

Lint Yourself

Дмитрий Кунин

FC

2018

**Frontend
Conf**

Профессиональная
конференция
фронтенд-разработчиков



Кто я?

- Дмитрий Кунин
- avito, tech lead, trust & safety
- @dkunin

В конце

- ссылки на материалы
- примеры кода
- ссылка на презентацию

Небольшой опрос

?

Небольшой опрос

- Человеческая ошибка

Небольшой опрос

- Человеческая ошибка
- Невнимательность

Небольшой опрос

- Человеческая ошибка
- Невнимательность
- Гремлины



Что пакостят гремлины

- `()`, `{`,
- `foo` is undefined
- `var foo = 1;var foo = 3;`
- `==` / `===`
- ...



План

- Чем помогает линтер
- Как он это делает
- С чем его приготовить
- Куда его положить
- Правила для esLint
- Кастомный линтер

План



Чем помогает линтер

- Как он это делает
- С чем его приготовить
- Куда его положить
- Правила для esLint
- Кастомный линтер

Чем поможет линтер

- ~~Головная боль~~

Чем поможет линтер

- ~~Головная боль~~
- Меньше ошибок

Чем поможет линтер

- ~~Головная боль~~
- Меньше ошибок
- Единый контракт

Чем поможет линтер

- ~~Головная боль~~
- Меньше ошибок
- Единый контракт
- Кастомизация

Популярные линтеры

- JSLint

▸ Commits on Nov 13, 2010

Add README



douglascrockford committed on 13 Nov 2010



7800939



first commit



douglascrockford committed on 13 Nov 2010



ca120a7



Популярные линтеры

- JSLint

▸ Commits on Nov 13, 2010

Add README



douglascrockford committed on 13 Nov 2010



7800939



first commit



douglascrockford committed on 13 Nov 2010



ca120a7









- JSHint

Популярные линтеры

- JSLint

Commits on Nov 13, 2010

Add README  douglascrockford committed on 13 Nov 2010	 7800939 
first commit  douglascrockford committed on 13 Nov 2010	 ca120a7 







- JSHint

- ESLint

Популярные линтеры

- JSLint

▸ Commits on Nov 13, 2010

Add README  douglascrockford committed on 13 Nov 2010	 7800939	
first commit  douglascrockford committed on 13 Nov 2010	 ca120a7	

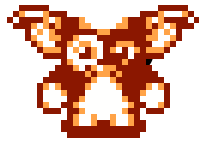
- JSHint
- ESLint
- TsLint
- styleLint
- textLint

Околоинтерье

- Prettier
- ShellCheck
- WebHint
- SonarTS

План

- Чем помогает линтер



Как он это делает

- С чем его приготовить
- Куда его положить
- Правила для esLint
- Кастомный линтер

- Исходный код

- Исходный код
- Парсер AST

- Исходный код
- Парсер AST
- Анализатор (+ плагины)

- Исходный код
- Парсер AST
- Анализатор (+ плагины)
- Список ошибок

- Исходный код
- Парсер AST
- Анализатор (+ плагины)
- Список ошибок
- (Опционально) Fix

```
var esprima = require('esprima');
```

```
var program = 'const answer = 42';
```

```
esprima.tokenize(program);
```

```
[ { type: 'Keyword', value: 'const' },  
  { type: 'Identifier', value: 'answer' },  
  { type: 'Punctuator', value: '=' },  
  { type: 'Numeric', value: '42' } ]
```

```
esprima.parseScript(program);
```

```
{ type: 'Program',  
  body:  
    [ { type: 'VariableDeclaration',
```

```
var esprima = require('esprima');  
var program = 'const answer = 42';
```

```
esprima.tokenize(program);  
[ { type: 'Keyword', value: 'const' },  
  { type: 'Identifier', value: 'answer' },  
  { type: 'Punctuator', value: '=' },  
  { type: 'Numeric', value: '42' } ]
```

```
esprima.parseScript(program);  
{ type: 'Program',  
  body:  
    [ { type: 'VariableDeclaration',
```

```
var esprima = require('esprima');  
var program = 'const answer = 42';
```

```
esprima.tokenize(program);
```

```
[ { type: 'Keyword', value: 'const' },  
  { type: 'Identifier', value: 'answer' },  
  { type: 'Punctuator', value: '=' },  
  { type: 'Numeric', value: '42' } ]
```

```
esprima.parseScript(program);
```

```
{ type: 'Program',  
  body:  
    [ { type: 'VariableDeclaration',
```

```
var esprima = require('esprima');  
var program = 'const answer = 42';
```

```
esprima.tokenize(program);  
[ { type: 'Keyword', value: 'const' },  
  { type: 'Identifier', value: 'answer' },  
  { type: 'Punctuator', value: '=' },  
  { type: 'Numeric', value: '42' } ]
```

```
esprima.parseScript(program);
```

```
{ type: 'Program',  
  body:  
    [ { type: 'VariableDeclaration',
```

