动化系统和任务运行器

把本地开发的项目目录转换为线上应用程序需要一系列步骤。每个步骤都需要细分为很多子任务，如构建和部署应用程序要涉及模块打包、编译、压缩和发布静态资源，等等。运行单元和集成测试也涉及初始化测试套件和控制无头浏览器。为了让管理和使用这些任务更容易，也出现了很多工具可以用来更高效地组织和拼接这些任务。

D.7.1　Grunt

Grunt 是在 Node.js 环境下运行的任务运行器，使用配置对象声明如何执行任务。Grunt 有庞大的社区和众多插件可以支持项目构建。

D.7.2　Gulp

与 Grunt 类似，Gulp 也是在 Node.js 环境下运行的任务运行器。Gulp 使用 UNIX 风格的管道方式定义任务，每个任务表现为一个 JavaScript 函数。Gulp 也有活跃的社区和丰富的扩展。

D.7.3　Brunch

Brunch 也是 Node.js 构建工具，旨在简化配置，方便使用。Brunch 虽然比 Gulp 和 Grunt 出现得晚，但仍有很多插件可以选择。

D.7.4　npm

npm 严格来讲不是构建工具，但它提供了脚本功能，很多项目会利用这个功能融合任务运行器。脚本是在 package.json 中定义的。

D.8　代码检查和格式化

JavaScript 代码调试有一个问题，没有多少 IDE 可以在输入代码时提示代码错误。大多数开发者是写一段代码，然后在浏览器里刷新看看有没有错误。在部署之前验证 JavaScript 代码可以显著减少线上错误。**代码检查器**（linter）可以检查基本的语法并提供关于风格的警告。

格式化器（formatter）是一种工具，可以分析语法规则并实现自动缩进、加空格和对齐代码等操作，也可以自定义完成对文件内容的其他操作。格式化器不会破坏或修改代码或者代码的语义，因为它们可以避免做出影响代码执行的修改。

D.8.1　ESLint

ESLint 是开源的 JavaScript 代码检查器，由本书前几版的作者 Nicholas Zakas 独立开发；完全“可插拔”，以常识化规则作为默认规则，支持配置；有大型可修改和可切换的规则库，可以用来调试工具的行为。

D.8.2　Google Closure Compiler

Google Closure Compiler 内置了一个代码检查工具，可以通过命令行参数激活。这个代码检查器基于代码的抽象语法树工作，因此不会检查空格、缩进或其他不影响代码执行代码组织问题。

D.8.3　JSLint

JSLint 是 Douglas Crockford 开发的 JavaScript 验证器。JSLint 从核心层面检查语法错误，以最大限度保证跨浏览器兼容作为最低要求。（JSLint 遵循最严格的规则以确保代码最大的兼容性。）可以启动 Crockford 关于代码风格的警告，包括代码格式、使用未声明的变量，等等。JSLint 虽然是使用 JavaScript 写的，但可以通过基于 Java 的 Rhino 解释器在命令行执行，也可以通过 WScript 或其他 JavaScript 解释器执行。它的网站提供了针对每个命令行解释器的自定义版。

D.8.4　JSHint

JSHint 是 JSLint 的分支，支持对检查规则更宽泛的自定义。与 JSLint 类似，JSHint 也先检查语法错误，然后再检查有问题的代码模式。JSLint 的每项检查 JSHint 中也都有，但开发者可以更好地控制应用哪些规则。同样与 JSLint 类似，JSHint 可以使用 Rhino 在命令行中执行。

D.8.5　ClangFormat

ClangFormat 是构建在 Clang 项目的 LibFormat 库基础上的格式化工具。它使用了 Clang 格式化规则自动重新组织代码（不会改变语义结构）。ClangFormat 可以在命令行中使用，也可以集成到编辑器里。

D.9　压缩工具

JavaScript 构建过程的一个重要环节就是压缩输出，剔除多余字符。这样可以保证只将最少的字节量传输到浏览器进行解析，用户体验会更好。有不少**压缩工具**，它们的压缩率有所不同。

D.9.1　Uglify

Uglify 现在是第 3 版[①]，是可以压缩、美化和最小化 JavaScript 代码的工具包。它可以在命令行运行，可以接收极为丰富的配置选项，实现满足需求的自定义压缩。

D.9.2　Google Closure Compiler

虽然严格来讲并不是压缩工具，但 Google Closure Compiler 也在其优化工具中提供了不同级别的优化，能够缩小代码体积。

D.9.3　JSMin

JSMin 是 Douglas Crockford 用 C 语言写的一个代码压缩程序，能对 JavaScript 进行基本的压缩。它主要用于删除空格和注释，确保结果可以正确运行。JSMin 也提供了 Window 可执行文件，有 C 语言和其他语言的源代码。

D.9.4　Dojo ShrinkSafe

Dojo Toolkit 团队开发的 ShrinkSafe 会使用 Rhino 先把 JavaScript 代码解析为符号流，然后再安全地压缩代码。与 JSMin 一样，ShrinkSafe 也会删除多余的空格（但不删除换行）和注释，且会更进一步将局部变量名替换为两个字符的变量名。因此结果比 JSMin 压缩后的更小，不会引入语法错误。

D.10　单元测试

大多数 JavaScript 库会使用某种形式的单元测试来测试自己的代码，有的还会将自己的单元测试框架公之于众，供他人使用。**测试驱动开发**（TDD，Test Driven Development）是以单元测试为中心的软件开发过程。

