LESS (Leaner Style Sheets, компактная таблица стилей)— это динамический язык стилей, который разработал Алексис Сельер (Alexis Sellier) в 2009г.

Он создан под влиянием языка стилей Sass, и, в свою очередь, оказал влияние на его новый синтаксис «SCSS», в котором также использован синтаксис, являющийся расширением CSS. CSS - это простой язык. Он позволяет нам задавать стили для HTML элементов с помощью селекторов. Однако, со временем размер вашего веб-проекта может вырасти, и вы получите большой объем повторяющегося кода CSS. Если вы столкнулись с этой проблемой, то самое время использовать препроцессор CSS.

Что такое препроцессор CSS?

Препроцессор (от англ. Preprocessor) — это инструмент, преобразующий код из одного синтаксиса в другой. Обычно, на вход препроцессора поступает код, написанный с использованием синтаксических конструкций, понятных этому препроцессору.

Стоит сказать, что препроцессор может полностью замещать синтаксические конструкции языка или частично расширять их, то есть добавлять новые конструкции. Так как препроцессор преобразует код, то помимо входа у него должен быть и выход. Иначе какой смысл в его работе, ведь другие программы не смогут воспользоваться его трудами.

Как правило, на выходе ожидается код с более низким уровнем. То есть код, лишённый синтаксических конструкций, вносимых препроцессором. Сейчас нам не важно, что происходит с такими конструкциями, допустим, что препроцессор их раскрывает или удаляет в соответствии с заложенными в него правилами. Иначе говоря, на выходе мы получаем код, понятный программе, которая будет использовать его после препроцессора.

Пример

Рассмотрим работу препроцессора на простейшем примере. Условия задачи у нас будут такими:

Пусть на вход препроцессора поступает строка, включающая в себя последовательность слов, разделённых пробелами, запятыми и точками. Основная функция препроцессора — замена ключевого слова «.CSS.» на его полную форму записи «Cascading Style Sheets». На выходе препроцессора имеем преобразованную строку.

Итак, исходя из условий мы понимаем, что «.CSS.» — это синтаксическая конструкция, которую должен обработать препроцессор.

Допустим, что на вход была подана следующая строка:

```
Developers use CSS preprocessors to build .CSS. faster.
```

Как мы видим, в этой строке два раза встречается слово «CSS». В первом случае это будет обычным словом как, например, «use» или «build», а во втором — синтаксической конструкцией препроцессора.

Далее препроцессор преобразует строку в соответствии с заложенным в него функционалом, и, в зависимости от настроек, на выходе имеется преобразованная строка.

```
Developers use CSS preprocessors to build Cascading Style Sheets faster.
```

Так работают препроцессоры в самом примитивном случае. В больших проектах все куда сложнее.

CSS препроцессор **(от англ. CSS preprocessor) — это надстройка над CSS, которая добавляет ранее недоступные возможности для CSS, с помощью новых синтаксических конструкций.**

Основная задача препроцессора — это предоставление удобных синтаксических конструкций для разработчика, чтобы упростить, и тем самым, ускорить разработку и поддержу стилей в проектах.

CSS препроцессоры преобразуют код, написанный с использованием препроцессорного языка, в чистый и валидный CSS-код.

При помощи препроцессоров вы можете писать код, который нацелен на:

- ***Читабельность для человека***
- ***Структурированность и логичность***
- ***Производительность***

Препроцессоры позволяют определить свойства один раз и затем повторно использовать их в нашем проекте, что в гораздо более функционально, чего простой CSS делать не может.

Какие бывают CSS-препроцессоры?

Пора перейти к более конкретным примерам, а именно к самим CSS-препроцессорам. На момент написания книги можно выделить три популярных препроцессора:

- Less
- Sass (SCSS)
- Stylus

Less

Собственно, герой этой книги. Самый популярный на момент написания книги препроцессор. Основан в 2009 году Алексис Сельер (Alexis Sellier) и написан на JavaScript (изначально был написан на Ruby, но Алексис вовремя сделал правильный шаг). Имеет все базовые возможности препроцессоров и даже больше, но не имеет условных конструкций и циклов в привычном для нас понимании. Основным плюсом является его простота, практически стандартный для CSS синтаксис и возможность расширения функционала за счёт системы плагинов.

LESS обеспечивает следующие расширения CSS: переменные, вложенные блоки, миксины, операторы и функции.

LESS может работать на стороне клиента (Internet Explorer 6+, WebKit, Firefox) или на стороне сервера под управлением **Node.js или Rhino.**

Sass (SCSS)

Самый мощный из CSS-препроцессоров. Имеет довольно большое сообщество разработчиков. Основан в 2007 году как модуль для HAML и написан на Ruby (есть порт на C++). Имеет куда больший ассортимент возможностей в сравнении с Less. Возможности самого препроцессора расширяются за счёт многофункциональной библиотеки Compass,

которая позволяет выйти за рамки CSS и работать, например, со спрайтами в автоматическом режиме.

Имеет два синтаксиса:

- Sass (Syntactically Awesome Style Sheets) — упрощённый синтаксис CSS, который основан на идентации. Считается устаревшим.
- SCSS (Sassy CSS) — основан на стандартном для CSS синтаксисе.

