

Лекция №8: Современный Front-End

Web-программирование / ПГНИУ

CSS-препроцессоры

- CSS препроцессоры – инструменты трансляции (пред-обработки) более высокоуровневых языков в CSS
- SASS (SCSS), LESS, Stylus
- Переменные, вычисления, переиспользование (миксины, модули), функции, условия, циклы и т.д.



PostCSS



- Программа, которая автоматизирует рутинные операции с CSS с помощью расширений, написанных на языке JavaScript.
- Конвейер для унифицированных плагинов, выполняющих некоторые операции с CSS
- **autoprefixer** – автоматически добавляет вендорные префиксы
- **postcss-preset-env** – позволяет писать на современном CSS
- **cssnano** – оптимизация и сжатие CSS

```

1  ::placeholder {
2    color: gray;
3  }
4
5  .image {
6    background-image: url(image@1x.png);
7  }
8  @media (min-resolution: 2dppx) {
9    .image {
10     background-image: url(image@2x.png);
11   }
12 }

```

```

1  ::-webkit-input-placeholder {
2    color: gray;
3  }
4  ::-moz-placeholder {
5    color: gray;
6  }
7  :-ms-input-placeholder {
8    color: gray;
9  }
10 ::-ms-input-placeholder {
11  color: gray;
12 }
13 ::placeholder {
14  color: gray;
15 }
16
17 .image {
18  background-image: url(image@1x.png);
19 }
20 @media (-webkit-min-device-pixel-ratio: 2),
21        (-o-min-device-pixel-ratio: 2/1),
22        (min-resolution: 2dppx) {
23  .image {
24    background-image: url(image@2x.png);
25  }
26 }

```

```

1 /* normalize selectors */
2 h1::before, h1:before {
3     /* reduce shorthand even further */
4     margin: 10px 20px 10px 20px;
5     /* reduce color values */
6     color: #ff0000;
7     /* remove duplicated properties */
8     font-weight: 400;
9     font-weight: 400;
10    /* reduce position values */
11    background-position: bottom right;
12    /* normalize wrapping quotes */
13    quotes: '«' '»';
14    /* reduce gradient parameters */
15    background: linear-gradient(to bottom, #ffe500 0%, #ffe50
    #121 50%, #121 100%);
16    /* replace initial values */
17    min-width: initial;
18 }
19 /* correct invalid placement */
20 @charset "utf-8";

```

```

1 @charset "utf-8";h1:before{margin:10px 20px;color:red;font-
weight:400;background-position:100% 100%;quotes:"«"
"»";background:linear-gradient(180deg,#ffe500,#ffe500 50%,#121
0,#121);min-width:0}

```

PostHTML

- Программа, которая автоматизирует рутинные операции с HTML с помощью расширений, написанных на языке JavaScript
- **posthtml-md**
- **posthtml-img-autosize**
- **posthtml-w3c**

Babel *BABEL*

- **JavaScript компилятор (транспайлер)**
- Toolchain для компиляции множеством плагинов
- **Транспайлер** — тип компилятора, который использует исходный код программы, написанной на одном языке программирования, в качестве исходных данных и производит эквивалентный исходный код на другом языке программирования
- Переводит современный JS в старый

Put in next-gen JavaScript

```
[1, 2, 3].map(n => n ** 2);|
```

Get browser-compatible JavaScript out

```
[1, 2, 3].map(function (n) {  
    return Math.pow(n, 2);  
});
```

Put in next-gen JavaScript

```
const x = [1, 2, 3];  
foo([...x]);|
```

Get browser-compatible JavaScript out

```
var x = [1, 2, 3];  
foo([].concat(x));
```

Put in next-gen JavaScript

```
var obj = {  
    shorthand,  
    method() {  
        return "😁";  
    }  
};|
```

Get browser-compatible JavaScript out

```
var obj = {  
    shorthand: shorthand,  
    method: function method() {  
        return "😁";  
    }  
};
```



```
1 class Animal {}
2 class Cat extends Animal {}
3 const cat = new Cat();
```

```
1 "use strict";
2
3 function _inheritsLoose(subClass, superClass) { subClass.prototype =
  Object.create(superClass.prototype); subClass.prototype.constructor =
  subClass; subClass.__proto__ = superClass; }
4
5 var Animal = function Animal() {};
6
7 var Cat =
8   /*#__PURE__*/
9   function (_Animal) {
10     _inheritsLoose(Cat, _Animal);
11
12     function Cat() {
13       return _Animal.apply(this, arguments) || this;
14     }
15
16     return Cat;
17   }(Animal);
18
19 var cat = new Cat();
```

