html academy
интерактивные
онлайн-курсы

Раздел 3:

Препроцессоры и автоматизация

1. Что такое препроцессоры?
2. Возможности Less и Sass.
3. Автоматизация.

Препроцессоры

Препроцессоры

- Профессиональные инструменты.
- Расширяют стандартные возможности языка.
- Бывают для CSS, HTML, JavaScript.

- Подключение файлов.
- Переменные.
- Математические операции.

- Управление цветами.
- Вложенные селекторы.


{less}


Sass

Less

- Написан на Ruby в 2009 году, портирован на JS.
- Работает прямо в Node.js и в браузере.
- Текущая версия 3.0.0.

lesscss.org и lesstester.com

Sass

- Написан на Ruby в 2006 году, портирован на C++.
- Есть: Dart Sass, LibSass, Ruby Sass.
- Текущая версия 3.5, наверняка в работе 4.0.

sass-lang.com и sassmeister.com

Sass и SCSS

```
a
  font:
    weight: bold
    family: serif
  &:hover
    background-color: #eee
```

```
a {
  font-weight: bold;
  font-family: serif;
  &:hover {
    background-color: #eee;
  }
}
```

Подключение

Подключение в CSS

```
/* style.css */
@import "normalize.css";

body {
  background-color: tomato;
}
```

```
/* normalize.css */
body {
  margin: 0;
}
```

Импортировать можно только в начале файла стилей.

Подключение в Less

```
/* style.less */
@import "normalize.less";

body {
  background-color: tomato;
}
```

```
/* normalize.less */
body {
  margin: 0;
}
```

Импортировать можно где угодно.

Подключение в Sass

```scss
/* style.scss */
@import "normalize.scss";

body {
  background-color: tomato;
}
```

```scss
/* normalize.scss */
body {
  margin: 0;
}
```

Импортировать можно где угодно.

Результат в Less и Sass

```
/* style.css */
body {
  margin: 0;
}
body {
  background-color: tomato;
}
```

Переменные

Переменные в CSS

```
01. html {
02.   --fancy-color: tomato;
03. }
04. body {
05.   color: var(--fancy-color);
06. }
```

Видимость в CSS

```
01. p {
02.   --fancy-color: tomato;
03. }
04. em {
05.   color: var(--fancy-color);
06. }
```

```
01. <p>
02.   <em>Я томат</em>
03. </p>
04.
05. <p></p>
06. <em>А я нет</em>
```

Переменные в Less

```
01. @fancy-color: tomato;
02.
03. body {
04.   color: @fancy-color;
05. }
```

```
01. body {
02.   color: tomato;
03. }
```

Видимость в Less

```
01. p {
02.   @fancy-color: tomato;
03. }
04. em {
05.   color: @fancy-color;
06. }
```

```
01. p {
02.   @fancy-color: tomato;
03.   em {
04.     color: @fancy-color;
05.   }
06. }
```

Переменные в Sass

```
$fancy-color: tomato;

body {
  color: $fancy-color;
}
```

```
body {
  color: tomato;
}
```

Видимость в Sass

```
01. p {
02.   $fancy-color: tomato;
03. }
04. em {
05.   color: $fancy-color;
06. }
```

```
01. p {
02.   $fancy-color: tomato;
03.   em {
04.     color: $fancy-color;
05.   }
06. }
```

Результат в Less и Sass

```
01. body {
02.   color: tomato;
03. }
```

- Простая замена псевдонимов на значения.
- Происходит только один раз во время сборки.
- Для тем нужно собирать отдельные файлы.

Интерполяция

Интерполяция переменных

```
@where: bottom;

.no-@{where} {
  border-@{where}: none;
}
```

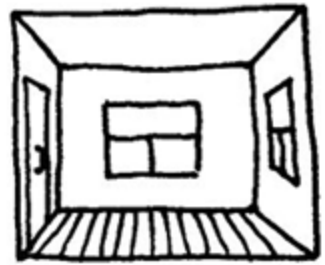
```
$where: bottom;

.no-#{$where} {
  border-#{$where}: none;
}
```

Результат интерполяции

```
01. .no-bottom {
02.   border-bottom: none;
03. }
```


МУЗЕЙ




5


4 5 1


1 3 6 5


10 р

Математика

Математические операции в CSS

```
01. html {
02.   --body-font-size: 16px;
03. }
04. body {
05.   margin-left: calc(-10px * 2);
06.   font-size: calc(var(--body-font-size) * 1.25);
07. }
```

Математические операции в CSS

```
01. body {
02.   width: calc(100% - 50px * 2);
03.   font-size: calc(1vw * 1em);
04. }
```

— Рассчитываются вживую, прямо в браузере.

