

ECMAScript 6 入门

作者：阮一峰

授权：署名-非商用许可证



目录

- 0.前言
- 1.ECMA Script 6简介
- 2.let 和 const 命令
- 3.变量的解构赋值
- 4.字符串的扩展
- 5.字符串的新增方法
- 6.正则的扩展
- 7.数值的扩展
- 8.函数的扩展
- 9.数组的扩展
- 10.对象的扩展
- 11.对象的新增方法
- 12.Symbol
- 13.Set 和 Map 数据结构
- 14.Proxy
- 15.Reflect
- 16.Promise 对象
- 17.Iterator 和 for...of 循环
- 18.Generator 函数的语法
- 19.Generator 函数的异步应用
- 20.async 函数
- 21.Class 的基本语法
- 22.Class 的继承
- 23.Module 的语法
- 24.Module 的加载实现
- 25.编程风格
- 26.读懂规格
- 27.异步遍历器
- 28.ArrayBuffer
- 29.最新提案
- 30.Decorator
- 31.参考链接

其他

- 源码
- 修订历史
- 反馈意见

参考链接

- 1.官方文件
- 2.综合介绍
- 3.let 和 const

4.解构赋值
5.字符串
6.正则
7.数值
8.数组
9.函数
10.对象
11.Symbol
12.Set 和 Map
13.Proxy 和 Reflect
14.Promise 对象
15.Iterator
16.Generator
17.异步操作和 Async 函数
18.Class
19.Decorator
20.Module
21.二进制数组
22.SIMD
23.工具

《ES6 实战教程》 深入学习一线大厂必备 ES6 技能。VIP 教程限时免费领取。 [← 立即查看](#)

1. 官方文件

- ECMAScript® 2015 Language Specification: ECMAScript 2015 规格
- ECMAScript® 2016 Language Specification: ECMAScript 2016 规格
- ECMAScript® 2017 Language Specification: ECMAScript 2017 规格（草案）
- ECMAScript Current Proposals: ECMAScript 当前的所有提案
- ECMAScript Active Proposals: 已经进入正式流程的提案
- ECMAScript proposals: 从阶段 0 到阶段 4 的所有提案列表
- TC39 meeting agendas: TC39 委员会历年的会议记录
- ECMAScript Daily: TC39 委员会的动态
- The TC39 Process: 提案进入正式规格的流程
- TC39: A Process Sketch, Stages 0 and 1: Stage 0 和 Stage 1 的含义
- TC39 Process Sketch, Stage 2: Stage 2 的含义

2. 综合介绍

- Axel Rauschmayer, Exploring ES6: Upgrade to the next version of JavaScript: ES6 的专著，本书的许多代码实例来自该书
- Sayanee Basu, Use ECMAScript 6 Today
- Ariya Hidayat, Toward Modern Web Apps with ECMAScript 6
- Dale Schouten, 10 EcmaScript-6 tricks you can perform right now
- Colin Toh, Lightweight ES6 Features That Push A Branch: ES6 的一些“轻量级”的特性介绍

- Domenic Denicola, [ES6: The Awesome Parts](#)
 - Nicholas C. Zakas, [Understanding ECMAScript 6](#)
 - Justin Drake, [ECMAScript 6 in Node.JS](#)
 - Ryan Dao, [Summary of ECMAScript 6 major features](#)
 - Luke Hoban, [ES6 features: ES6 新语法点的罗列](#)
 - Traceur-compiler, [Language Features](#): Traceur 文档列出的一些 ES6 例子
 - Axel Rauschmayer, [ECMAScript 6: what's next for JavaScript?](#): 关于 ES6 新增语法的综合介绍，有很多例子
 - Axel Rauschmayer, [Getting started with ECMAScript 6](#): ES6 语法点的综合介绍
 - Toby Ho, [ES6 in io.js](#)
 - Guillermo Rauch, [ECMAScript 6](#)
 - Benjamin De Cock, [Frontend Guidelines: ES6 最佳实践](#)
 - Jani Hartikainen, [ES6: What are the benefits of the new features in practice?](#)
 - kangax, [JavaScript quiz. ES6 edition: ES6 小测试](#)
 - Jeremy Fairbank, [HTML5DevConf ES7 and Beyond!:](#) ES7 新增语法点介绍
 - Timothy Gu, [How to Read the ECMAScript Specification](#): 如何读懂 ES6 规格
-