Другие языки

- TypeScript 
- Dart
- Elm
- Reason
- Kotlin.js
- Scala.js

```
1 class Student {
2     fullName: string;
3
4     constructor(public firstName: string, public middleInitial: string, public lastName: string) {
5         this.fullName = firstName + " " + middleInitial + " " + lastName;
6     }
7 }
8
9 interface Person {
10     firstName: string;
11     lastName: string;
12 }
13
14 function greeter(person: Person) {
15     return "Hello, " + person.firstName + " " + person.lastName;
16 }
17
18 let user = new Student("Jane", "M.", "User");
19
20 document.body.textContent = greeter(user);
```

Другие задачи

- Минификация - уменьшение размера кода
- Обфускация - запутывание кода
- Оптимизация кода - замена конструкций на более эффективные эквивалентные

```
1 var i = 0;
2 while (i++ < 10) {
3   alert( i );
4 }
5
6 if (i) {
7   alert( i );
8 }
9
10 if (i === '1') {
11   alert( 1 );
12 } else if (i === '2') {
13   alert( 2 );
14 } else {
15   alert( i );
16 }
```



```
1 for(var i=0;10>i++;)alert(i);i%6?alert(i):
2 "1"—i?alert(1):"2"—i?alert(2):alert(i);
```

Browserslist

- Как понять, в какую версию ES нам транспайлить код?
- Какие нужны полифилы? Префиксы в CSS?
- Нужно проверять всё на caniuse.com?

 .browserslistrc

```
1 last 2 versions
2 > 5% in USA
3 maintained node versions
4 not dead
5 IE10
```

Качество кода

Линтер – инструмент анализа кода, соблюдение стиля кода и поиска проблемных мест.

Самый популярный - **eslint** 

Форматер – инструмент, форматирующий код по определённым правилам.

Самый популярный - **prettier** 

NodeJS

- Node или Node.js — программная платформа, основанная на движке V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. (wikipedia)
- NodeJS – среда исполнения JavaScript и API для взаимодействия с ОС
- Позволяет писать на JS консольные утилиты, серверную часть, десктопные приложения, serverless и т.д.



Раньше фронтендерам нужны были только HTML, CSS и JS! А теперь им нужно изучать Node, npm, Grunt, gulp, Webpack, Babel... погодите...



На кой чёрт мне сдался Node во фронтенде?



Проблема управлением зависимостями

- Есть куча библиотек, каждую надо найти на её сайте (вспомнив или найдя сайт) и скачать / подключить
- Иногда надо проверять, а не вышла ли новая версия
- Обновили – всё сломалось, ведь какой-то другой библиотеке была нужна обязательно старая версия
- Решение - управление зависимостями через пакетный менеджер



- **NodeJS Package Manager**
- Приложение = пакет. Его описание хранится в `package.json`
 - `scripts` - различные скрипты, команды, использующиеся при разработке пакета
 - `dependencies` - ЗАВИСИМОСТИ
 - `devDependencies` - зависимости для разработки
- `node_modules` - директория, в которой хранятся установленные ЗАВИСИМОСТИ

