

云水

MyGitee - <https://gitee.com/lsgx/>
MyGithub - <https://github.com/lsgxeva/>
ProxyDownload - <https://d.serctl.com/>
translate - <https://translate.google.com/>
nand2tetris - <https://www.nand2tetris.org/>
fuchsia - <https://cs.opensource.google/fuchsia/fuchsia>
firefox - <https://www.mozilla.org/zh-CN/firefox>

博客园

首页

新随笔

联系

订阅

管理

随笔 - 1568 文章 - 0 评论 - 99 阅读 - 353万

eslint的安装与使用

eslint的安装与使用

什么是 ESLint

ESLint (中文站点) 是一个开源的 JavaScript 代码检查工具, 使用 Node.js 编写, 由 Nicholas C. Zakas 于 2013 年 6 月创建。ESLint 的初衷是为了让程序员可以创建自己的检测规则, 使其可以在编码的过程中发现问题而不是在执行的过程中。ESLint 的所有规则都被设计成可插入的, 为了方便使用, ESLint 内置了一些规则, 在这基础上也可以增加自定义规则。

安装 ESLint 扩展

安装环境

- [VSCode V1.11.1](#)
- Windows 10

安装 ESLint 扩展

首先, 打开 VSCode 扩展面板并搜索 ESLint 扩展, 然后点击安装

安装完毕之后点击 **重新加载** 以激活扩展, 但想要让扩展进行工作, 我们还需要先进行 ESLint 的安装配置。

安装 ESLint

如果你仅仅想让 ESLint 成为你项目构建系统的一部分, 我们可以在项目根目录进行本地安装:

```
$ npm install eslint --save-dev
```

如果想使 ESLint 适用于你所有的项目, 我们建议使用全局安装, 使用全局安装 ESLint 后, 你使用的任何 ESLint 插件或可分享的配置也都必须在全局安装。

这里我们使用全局安装:

```
$ npm install -g eslint
```

安装完毕后, 我们使用 `eslint --init` 命令在用户目录中生成一个配置文件 (也可以在任何你喜欢的位置进行生成)

我们在第一个选项中选择自定义代码风格, 之后根据需要自行选择。

设置完成后我们会得到一份文件名为 `.eslintrc.js` 的配置文件:

公告

昵称: lsgxeva
园龄: 5年11个月
粉丝: 216
关注: 1
[+加关注](#)

2021年9月						
日	一	二	三	四	五	六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

搜索

常用链接

[我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

随笔分类

a10(4)
AI(1)
algorithm(1)
Android(38)
ans(55)
bitcoin(4)
c++(107)
c++11(38)
ccna(36)
cmake(5)
cnns(18)
cocos(1)
CSharp(29)

```
module.exports = {
  "env": {
    "browser": true,
    "commonjs": true,
    "es6": true
  },
  "extends": "eslint:recommended",
  "parserOptions": {
    "sourceType": "module"
  },
  "rules": {
    "indent": [
      "error",
      4
    ],
    "linebreak-style": [
      "error",
      "windows"
    ],
    "quotes": [
      "error",
      "single"
    ],
    "semi": [
      "error",
      "never"
    ]
  }
};
```

配置 ESLint

配置文件生成之后，我们接着可以进行自定义修改，这里我们只粗略讲解常用的配置项，完整的可配置项可访问[官方文档](#)

配置环境

在上文生成的配置文件中可以使用 `env` 属性来指定要启用的环境，将其设置为 `true`，以保证在进行代码检测时不会把这些环境预定义的全局变量识别成未定义的变量而报错：

```
"env": {
  "browser": true,
  "commonjs": true,
  "es6": true,
  "jquery": true
}
```

设置语言选项

默认情况下，ESLint 支持 ECMAScript 5 语法，如果你想启用对 ECMAScript 其它版本和 JSX 等的支持，ESLint 允许你使用 `parserOptions` 属性进行指定想要支持的 JavaScript 语言选项，不过你可能需要自行安装 `eslint-plugin-react` 等插件。

```
"parserOptions": {
  "ecmaVersion": 6,
  "sourceType": "module",
  "ecmaFeatures": {
    "jsx": true
  }
}
```

