ESLint JavaScript linter

START HERE

npm init @eslint/config@latest
Create a config

npx eslint .
Lint the project

npx eslint --fix .
Auto-fix problems

npx eslint *<file>*
Lint one file

npx eslint --quiet .
Errors only

npx eslint --max-warnings 0 .
Fail on warnings

npx eslint --cache .
Skip unchanged files

npx eslint --print-config *<file>*
Show effective config

Files globs from CLI → **Flat config** eslint.config.js → **Problems** error / warn → **Report** -f, exit code

CONCEPTS

ESLint 9+ reads only the flat config eslint.config.js (or .mjs/.cjs/.ts). Old .eslintrc files are ignored.

Rules are off, warn or error. Only errors (and warnings over --max-warnings) give exit code 1.

RUNNING

npx eslint .
Lint the current directory

npx eslint *<file> <dir>*
Lint specific files or folders

npx eslint "src//*.{js,ts}"**
Lint a quoted glob
Quote globs so ESLint expands them, not the shell.

npx eslint --stdin --stdin-filename *<path>*
Lint code from stdin

npx eslint --ignore-pattern *<glob>* **.**
Skip files for this run

npx eslint --no-warn-ignored *<file>*
Silence the warning for ignored files
Useful when a pre-commit hook passes ignored files.

npx eslint --no-error-on-unmatched-pattern *<glob>*
Do not fail when a glob matches nothing

npx eslint --concurrency auto .
Lint in parallel worker threads

FIXING

npx eslint --fix .
Apply automatic fixes

FIXING (cont.)

npx eslint --fix --fix-type suggestion,layout .
Only apply some fix kinds
Types: problem, suggestion, layout, directive.

npx eslint --cache --fix .
Fix and remember what is clean

WARNINGS AND EXIT CODES

npx eslint --quiet .
Hide warnings, show errors

npx eslint --max-warnings 0 .
Exit 1 if any warning appears

echo $?
0 clean, 1 lint failure, 2 config or crash

npx eslint --rule '*rule***: error' .**
Set a rule for this run only
Quote the value; the syntax is rule-name: severity or JSON.

npx eslint --report-unused-disable-directives .
Flag useless disable comments

OUTPUT

npx eslint -f json .
Machine-readable JSON

npx eslint -f *<name>* **.**
Pick a formatter
Built in: stylish (default), json, json-with-metadata, html.

npx eslint -f json -o *<file>* **.**
Write the report to a file

CACHE AND SPEED

npx eslint --cache .
Only lint changed files

CACHE AND SPEED (cont.)

npx eslint --cache --cache-strategy content .
Detect changes by content, not mtime
Handy in CI where checkouts reset modification times.

Add .eslintcache to .gitignore. Delete it if results look stale after a plugin or config change.

CONFIG

npm init @eslint/config@latest
Create eslint.config.js interactively

npx eslint -c *<file>* **.**
Use a specific config file

npx eslint --print-config *<file>*
Show the config a file gets

npx eslint --debug *<file>*
Print debug output

eslint.config.js

```
import js from "@eslint/js";
import globals from "globals";

export default [
  { ignores: ["dist/", "coverage/"] },
  js.configs.recommended,
  {
    files: ["**/*.js"],
    languageOptions: { globals: globals.node },
    rules: { "no-unused-vars": "warn", eqeqeq: "error" },
  },
];
```

defineConfig and extends

```
import { defineConfig } from "eslint/config";
import js from "@eslint/js";

export default defineConfig([
  { files: ["src/**/*.js"], plugins: { js }, extends: ["js/recommended"] },
  { files: ["test/**"], rules: { "no-console": "off" } },
]);
```

CONFIG (cont.)

{ ignores: ["dist/"] }
Global ignore (only key in the object)
With other keys it only ignores for that one config object.

{ files: ["/*.ts"], ... }**
Limit a config object to matching files

INLINE COMMENTS

// eslint-disable-next-line *<rule>*
Silence a rule on the next line

// eslint-disable-line *<rule>*
Silence a rule on this line

/* eslint-disable *<rule>* ***/**
Disable from here to the end of file

// eslint-disable-next-line *<rule>* **--** *<reason>*
Document why after --

RECIPES

CI lint step

```
npx eslint --max-warnings 0 --cache --cache-strategy content .
```

Staged files only (pre-commit)

```
git diff --name-only --cached --diff-filter=d -- '*.js' \
  | xargs npx eslint --max-warnings 0
```

Adopt a rule in a legacy repo

```
npx eslint --suppress-all .
# fix gradually, then drop stale entries
npx eslint --prune-suppressions
```