Engineer

ESLint

JavaScriptを規則に照らして調べるリンター

どのようなものか

ESLintは、JavaScriptのソースコードを読み、設定した規則に照らし、違反する場所を一つずつ報告する、オープンソースの静的解析ツールです。求められれば、簡単なものは自動で修正します。コマンドラインでは `eslint` というコマンドで、検査するプロジェクトの開発用依存パッケージとして、npmからインストールします。Nicholas C. Zakasが2013年6月に始めたプロジェクトで、現在も、自らを設定可能なJavaScriptリンターと紹介しています。

対象のプログラムは、一度も実行しません。パーサーが各ファイルをESTree形式の抽象構文木に変えます。各規則は、小さいモジュールです。関数、比較、`await` など、特定の種類のノードごとに呼ばれるよう要求します。渡されたものを調べ、問題があれば、メッセージと位置を報告します。自動修正を持つ規則もあります。プロジェクトが修正として認めるのは、プログラムの動作を変えずに問題を直す変更だけで、`eslint --fix` がファイルへ適用します。代わりに提案を出す規則もあります。動作を変える場合があるため、人がエディターで一つずつ適用します。

この中心部分の周りは、ほぼすべて交換できます。プラグインは、規則、共有できる設定、プロセッサー、ほかの言語への対応を追加するnpmパッケージです。別のパーサーを使えば、TypeScriptなど、既定のパーサーが理解しない構文も読めます。プロセッサーは、Markdown文書のコードブロックなど、別のファイルからJavaScriptを取り出します。フォーマッターは、報告の表示方法を変えます。プロジェクト自身が述べる考え方は、各規則が独立し、無効にでき、それぞれの基準で警告や失敗を設定できる、というものです。

すべてを設定するのは、プロジェクトのルートに置く、一つのJavaScriptファイルです。通常は `eslint.config.js` か、 `.mjs` 、 `.cjs` の形式で、オブジェクトの配列をエクスポートします。各オブジェクトが、対象ファイルと、適用する規則、グローバル変数、パーサーのオプションを示し、後のオブジェクトが前を上

書きします。この構成をフラット設定と呼び、2024年4月5日のESLint v9.0.0で既定になりました。2026年2月6日公開のESLint v10.0.0は、古い `.eslintrc` 形式と `.eslintignore` ファイルを、完全に廃止しました。

概念を判断に生かす

「ESLintとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 3社の間で受け渡す代わりに、1つのチームがアイデアのスケッチから、お客様がログインして使える製品まで担当します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 GIS地図アプリの開発を主導し売上を10倍に — 10x 地図製品の開発を主導して実現した売上成長

GIS地図アプリの開発を主導し売上を10倍に お問い合わせ →

なぜ重要なのか

リンターが役立つ決定的な場面は、変更のマージを止めるときです。ESLintは、終了コードで判断を伝え、その値は重大度に従います。各規則は、`"off"` 、 `"warn"` 、 `"error"` のいずれかです。エラーがあれば、コマンドは1で終了します。警告は表示しても、 `--max-warnings` で上限を設けなければ、終了コードは0のままです。この上限は、既定では無効です。終了コード2は、設定の不備か内部の異常終了に使います。

つまり、ESLintを使うチームが本当に決めるのは、警告の意味です。ビルドを失敗させない警告は、助言です。端末に四十行も並べば、読み飛ばされます。リンターに基準を守らせたいチームは、必要な規則をすべて `"error"` にするか、品質ゲートで `--max-warnings 0` を使います。どちらでも、健全なブランチは、何の指摘もなく検査を通ります。新しい一行が出れば、気付いて読まれる知らせになります。

効果が分かるのは、JavaScriptが通してしまうコードです。宣言のない変数を参照する、オブジェクトリテラルで同じキーを二回定義する、 `NaN` を `===` で比較する、 `return` の後に実行不能なコードを残す。言語は、こうしたファイルも読み込みます。問題が読み込みを通過して気付かれないことも、実行時に、本番を含めて誤動作を起こすこともあります。ESLintの推奨規則は、四つともレビュー前に検出します。レビューする人は、設計を読むことへ集中できます。

負担もあります。設定には、好みが積み重なります。正しいコードに反応する規則は、出力を読まない習慣を付け、規則がないより悪い結果になります。リンターの価値は、いつ実行するかによっても制限されます。一部のマシンのエディターでしか動かないものは、限られた機会に、一部のコードしか調べません。

実際の活用例

ESLintは、六つの言語へ警告ゼロの基準を設けた仕事で使った、六つのリンターの一つでした。TypeScriptを担当し、Goのgolangci-lint、SQLのSQLFluff、PythonのRuff、シェルスクリプトのShellCheck、Markdownのmarkdownlintと並びました。基準を支えたのは、重大度について、全員に共通する答えを一度に決めたことです。すべての指摘はエラーで、逃げ込める「warn」の段階はなく、コード内の抑制指定は禁止しました。コミット前とプッシュ前のフックが、リンターとテストを実行します。対象となる規則やテストに違反すると、対応するコミットまたはプッシュを止めます。