配置规则

在上文的配置文件中，`"extends": "eslint:recommended"` 选项表示启用推荐规则，在推荐规则的基础上我们还可以根据需要使用 `rules` 新增自定义规则，每个规则的第一个值都是代表该规则检测后显示的错误级别：

- `"off"` 或 `0` - 关闭规则
- `"warn"` 或 `1` - 将规则视为一个警告
- `"error"` 或 `2` - 将规则视为一个错误

DataStructure(6)
DesignPattern(39)
[更多](#)

随笔档案

- 2021年9月(9)
- 2021年8月(8)
- 2021年7月(25)
- 2021年6月(3)
- 2021年5月(11)
- 2021年4月(9)
- 2021年3月(6)
- 2021年2月(4)
- 2021年1月(39)
- 2020年12月(34)
- 2020年11月(48)
- 2020年10月(38)
- 2020年9月(89)
- 2020年8月(20)
- 2020年7月(28)
- [更多](#)

阅读排行榜

1. Docker 创建镜像、修改、上传镜像(189374)
2. Git Submodule使用完整教程(111180)
3. git diff命令详解(57408)
4. C++标准转换运算符reinterpret_cast(56171)
5. Qt基本控件及三大布局(54345)

评论排行榜

1. Git Submodule使用完整教程(9)
2. babel从入门到入门(6)
3. libuv 简单使用(5)
4. MIT Scheme 的基本使用(4)
5. 关于qt中的tr () 函数(4)

推荐排行榜

1. Git Submodule使用完整教程(13)
2. javaee, javaweb和javase的区别以及各自的知识体系(8)
3. Docker 创建镜像、修改、上传镜像(7)
4. c++11 类默认函数的控制: "=default" 和 "=delete"函数(7)
5. c++11 智能指针 unique_ptr、shared_ptr 与weak_ptr(6)

最新评论

1. Re:Xinetd服务的安装与配置图没了
--ageovb
2. Re:Linux高并发内核优化-TougheRadius大神，这玩意如何跟windows AD集成身份验证，windows 客户大多使用mschap
--蓉城浪子
3. Re:Unicode 和 UTF-8 的区别你好，文章开头的：U+10000 ~ U+1FFFF: 1110XXX 10XXXXXX 10XXXXXX 10XXXXXX 应该为：U+10000 ~ U+10FFFF: 1110XXXX ...
--ukeyim
4. Re:英特尔vPro博锐技术激活这个地址是内网地址吧，如果是外网地址，怎么连呢？我在家，故障的电脑在别的有网

```
"rules": {
  "indent": [
    "error",
    4
  ],
  "linebreak-style": [
    "error",
    "windows"
  ],
  "quotes": [
    "error",
    "single"
  ],
  "semi": [
    "error",
    "never"
  ]
}
```

络的地方，那地址要怎么填？ DHCP？ 知道故障电脑的外网IP？

--weatherz

5. Re:C调用C++（C++封装以及C对其调用）

学习了

--itfanr

完整的可配置规则列表可访问：<http://eslint.cn/docs/rules/>

其中带  标记的表示该规则为推荐规则。

设置 ESLint 扩展

安装并配置完成 ESLint 后，我们继续回到 VSCode 进行扩展设置，依次点击 文件 > 首选项 > 设置 打开 VSCode 配置文件

从左侧系统设置中可以看到，ESLint 扩展默认已经启用，我们现在只需在右侧用户设置中添加配置来指定我们创建的 .eslintrc.js 配置文件路径即可启用自定义规则检测，ESLint 会查找并自动读取它们：

```
"eslint.options": {
  "configFile": "E:/git/github/styleguide/eslint/.eslintrc.js"
},
```

