УДК 004.435

СРАВНИТЕЛЬНЫЙ АНАЛИЗ CSS-ПРЕПРОЦЕССОРОВ

А. Ю. Радченко
Донской государственный технический университет (г. Ростов-на-Дону, Российская Федерация)

Веб-разработчики довольно часто отказываются от препроцессоров в пользу нативного CSS (формальный язык описания внешнего вида документа, от англ. cascading style sheets — каскадные таблицы стилей). Статья посвящена функциям, преимуществам и сравнению самых популярных препроцессоров. Выявлены несколько критериев для сравнивания, определены условия, когда можно использовать только нативный CSS, и обоснованно выбран оптимальный вариант.

Ключевые слова: веб-программирование, CSS, CSS-препроцессоры.

COMPARATIVE ANALYSIS OF CSS PREPROCESSORS

A. Yu. Radchenko
Don State Technical University (Rostov-on-Don, Russian Federation)

Web developers quite often abandon preprocessors in favor of native CSS (a formal language for describing the appearance of a document, from the English language — cascading style sheets). The article is devoted to the functions, advantages and comparison of the most popular preprocessors. Several criteria for the comparison are identified, the conditions are defined when only native CSS can be used, and the optimal option is reasonably chosen.

Keywords: Web programming, CSS, CSS-preprocessors.

Введение. Препроцессоры CSS (формальный язык описания внешнего вида документа, от англ. cascading style sheets — каскадные таблицы стилей) — это языки, специально написанные, чтобы добавить ценные функции в CSS без нарушения совместимости браузера. При этом написанный код компилируется в обычный CSS, который можно использовать в любом браузере [1].

Цель исследования — разработать критерии сравнивания и выяснить, какой препроцессор лучше подходит для работы и в каких случаях нужно использовать нативный CSS.

Самые популярные препроцессоры для веб-разработки:

— SASS (SCSS),

— LESS,

— Stylus.

SASS (SCSS) — это метаязык на основе CSS, предназначенный для повышения уровня абстракции CSS-кода и упрощения файлов каскадных таблиц стилей.

LESS — это динамический язык стилей. Он создан под влиянием языка стилей SASS и оказал влияние на его новый синтаксис — SCSS.

Stylus — это препроцессор CSS, который был написан на JavaScript для Node.js [2].

Все перечисленные препроцессоры имеют одинаковые возможности:

— объявление переменных,

— вложенность свойств и селекторов,

— создание и использование миксинов,

— импорт других стилей в текущий файл,

— использование цветовых функций,

— выполнение математических операций.

Однако у них есть и различия, которые рассмотрим ниже.

Критерии сравнивания:

— синтаксис,

— наследование,

— скорость работы.

Синтаксис. Наиболее важной частью написания кода с использованием препроцессора CSS является понимание его синтаксиса. Синтаксис SCSS и LESS идентичен стандартному CSS, а для Stylus он более подробный. Stylus имеет три стиля синтаксиса в одном. Первый идентичен CSS. Второй предполагает, что можно не писать фигурные скобки. В третьем можно не писать двоеточия и точку с запятой. Можно использовать в одном файле разные варианты синтаксиса Stylus. Нецелесообразен симбиоз нескольких стилей в одном файле или технологии, поэтому следует предпочесть SASS (SCSS) и LESS. На рис. 1 — примеры синтаксиса для SCSS, LESS и Stylus.

```
/* style.scss или style.less */
h1 {
  color: #0982C1;
}

/* style.styl */
h1 {
  color: #0982C1;
}

/* style.styl - без фигурных скобок */
h1
  color: #0982C1;

/* style.styl - без двоеточия и точки с запятой */
h1
  color #0982C1
```

Рис. 1. Примеры синтаксиса препроцессоров

Наследование — это способность одного селектора CSS получать свойства другого. Такой функционал отлично работает в SASS (SCSS) и Stylus (рис. 2).

```
.block {
  margin: 10px 5px;
  padding: 2px;
}

p {
  @extend .block; /* унаследованные стили от '.block' */
  border: 1px solid #EEE;
}
ul, ol {
  @extend .block; /* унаследованные стили от '.block' */
  color: #333;
  text-transform: uppercase;
}
```