Secure | https://astexplorer.net

AST Explorer

Snippet

JavaScript

acorn-to-esprima

Transform

Parser: [acorn-to-esprima-1.0.7](#)

```
1 const someNode = document.querySelector('some-string');
2 const someOtherNode = document.querySelector('js-correct-string');
```

TreeJSON343ms

☒ Autofocus☒ Hide methods☐ Hide empty keys☐ Hide location data☐ Hide type keys

```
- Program {
  type: "Program"
  start: 0
  end: 122
+ loc: {start, end}
  sourceType: "module"
- body: [
  - VariableDeclaration = $node {
    type: "VariableDeclaration"
    start: 0
    end: 55
+ loc: {start, end}
  - declarations: [
    + VariableDeclarator {type, start, end, loc, id, ... +2}
  ]
    kind: "const"
+ range: [2 elements]
  }
+ VariableDeclaration {type, start, end, loc, declarations, ... +2}
]
+ tokens: [21 elements]
  comments: [ ]
+ range: [2 elements]
}
```

FC

2018

Frontend Conf

План

- Чем помогает линтер
- Как он это делает



С чем его приготовить

- Куда его положить
- Правила для esLint
- Кастомный линтер

Настройки линтеров

- .eslintrc
- .stylelintrc
- tslint.json / tslint.yaml
- .textlintrc
- package.json (!)

eslint-plugin-compat

package.json:

```
{  
  // ...  
  "browserslist": ["last 1 versions", "not ie  
}
```

Тип запуска CLI

```
eslint index.js
```

```
🐛 optimizilla-cli (master) eslint index.js

/Users/dikunin/Projects/optimizilla-cli/index.js
  86:22  error    '?' should be placed at the end of the line  operator-linebreak
  87:22  error    ':' should be placed at the end of the line  operator-linebreak
 177:9   warning  Unexpected console statement                no-console

✖ 3 problems (2 errors, 1 warning)
  2 errors, 0 warnings potentially fixable with the `--fix` option.
```

Тип запуска Editor Plugin

```
4
• 5 RemarkJSSlider.grammar = ohm.grammar(`
6   • RemarkJSSlider {
7     • • Rules
8     • • • = ListOf<Rule, "\\n">
9     • Rule
```

eslint: no-undef - 'ohm' is not defined.

```
const CLIEngine = require('eslint').CLIEngine
const cli = new CLIEngine({
  parserOptions: {
    ecmaVersion: 6,
  },
  rules: {
    'no-unused-vars': 'off',
  }
});

const report = cli.executeOnText("let foo = '

if (report.errorCount) {
```

```
const CLIEngine = require('eslint').CLIEngine
const cli = new CLIEngine({
  parserOptions: {
    ecmaVersion: 6,
  },
  rules: {
    'no-unused-vars': 'off',
  }
});

const report = cli.executeOnText('let foo = '

if (report.errorCount) {
```

```
const CLIEngine = require('eslint').CLIEngine
const cli = new CLIEngine({
  parserOptions: {
    ecmaVersion: 6,
  },
  rules: {
    'no-unused-vars': 'off',
  }
});
```

```
const report = cli.executeOnText("let foo = "
```

```
if (report.errorCount) {
```