至此，我们已经可以使用 ESLint 扩展来检测我们的 js 文件了。

让 ESLint 支持 Vue 单文件组件

由于 ESLint 默认只支持 js 文件的脚本检测，如果我们需要支持类 html 文件（如 vue ）的内联脚本检测，还需要安装 eslint-plugin-html 插件。

因为我们使用全局安装了 ESLint，所以 eslint-plugin-html 插件也必须进行全局安装：

```
$ npm install -g eslint-plugin-html
```

安装完成后，我们再次打开 文件 > 首选项 > 设置 ，在右侧用户设置中修改 ESLint 的相关配置并保存：

```
"eslint.options": {
  "configFile": "E:/git/github/styleguide/eslint/.eslintrc.js",
  "plugins": ["html"]
},
"eslint.validate": [
  "javascript",
  "javascriptreact",
  "html",
  "vue"
]
```

最后，我们打开一个 vue 文件，可以发现 ESLint 扩展已经正常工作了

- "no-alert": 0, //禁止使用alert confirm prompt
- "no-array-constructor": 2, //禁止使用数组构造器
- "no-bitwise": 0, //禁止使用按位运算符
- "no-caller": 1, //禁止使用arguments.caller或arguments.callee
- "no-catch-shadow": 2, //禁止catch子句参数与外部作用域变量同名
- "no-class-assign": 2, //禁止给类赋值
- "no-cond-assign": 2, //禁止在条件表达式中使用赋值语句
- "no-console": 2, //禁止使用console

"no-const-assign": 2, //禁止修改const声明的变量

"no-constant-condition": 2, //禁止在条件中使用常量表达式 if(true) if(1)

"no-continue": 0, //禁止使用continue

"no-control-regex": 2, //禁止在正则表达式中使用控制字符

"no-debugger": 2, //禁止使用debugger

"no-delete-var": 2, //不能对var声明的变量使用delete操作符

"no-div-regex": 1, //不能使用看起来像除法的正则表达式 /=foo/

"no-dupe-keys": 2, //在创建对象字面量时不允许键重复 {a:1,a:1}

"no-dupe-args": 2, //函数参数不能重复

"no-duplicate-case": 2, //switch中的case标签不能重复

"no-else-return": 2, //如果if语句里面有return,后面不能跟else语句

"no-empty": 2, //块语句中的内容不能为空

"no-empty-character-class": 2, //正则表达式中的[]内容不能为空

"no-empty-label": 2, //禁止使用空label

"no-eq-null": 2, //禁止对null使用==或!=运算符

"no-eval": 1, //禁止使用eval

"no-ex-assign": 2, //禁止给catch语句中的异常参数赋值

"no-extend-native": 2, //禁止扩展native对象

"no-extra-bind": 2, //禁止不必要的函数绑定

"no-extra-boolean-cast": 2, //禁止不必要的bool转换

"no-extra-parens": 2, //禁止非必要的括号

"no-extra-semi": 2, //禁止多余的冒号

"no-fallthrough": 1, //禁止switch穿透

"no-floating-decimal": 2, //禁止省略浮点数中的0 .5 3.