同じリポジトリには、その後、公開パッケージにはない規則も必要になりました。そのコードベース特有の作り方を守る規則です。独自のESLint規則十五個で、TypeScriptのプロジェクト固有のパターンを守りました。ほかに274個の検査スクリプトがあり、268個をコミットフックへ登録しました。一つのバリデーターは、コード内の抑制指定を、気付かれないまま追加していないかを確かめます。検査機構自体にも、146個のテストがあります。壊れた検査は、すべてを通し、誰も気付かないためです。事例は代償も明記します。コミットに時間がかかります。

このサイトも、より小さい規模で、ビルドスクリプトに同じ基準を適用します。`eslint.config.mjs` は `js.configs.recommended` から始め、測定時に、コードがすでに満たしていた規則だけを加えます。コメントには、2026-10-15に、非推奨に指定されていない199個の規則全体を再測定し、好みではなく正しさを調べる規則を、さらに21個有効にした記録があります。すべての規則は、違反をエラーとします。既定では警告の `reportUnusedDisableDirectives` も、エラーへ上げます。何も抑制しなくなった無効化コメントが、検査を失敗させるためです。スクリプトは、Nodeとブラウザーのグローバル変数を、両方宣言します。 `page.evaluate` へ渡す関数本体は、Nodeで動くコードの数行隣にあっても、ブラウザー内で動くためです。

問題が起きるところ

ESLintの問題の多くは、リンターが、設定どおりに動いていることです。その設定の意味が、チームの理解と一致するとは限りません。

ビルドは緑なのに、端末が警告で埋まります。 警告では終了コードが0のままなので、継続的インテグレーションは通ります。一覧は毎週数行ずつ増え、やがて本物の欠陥が、読まれないまま紛れ込みます。必要な規則を決め、 `"error"` に

し、ほかは無効にします。品質ゲートに `--max-warnings 0` も加え、警告を蓄積できなくします。

無効化コメントだけが、理由を失って残ります。 かつて正当な理由で、その行の指摘を抑えました。下のコードが変わってもコメントは残り、その行の次の問題を隠すか、何も抑えなくなります。 `linterOptions.reportUnusedDisableDirectives` を `"error"` にし、何も抑えない指定が検査を失敗させるようにします。残すものには、 `--` の後に理由を求め、レビューで判断できるようにします。

```
// eslint-disable-next-line no-console -- 出力を表示するのがこのCLIの役割
console.log(report);
```

更新で、検査するファイルが変わります。 v10へ移ると、 `.eslintignore` で除外していたパスが、突然報告されたり、一つのフォルダーが違う規則に従い始めたりします。バージョン10は、 `.eslintrc.*` も `.eslintignore` も使いません。検査するファイルごとに、そのファイルのディレクトリーから `eslint.config.*` を探します。そのため、サブフォルダーに残った予定外の設定が、そのフォルダーに適用されます。除外は、グローバルな `ignores` 項目へ移します。 `npx @eslint/migrate-config .eslintrc.json` で古いファイルを変換し、出力をレビューします。その後、ツリー内のすべての `eslint.config.*` を一覧にし、残すつもりのなかったものを削除します。

ブラウザーのコードが、未定義と報告されます。 ブラウザーへ関数を渡すNodeスクリプトで、 `document` や `window` に `no-undef` エラーが出ます。そこで `no-undef` を無効にすると、本物のタイプミスも隠れます。各ファイルが実際に動くすべての環境を、 `languageOptions.globals` で、そのファイルに対して宣言します。

独自の規則が、すべてを通します。 規則のビジターが、対象のコードに一度も現れない種類のノードを指定したり、条件が逆だったりすると、何も報告しません。沈黙は、規則を守った結果と同じに見えます。各独自規則をESLintの `RuleTester` で試し、通るべきコードと、失敗すべきコードを渡します。報告しなくなった規則が、コードベースを守れなくなる前に、自分のテストで失敗するようにします。

参考資料

- 基本概念 — プロジェクトが定義する、規則、修正、プラグイン、パーサー、プロセッサーです。

- 設定ファイル — フラット設定の配列、ファイルパターン、グローバルな除外、オブジェクトの結合方法です。
- 規則の設定 — 重大度、規則のオプション、特定の行で規則を無効にするコメントです。
- コマンドラインインターフェースのリファレンス — `--max-warnings` を含むすべてのオプションと、各終了コードの意味です。
- ESLint v10.0.0公開 — eslintrcの廃止と、新しい設定の探索方法を説明するリリースノートです。
- 独自規則 — 構文木を調べる規則を書き、`RuleTester` で確かめる方法です。

用語集

実際の制約に合わせて選ぶ

業務の流れ、既存のインターフェース、変更を必要とする利用者をお知らせください。検証可能な納品内容と、お客様のチームが保守を続けられる引き継ぎ内容を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「ESLintとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/eslint/