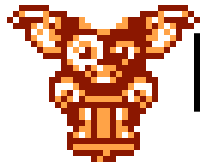
```
const CLIEngine = require('eslint').CLIEngine
const cli = new CLIEngine({
  parserOptions: {
    ecmaVersion: 6,
  },
  rules: {
    'no-unused-vars': 'off',
  }
});

const report = cli.executeOnText('let foo = '

if (report.errorCount) {
```

План

- Чем помогает линтер
- Как он это делает
- С чем его приготовить



Куда его положить

- Правила для esLint
- Кастомный линтер

Как встроить его во флоу

- при сохранении
- `git precommit/prepush/prereceive`
- CI-step

План

- Чем помогает линтер
- Как он это делает
- С чем его приготовить
- Куда его положить



Правила для esLint

- Кастомный линтер

Ингредиенты для кастомного плагина eslint

- Конфиг
- Функция
- Фикс
- Пре/постпроцессинг
- Тесты

Какую проблему решаем

// 👎

```
document.querySelector( '.items' );
```

// 👍

```
document.querySelector( '.js-items' );
```

```
{  
  "plugins": [  
    "@avito/custom-rules"  
  ],  
  "rules": {  
    "eqeqeq": "off",  
    "curly": "error",  
    "quotes": ["error", "double"],  
    "@avito/custom-rules/prefer-js-prefix"  
    ...  
  }  
}
```

```
{  
  "plugins": [  
    "@avito/custom-rules"  
  ],  
  "rules": {  
    "eqeqeq": "off",  
    "curly": "error",  
    "quotes": ["error", "double"],  
    "@avito/custom-rules/prefer-js-prefi  
    ...  
  }  
}
```



```
module.exports = {  
  create: function(context) {  
    // context => parserOptions/ruleId/option  
    return {  
      CallExpression: function(node) {  
        if (  
          getNestedProperty(node, 'callee.p  
            'querySelector' &&  
          !getNestedProperty(node, 'argumen  
            '.js-'  
        )  
      ) {  
        context.report(  

```

```
module.exports = {  
  create: function(context) {  
    // context => parserOptions/ruleId/option  
    return {  
      CallExpression: function(node) {  
        if (  
          getNestedProperty(node, 'callee.p  
            'querySelector' &&  
          !getNestedProperty(node, 'argumen  
            '.js-'  
        )  
      ) {  
        context.report(  

```

```
module.exports = {  
  create: function(context) {  
    // context => parserOptions/ruleId/option  
    return {  
      CallExpression: function(node) {  
        if (  
          getNestedProperty(node, 'callee.p  
            'querySelector' &&  
          !getNestedProperty(node, 'argumen  
            '.js-'  
        )  
      ) {  
        context.report(  

```

```
module.exports = {  
  create: function(context) {  
    // context => parserOptions/ruleId/option  
    return {  
      CallExpression: function(node) {  
        if (  
          getNestedProperty(node, 'callee.p  
            'querySelector' &&  
          !getNestedProperty(node, 'argumen  
            '.js-'  
        )  
      ) {  
        context.report(  

```

```
const rule = require('./prefer-node-suffix');  
const RuleTester = require('eslint').RuleTest  
const ruleTester = new RuleTester();  
ruleTester.run('prefer-node-suffix', rule, {  
  valid: [  
    'const itemsNode = document.querySele  
    'const obj = {}; obj.tabsNode = items  
  ],  
  invalid: [  
    {  
      code: 'const items = document.que  
      errors: [...]  
    }  
  ]  
}
```

```
const rule = require('./prefer-node-suffix');  
const RuleTester = require('eslint').RuleTest  
const ruleTester = new RuleTester();  
ruleTester.run('prefer-node-suffix', rule, {  
  valid: [  
    'const itemsNode = document.querySele  
    'const obj = {}; obj.tabsNode = items  
  ],  
  invalid: [  
    {  
      code: 'const items = document.que  
      errors: [...]  
    }  
  ]  
}
```

Пре/постпроцессинг

```
processors: {  
  ".ext": {  
    preprocess: function(text, filename)  
      return [string];  
  },  
  postprocess: function(messages, file)  
    return [Message];  
  },  
  supportsAutofix: true  
}  
}
```

