CSS - это простой язык. Он позволяет нам задавать стили для HTML элементов с помощью селекторов. Это просто даже для начинающих.

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.

В этом уроке мы объясним, что такое CSS-препроцессор и рассмотрим один из самых популярных: LESS.

Что такое препроцессор CSS?

CSS препроцессоры не заменяют CSS. На самом деле, препроцессоры просто дают нам дополнительные функциональные возможности, такие как переменные,  операторы и функции, с которыми мы можем упростить создание и управление файлами CSS.

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.

Что такое LESS?

LESS относительно новый препроцессор, ему около 4 лет. Его часто сравнивают с SASS, более старым препроцессором. Как и LESS, так и SASS, имеют свои корни в языке программирования Ruby, но в настоящее время используются гораздо шире. LESS теперь основан на Javascript.

Люди, использующие LESS, могут создать заранее определенный набор цветов для своего сайта. Они могут определить особый стиль один раз и затем многократно использовать его везде, где необходимо.

Использование LESS

Чтобы воспользоваться тем, что LESS может предложить, мы сначала должны сохранить наш CSS-код в файл с расширением *.less*. Использовать LESS очень просто. Он работает как на веб-сервере, так и на стороне клиента.

Самый простой способ [начать работу на стороне клиента](http://www.webmasters.by/articles/html-coding/3407-set-up-less.html). Добавьте ссылку на less-файл на вашей веб-странице, так же как добавляете CSS-файл, но с атрибутом rel установленным как "stylesheet/less", а затем файл less.js, который можно загрузить с [www.lesscss.org](http://www.lesscss.org/)

**<link** rel="stylesheet/less" type="text/css" href="/styles.less" **/>**   
**<script** src="/less.js" type="text/javascript"**></script>**

Этот метод работает только в современных браузерах и подходит только для использования на локальном сервере. Для рабочего сайта необходима предварительная компиляция с применением Node на веб-сервере или различные инструменты от сторонних производителей.

Некоторые из инструментов сторонних производителей:

* [WinLess](http://winless.org/)
* [Simpless](http://wearekiss.com/simpless)
* [Crunch](http://crunchapp.net/)
* и многие другие ...

LESS Синтаксис: Переменные

Одни из вещей, которые часто повторяются в файле CSS - это цвета. Одни и те же цвета повторяются в файлах CSS для разных элементов, заголовков, ссылок и т.д.

#header {   
    **background-color**: #CCCCCC   
}  
.aside {   
    **background-color**: #CCCCCC;   
    **color**: #000000;  
.sidebar a {    
    **border-bottom**: 1px solid #CCCCCC;    
}

В LESS, мы можем просто объявить переменную для сохранения цвета, вот так:

@grey: #CCCCCC;

И затем повторно использовать это так:

#header {    
    **background-color**: @grey;    
}    
.aside {    
    **background-color**: @grey;    
    **color**: #000000;    
}    
.sidebar a {    
    **border-bottom**: 1px solid @grey;    
}

Мы можем также использовать переменные, чтобы сохранить другие свойства, такие как ширина, высота, размер шрифта и многие другие значения, которые вы планируете повторно использовать в различных местах стилей.

LESS Синтаксис: Mixins

Переменные полезны для повторного использования отдельных значений, таких как цвет, размер шрифта. В LESS с mixins мы можем использовать набор свойств из одного или нескольких правил. Mixins могут быть использованы, например, для определения стиля границ, размер шрифта и т.д.

.border\_top\_bottom {  
    **border-top**: dotted 1px #000;  
    **border-bottom**: solid 2px #000;  
}

В приведенном выше коде мы определили верхнюю и нижнюю границу внутри класса border\_top\_bottom. Теперь, когда мы хотим добавить эти стили к другими элементами, мы можем использовать их так:

#header {  
    **color**: #000000;  
    .border\_top\_bottom;  
}  
.**content** a {  
    **color**: #000000;  
    .border\_top\_bottom;  
}

Приведенный выше код выведет следующее:

#header {   
    **color**: #000000;   
    **border-top**: dotted 1px #000000;   
    **border-bottom**: solid 2px #000000;   
}  
.**content** a {   
    **color**: #000000;   
    **border-top**: dotted 1px #000000;   
    **border-bottom**: solid 2px #000000;   
}

Чтобы добавить больше гибкости, mixins также позволят нам передавать переменные (называемые здесь параметры). Например:

.border-radius (@radius: 5px) {   
    border-radius: @radius;   
    -moz-border-radius: @radius;   
    -webkit-border-radius: @radius;   
}

В приведенном выше примере, мы задали параметр для радиуса. Мы также дали ему значение по умолчанию равное 5px, которое не является обязательным. Теперь мы хотим добавить это для класса button:

.button {   
    .border-radius(6px);   
}

Если мы уберем 6px из приведенного выше примера, границы радиуса будут заданы по умолчанию, который указан в нашем классе border-radius и равен 5px.

Синтаксис LESS: Nesting

Одна из вещей, распространеных в CSS, это длинные селекторы, которые нам часто приходится писать для стилей дочерних элементов:

nav { }  
nav li { }  
nav li a { }  
nav li a:hover { }  
nav li a.active { }  
nav li a.visited { }

И если у вас есть несколько подуровней выпадающего меню, то это становится более сложным для понимания. В LESS это может быть записано как:

nav {  
    li {  
        a {  
            &amp;:hover { }  
            &amp;:active { }  
            &amp;:visited { }  
        }  
    }  
}

Символ "&" используется перед псевдо-классами в нашей вложенной структуре. Описанную выше конструкцию гораздо легче понять и показать иерархию именно так, как она есть.

Синтаксис LESS: Операции

LESS также позволяет нам выполнять такие операции как сложение, вычитание, умножение и деление числовых значений, цветов и переменных в таблице стилей. Допустим, мы объявили переменную *padding*, которая применяется ко всем нашим элементам H1. Но теперь мы хотим добавить дополнительный отступ для нашего заголовка на главной странице. Мы можем сделать это следующим образом:

@padding: 5px;  
h1 {  
    **padding**: @padding;  
}  
h1.page-title {  
    **padding**: (@padding \* 2);  
}

Значение по умолчанию будет умножено на 2, что задаст заголовку на главной странице padding равный 10px. Операции должны выполняться в круглых скобках, однако они могут также работать без скобок.

Синтаксис LESS: Scope

Область действия в LESS похожа на область действия в других языках программирования. Переменные и mixins сначала ищутся на локальном уровне, и если они не найдены, то компилятор будет искать их в родительской области и так далее.

@var: red;  
  
#page {  
@var: white;  
  
#header {  
**color**: @var; // выведет white  
    }  
}  
#footer {  
    **color**: @var; // выведет red  
}

Заключение

Эта статья не охватывает все что LESS может предложить. Цель состоит в том, чтобы объяснить, как используя LESS можно помочь нам сделать наш CSS более динамичным. CSS препроцессоры позволяют нам создавать и управлять CSS более эффективно. Даже имея хорошее понимание CSS, LESS позволит нам использовать его более продуктивно.