"no-func-assign": 2, //禁止重复的函数声明

"no-implicit-coercion": 1, //禁止隐式转换

"no-implied-eval": 2, //禁止使用隐式eval

"no-inline-comments": 0, //禁止行内备注

"no-inner-declarations": [2, "functions"], //禁止在块语句中使用声明（变量或函数）

"no-invalid-regexp": 2, //禁止无效的正则表达式

"no-invalid-this": 2, //禁止无效的this，只能用在构造器，类，对象字面量

"no-irregular-whitespace": 2, //不能有不规则的空格

"no-iterator": 2, //禁止使用__iterator__属性

"no-label-var": 2, //label名不能与var声明的变量名相同

"no-labels": 2, //禁止标签声明

"no-lone-blocks": 2, //禁止不必要的嵌套块

"no-lonely-if": 2, //禁止else语句内只有if语句

"no-loop-func": 1, //禁止在循环中使用函数（如果没有引用外部变量不形成闭包就可以）

"no-mixed-requires": [0, false], //声明时不能混用声明类型

"no-mixed-spaces-and-tabs": [2, false], //禁止混用tab和空格

"linebreak-style": [0, "windows"], //换行风格

"no-multi-spaces": 1, //不能用多余的空格

"no-multi-str": 2, //字符串不能用换行

"no-multiple-empty-lines": [1, {"max": 2}], //空行最多不能超过2行

"no-native-reassign": 2, //不能重写native对象

"no-negated-in-lhs": 2, //in 操作符的左边不能有!

"no-nested-ternary": 0, //禁止使用嵌套的三目运算

"no-new": 1, //禁止在使用new构造一个实例后不赋值

"no-new-func": 1, //禁止使用new Function

"no-new-object": 2, //禁止使用new Object()

"no-new-require": 2, //禁止使用new require

"no-new-wrappers": 2, //禁止使用new创建包装实例， new String new Boolean new Number

"no-obj-calls": 2, //不能调用内置的全局对象，比如Math() JSON()

"no-octal": 2, //禁止使用八进制数字

"no-octal-escape": 2, //禁止使用八进制转义序列

"no-param-reassign": 2, //禁止给参数重新赋值

"no-path-concat": 0, //node中不能使用__dirname或__filename做路径拼接

"no-plusplus": 0, //禁止使用++，--

"no-process-env": 0, //禁止使用process.env

"no-process-exit": 0, //禁止使用process.exit()

"no-proto": 2, //禁止使用__proto__属性

"no-redeclare": 2, //禁止重复声明变量

"no-regex-spaces": 2, //禁止在正则表达式字面量中使用多个空格 /foo bar/

"no-restricted-modules": 0, //如果禁用了指定模块，使用就会报错

"no-return-assign": 1, //return 语句中不能有赋值表达式
"no-script-url": 0, //禁止使用javascript:void(0)
"no-self-compare": 2, //不能比较自身
"no-sequences": 0, //禁止使用逗号运算符
"no-shadow": 2, //外部作用域中的变量不能与它所包含的作用域中的变量或参数同名
"no-shadow-restricted-names": 2, //严格模式中规定的限制标识符不能作为声明时的变量名使用
"no-spaced-func": 2, //函数调用时 函数名与()之间不能有空格
"no-sparse-arrays": 2, //禁止稀疏数组, [1,,2]
"no-sync": 0, //nodejs 禁止同步方法
"no-ternary": 0, //禁止使用三目运算符
"no-trailing-spaces": 1, //一行结束后面不要有空格
"no-this-before-super": 0, //在调用super()之前不能使用this或super
"no-throw-literal": 2, //禁止抛出字面量错误 throw "error";
"no-undef": 1, //不能有未定义的变量
"no-undef-init": 2, //变量初始化时不能直接给它赋值为undefined
"no-undefined": 2, //不能使用undefined
"no-unexpected-multiline": 2, //避免多行表达式
"no-underscore-dangle": 1, //标识符不能以_开头或结尾
"no-unneeded-ternary": 2, //禁止不必要的嵌套 var isYes = answer === 1 ? true : false;
"no-unreachable": 2, //不能有无法执行的代码
"no-unused-expressions": 2, //禁止无用的表达式
"no-unused-vars": 2, { "vars": "all", "args": "after-used" }, //不能有声明后未被使用的变量或参数
"no-use-before-define": 2, //未定义前不能使用
"no-useless-call": 2, //禁止不必要的call和apply
"no-void": 2, //禁用void操作符
"no-var": 0, //禁用var, 用let和const代替
"no-warning-comments": [1, { "terms": ["todo", "fixme", "xxx"], "location": "start" }], //不能有警告备注
"no-with": 2, //禁用with