D.10.1　Mocha

Mocha 是目前非常流行的单元测试框架，为开发单元测试提供了优秀的配置能力和可扩展性。Mocha 的测试非常灵活，顺序执行可以保证生成准确的报告且更容易调试。

D.10.2　Jasmine

Jasmine 虽然是比较老的单元测试框架，但仍非常流行。它内置了单元测试所需的一切，没有外部依赖，而且语法简单易读。

---

① 编辑本书时仍是第 3 版。——编者注

D.10.3　qUnit

qUnit 是为 jQuery 设计的单元测试框架。事实上，jQuery 本身在所有测试中都使用 qUnit。除此之外，qUnit 对 jQuery 没有依赖，可用于测试任何 JavaScript 代码。qUnit 非常简单，容易上手。

D.10.4　JsUnit

JsUnit 是早期的 JavaScript 单元测试库，不依赖任何 JavaScript 库。JsUnit 是流行的 Java 测试框架 JUnit 的端口。测试在页面中运行，可以设置为自动测试并将结果提交给服务器。JsUnit 的网站上包含示例和文档。

D.10.5　Dojo Object Harness

Dojo Object Harness（DOH）最初是 Dojo 内部的单元测试工具，后来开放给所有人使用。与其他框架一样，DOH 的测试也是在浏览器中运行的。

D.11　文档生成器

大多数 IDE 包含主语言的文档生成器。因为 JavaScript 没有官方 IDE，所以过去文档要么手动生成，要么借用其他语言的文档生成器生成。不过，目前已出现了一些面向 JavaScript 的文档生成器。

D.11.1　ESDoc

ESDoc 能够为 JavaScript 代码生成非常高级的文档页面，包括从文档页面链接到源代码的功能。ESDoc 还有一个插件库可以扩展其功能。不过，ESDoc 要求代码必须使用 ES6 模块。

D.11.2　documentation.js

documentation.js 可以处理代码中的 JSDoc 注释，自动生成 HTML、Markdown 或 JSON 格式的文档。它兼容最新版本的 ECMAScript 和所有主流构建工具，也支持 Flow 的注解。

D.11.3　Docco

按照其网站的描述，Docco 是“简单快捷”的文档生成器。这个工具的理念是以简单的方式生成描述代码的 HTML 页面。Docco 在某些情况下会出问题，但它确实是生成代码文档的极简方法。

D.11.4　JsDoc Toolkit

JsDoc Toolkit 是早期的 JavaScript 文档生成器。它要求代码中包含 Javadoc 风格的注释，然后可以基于这些注释生成 HTML 文件。可以使用预置的 JsDoc 模板或自己创建的模式来自定义生成的 HTML 页面格式。JsDoc Toolkit 是个 Java 包。

D.11.5　YUI Doc

YUI Doc 是 YUI 的文档生成器。该生成器是用 Python 写的，因此要求安装 Python 运行时。YUI Doc 输出的 HTML 文件中集成了基于 YUI 的自动完成部件的属性和方法搜索功能。与 JsDoc 一样，YUI Doc

要求代码中包含 Javadoc 风格的注释。可以通过修改默认 HTML 模板和关联的样式表来修改默认的 HTML 输出。

D.11.6　AjaxDoc

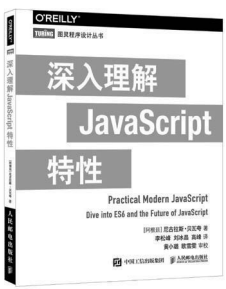
AjaxDoc 的目标与前面的文档生成器稍有不同。它不会为 JavaScript 代码创建 HTML 文件，而是会创建与.NET 语言（如 C#、Visual Basic）兼容的 XML 格式。这样就可以使用标准.NET 文档生成器来创建 HTML 文档。AjaxDoc 要求所有文档注释的格式与.NET 语言的文档注释格式类似。AjaxDoc 是为 ASP.NET Ajax 解决方案而创建的，但也可以用于独立的项目。

技术改变世界 · 阅读塑造人生

你不知道的 JavaScript

- **深入挖掘JavaScript语言本质，简练形象地解释抽象概念，打通JavaScript的任督二脉**

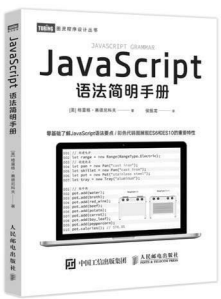
（上卷）书号：978-7-115-38573-4 定价：49.00 元
（中卷）书号：978-7-115-43116-5 定价：79.00 元
（下卷）书号：978-7-115-47165-9 定价：79.00 元



深入理解 JavaScript 特性

- **JavaScript之父Brendan Eich作序推荐**
- **将JavaScript新特性融入简单易懂的示例中，助你大幅提升代码表达能力**

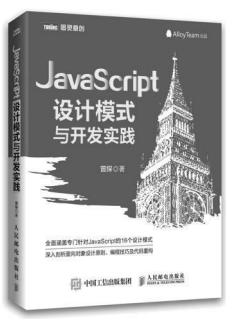
书号：978-7-115-51040-2
定价：79.00 元



JavaScript 语法简明手册

- **零基础了解JavaScript语法要点**
- **彩色代码图展现ES6和ES10的重要特性**

书号：978-7-115-53992-2
定价：79.00 元



JavaScript 设计模式与开发实践

- **腾讯前端Alloy Team团队出品，资深前端工程师曾探力作**
- **全面涵盖专门针对JavaScript的16个设计模式**
- **深入剖析面向对象设计原则、面向对象编程技巧及代码重构**

书号：978-7-115-38888-9
定价：59.00 元

微信连接

回复“JavaScript”查看相关书单

微博连接

关注@图灵教育 每日分享IT好书

QQ连接

图灵读者官方群I：218139230
图灵读者官方群II：164939616

图灵社区
iTuring.cn

在线出版，电子书，《码农》杂志，图灵访谈