3. let 和 const

- Kyle Simpson, [For and against let](#): 讨论 let 命令的作用域
 - kangax, [Why typeof is no longer "safe"](#): 讨论在块级作用域内，let 命令的变量声明和赋值的行为
 - Axel Rauschmayer, [Variables and scoping in ECMAScript 6](#): 讨论块级作用域与 let 和 const 的行为
 - Nicolas Bevacqua, [ES6 Let, Const and the "Temporal Dead Zone" \(TDZ\) in Depth](#)
 - acorn, [Function statements in strict mode](#): 块级作用域对严格模式的函数声明的影响
 - Axel Rauschmayer, [ES proposal: global](#): 顶层对象 `global`
 - Mathias Bynens, [A horrifying `globalThis` polyfill in universal JavaScript](#): 如何写 `globalThis` 的垫片库
-

4. 解构赋值

- Nick Fitzgerald, [Destructuring Assignment in ECMAScript 6](#): 详细介绍解构赋值的用法
 - Nicholas C. Zakas, [ECMAScript 6 destructuring gotcha](#)
-

5. 字符串

- Nicholas C. Zakas, [A critical review of ECMAScript 6 quasi-literals](#)
- Mozilla Developer Network, [Template strings](#)
- Addy Osmani, [Getting Literal With ES6 Template Strings](#): 模板字符串的介绍
- Blake Winton, [ES6 Templates](#): 模板字符串的介绍
- Peter Jaskowiak, [How to write a template compiler in JavaScript](#): 使用模板字符串，编写一个模板编译函数
- Axel Rauschmayer, [ES.stage3: string padding](#)

6. 正则

- Mathias Bynens, [Unicode-aware regular expressions in ES6](#): 详细介绍正则表达式的 u 修饰符
 - Axel Rauschmayer, [New regular expression features in ECMAScript 6](#): ES6 正则特性的详细介绍
 - Yang Guo, [RegExp lookbehind assertions](#): 介绍后行断言
 - Axel Rauschmayer, [ES proposal: RegExp named capture groups](#): 具名组匹配的介绍
 - Mathias Bynens, [ECMAScript regular expressions are getting better!](#): 介绍 ES2018 添加的多项正则语法
-

7. 数值

- Nicolas Bevacqua, [ES6 Number Improvements in Depth](#)
 - Axel Rauschmayer, [ES proposal: arbitrary precision integers](#)
 - Mathias Bynens, [BigInt: arbitrary-precision integers in JavaScript](#)
-

8. 数组

- Axel Rauschmayer, [ECMAScript 6's new array methods](#): 对 ES6 新增的数组方法的全面介绍
 - TC39, [Array.prototype.includes](#): 数组的 includes 方法的规格
 - Axel Rauschmayer, [ECMAScript 6: holes in Arrays](#): 数组的空位问题
-

9. 函数

- Nicholas C. Zakas, [Understanding ECMAScript 6 arrow functions](#)
 - Jack Franklin, [Real Life ES6 - Arrow Functions](#)
 - Axel Rauschmayer, [Handling required parameters in ECMAScript 6](#)
 - Dmitry Soshnikov, [ES6 Notes: Default values of parameters](#): 介绍参数的默认值
 - Ragan Wald, [Destructuring and Recursion in ES6](#): rest 参数和扩展运算符的详细介绍
 - Axel Rauschmayer, [The names of functions in ES6](#): 函数的 name 属性的详细介绍
 - Kyle Simpson, [Arrow This](#): 箭头函数并没有自己的 this
 - Derick Bailey, [Do ES6 Arrow Functions Really Solve "this" In JavaScript?](#): 使用箭头函数处理 this 指向, 必须非常小心
 - Mark McDonnell, [Understanding recursion in functional JavaScript programming](#): 如何自己实现尾递归优化
 - Nicholas C. Zakas, [The ECMAScript 2016 change you probably don't know](#): 使用参数默认值时, 不能在函数内部显式开启严格模式
 - Axel Rauschmayer, [ES proposal: optional catch binding](#)
 - Cynthia Lee, [When you should use ES6 arrow functions—and when you shouldn't](#): 讨论箭头函数的适用场合
 - Eric Elliott, [What is this?](#): 箭头函数内部的 this 的解释。
-