— Можно использовать относительные единицы: `%`, `em`, `vw`.

Математические операции в Less

```
01. @body-font-size: 16px;
02.
03. body {
04.   margin-left: -10px * 2;
05.   font-size: @body-font-size * 1.25;
06. }
```

Математические операции в Sass

```
01. $body-font-size: 16px;
02.
03. body {
04.   margin-left: -10px * 2;
05.   font-size: $body-font-size * 1.25;
06. }
```

Результат в Less и Sass

```
01. body {
02.   margin-left: -20px;
03.   font-size: 20px;
04. }
```

- Рассчитываются только один раз во время сборки.
- Ничего не знают про относительные единицы `%`, `em`, `vw`.
- Будьте осторожны при сложении разных единиц.

Цвета

Операции с цветами в CSS

```
01. body {
02.   color: #302682;
03.   color: rgba(47, 38, 130, 0.5);
04.   color: hsla(246, 55%, 33%, 0.5);
05. }
```

HSL — Hue, Saturation, Lightness (оттенок, насыщенность, светлость).

```
hsl(246, 55%, 33%)
```

Операции с цветами в Less

```
01. body {
02.   color: fade(#302682, 50%);
03.   color: desaturate(#302682, 30%);
04.   color: lighten(#302682, 20%);
05. }
```

Операции с цветами в Sass

```
01. body {
02.   color: rgba(#302682, 0.5%);
03.   color: desaturate(#302682, 30%);
04.   color: lighten(#302682, 20%);
05. }
```

Результат в Less и Sass

```
01. body {
02.   color: rgba(48, 38, 130, 0.5);
03.   color: #443f69;
04.   color: #5445c9;
05. }
```

— Рассчитывается финальное значение в HEX.

— Широкий набор функций по настройке цветов.

Вложенность

Псевдоклассы

```
.link {
  color: tomato;
  &:hover,
  &:focus {
    color: plum;
  }
}
```

```
.link {
  color: tomato;
}
.link:hover,
.link:focus {
  color: plum;
}
```

Псевдоэлементы

```
.item {
  color: tomato;
  &::before {
  content: "—";
  }
}
```

```
.item {
  color: tomato;
}
.item::before {
  content: "—";
}
```

Модификаторы

```
.button {
  width: 100px;
  &--wide {
    width: 500px;
  }
}
```

```
.button {
  width: 100px;
}
.button--wide {
  width: 500px;
}
```

❌ Элементы

```
01. .button {
02.   color: tomato;
03.   &__icon {
04.     width: 16px;
05.   }
06. }
```

```
01. .button {
02.   color: tomato;
03. }
04. .button__icon {
05.   width: 16px;
06. }
```

```
01. a {
02.   color: tomato;
03.   b {
04.     color: plum;
05.     c {
06.       color: wheat;
07.     }
08.   }
09. }
```

```
01. a {
02.   color: tomato;
03. }
04. a b {
05.   color: plum;
06. }
07. a b c {
08.   color: wheat;
09. }
```

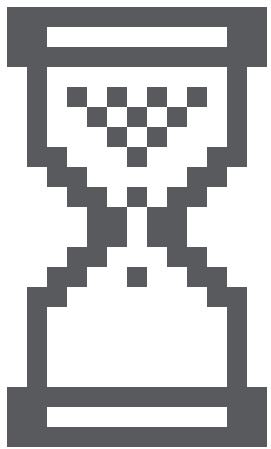
Кроличья нора

```
01. .block {
02.   &__elem {
03.     &--mod {
04.       &::before { … }
05.     }
06.   }
07. }
```

```
01. .block__elem {
02.   &--mod {
03.     &::before { … }
04.   }
05. }
```

Д10. Нет вложенности больше двух уровней.

Д11. Верное использование `&` в стилевых файлах.