Команды npm

Основные команды

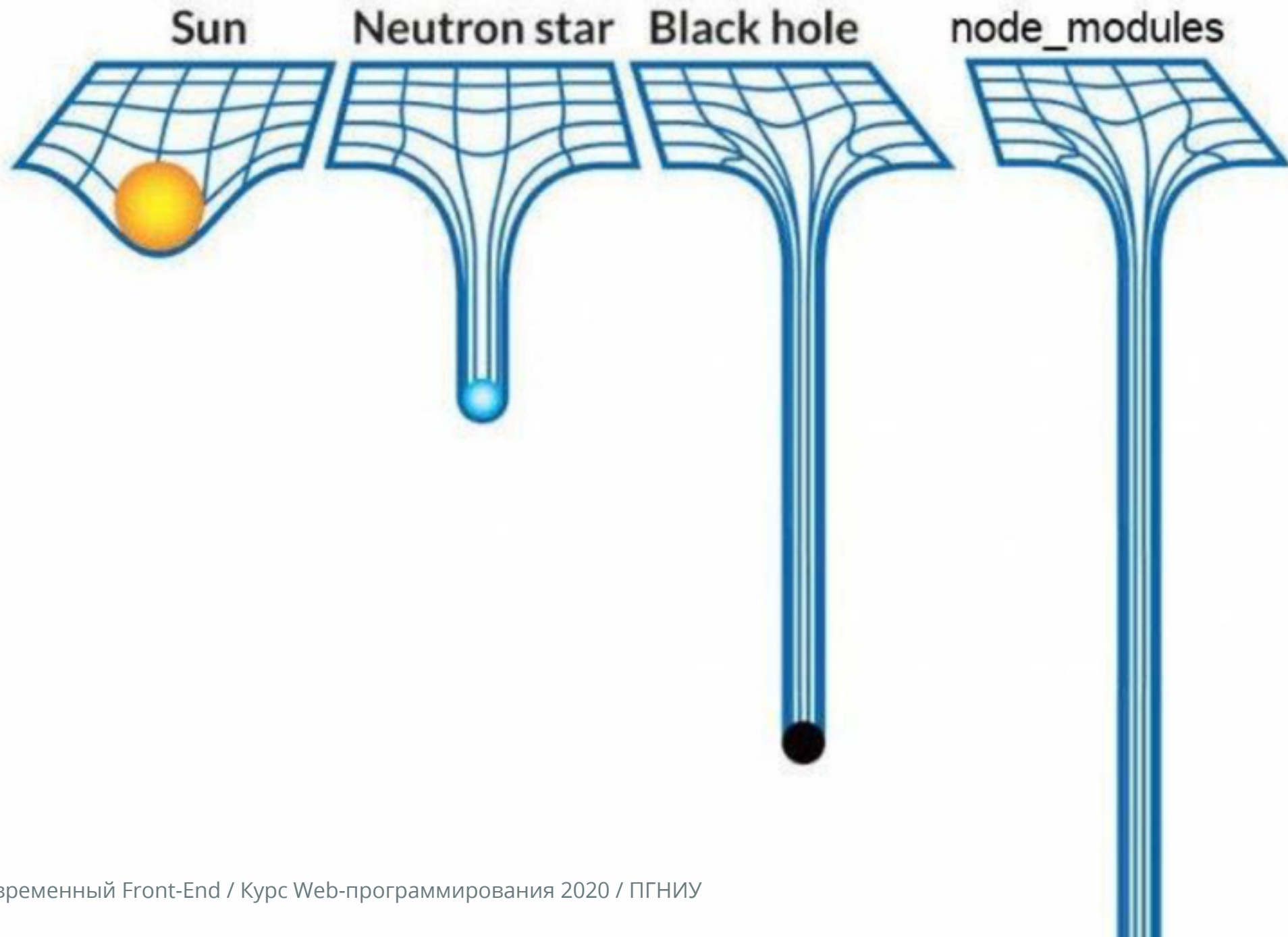
```
$ npm init`  
$ npm install  
$ npm install <package> [ <package>]  
$ npm install --save-dev <package> [ <package>]  
$ npm install --global <package>  
$ npm uninstall <package> [ <package>]  
$ npm run <script name>  
$ npx <bin>
```