"array-bracket-spacing": [2, "never"], //是否允许非空数组里面有多余的空格
"arrow-parens": 0, //箭头函数用小括号括起来
"arrow-spacing": 0, //>的前/后括号
"accessor-pairs": 0, //在对象中使用getter/setter
"block-scoped-var": 0, //块语句中使用var
"brace-style": [1, "1tbs"], //大括号风格
"callback-return": 1, //避免多次调用回调什么的
"camelcase": 2, //强制驼峰法命名
"comma-dangle": [2, "never"], //对象字面量项尾不能有逗号
"comma-spacing": 0, //逗号前后的空格
"comma-style": [2, "last"], //逗号风格, 换行时在行首还是行尾
"complexity": [0, 11], //循环复杂度
"computed-property-spacing": [0, "never"], //是否允许计算后的键名什么的
"consistent-return": 0, //return 后面是否允许省略
"consistent-this": [2, "that"], //this别名
"constructor-super": 0, //非派生类不能调用super, 派生类必须调用super
"curly": [2, "all"], //必须使用 if(){} 中的{}
"default-case": 2, //switch语句最后必须有default
"dot-location": 0, //对象访问符的位置, 换行的时候在行首还是行尾
"dot-notation": [0, { "allowKeywords": true }], //避免不必要的方括号
"eol-last": 0, //文件以单一的换行符结束
"eqeqeq": 2, //必须使用全等
"func-names": 0, //函数表达式必须有名字
"func-style": [0, "declaration"], //函数风格, 规定只能使用函数声明/函数表达式
"generator-star-spacing": 0, //生成器函数*的前后空格
"guard-for-in": 0, //for in循环要用if语句过滤
"handle-callback-err": 0, //nodejs 处理错误
"id-length": 0, //变量名长度
"indent": [2, 4], //缩进风格
"init-declarations": 0, //声明时必须赋初值
"key-spacing": [0, { "beforeColon": false, "afterColon": true }], //对象字面量中冒号的前后空格
"lines-around-comment": 0, //行前/行后备注
"max-depth": [0, 4], //嵌套块深度
"max-len": [0, 80, 4], //字符串最大长度

```

"max-nested-callbacks": [0, 2],//回调嵌套深度
"max-params": [0, 3],//函数最多只能有3个参数
"max-statements": [0, 10],//函数内最多有几个声明
"new-cap": 2,//函数名首行大写必须使用new方式调用，首行小写必须用不带new方式调用
"new-parens": 2,//new时必须加小括号
"newline-after-var": 2,//变量声明后是否需要空一行
"object-curly-spacing": [0, "never"],//大括号内是否允许不必要的空格
"object-shorthand": 0,//强制对象字面量缩写语法
"one-var": 1,//连续声明
"operator-assignment": [0, "always"],//赋值运算符 += -= 什么的
"operator-linebreak": [2, "after"],//换行时运算符在行尾还是行首
"padded-blocks": 0,//块语句内行首行尾是否要空行
"prefer-const": 0,//首选const
"prefer-spread": 0,//首选展开运算
"prefer-reflect": 0,//首选Reflect的方法
"quotes": [1, "single"],//引号类型 ` "" "
"quote-props": [2, "always"],//对象字面量中的属性名是否强制双引号
"radix": 2,//parseInt必须指定第二个参数
"id-match": 0,//命名检测
"require-yield": 0,//生成器函数必须有yield
"semi": [2, "always"],//语句强制分号结尾
"semi-spacing": [0, {"before": false, "after": true}],//分号前后空格
"sort-vars": 0,//变量声明时排序
"space-after-keywords": [0, "always"],//关键字后面是否要空一格
"space-before-blocks": [0, "always"],//不以新行开始的块{前面要不要有空格
"space-before-function-paren": [0, "always"],//函数定义时括号前面要不要有空格
"space-in-parens": [0, "never"],//小括号里面要不要有空格
"space-infix-ops": 0,//中缀操作符周围要不要有空格
"space-return-throw-case": 2,//return throw case后面要不要加空格
"space-unary-ops": [0, {"words": true, "nonwords": false}],//一元运算符的前/后要不要加空格
"spaced-comment": 0,//注释风格要不要有空格什么的
"strict": 2,//使用严格模式
"use-isnan": 2,//禁止比较时使用NaN，只能用isNaN()
"valid-jsdoc": 0,//jsdoc规则
"valid-typeof": 2,//必须使用合法的typeof的值
"vars-on-top": 2,//var必须放在作用域顶部
"wrap-iife": [2, "inside"],//立即执行函数表达式的小括号风格
"wrap-regex": 0,//正则表达式字面量用小括号包起来
"yoda": [2, "never"]//禁止尤达条件

