intershop

ESLint

and the Intershop PWA

Max Kless

January 2022

intershop

Motivation

Static code analysis tool to flag programming and stylistic errors, bugs and suspicious constructs

„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.” – eslint.org

intershop

How does ESLint work? – Parsers and ASTs

Source Code

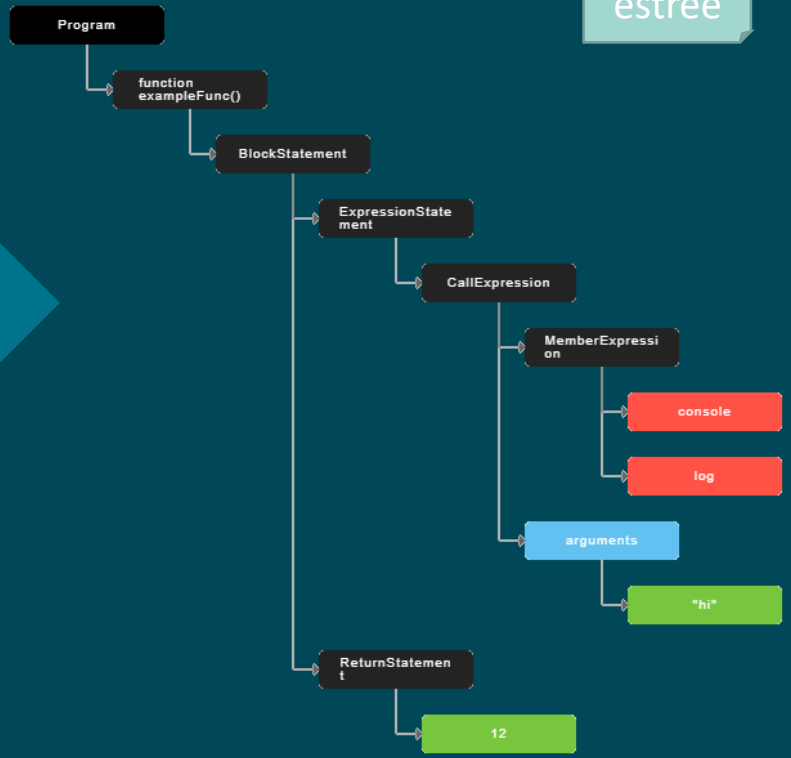
```
function exampleFunc(){
  console.log("hi");
  return 12;
}
```

Parser

espree

Abstract Syntax Tree

estree

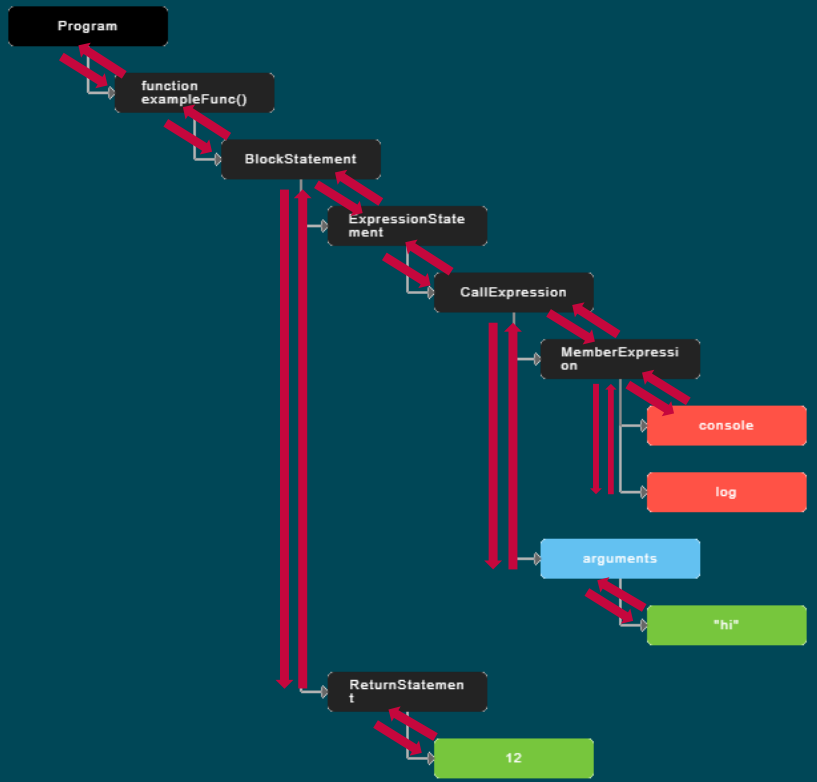


Playground on typescript-eslint.io/play

intershop

How does ESLint work – Traversing the AST

- Traversing with estraverse
- Uses Depth-First Search
- Nodes are selected using esquery, `:enter` and `:leave` operations possible



intershop

ESLint + Typescript

@typescript-eslint/plugin – Core ESLint plugin with rules and configurations

@typescript-eslint/parser – takes configuration and source code and produces estree AST

@typescript-eslint/estree – used by the parser to transform a typescript AST to an estree AST

@typescript-eslint/utils – used for writing rules and plugins in typescript, includes type information

@typescript-eslint/eslint-plugin-tslint – seperate plugin that allows running tslint rules in ESLlint

For backwards compatibility only!

intershop

ESLint + Angular

@angular-eslint/builder – use ESLint in ng lint

@angular-eslint/eslint-plugin – custom rules and configurations specific to Angular

@angular-eslint/template-parser – use ESLint for Angular templates

@angular-eslint/eslint-plugin-template – custom rules specific to Angular templates

@angular-eslint/schematics – utility schematics

intershop

ESLint in the Intershop PWA - `.eslintrc.json`

- Angular + ESLint is a complex use-case ➡ nonstandard configuration
- Different blocks of config for different file types ➡ overrides array
- True JSON, not JavaScript
- Different HTML configs <u>because of prettier</u>

```
{
  "root": true,
  "overrides:" : [
    {
      "files" : ["*.ts"],
      …
    },
    {
      "files" : ["*.html"],
      …
    },
    {
      "files: ["*inline-template-*.component.html"],
       …
    }
  ]
}
```

intershop

ESLint in the Intershop PWA - `.eslintrc.json`

- `extends`: extends by one or more predefined configurations, merges configurations
- `plugins`: gives access to custom rules and more
- `rules`: „off“, „warn“, „error“ settings as well as rule-specific configuration

```
{
  "files" : ["*.ts"],
  "extends" : [
    “example-config”,
    …,
    "plugin:prettier/recommended"
  ],
  "plugins" : [
    “rxjs”,
    …,
    “ish-custom-rules”
],
  "rules" : {
    “eqeqeq”: [“error”, “always”],
    …
 }
},
```

= eslint-config-example-config

= eslint-plugin-rxjs

intershop

ESLint in the Intershop PWA – Custom Rules

- Located in `/eslint-rules`
- List of all rules in index.ts
- npm scripts for building rules and running tests – integrated with `npm install`
- Tests are run by `tests/_execute-tests.ts`
- Built with @typescript-eslint/experimental-utils ➡ could change in the future
- Local import in package.json

intershop

ESLint in the Intershop PWA – Writing a custom rule

DEMO TIME

intershop

Extras

- Well documented Node.js API, refer to `fix-imports.js` for an example
- Performance measuring with `TIMING=1, TIMING=50, TIMING=all`
- Prettier is integrated as a rule via `plugin:prettier`, might be changed with a later release to standalone prettier
- Commit structure designed for easy upgrading, refer to Migration Guide for details