Алиасы

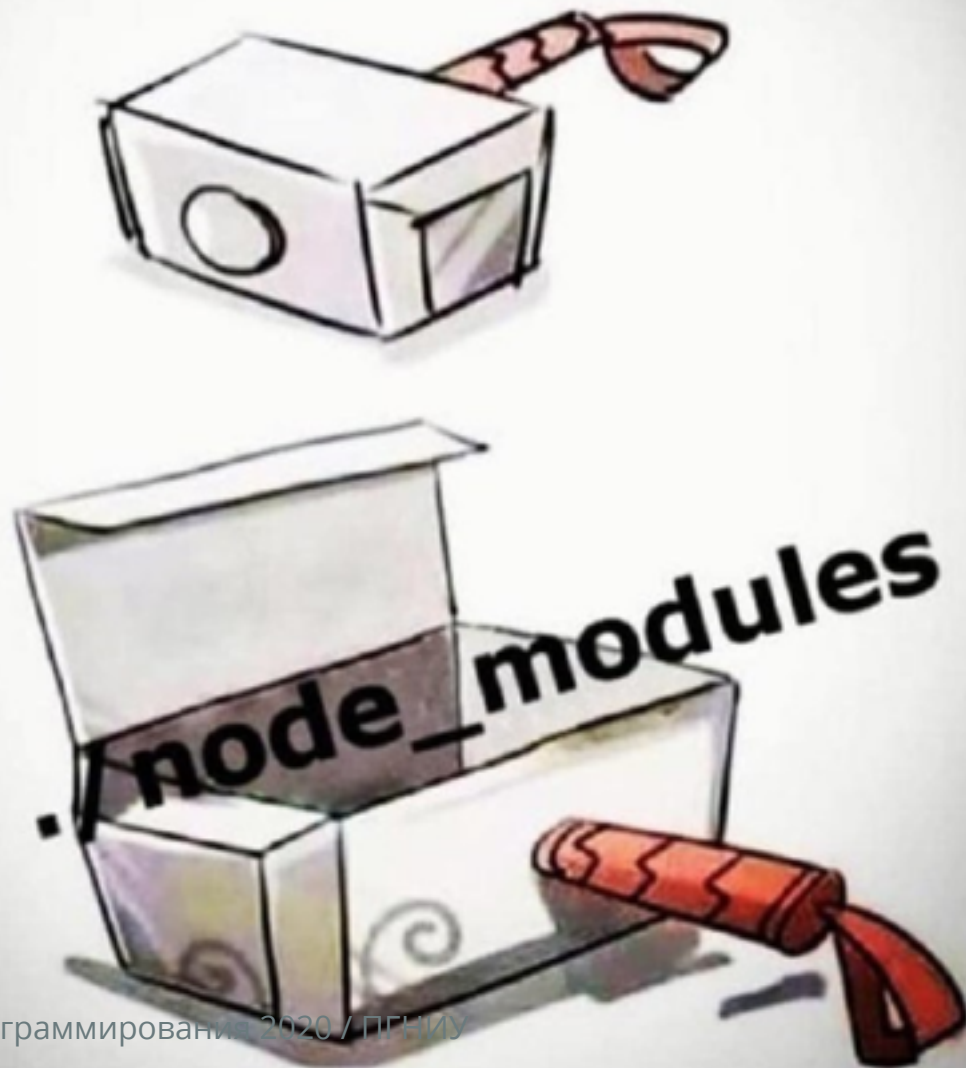
```
npm i === npm install  
npm i -D == npm install --save-dev  
npm i -g == npm install --global  
npm start === npm run start
```

- node_modules
- grunt-spritesmith
- node_modules
- spritesmith
- node_modules
- pixelsmith
- node_modules
- ndarray-fill
- node_modules
- cwise
- node_modules
- uglify-js
- node_modules
- yargs
- node_modules
- cliui
- node_modules
- center-align
- node_modules
- align-text
- node_modules
- kind-of
- node_modules
- is-buffer

node_modules



The secret behind Thor's hammer





node_modules

GitHub

mentos

Coca-Cola

Запуск задач

- Системы автоматизации задач
- `$ gulp js`



```
1 const { src, dest, parallel } = require('gulp');
2 const pug = require('gulp-pug');
3 const less = require('gulp-less');
4 const minifyCSS = require('gulp-cssso');
5 const concat = require('gulp-concat');
6
7 function html() {
8   return src('client/templates/*.pug')
9     .pipe(pug())
10    .pipe(dest('build/html'))
11 }
12
13 function css() {
14   return src('client/templates/*.less')
15     .pipe(less())
16     .pipe(minifyCSS())
17     .pipe(dest('build/css'))
18 }
19
20 function js() {
21   return src('client/javascript/*.js', { sourcemaps: true })
22     .pipe(concat('app.min.js'))
23     .pipe(dest('build/js', { sourcemaps: true }))
24 }
25
26 exports.js = js;
27 exports.css = css;
28 exports.html = html;
29 exports.default = parallel(html, css, js);
```