Пример правила tslint

// 👎

```
class Barman {  
    ...  
}
```

// 👍

```
class Barista {  
    ...  
}
```



```
import * as ts from "typescript";  
import * as Lint from "tslint";
```

```
class NoBarmenWalker extends Lint.RuleWalker  
    public visitClassDeclaration(node: ts.I  
        if (node.name.text === 'Barman') {  
            this.addFailure(this.createFail  
                node.name.getStart(),  
                node.name.getWidth(),  
                Rule.FAILURE_STRING)  
        }  
    );  
};  
// ...
```

```
import * as ts from "typescript";  
import * as Lint from "tslint";
```

```
class NoBarmanWalker extends Lint.RuleWalker  
    public visitClassDeclaration(node: ts.  
        if (node.name.text === 'Barman') {  
            this.addFailure(this.createFail  
                node.name.getStart(),  
                node.name.getWidth(),  
                Rule.FAILURE_STRING)  
        }  
    );  
};  
// ...
```

```
import * as ts from "typescript";  
import * as Lint from "tslint";
```

```
class NoBarmenWalker extends Lint.RuleWalker  
    public visitClassDeclaration(node: ts.I  
        if (node.name.text === 'Barman') {  
            this.addFailure(this.createFail  
                node.name.getStart(),  
                node.name.getWidth(),  
                Rule.FAILURE_STRING)  
        };  
};  
// ...
```

```
import * as ts from "typescript";  
import * as Lint from "tslint";
```

```
class NoBarmenWalker extends Lint.RuleWalker  
  public visitClassDeclaration(node: ts.I  
    if (node.name.text === 'Barman') {  
      this.addFailure(this.createFailure  
        node.name.getStart(),  
        node.name.getWidth(),  
        Rule.FAILURE_STRING)  
    }  
  }  
  // ...
```

План

- Чем помогает линтер
- Как он это делает
- С чем его приготовить
- Куда его положить
- Правила для esLint



Кастомный линтер

Что нужно уметь линтеру

- Разбирать на AST
- Бегать по AST
- Выводить ошибки
- (Опционально) Исправлять ошибки

Подходы

- Новый синтаксис
- Расширенный синтаксис

class: center, middle

Подходы

- Новый синтаксис
- Расширенный синтаксис

???

- Кастомный токенизатор (напр. Ohmlang) + walker
- Универсальный парсер с плагинами

Кастомный токенизатор

- Текст в AST
- Грамматика
- Функция хождения
- Вывод ошибок

```

NLDataLog {
  Rules
    = ListOf<Rule, "\\n">
  Rule
    = Clause          -- fact
  Clause
    = ( classRow | word )+
  word = wordChar+
  classDecl = "class: "
  classRow = classDecl (className ("," clas
  className = classChar+
  classChar = ~(eol | "," | "\\n") any
  wordChar = any

```

```

NLDataLog {
  Rules
    = ListOf<Rule, "\\n">
  Rule
    = Clause          -- fact
  Clause
    = ( classRow | word )+
  word = wordChar+
  classDecl = "class: "
  classRow = classDecl (className ("," clas
  className = classChar+
  classChar = ~(eol | "," | "\\n") any
  wordChar = any

```

```

NLDataLog {
  Rules
    = ListOf<Rule, "\\n">
  Rule
    = Clause          -- fact
  Clause
    = ( classRow | word )+
  word = wordChar+
  classDecl = "class: "
  classRow = classDecl (className ("," cla
  className = classChar+
  classChar = ~(eol | "," | "\\n") any
  wordChar = any