Рис. 2. Пример наследования в SASS и Stylus

На рисунке 3 показан скомпилированный результат в CSS.

```
.block, p, ul, ol {
  margin: 10px 5px;
  padding: 2px;
}
p {
  border: 1px solid #EEE;
}
ul, ol {
  color: #333;
  text-transform: uppercase;
}
```

Рис. 3. Скомпилированный в CSS результат наследования в SASS и Stylus

Результат наследования в LESS будет иным. Вместо добавления нескольких селекторов к одному набору свойств LESS рассматривает наследование как миксин без аргументов и импортирует стили в свои собственные селекторы (рис. 4). У такого решения есть недостаток: свойства повторяются в скомпилированном CSS.

```
.block {
  margin: 10px 5px;
  padding: 2px;
}
p {
  margin: 10px 5px;
  padding: 2px;
  border: 1px solid #EEE;
}
ul,
ol {
  margin: 10px 5px;
  padding: 2px;
  color: #333;
  text-transform: uppercase;
}
```

Рис. 4. Скомпилированный в CSS результат наследования в LESS

Скорость работы. SASS (SCSS) написан на языке программирования Си, остальные — на JavaScript. Си компилируется заранее и затем просто выполняется, а JavaScript интерпретируется и иногда компилируется во время выполнения с помощью JIT-компилятора, поэтому SASS (SCSS) будет работать немного быстрее [3].

Нативный CSS. В каких же случаях использовать нативный CSS? Из трех ключевых возможностей (переменные, вложенность, миксины) в CSS реализована только одна — переменные, которые пока неприменимы из-за кроссбраузерности [4]. В 2021 году CSS переменные не поддерживаются в Internet Explorer, Opera Mini и Opera Mobile. Поэтому только CSS и его нативные возможности можно задействовать в простых проектах, например для лендингов. В серьезных же проектах такой подход, скорее всего, будет неэффективным. Отсутствие вложенности и примесей как средства абстракции сильно усложняет поддержку кода, поэтому данную нишу прочно занимают препроцессоры [5].

Заключение. Каждый рассмотренный препроцессор CSS (SASS, LESS и Stylus) имеет собственный уникальный способ выполнения задачи и позволяет разработчикам использовать полезные неподдерживаемые функции, сохраняя при этом совместимость браузера и чистоту кода.

Анализ показал, что Stylus и LESS уступают по критериям скорости, синтаксиса и наследования. Следовательно, в 2021 году оптимальным выбором будет SASS (SCSS). Кроме того, SASS (SCSS), по статистике, популярнее и чаще используется в проектах, поэтому разработчики активно поддерживают и развивают данный препроцессор.

Библиографический список

1. Croom, J. Sass, LESS, Stylus — какой лучше: препроцессор Shootout / J. Croom // Envato Tuts+ : [сайт]. — URL: https://code.tutsplus.com/ru/tutorials/sass-vs-less-vs-stylus-preprocessor-shootout--net-24320 (дата обращения: 13.01.2021).

2. Neretin-Trike. Препроцессор Stylus / Neretin-Trike // GitHub : [сайт]. — URL: https://gist.github.com/neretin-trike/214fe69cf632fbd9db04d702b7f303c1 (дата обращения: 15.01.2021).

3. Lukoie. Стоит ли использовать препроцессор, отличный от SCSS? // Хабр Q&A : [сайт]. — URL: https://qna.habr.com/q/592472 (дата обращения: 29.01.2021).

4. CSS Environment Variables env // Can I use : [сайт]. — URL: https://caniuse.com/?search=var%20css (дата обращения: 10.02.2021).

5. Романов, А. Нужны ли CSS-препроцессоры, или Насколько мы близки к ванильному CSS // andrew-r.ru : [сайт]. — URL: https://andrew-r.ru/notes/preprocessors-vs-vanilla-css/ (дата обращения: 17.02.2021).

Об авторах:

Радченко Александр Юрьевич, магистрант кафедры «Медиапроизводство» Донского государственного технического университета (344000, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), radch.alex97@gmail.com.

Author:

Radchenko, Aleksandr Yu., Master's degree student, Department of Media Production, Don State Technical University (1, Gagarin sq., Rostov-on-Don, RF, 344003), radch.alex97@gmail.com.