```

支持react，要加上

```

// "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 is 1 when triggered)
module.exports = {
  "env": {
    "browser": true,
    "commonjs": true,
    "es6": true,
    "node": true,
    "jest": true,
    "jquery": true
  },
  "parser": "Espree",
  "globals": {
    "__dirname": "__dirname"
  },
  "extends": "eslint:recommended",
  "parserOptions": {
    "ecmaVersion": 6,
    "ecmaFeatures": {
      "experimentalObjectRestSpread": true,

```

```
"jsx": true,
"globalReturn": true //allow return statements in the global scope
},
"sourceType": "module"
},
"plugins": [
"react"
],
"rules": {
"react/jsx-uses-react": "error",
"react/jsx-uses-vars": ["error"],
"indent": [
2, 4
],
"linebreak-style": [
"error",
"windows"
],
"quotes": [
"error",
"single"
],
"semi": "error",
"eqeqeq": "error", // require the use of === and !==
"no-alert": "error", //require the use of === and !==
"no-multi-spaces": "error", //disallow multiple spaces
"no-redeclare": "error", //disallow variable redeclaration

}
};
```

扩展：1：

如果有个全局变量，eslint报错可以用`/* global var1, var2 */`来消除
或者在eslint中定义global

```
"globals":{
  "__dirname": "__dirname"
},
或者
/* eslint-disable */

let a = 12;


/* eslint-enable */
```

分类: nodejs

好文要顶

关注我

收藏该文

 **lsgxeva**

关注 - 1

粉丝 - 216

+加关注

« 上一篇: [NodeJS优缺点及适用场景讨论](#)
» 下一篇: [npm+webpack+babel+react安装](#)

posted @ 2017-12-06 19:46 lsgxeva 阅读(23847) 评论(0) 编辑 收藏 举报

20

刷新评论 刷新页面 返回顶部

登录后才能查看或发表评论，立即 [登录](#) 或者 [逛逛](#) 博客园首页

【推荐】百度智能云超值优惠：新用户首购云服务器1核1G低至69元/年

【推荐】跨平台组态工控仿真\CAD 50万行C++源码全开放免费下载！

【推荐】阿里云云大使特惠：新用户购ECS服务器1核2G最低价87元/年

【推荐】和开发者在一起：华为开发者社区，入驻博客园科技品牌专区
【推广】园子与爱卡汽车爱宝险合作，随手就可以买一份的百万医疗保险



编辑推荐：

- 记一次 .NET 某上市工业智造 CPU+内存+挂死 三高分析
- 深入 xLua 实现原理之 C# 如何调用 Lua
- 记一次 k8s pod 频繁重启的优化之旅
- .Net Core with 微服务：分布式事务 - 可靠消息最终一致性
- 通过 Wireshark 抓包分析谈谈 DNS 域名解析的那些事儿

最新资讯：

- 明星代言，钱不好赚了 (2021-09-27 13:40)
 - 凭空制造淀粉：什么人，竟然破了十几亿年光合“铁律” (2021-09-27 13:25)
 - 正失去年轻人的老字号，想要翻红有多拼？ (2021-09-27 13:13)
 - 美团战火，烧向拼多多 (2021-09-27 13:00)
 - 红杉沈南鹏：北交所将成为中国软件企业规模化上市的一大重要承接平台 (2021-09-27 12:45)
- » 更多新闻...