```

```
RemarkJSSlider.semantics = RemarkJSSlider.gra
    .createSemantics()
    .addOperation('toAST', {
        Rules(rules) {
            return new Program(rules.toAST());
        },

        Rule_fact(head) {
            return new Rule(head.toAST(), []);
        },
        ...
    })
```

```
RemarkJSSlider.semantics = RemarkJSSlider.gra
    .createSemantics()
    .addOperation('toAST', {
        Rules(rules) {
            return new Program(rules.toAST());
        },

        Rule_fact(head) {
            return new Rule(head.toAST(), []);
        },
        ...
    })
```

```
const walker = new ASTWalker();
```

```
walker.traverse(ast,  
{  
  enterNode: (node, parent) =>  
  {  
    if (node.type && linter[node.type]) {  
      linter[node.type](node, parent)  
    }  
  }  
});
```

```
const linter = {
```

```
const walker = new ASTWalker();
```

```
walker.traverse(ast,  
{
```

```
  enterNode: (node, parent) =>
```

```
  {
```

```
    if (node.type && linter[node.type]) {  
      linter[node.type](node, parent)
```

```
    }
```

```
  }
```

```
});
```

```
const linter = {
```



```
const walker = new ASTWalker();
```

```
walker.traverse(ast,
```

```
{
```

```
  enterNode: (node, parent) =>
```

```
  {
```

```
    if (node.type && linter[node.type]) {
```

```
      linter[node.type](node, parent)
```

```
    }
```

```
  }
```

```
});
```

```
const linter = {
```

```
const walker = new ASTWalker();

walker.traverse(ast,
{
  enterNode: (node, parent) =>
  {
    if (node.type && linter[node.type]) {
      linter[node.type](node, parent)
    }
  }
});

const linter = {
```

```
const walker = new ASTWalker();

walker.traverse(ast,
{
  enterNode: (node, parent) =>
  {
    if (node.type && linter[node.type]) {
      linter[node.type](node, parent)
    }
  }
});

const linter = {
```

Кастомизация существующего (unified)

- MD => доп. токены => AST
- Правило на доп. токены
- ~~Функция хождения~~
- ~~Вывод ошибок~~

```
function tokenClassListSeparator(process, val
  const match = /^class:(.+)/.exec(value);
  if (match) {
    return process(match[0])({
      type: 'classRow',
      children: [{ type: 'classList', v
    });
  }
}
```

```
function tokenClassListSeparator(process, val
  const match = /^class:(.+)/.exec(value);
  if (match) {
    return process(match[0])({
      type: 'classRow',
      children: [{ type: 'classList', v
    });
  }
}
```

```
const visit = require('unist-util-visit');  
const reason = 'ClassName cannot be empty';
```

```
function noEmptyClassList(tree, file) {  
  visit(tree, 'classList', (file) => {  
    return (node) => {  
      const classList = node.value  
        .replace('class:', '')  
        .split(',')  
        .map(singleClassName => singleC  
        .filter(Boolean);
```

```
  if (classList.length === 0) {
```

```
const visit = require('unist-util-visit');  
const reason = 'ClassName cannot be empty';
```

```
function noEmptyClassList(tree, file) {  
  visit(tree, 'classList', (file) => {  
    return (node) => {  
      const classList = node.value  
        .replace('class:', ' ')  
        .split(',')  
        .map(singleClassName => singleC  
        .filter(Boolean);
```

```
    if (classList.length === 0) {
```



```
const visit = require('unist-util-visit');
const reason = 'ClassName cannot be empty';

function noEmptyClassList(tree, file) {
  visit(tree, 'classList', (file) => {
    return (node) => {
      const classList = node.value
        .replace('class:', '')
        .split(',')
        .map(singleClassName => singleC
        .filter(Boolean);
```

```
if (classList.length === 0)
```

```
//...
```

```
const noEmptyClassListToken = require('./no-e
```

```
const noEmptyClassListRule = require('./no-er
```

```
//...
```

```
const extraRule = rule('remark-lint:no-empty-  
guide.plugins = guide.plugins.concat(extraRul
```

```
remark( )
```

```
  .use(markdown)
```

```
  .use(noEmptyClassListToken)
```

```
  .use(guide)
```

```
  .use(html)
```

```
  .process(slides, function(err, file)
```

```
//...
const noEmptyClassListToken = require('./no-e
const noEmptyClassListRule = require('./no-er
//...
const extraRule = rule('remark-lint:no-empty-
guide.plugins = guide.plugins.concat(extraRul

remark( )
  .use(markdown)
  .use(noEmptyClassListToken)
  .use(guide)
  .use(html)
  .process(slides, function(err, file)
```

```
//...
const noEmptyClassListToken = require('./no-e
const noEmptyClassListRule = require('./no-er
//...
const extraRule = rule('remark-lint:no-empty-
guide.plugins = guide.plugins.concat(extraRul
remark( )
  .use(markdown)
  .use(noEmptyClassListToken)
  .use(guide)
  .use(html)
  .process(slides, function(err, file)
```

```
//...
const noEmptyClassListToken = require('./no-e
const noEmptyClassListRule = require('./no-er
//...
const extraRule = rule('remark-lint:no-empty-
guide.plugins = guide.plugins.concat(extraRul
remark( )
  .use(markdown)
  .use(noEmptyClassListToken)
  .use(guide)
  .use(html)
  .process(slides, function(err, file)
```

```
//...
const noEmptyClassListToken = require('./no-e
const noEmptyClassListRule = require('./no-er
//...
const extraRule = rule('remark-lint:no-empty-
guide.plugins = guide.plugins.concat(extraRul

remark( )
  .use(markdown)
  .use(noEmptyClassListToken)
  .use(guide)
  .use(html)
  .process(slides, function(err, file)
```

```
//...
const noEmptyClassListToken = require('./no-e
const noEmptyClassListRule = require('./no-er
//...
const extraRule = rule('remark-lint:no-empty-
guide.plugins = guide.plugins.concat(extraRul

remark( )
  .use(markdown)
  .use(noEmptyClassListToken)
  .use(guide)
  .use(html)
  .process(slides, function(err, file)
```