Webpack webpack

- Сборщик модулей JavaScript приложения
- Не только js модулей, но и других зависимостей приложения
- Исследование графа зависимостей, разделение бандла на части (chunks), TreeShaking
- webpack-dev-server
 - Локальный сервер с хостингом файлов
 - Прокси к API
 - Hot Module Replacement



```
1 const path = require('path');
2
3 module.exports = {
4   entry: './path/to/main.js',
5
6   output: {
7     path: path.resolve(__dirname, 'dist'),
8     filename: 'bundle.js'
9   },
10
11   rules: [ /* ... */ ],
12
13   plugins: [ /* ... */ ],
14
15   mode: 'development',
16 };
```

Правила (rules)

Правило определяется файлами, на которые оно действует (test) и списком загрузчиков (use), которые обрабатывают файлы

A screenshot of a code editor window titled 'webpack.config.js'. The code is written in JavaScript and defines a Webpack configuration. It shows the 'module' section with a 'rules' array. The first rule has a 'test' property with a regular expression for CSS files and a 'use' property pointing to 'css-loader'. The second rule has a 'test' property for TypeScript files and a 'use' property pointing to 'ts-loader'. The code is as follows:

```
1 module.exports = {  
2   module: {  
3     rules: [  
4       { test: /\.css$/, use: 'css-loader' },  
5       { test: /\.ts$/, use: 'ts-loader' }  
6     ]  
7   }  
8 };
```

Загрузчики (loaders)

- style-loader, css-loader
- less-loader, sass-loader, postcss-loader
- babel-loader, ts-loader
- json-loader, file-loader, csv-loader
- posthtml-loader
- eslint-loader

Плагины

Переиспользуемые модули, имеющие ту же систему конфигурации, что и сам webpack

- clean-webpack-plugin
- mini-css-extract-plugin
- hot-module-replacement-plugin
- uglifyjs-webpack-plugin

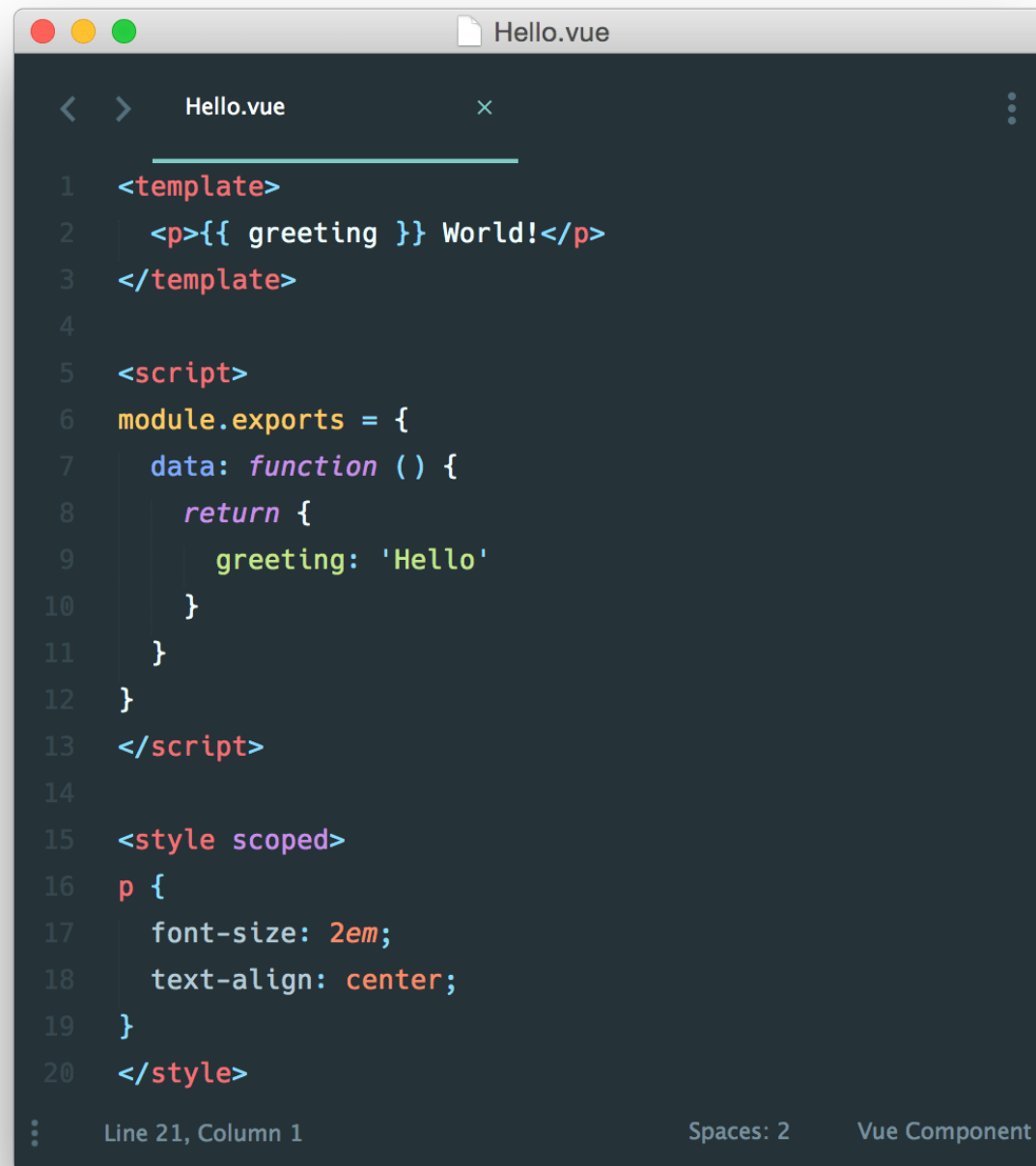
```

/* JSX */

class TodoApp extends React.Component {
  //...
  render() {
    return (
      <div>
        <h3>Список дел</h3>
        <TodoList items={this.state.items} />
        <form onSubmit={this.handleSubmit}>
          <label htmlFor="new-todo"> Что нужно сделать? </label>
          <input id="new-todo" onChange={this.handleChange} value={this.state.text} />
          <button> Добавить #{this.state.items.length + 1}</button>
        </form>
      </div>
    );
  }
}

class TodoList extends React.Component {
  render() {
    return (
      <ul>
        {this.props.items.map(item => (
          <li key={item.id}>{item.text}</li>
        ))}
      </ul>
    );
  }
}

