

Version

v8.23.0

Search

USER GUIDE

Getting Started

- Configuring
- Configuration Files (New)
- Configuration Files
- Configuring Language Options
- Configuring Rules
- Configuring Plugins
- Ignoring Code
- Command Line Interface
- Rules
- Formatters
- Integrations
- Migrating to v8.x

DEVELOPER GUIDE

- Architecture
- Getting the Source Code
- Set Up a Development Environment
- Run the Tests
- Working with Rules
- Working with Plugins
- Working with Custom Formatters
- Working with Custom Parsers
- Shareable Configs
- Node.js API
- Contributing

MAINTAINER GUIDE

- Managing Issues
- Reviewing Pull Requests
- Managing Releases
- Governance

Getting Started with ESLint

ESLint is a tool for identifying and reporting on patterns found in ECMAScript/JavaScript code, with the goal of making code more consistent and avoiding bugs. In many ways, it is similar to JSLint and JSHint with a few exceptions:

- ESLint uses [Espree](#) for JavaScript parsing.
- ESLint uses an AST to evaluate patterns in code.
- ESLint is completely pluggable, every single rule is a plugin and you can add more at runtime.

Installation and Usage

Prerequisites: [Node.js](#) ([^12.22.0](#), [^14.17.0](#), or [>=16.0.0](#)) built with SSL support. (If you are using an official Node.js distribution, SSL is always built in.)

You can install and configure ESLint using this command:

```
1 npm init @eslint/config
```

Note: `npm init @eslint/config` assumes you have a `package.json` file already. If you don't, make sure to run `npm init` or `yarn init` beforehand.

After that, you can run ESLint on any file or directory like this:

```
1 npx eslint yourfile.js
2
3 # or
4
5 yarn run eslint yourfile.js
```

It is also possible to install ESLint globally rather than locally (using `npm install eslint --global`). However, this is not recommended, and any plugins or shareable configs that you use must be installed locally in either case.

Configuration

Note: If you are coming from a version before 1.0.0 please see the [migration guide](#).

After running `npm init @eslint/config`, you'll have a `.eslintrc.{js,yml,json}` file in your directory. In it, you'll see some rules configured like this:

```
1 {
2   "rules": {
3     "semi": ["error", "always"],
4     "quotes": ["error", "double"]
5   }
6 }
```

The names `"semi"` and `"quotes"` are the names of [rules](#) in ESLint. The first value is the error level of the rule and can be one of these values:

- `"off"` or `0` - turn the rule off
- `"warn"` or `1` - turn the rule on as a warning (doesn't affect exit code)
- `"error"` or `2` - turn the rule on as an error (exit code will be 1)

The three error levels allow you fine-grained control over how ESLint applies rules (for more configuration options and details, see the [configuration docs](#)).

Your `.eslintrc.{js,yml,json}` configuration file will also include the line:

```
1 {
2   "extends": "eslint:recommended"
3 }
```

Because of this line, all of the rules marked "(recommended)" on the [rules page](#) will be turned on. Alternatively, you can use configurations that others have created by searching for "eslint-config" on [npmjs.com](#). ESLint will not lint your code unless you extend from a shared configuration or explicitly turn rules on in your configuration.

Next Steps

- Learn about [advanced configuration](#) of ESLint.
- Get familiar with the [command line options](#).
- Explore [ESLint integrations](#) into other tools like editors, build systems, and more.
- Can't find just the right rule? Make your own [custom rule](#).
- Make ESLint even better by [contributing](#).

Edit this page

Table of Contents

Installation and Usage

Configuration

Next Steps