2. More About Less

Top Three Features

Once you get done reading the Less Features Guide online (at http://lesscss.org/features/ (http://lesscss.org/features/)) you will truly understand the power of a CSS pre-processor. But before that happens this week, let me whet your appetite with an overview of the top three features in my humble opinion.

The syntax for Less is almost identical to CSS. The authors of Less were smart and made Less compatible with CSS, meaning you can write CSS in your Less files. But do remember to save your Less files names with the suffix .less and not .css.

1. Variables

So let's take a look at using variables with Less, and demonstrate what you can do with them.

Variables in Less begin with the @ sign. What follows that can be any combination of letters and numbers, dashes and underscores. Once you've named your variable, follow with a colon the variable's stored value. This can include a hex code for a color (very common) or a string enclosed in quotes. Let's take a look at a couple of variables to see how they'll become CSS once compiled.

```

@color1: #00214D;
@size1: 15px;
```

Now we can use those variables in any other Less files throughout your project. This is a way to keep your code minimal without repetitions.

When you use variables in Less files, they compile to be perfectly clean CSS.

```
h1 {
 color: @color1;
}
h3 {
 font-size: @size1;
}
```

These compile to:

```
h1 {
 color: #00214D;
}
h3 {
 font-size: 15px;
}
```

2. Nested Rules

While we know the HTML language is filled with parent and child relationships representing nested elements, we also know CSS does not support nested rules. With Less, we're basically doing the same concept of nesting rules within other rules.

In your HTML you might have the following:

```
<header class="x">
    <nav> ... </nav>
</header>
```

If you want to write a CSS rule for the nav element inside this specific header, you could write

```
header.x nav { ... }
```

With Less, you can write the more transparent nested rules:

```
header.x {
   nav { ... }
}
```

Which compiles to the CSS example above.

3. Mix-ins

A Mixin in Less is basically a common group of CSS properties grouped into one class-based rule, which can then be inserted into various other Less rules. You can think of it like a reusable Less rule that other Less rules can incorporate.

For example, what if your design incorporated a common border and background treatment in many element's style rules? With a Less mix-in you could write one rule and add it to other element's rules.

```
.mixit1 {
   border:2px solid red;
   background-color:yellow;
}

#element1 {
   font-size:20px;
   .mixit1
}
```

The CSS compiled rule for #element1 would be

```
#element1 {
   font-size:20px;
   border:2px solid red;
   background-color:yellow;
}
```

Next Steps

Hopefully, the three features discussed here have gotten you ready for the deep dive on Less. Ready? **Dive into the Less Language Features Guide. (http://lesscss.org/features/)**