```

```
1 <template>
2   <p>{{ greeting }} World!</p>
3 </template>
4
5 <script>
6   module.exports = {
7     data: function () {
8       return {
9         greeting: 'Hello'
10      }
11    }
12  }
13 </script>
14
15 <style scoped>
16   p {
17     font-size: 2em;
18     text-align: center;
19   }
20 </style>
```

Line 21, Column 1 Spaces: 2 Vue Component

Source Map

Специальные файлы, которые позволяют в процессе разработки и отладки соотносить исполняемый код и исходный код.

Ссылки: интересные статьи

- Каково оно учить JavaScript в 2016:
<https://habr.com/ru/post/312022/>
- Объясняем современный JavaScript динозавру:
<https://habr.com/ru/company/mailru/blog/340922/>

Ссылки: доклады

- HolyJS 2017 Moscow | Михаил Башуров – Yarn, npm v5 или pnpm — кто круче?
https://www.youtube.com/watch?v=hq_glihAs5A
- HolyJS 2017 Moscow | Модульный CSS — Андрей Оконечников:
<https://www.youtube.com/watch?v=vYmSYsj-s5w>
- HolyJS 2018 Moscow | Стас Курилов — Глубокое погружение в webpack:
<https://www.youtube.com/watch?v=aiYkJOPD9v8>
- HolyJS 2019 Moscow | Nicolò Ribaud — @babel/how-to:
https://www.youtube.com/watch?v=UeVq_U5obnE

Ссылки: инструменты

- NodeJS: <https://nodejs.org/>
- NPM: <https://www.npmjs.com/>
 - Документация по package.json: <https://docs.npmjs.com/configuring-npm/package-json.html>
 - Документация по CLI: <https://docs.npmjs.com/cli-documentation/cli>
- LESS, SASS: <http://lesscss.org/>, <https://sass-lang.com/>
- PostCSS: <https://postcss.org/>, <https://github.com/postcss/postcss>
- Babel: <https://babeljs.io/>

Ссылки: инструменты

- Browserslist: <https://github.com/browserslist/browserslist>,
<https://browserl.ist/>
- Webpack: <https://webpack.js.org/>
- Webpack DevServer: <https://webpack.js.org/configuration/dev-server/>
- ESLint: <https://eslint.org/>
- Prettier: <https://prettier.io/>
- core-js: <https://www.npmjs.com/package/core-js>