Stylus

Самый молодой, но в тоже время самый перспективный CSS-препроцессор. Основан в 2010 году небезызвестной в наших кругах личностью TJ Holowaychuk. Говорят, это самый удобный и расширяемый препроцессор, а ещё он гибче Sass. Написан на JavaScript. Поддерживает уйму вариантов синтаксиса от подобного CSS до упрощённого (отсутствуют :, ;, {} и некоторые скобки).

Компиляция

Если вы помните, то препроцессоры предлагают нам свой вариант синтаксиса для некоторых или всех конструкций языка, надстройкой над которым они являются. И CSS-препроцессоры не исключение.

Для того, чтобы браузер понимал код, написанный с использованием синтаксических конструкций препроцессора, его нужно компилировать в понятный для него язык. Таким языком для браузера, как не сложно догадаться, является CSS.

Существует несколько вариантов того, как можно перейти от Less к CSS.

1 способ Компиляция в браузере (less.js)

Наиболее простой способ использования CSS-препроцессора, но в тоже время малопопулярный. Альтернативные решения удобнее и предоставляют наиболее интересный функционал. Применяется на этапе разработки или отладки проекта, когда важен результат компиляции, а не её скорость.

Основан на идее подключения стилей с расширением .less к документу, используя стандартный тег <link>, но с измененным атрибутом rel. А также осуществляется подключение файла библиотеки.

Для начала свяжите ваши .less таблицы стилей с rel атрибутом, установленным в " stylesheet/less":

```
<link rel="stylesheet/less" type="text/css" href="styles.less" />
```

Затем загрузите less.js (http://lesscss.org/)и включите его в <script></script>тег в <head>элементе вашей страницы:

```
<script src="less.js" type="text/javascript"></script>
```

После компиляции скрипт проводит инъекцию полученного CSS-кода в секцию head документа посредством тега <style>.

Способ не желателен к применению на так называемом «продакшене» в виду того, что имеет серьезные проблемы со скоростью и сильно зависит от производительности устройства, а также скорости интернет-соединения. Помимо этого увеличивается объем загружаемых данных, так как браузеру пользователя приходится загружать less-файлы и файл библиотеки. Только после полной загрузки необходимых ресурсов начинается процесс компиляции less-кода в CSS.

2 способ Через системы компиляторы (less в css) и подключаем css

```
<link rel="stylesheet " type="text/css" href="styles.css" />
```

Установить http://koala-app.com/

материалы https://mrmlnc.gitbooks.io

3 способ Компиляция через Терминал

Через плагины

1) **Устанавливаем Node.js https://nodejs.org/en/ и**

устанавливаем NPM (устанавливается вместе с Node.js, **npm** – это менеджер пакетов, который входит в состав Node.js.)

2) **Устанавливаем Less Глобально через терминал**. Открываем Windows консоль

- ➔ (Нажмите значок поиска на Панели задач или кнопку Пуск
- ➔ В строке поиска напечатайте cmd. ...
- ➔ В результатах поиска нажмите правую кнопку мыши на классическом приложении Командная строка
- ➔ В открывшемся меню выберите пункт **Запустить** от имени администратора
- ➔ *Убедимся, что node установился. Для этого в cmder проверим версию Node.js с помощью команды.* Данные команды выводят версию node.js и npm
  - node –v
  - npm -v
- ➔ *Устанавливаем LESS , для этого прописываем*
  - npm install -g less

Рассмотрим синтаксис команд npm:

npm - пакетный менеджер;
install, то есть «установить»;
-g - флаг, который указывает на то, что пакет будет установлен глобально;
less - имя устанавливаемого пакета;

➔ *После* установки всех компонентов проверяем работу LESS, для этого в командной строке вводим:

- lessc -v

Мы должны получить предупреждение *lessc: no input files*, которое свидетельствует что LESS работает и ругается на отсутствие входящего файла.

Далее выполняете 3 или 4 пункт

3) **Устанавливаем и настраиваем плагины в Sublime Text**

➔ Устанавливаем плагин Sublime **LESS** (syntax highlighting — подсветка кода less)

➔ Установим плагины Sublime **Less2Css** (компиляция LESS в CSS)

➔ Установим плагины Sublime **Sublime OnSaveBuild (**SublimeOnSaveBuild – работает с препроцессорами LESS, SASS, JADE, выполняет компиляцию в момент сохранения файла. **)**

➔ Откроем настройки **Sublime : Preferences → Settings — Default,** и в конец пишем строку:

```
"lesscCommand": "/usr/local/share/npm/bin/lessc"
```

➔ Открываем Windows консоль и вводим :

- npm install -g less-plugin-clean-css

➔ В user-настройках плагина SublimeOnSaveBuild вставляем

```
{
"filename_filter": "\\.(css|js|sass|less|scss)$", "build_on_save": 1
 }
```

➔ Отключаем устаревший “css minifier”, идём в настройки **Preferences → Package Settings → Less2Css → Settings — User**. И добавляем строчку:

```
{
 "minify": false
}
```

4) **Устанавливаем плагины в VSC**

➔ Устанавливаем плагин **Easy Less**

5) **Сохраните** .less файл (ctrl+s). Произошел билд less файла, и рядом с style.less файлом появился готовый минифицированный style.css. К проекту подключаете style.css.

```
<link rel="stylesheet " type="text/css" href="styles.css" />
```

Дополнительный плагин lesshint для вывода подсказок