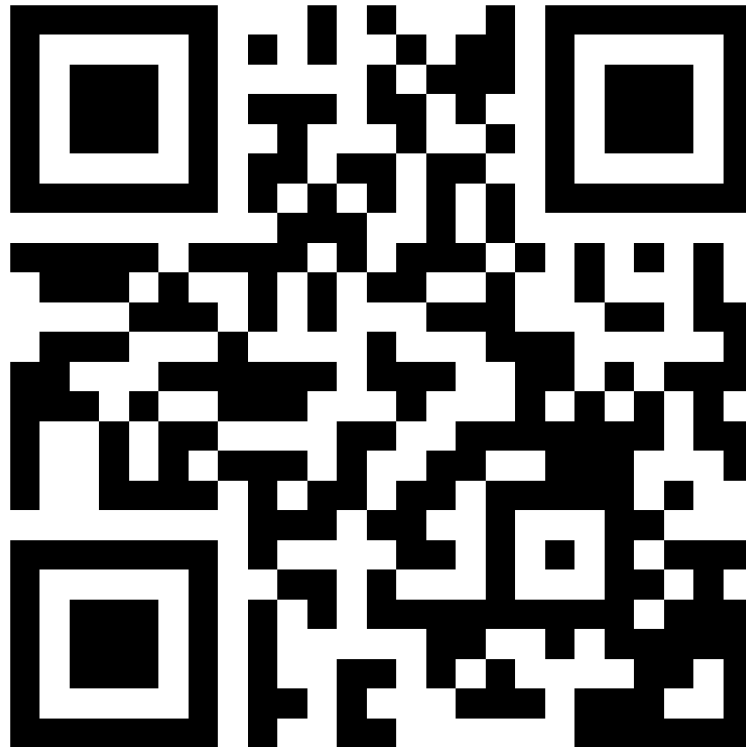
Итого

- Чем помогает линтер
- Как он это делает
- С чем его приготовить
- Куда его положить
- Правила для esLint
- Кастомный линтер



Вопросы?

- dkun.in
- @DKunin
- <https://bit.ly/2XpYUw1>



Полезные материалы

- <https://eslint.org/docs/developer-guide/working-with-plugins>
- <https://stylelint.io/developer-guide/plugins/>
- <https://palantir.github.io/tslint/develop/custom-rules/>
- <https://github.com/dustinspecker/awesome-eslint>
- <https://github.com/camelomartins/awesome-linters>
- <https://github.com/SAP/chevrotain>
- <https://github.com/acornjs/acorn>
- <https://github.com/jquery/esprima>
- <https://github.com/sindresorhus/awesome-lint>
- <https://github.com/syntax-tree/unists>
- <https://github.com/unifiedjs/unified>
- <https://github.com/DKunin/ohm-example>
- <https://github.com/DKunin/unified-test>
- <https://github.com/DKunin/tslint-example>
- <https://github.com/SonarSource/SonarTS--->