Примеси

Примеси в Less

```
.heading() {
  margin-bottom: 16px;
  font-size: 32px;
}
```

```
.clearfix() {
  &::after {
    display: table;
    clear: both;
    content: "";
  }
}
```

Примеси в Sass

```
01. @mixin heading {
02.   margin-bottom: 16px;
03.   font-size: 32px;
04. }
```

```
01. @mixin clearfix {
02.   &::after {
03.     display: table;
04.     clear: both;
05.     content: "";
06.   }
07. }
```

Примеси в Less и Sass

```
01. .title {
02.   color: tomato;
03.   .heading();
04.   .clearfix();
05. }
```

```
01. .title {
02.   color: tomato;
03.   @include heading;
04.   @include clearfix;
05. }
```

Результат в Less и Sass

```
.title {
  color: tomato;
  margin-bottom: 16px;
  font-size: 32px;
}
```

```
.title::after {
  display: table;
  clear: both;
  content: "";
}
```

Примеси в HTML

```
.heading {
  margin-bottom: 16px;
  font-size: 32px;
}
.clearfix::after {
  clear: both;
}
```

```
<h2 class="
    title
    heading
    clearfix
">
```

Примеси с параметрами в Less

```
01. .size(@width: 50px, @height: 50px) {
02.   width: @width;
03.   height: @height;
04. }
05. .square {
06.     .size(100px, 100px);
07. }
```

Примеси с параметрами в Sass

```
01. @mixin size($width: 50px, $height: 50px) {
02.   width: $width;
03.   height: $height;
04. }
05. .square {
06.     @include size(100px, 100px);
07. }
```

Результат в Less и Sass

```
01. .square {
02.     width: 100px;
03.     height: 100px;
04. }
```

Расширения

Расширения в Less

```
.list-reset {
  list-style: none;
}
.main-nav__items {
  margin: 0;
  &:extend(.list-reset);
}
```

Расширения в Sass

```
.list-reset {
  list-style: none;
}
.main-nav__items {
  margin: 0;
  @extend .list-reset;
}
```

❌ Результат в Less и Sass

```
.list-reset,
.main-nav__items {
  list-style: none;
}
.main-nav__items {
  margin: 0;
}
```

§28. Знакомство с Less

§37. Примеси в Less

Рекомендации

- Для каждого блока — отдельный файл стилей.
- Не создавайте вложенность больше двух уровней.
- Не используйте цветовые функции для получения цветов из дизайна.
- Не используйте расширения, совсем.
- Не используйте сразу всё, пробуйте постепенно.

Автоматизация

```
img/
fonts/
less/
  blocks/
    main-nav.less
    page-main.less
  style.less
  mixins.less
  variables.less
```

```
img/
fonts/
sass/
  blocks/
    main-nav.scss
    page-main.scss
  style.scss
  mixins.scss
  variables.scss
```

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS | FOUNDATION

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

Important March 2018 security upgrades now available

Download for macOS (x64)

8.11.1 LTS
Recommended For Most Users

Other Downloads | Changelog | API Docs

9.11.1 Current
Latest Features

Other Downloads | Changelog | API Docs

Or have a look at the LTS schedule.

Sign up for Node.js Everywhere, the official Node.js Weekly Newsletter.


Gulp

Неудобный CSS

```
.wait {
  -ms-interpolation-mode: nearest-neighbor;
  image-rendering: -webkit-optimize-contrast;
  image-rendering: -moz-crisp-edges;
  image-rendering: -o-pixelated;
  image-rendering: pixelated;
}
```

Удобный CSS

```
.wait {
  image-rendering: pixelated;
}
```

Плагин autoprefixer для PostCSS.

PostCSS

```
01. var postcss = require("gulp-postcss");
02. var autoprefixer = require("autoprefixer");
03. gulp.task("css", function () {
04.   return gulp.src("*.css")
05.     .pipe(postcss([autoprefixer()]))
06.     .pipe(gulp.dest("build"));
07. });
```

Неудобный SVG

```
01. <body>
02.   <svg viewBox="0 0 547.769 410.595" fill="#cd6799">
03.     <path d="M471.418 235.99a112.63 112.63 0 0 0-49.627 11…
04.       9.729-4.561 16.32-29 66.16-36 81.6a467.21 467.21 0 0…
05.       13.92 44.64c8.013 0 16.075-10.391 19.664-15.688l.015…
06.   </svg>
07. </body>
```

Удобный SVG

```
01. <body>
02.   <include src="images/icon.svg"></include>
03. </body>
```

Плагин posthtml-include для PostHTML.

PostHTML

```
01. var posthtml = require("gulp-posthtml");
02. var include = require("posthtml-include");
03. gulp.task("html", function () {
04.   return gulp.src("*.html")
05.     .pipe(posthtml([include()]))
06.     .pipe(gulp.dest("build"));
07. });
```

1. Less + Gulp
2. Less + Grunt
3. Sass + Gulp

4. Sass + Grunt

Задание на дом

1. Пройти дополнительные курсы.
2. Выбрать препроцессор и сборщик.
3. Получить обновления от Кекса.
4. Проверить проект с помощью линтеров.
5. Начать вёрстку с CSS-препроцессором.