10. 对象

- Addy Osmani, [Data-binding Revolutions with Object.observe\(\)](#): 介绍 `Object.observe()` 的概念
 - Sella Rafaeli, [Native JavaScript Data-Binding](#): 如何使用 `Object.observe` 方法, 实现数据对象与 DOM 对象的双向绑定
 - Axel Rauschmayer, [__proto__ in ECMAScript 6](#)
 - Axel Rauschmayer, [Enumerability in ECMAScript 6](#)
 - Axel Rauschmayer, [ES proposal: Object.getOwnPropertyDescriptors\(\)](#)
 - TC39, [Object.getOwnPropertyDescriptors Proposal](#)
 - David Titarenco, [How Spread Syntax Breaks JavaScript](#): 扩展运算符的一些不合理的地方
-

11. Symbol

- Axel Rauschmayer, [Symbols in ECMAScript 6: Symbol 简介](#)
 - MDN, [Symbol](#): Symbol 类型的详细介绍
 - Jason Orendorff, [ES6 In Depth: Symbols](#)
 - Keith Cirkel, [Metaprogramming in ES6: Symbols and why they're awesome](#): Symbol 的深入介绍
 - Axel Rauschmayer, [Customizing ES6 via well-known symbols](#)
 - Derick Bailey, [Creating A True Singleton In Node.js, With ES6 Symbols](#)
 - Das Surma, [How to read web specs Part IIa – Or: ECMAScript Symbols](#): 介绍 Symbol 的规格
-

12. Set 和 Map

- Mozilla Developer Network, [WeakSet](#): 介绍 WeakSet 数据结构
 - Dwayne Charrington, [What Are Weakmaps In ES6?: WeakMap 数据结构介绍](#)
 - Axel Rauschmayer, [ECMAScript 6: maps and sets](#): Set 和 Map 结构的详细介绍
 - Jason Orendorff, [ES6 In Depth: Collections](#): Set 和 Map 结构的设计思想
 - Axel Rauschmayer, [Converting ES6 Maps to and from JSON](#): 如何将 Map 与其他数据结构互相转换
-

13. Proxy 和 Reflect

- Nicholas C. Zakas, [Creating defensive objects with ES6 proxies](#)
- Axel Rauschmayer, [Meta programming with ECMAScript 6 proxies](#): Proxy 详解
- Daniel Zautner, [Meta-programming JavaScript Using Proxies](#): 使用 Proxy 实现元编程
- Tom Van Cutsem, [Harmony-reflect](#): Reflect 对象的设计目的
- Tom Van Cutsem, [Proxy Traps](#): Proxy 拦截操作一览
- Tom Van Cutsem, [Reflect API](#)
- Tom Van Cutsem, [Proxy Handler API](#)
- Nicolas Bevacqua, [ES6 Proxies in Depth](#)
- Nicolas Bevacqua, [ES6 Proxy Traps in Depth](#)
- Nicolas Bevacqua, [More ES6 Proxy Traps in Depth](#)
- Axel Rauschmayer, [Pitfall: not all objects can be wrapped transparently by proxies](#)
- Bertalan Miklos, [Writing a JavaScript Framework - Data Binding with ES6 Proxies](#): 使用 Proxy 实现观察者模式
- Keith Cirkel, [Metaprogramming in ES6: Part 2 - Reflect](#): Reflect API 的详细介绍

14. Promise 对象

- Jake Archibald, [JavaScript Promises: There and back again](#)
 - Jake Archibald, [Tasks, microtasks, queues and schedules](#)
 - Tilde, [rsvp.js](#)
 - Sandeep Panda, [An Overview of JavaScript Promises: ES6 Promise 入门介绍](#)
 - Dave Atchley, [ES6 Promises: Promise 的语法介绍](#)
 - Axel Rauschmayer, [ECMAScript 6 promises \(2/2\): the API](#): 对 ES6 Promise 规格和用法的详细介绍
 - Jack Franklin, [Embracing Promises in JavaScript: catch 方法的例子](#)
 - Ronald Chen, [How to escape Promise Hell](#): 如何使用 `Promise.all` 方法的一些很好的例子
 - Jordan Harband, [proposal-promise-try: Promise.try\(\) 方法的提案](#)
 - Sven Slootweg, [What is Promise.try, and why does it matter?: Promise.try\(\) 方法的优点](#)
 - Yehuda Katz, [TC39: Promises, Promises: Promise.try\(\) 的用处](#)
-

