



ESLint

<https://eslint.bootcss.com/>



你将学到 Get

- 什么是 ESLint
- 如何配置 ESLint

ESLint 是什么

ESLint 最初是由 Nicholas C. Zakas 于 2013 年 6 月创建的开源项目。它的目标是提供一个插件化的 javascript 代码检测工具。

Eslint 是国外的前端大牛 Nicholas C. Zakas 在 2013 年发起的一个开源项目，有一本书被誉为前端界的"圣经"，叫《JavaScript 高级程序设计》(即红宝书)，他正是这本书的作者

尼古拉斯·泽卡斯 Nicholas C.Zakas的作品 (10)



按收藏人数排序 按评分排序 按出版时间排序



JavaScript高级程序设计 (第3版) (2012)

[> 更多版本 \(8\)](#)

[美] Nicholas C. Zakas / 李松峰 / 曹力 / 人民邮电出版社 / 图灵教育

★★★★★ 9.3 / 2590人评价



高性能JavaScript (2010)

[> 更多版本 \(3\)](#)

Nicholas C.Zakas / 丁琛 / 赵泽欣 / 电子工业出版社

★★★★★ 9.0 / 593人评价



编写可维护的JavaScript (2013)

[> 更多版本 \(2\)](#)

扎卡斯 / 李晶 / 郭凯 / 张散集 / 人民邮电出版社

★★★★★ 8.1 / 426人评价



深入理解ES6 (2017)

[> 更多版本 \(2\)](#)

【美】 Nicholas C. Zakas / 刘振涛 / 电子工业出版社 / 博文视点

★★★★★ 9.3 / 376人评价

- ESLint 使用 Espree 解析 JavaScript。
- ESLint 使用 AST 去分析代码中的模式
- ESLint 是完全插件化的。每一个规则都是一个插件并且你可以在运行时添加更多的规则。

初始化 – lint

通过 eslint cli 初始化配置

```
npx eslint --init
```

Bash

- **JavaScript** – 使用 `.eslintrc.js` 然后输出一个配置对象。
- **YAML** – 使用 `.eslintrc.yaml` 或 `.eslintrc.yml` 去定义配置的结构。
- **JSON** – 使用 `.eslintrc.json` 去定义配置的结构，ESLint 的 JSON 文件允许 JavaScript 风格的注释。
- **(弃用)** – 使用 `.eslintrc`，可以使 JSON 也可以是 YAML。
- **package.json** – 在 `package.json` 里创建一个 `eslintConfig` 属性，在那里定义你的配置。

如果同一个目录下有多个配置文件，ESLint 只会使用一个。优先级顺序如下：

1. `.eslintrc.js`
2. `.eslintrc.yaml`
3. `.eslintrc.yml`
4. `.eslintrc.json`
5. `.eslintrc`
6. `package.json`

解析配置 – parserOptions

`.eslintrc.cjs`

```
"parserOptions": {  
  "ecmaVersion": 2015,  
},
```

JSON

- `ecmaVersion` 默认设置为 3, 5（默认），你可以使用 6、7、8、9 你也可以用使用年份命名的版本号指定为 2015（同 6），2016（同 7），或 2017（同 8）或 2018（同 9）或 2019（same as 10）

- `sourceType` 设置为 `"script"` (默认) 或 `"module"` (如果你的代码是 ECMAScript 模块)
- `ecmaFeatures` – 这是个对象, 表示你想使用的额外的语言特性:
 - `globalReturn` – 允许在全局作用域下使用 `return` 语句
 - `impliedStrict` – 启用全局 strict mode (如果 `ecmaVersion` 是 5 或更高)
 - `jsx` – 启用 JSX
 - `experimentalObjectRestSpread` – 启用实验性的 object rest/spread properties 支持。(重要: 这是一个实验性的功能, 在未来可能会有明显改变。建议你写的规则 **不要** 依赖该功能, 除非当它发生改变时你愿意承担维护成本。)

JavaScript

```
var a = 1
// es6
const b = 2
// 模块化方案
export const c = 3
// JSX
export const d = <div></div>
```

解析器设置 – parser

TypeScript

```
{
  parser: '@typescript-eslint/parser', // TS
  parser: 'vue-eslint-parser', // Vue
}
```

env 开发环境与 globals 全局变量


```
env: {
  browser: true,
  es2020: true,
  node: true,
  jest: true
},
globals: {
  ga: true,
  chrome: true,
  __DEV__: true
},
```

.JavaScript

扩展规则 – extends


```
extends: [
  'eslint:recommended',
],
```

JavaScript

值为 `"eslint:recommended"` 的 `extends` 属性启用一系列核心规则，这些规则报告一些常见问题，在 [规则页面](#) 中被标记为 。这个推荐的子集只能在 ESLint 主要版本进行更新。

<https://cn.eslint.org/docs/rules/>

继承配置 – extends

`"eslint:recommended"` 这个扩展包帮我们启用了一系列核心规则，这些规则是在 [rules 页面](#) 中被标记为  的常见问题。

定义规则 – rule

```
rules: {  
  'no-console': process.env.NODE_ENV === 'production' ? 'warn' : 'off'  
  'no-debugger': process.env.NODE_ENV === 'production' ? 'warn' : 'off'  
}
```

JavaScript

rule 配置规则与跳过规则

跳过规则

```
console.log(1) // eslint-disable-line
```

Bash

插件 – plugins

```
pnpm i @typescript-eslint/parser
```

Bash

```
plugins: ['@typescript-eslint']
```

JSON