Linter是一种工具，主要用于检查CodeBase中潜在的问题。例如语法错误， 配置代码风格（包括缩进，每行代码长度，循环的复杂性等）, 安全检查等。

Linter起源为编译语言C, 后来发展到解释型语言例如Python, JavaScript当中。Linter已经发展为可以为其配置，指定项目自身的风格，并且融入到IDE。

JavaScript中主要主流的Linter有JSLint, JSHint, 和ESLint。

**JSLint**

[JSLint](https://en.wikipedia.org/wiki/JSLint)是由[Douglas Crockford](https://en.wikipedia.org/wiki/Douglas_Crockford)开发的，可能是最早的JavaScript Lint工具了吧，它的名字来源于著名的C语言工具Lint。

**JSHint**

[JSHint](https://en.wikipedia.org/wiki/JSHint)是由[Anton Kovalyov](http://anton.kovalyov.net/)基于JSLint的代码实现的开源项目，由于JSLint时期大多数人都在受JSLint压迫，JSHint相比较之下，更友好，也更容易配置，所以很快就发展了起来，也得到了众多IDE和编辑器的[支持](http://jshint.com/install/)。但是，由于它是基于JSLint开发的，自然原有的一些问题它也继承下来了，比如不易扩展，不容易直接根据报错定位到具体的规则配置等，虽然之前好像是有过相关的讨论，但是现在仍然没有什么好的解决办法。好在它发展的不错，很多时候遇到的问题都可以在网上找到相关的解决方案，而且文档也是非常不错的。

JSHint的配置文件为jshintrc.通常包含在项目当中。JSHint不原生支持JSX语言的检查。

**ESLint**

[ESLint](http://eslint.org/)是由[Nicholas C. Zakas](https://www.nczonline.net/)在2013年开始开发的，它的初衷就是为了能让开发者能自定义自己的linting rules，而且它提供了一套相当完善的插件机制，可以自由的扩展，动态加载配置规则，同时可以方便的根据报错定位到具体的规则配置。而且我比较喜欢它的一点是[文档](http://eslint.org/docs/user-guide/configuring)非常详细，可能这也是灵活所必须的吧。在这里还要提一点，ESLint最初并不是为了造一个重复的轮子，而是作者在实际使用中的需求[没有能得到JSHint团队的回应](https://github.com/eslint/eslint#why-dont-you-like-jshint)，所以他就结合当时的JSHint和另一个代码风格的检查工具JSCS写出来了现在具备代码风格检查，自定义插件扩展功能的ESLint了。

ESLint原生支持JSX语言的检查。配置文件为. eslint。