15. Iterator

- Mozilla Developer Network, [Iterators and generators](#)
 - Mozilla Developer Network, [The Iterator protocol](#)
 - Jason Orendorff, [ES6 In Depth: Iterators and the for-of loop](#): 遍历器与 `for...of` 循环的介绍
 - Axel Rauschmayer, [Iterators and generators in ECMAScript 6](#): 探讨 Iterator 和 Generator 的设计目的
 - Axel Rauschmayer, [Iterables and iterators in ECMAScript 6](#): Iterator 的详细介绍
 - Kyle Simpson, [Iterating ES6 Numbers](#): 在数值对象上部署遍历器
-

16. Generator

- Matt Baker, [Replacing callbacks with ES6 Generators](#)
- Steven Sanderson, [Experiments with Koa and JavaScript Generators](#)
- jmar777, [What's the Big Deal with Generators?](#)
- Marc Harter, [Generators in Node.js: Common Misconceptions and Three Good Use Cases](#): 讨论 Generator 函数的作用
- StackOverflow, [ES6 yield : what happens to the arguments of the first call next\(\)?](#): 第一次使用 `next` 方法时不能带有参数
- Kyle Simpson, [ES6 Generators: Complete Series](#): 由浅入深探讨 Generator 的系列文章, 共四篇
- Gajus Kuizinas, [The Definitive Guide to the JavaScript Generators](#): 对 Generator 的综合介绍
- Jan Krems, [Generators Are Like Arrays](#): 讨论 Generator 可以被当作数据结构看待
- Harold Cooper, [Coroutine Event Loops in JavaScript](#): Generator 用于实现状态机
- Ruslan Ismagilov, [learn-generators](#): 编程练习, 共 6 道题
- Steven Sanderson, [Experiments with Koa and JavaScript Generators](#): Generator 入门介绍, 以 Koa 框架为例
- Mahdi Dibaiee, [ES7 Array and Generator comprehensions](#): ES7 的 Generator 推导
- Nicolas Bevacqua, [ES6 Generators in Depth](#)
- Axel Rauschmayer, [ES6 generators in depth: Generator 规格的详尽讲解](#)
- Derick Bailey, [Using ES6 Generators To Short-Circuit Hierarchical Data Iteration](#): 使用 `for...of` 循环完成预定的操作步骤

17. 异步操作和 Async 函数

- Luke Hoban, [Async Functions for ECMAScript](#): Async 函数的设计思想, 与 Promise、Generator 函数的关系
 - Jafar Husain, [Asynchronous Generators for ES7](#): Async 函数的深入讨论
 - Nolan Lawson, [Taming the asynchronous beast with ES7](#): async 函数通俗的实例讲解
 - Jafar Husain, [Async Generators](#): 对 async 与 Generator 混合使用的一些讨论
 - Daniel Brain, [Understand promises before you start using async/await](#): 讨论 async/await 与 Promise 的关系
 - Jake Archibald, [Async functions - making promises friendly](#)
 - Axel Rauschmayer, [ES proposal: asynchronous iteration](#): 异步遍历器的详细介绍
 - Dima Grossman, [How to write async await without try-catch blocks in JavaScript](#): 除了 try/catch 以外的 async 函数内部捕捉错误的方法
 - Mostafa Gaafa, [6 Reasons Why JavaScript's Async/Await Blows Promises Away](#): Async 函数的6个好处
 - Mathias Bynens, [Asynchronous stack traces: why await beats Promise#then\(\)](#): async 函数可以保留错误堆栈
-

18. Class

- Sebastian Porto, [ES6 classes and JavaScript prototypes](#): ES6 Class 的写法与 ES5 Prototype 的写法对比
 - Jack Franklin, [An introduction to ES6 classes](#): ES6 class 的入门介绍
 - Axel Rauschmayer, [ECMAScript 6: new OOP features besides classes](#)
 - Axel Rauschmayer, [Classes in ECMAScript 6 \(final semantics\)](#): Class 语法的详细介绍和设计思想分析
 - Eric Faust, [ES6 In Depth: Subclassing](#): Class 语法的深入介绍
 - Nicolás Bevacqua, [Binding Methods to Class Instance Objects](#): 如何绑定类的实例中的 this
 - Jamie Kyle, [JavaScript's new #private class fields](#): 私有属性的介绍。
 - Mathias Bynens, [Public and private class fields](#): 实例属性的新写法的介绍。
-

