Hareesh N Nampoothiri
Research Scholar, University of Kerala, Thiruvananthapuram
http://www.newnmedia.com/~haree

Let's do LESS

CSS dynamic languages are the new buzz in the web development world. LESS, developed by Alexis Sellier (alias cloudhead), is an open-source CSS dynamic language which adds mechanisms like variables, nesting, mixins, operators and functions to normal CSS.

Since the beginning of SGML in the 1980s, style sheet languages were in use in one form or the other. Cascading style sheets were developed as a consistent approach in providing style information for web documents. In 1994, Håkon Wium Lie proposed Cascading HTML Style Sheets (CHSS) which can be considered as the first version of CSS. Later, the 'HTML' part was removed from the name, since the style sheets can be used with other markup languages besides HTML. In December 1996, W3C officially released the level 1 version of CSS. W3C Working Group addressed the various issues faced in CSS level 1 and released the CSS level 2 in November 1994. CSS level 3, which was started in 1998, is in its final stage of development as of now. The 3rd level of CSS extends the features defined in CSS 2 and incorporates many modules which adds new capabilities. CSS dynamic languages (*LESS, Sass, Stylus etc.*) extends the CSS 3 functionality further by adding dynamic behavior such as variables, mixins, operations, and functions. Sometimes, these languages are also referred as CSS Preprocessor languages.

Using LESS

Those who are familiar with CSS can instantly start working with LESS. Since both CSS and LESS use the same semantics, any valid code in CSS is valid in LESS as well. The major advantage of LESS (*in comparison with other similar languages*) is that it has a smaller learning curve (*almost nil*) for designers and developers as the familiar CSS syntax feels very natural.

Requirements

You may use any normal text editor to write LESS. But, it is highly recommended to use a specialized editor. **Crunch!** is an editor of that kind. It also serves as a compiler which can be used to convert a file written in LESS to normal CSS. The application can be downloaded from here: http://crunchapp.net/

If you are a Notepad++ user, then LESS can be enabled using the User-Defined Dialogue... option available in the View menu. Download the user-defined language file for LESS and then import it in the window opened by selecting the above option. The user-defined language file for LESS can be downloaded from here: http://bit.ly/notepad-language-files

You also need the latest version of `less.js` file, if you intend to use the LESS style sheet directly in an HTML file. It can be downloaded from here: http://lesscss.org

Features

We all love to achieve more by doing less and in less time. LESS exactly enables us to do the same. Let's move on to the features of LESS, which extends the basic functionality of CSS and help us to achieve more by doing less.

Variables

An explanation seems unnecessary here. CSS do not provide an option to use variables and LESS simply adds that functionality. The official website gives this description:

> *Variables allow you to specify widely used values in a single place, and then re-use them throughout the style sheet, making global changes as easy as changing one line of code.*

```
// LESS
@my-color: #43464C;
header{
    background: @my-color;
}
footer{
    background: @my-color;
}
```

When compiled to CSS it will give the following result.

```
/* Compiled CSS */
header{
    background: #43464C;
}
footer{
    background: #43464C;
}
```

Mixins

Using mixins we can mix things up. Here we will define a class (mixin class) and then mix it in other classes to achieve the properties defined in the former class. Mixins can also act similar to a function. We can pass arguments to a mixin class and get custom results based on the passed values.

> *Mixins allow you to embed all the properties of a class into another class by simply including the class name as one of its properties. It's just like variables, but for whole classes.*

```
// LESS
.fullround (@radius: 10px) {
   border-radius: @radius;
   -webkit-border-radius: @radius;
   -moz-border-radius: @radius;
}
header{
   .fullround;
}
#header-inner{
   .fullround (5px);
}
```

```
/* Compiled CSS */
header{
    border-radius: 10px;
    -webkit-border-radius: 10px;
    -moz-border-radius: 10px;
}
```

```
#header-inner{
    border-radius: 5px;
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
}
```

Nested Rules

Using LESS it is possible to nest selectors inside other selectors. This helps the designer to avoid constructing long selector names. Another advantage is that it makes inheritance clear and style sheets shorter.

```
// Less
header{
   background:#FFF;
   color: #000;
   h1{ font-size: 20pt;
       font-weight: bold;
   }
   p{
       font-size: 12pt;
       line-height: 14pt;
       font-weight: normal;
       a: {
          text-decoration: none;
          color: #00F;
          &:hover{ color: #0FF; }
       }
   }
}
```

```
/* Compiled CSS */
header{
    background: #FFF;
    color: #000;
}
header h1{
    font-size: 20pt;
    font-weight: bold;
}
header p{
    font-size: 12pt;
    line-height: 14pt;
    font-weight: normal;
}
```

```
header p a{
    text-decoration: none;
    color: #00F;
}
header p a:hover{
    color: #0FF;
}
```

Functions & Operations

LESS provides some inbuilt functions and operations. Operations will help to define elements proportional to other elements. Functions permits the designer to manipulate the values in different ways.

> *Operations let you add, subtract, divide and multiply property values and colors, giving you the power to create complex relationships between properties. Functions map one-to-one with JavaScript code, allowing you to manipulate values however you want.*

```
// LESS
@my-color: #43464C;
@default-height:150px;

header{
    background: @my-color;
    height: @default-height;
}
footer{
    background: fade(@my-color,50%);
    height: @default-height * 2;
}
```

```
/* Compiled CSS */
header{
    background: #43464C;
    height:150px;
}
footer{
    background: rgba(67,70,76,0.5);
    height:300px;
}
```

Usage

First ink the `*.less` file in the HTML with rel set to `"stylesheet/less"`. Then include the `less.js` file in the `<head>` element of the page. The style sheets should be included before the script file. Sample HTML code is given below.

```
<!DOCTYPE html>
<head>
    <link rel="stylesheet/less"
type="text/css" href="styles.less">
    <script src="less.js"
type="text/javascript"></script>
</head>
<body>
</body>
</html>
```

This method is often recommended only in the development stages. Once a website is fully developed, it is advisable to compile LESS files to normal CSS and link it in the HTML files. Otherwise, if the user prefers not to run JavaScript, the styles will not get applied to the pages. Another option is to compile it in the server itself, when each time the page is requested. It requires configuring the server as well as it will increase the time required to load the page.

To compile LESS, Crunch! can be used in Windows/Mac machines. **WinLess** is another application for Windows machines which will help you to compile LESS files. WinLess can be downloaded from here: http://winless.org/

SimpLESS is another compiler which supports Windows, Mac and Linux platforms. Download the application from here: http://wearekiss.com/simpless

About the Author

Hareesh N Nampoothiri is a visual design consultant with an experience of more than a decade and worked with government organizations like C-DIT, C-DAC, University of Kerala and other private organizations. Currently he is doing research in University of Kerala, on communication design with special reference to the aesthetic principles of Indian art. He is an author of two books on graphic design and a regular columnist in leading technology magazines including CSI Communications. Kathakali, blogging and photography are his passions. He has directed a documentary feature on Kathakali and also directed an educational video production for IGNOU, New Delhi.