19. Decorator

- Maximiliano Fierro, [Declarative vs Imperative: Decorators 和 Mixin 介绍](#)
 - Justin Fagnani, ["Real" Mixins with JavaScript Classes](#): 使用类的继承实现 Mixin
 - Addy Osmani, [Exploring ES2016 Decorators](#): Decorator 的深入介绍
 - Sebastian McKenzie, [Allow decorators for functions as well](#): 为什么修饰器不能用于函数
 - Maximiliano Fierro, [Traits with ES7 Decorators](#): Trait 的用法介绍
 - Jonathan Creamer, [Using ES2016 Decorators to Publish on an Event Bus](#): 使用修饰器实现自动发布事件
-

20. Module

- Jack Franklin, [JavaScript Modules the ES6 Way](#): ES6 模块入门
- Axel Rauschmayer, [ECMAScript 6 modules: the final syntax](#): ES6 模块的介绍, 以及与 CommonJS 规格的详细比较
- Dave Herman, [Static module resolution](#): ES6 模块的静态化设计思想
- Jason Orendorff, [ES6 In Depth: Modules](#): ES6 模块设计思想的介绍
- Ben Newman, [The Importance of import and export](#): ES6 模块的设计思想
- ESDiscuss, [Why is "export default var a = ..."](#) 上一章 S 下一章

- Bradley Meck, [ES6 Module Interoperability](#): 介绍 Node 如何处理 ES6 语法加载 CommonJS 模块
 - Axel Rauschmayer, [Making transpiled ES modules more spec-compliant](#): ES6 模块编译成 CommonJS 模块的详细介绍
 - Axel Rauschmayer, [ES proposal: import\(\) – dynamically importing ES modules](#): import() 的用法
 - Node EPS, [ES Module Interoperability](#): Node 对 ES6 模块的处理规格
-

21. 二进制数组

- Ilmari Heikkinen, [Typed Arrays: Binary Data in the Browser](#)
 - Khronos, [Typed Array Specification](#)
 - Ian Elliot, [Reading A BMP File In JavaScript](#)
 - Renato Mangini, [How to convert ArrayBuffer to and from String](#)
 - Axel Rauschmayer, [Typed Arrays in ECMAScript 6](#)
 - Axel Rauschmayer, [ES proposal: Shared memory and atomics](#)
 - Lin Clark, [Avoiding race conditions in SharedArrayBuffers with Atomics](#): Atomics 对象使用场景的解释
 - Lars T Hansen, [Shared memory - a brief tutorial](#)
 - James Milner, [The Return of SharedArrayBuffers and Atomics](#)
-

22. SIMD

- TC39, [SIMD.js Stage 2](#)
 - MDN, [SIMD](#)
 - TC39, [ECMAScript SIMD](#)
 - Axel Rauschmayer, [JavaScript gains support for SIMD](#)
-

23. 工具

- Babel, [Babel Handbook](#): Babel 的用法介绍
- Google, [traceur-compiler](#): Traceur 编译器
- Casper Beyer, [ECMAScript 6 Features and Tools](#)
- Stoyan Stefanov, [Writing ES6 today with jstransform](#)
- ES6 Module Loader, [ES6 Module Loader Polyfill](#): 在浏览器和 node.js 加载 ES6 模块的一个库，文档里对 ES6 模块有详细解释
- Paul Miller, [es6-shim](#): 一个针对老式浏览器，模拟 ES6 部分功能的垫片库（shim）
- army8735, [JavaScript Downcast](#): 国产的 ES6 到 ES5 的转码器
- esnext, [ES6 Module Transpiler](#): 基于 node.js 的将 ES6 模块转为 ES5 代码的命令行工具
- Sebastian McKenzie, [BabelJS](#): ES6 转译器
- SystemJS, [SystemJS](#): 在浏览器中加载 AMD、CJS、ES6 模块的一个垫片库
- Modernizr, [HTML5 Cross Browser Polyfills](#): ES6 垫片库清单
- Facebook, [regenerator](#): 将 Generator 函数转为 ES5 的转码器

