cssless用法

Less 是一种动态的样式语言。Less扩展了CSS的动态行为，比如说，设置变量（Variables）、混合书写模式（mixins）、操作（operations）和功能（functions）等等，最棒的是，Less使用了现有的CSS语法，也就是说，你可以直接把你现成的样式文件“style.css”直接改成“style.less”，他也能正常工作。如：

```
<link rel="stylesheet/less" href="less/style.less" />
```

Less现在可以在客户端（如：IE+,Webkit,Firefox）和服务器（如node.js）上运行。前面也说过Less是CSS的一种扩展，他不但向后兼容，而且在现有的CSS语法基础上增加许多额外的功能。如果你具有一定的CSS语法基础，学习Less将是一件轻而易举的事情，那么我们现在就开始吧，首先一起看一段用Less语法的CSS代码：

```
.box-shadow (@x: 0, @y: 0, @blur: 1px, @alpha) {
    @val: @x @y @blur rgba(0, 0, 0, @alpha);
    box-shadow:         @val;
    -webkit-box-shadow: @val;
    -moz-box-shadow:    @val;
}
.box { @base: #f938ab;
    color:        saturate(@base, 5%);
    border-color: lighten(@base, 30%);
    div { .box-shadow(0, 0, 5px, 0.4) }
}
```

看到这里或许你现在并不知道这些代码表示的是什么意思？不过不要紧张，我们会一步一步一介绍这些语法表示的是什么意思。别的先不说，我们一起动起来吧。

如何使用Less

要成功的使用Less，需要一个脚本的支持，这个脚本我们把他叫做less.js。大家可以在点击 这里 下载这个less脚本,并放到你的项目中。下载好以后我们需要把less.js引用到文件中，引用方式很简单：

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

Less包含些什么？

Less具体有哪些东东呢？其实这个问题我在前面早就有说过，Less要CSS语法的基础上进行了扩展，主要包含： **Variables，Mixins、Nested Rules、Functions & Operations、Client-side usage、Server-side usage** 等等，下面我们就针对这几个部分，更好帮助我们了解和深入的学习Less。

1、变量——Variables

Less中的变量充许你在样式中的某个地方对常用的值进行定义，然后应用到样式中，这样只要改变你定义的变量参数值就可以达到改变全局的效果，我们先来看一段代码：

Less Code

```
/*======== 定义变量===========*/
@color: #4d926f;
/*======== 应用到元素中 ========*/
#header {
    color: @color;
}
h2 {
    color: @color;
}
```

Compiled Css code:

```
/*======= Less 编译成 css ======*/
#header {
```

```
    color: #4d926f;
}
h2 {
    color: #4d926f;
}
```

Less中的变量还具有计算功能，如： **Less Code:**

```
@nice-blue: #5b83ad;
@light-blue: @nice-blue + #111;
#header {
    color: @light-blue;
}
```

Compiled Css Code:

```
#header {color: #6c94be;}
```

我们还可以定义一个变量名为变量，如

Less Code:

```
@color: #253636;
@highlight: "color";
#header {color: @@highlight;}
```

Compiled Css Code:

```
#header {color: #253636;}
```

注：在Less中的变量实际上就是一个“常量”，因为它们只能被定义一次。

Less Code:

```
    @color: #253636;
    @highlight: "color";
    @color: #ff3636;
    #header {color: @@highlight;}
```

Compiled Css Code:

```
#header {color: #ff3636;}
```

上面代码很明显说明了后的@color覆盖了前面的@color。

2、混入——Mixins

混入其实就是一种嵌套，它充许你将一个类嵌入到另一个类中，而被嵌入的这个类也称为是一个变量。换句话说，你可以用一个类定义CSS，然后把整个为当作一个变量来使用，嵌入到另一人类中当作他的属性；另外混入也像一个带有参数的functions，如下在的例子：

Less Code:

```
/*========== 定义一个类 ===========*/
.roundedCorners(@radius:5px) {
    -moz-border-radius: @radius;
    -webkit-border-radius: @radius;
    border-radius: @radius;
}
/*============ 定义的类应用到另个一个类中 ============*/
#header {
    .roundedCorners;
}
#footer {
    .roundedCorners(10px);
}
```

Compiled Css Code:

```
#header {
    -moz-border-radius:5px;
    -webkit-border-radius:5px;
    border-radius:5px;
}
#footer {
    -moz-border-radius:10px;
    -webkit-border-radius:10px;
    border-radius:10px;
}
```

注：这样任何CSS的类或ID下的样式都可以当作变量，使用混入模式用来当作另一个元素的属性值。

混入(Mixin)有一个名词叫“混入参数（Parametric Mixins）”，上面也说过。Less具有一个特殊类型的规则集，那就是一个类可以当作另一个元素的属生值，并且还可以接受其自己的参数，我们来看一个典型的实例：

Less Code:

```
/*========== 定义一个规则，并且不设置默认参数值 ============*/
.borderRadius(@radius){
    -moz-border-radius: @radius;
    -webkit-border-radius: @radius;
    border-radius: @radius;
}
/*============ 应用到元素中 ============*/
#header {
    .borderRadius(10px); /*把10px传给变量@radius*/
}
.btn {
    .borderRadius(3px);/*把3px传给变量@radius*/
}
```

Compiled Css Code:

```
#header {
    -moz-border-radius: 10px;
    -webkit-border-radius: 10px;
    border-radius: 10px;
}
.btn {
    -moz-border-radius: 3px;
    -webkit-border-radius: 3px;
    border-radius: 3px;
}
```

我们还可以给Mixins的参数定义一人默认值，如

Less Code:

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

.btn {
    .borderRadius;
}
```

Compiled Css Code:

```
.btn {
    -moz-border-radius: 5px;
    -webkit-border-radius: 5px;
    border-radius: 5px;
}
```

还有一种方法就是给Mixins不定我任何参数，特别是在你想隐藏输出的CSS规则，但又想在别的规则中包含他的属性，使用这种不带参数的Mixins将非常有用的，我们来看一段代码：

Less Code:

```
.wrap(){
    text-wrap: wrap;
    white-space: pre-wrap;
    white-space: -moz-pre-wrap;
    word-wrap: break-word;
}
pre {
    .wrap;
}
```

Compiled Css Code:

```
pre {
    text-wrap: wrap;
    white-space: pre-wrap;
    white-space: -moz-pre-wrap;
    word-wrap: break-word;
}
```

Mixins还有一个重要的变量：@arguments。@arguments在Mixins中具是一个很特别的参数，当Mixins引用这个参数时，他将表示所有的变量，当你不想处理个别的参数时，这个将很有用，我们来看一个阴影的实例：

Less Code:

```
.boxShadow(@x:0,@y:0,@blur:1px,@color:#000){
    -moz-box-shadow: @arguments;
    -webkit-box-shadow: @arguments;
    box-shadow: @arguments;
}

#header {
    .boxShadow(2px,2px,3px,#f36);
}
```

Compiled Css Code:

```
#header {
    -moz-box-shadow: 2px 2px 3px #FF36;
    -webkit-box-shadow: 2px 2px 3px #FF36;
    box-shadow: 2px 2px 3px #FF36;
}
```

3、嵌套规则——Nested Rules

嵌套规则主要是针对一多层元素的样式规则写法，以前我们在多层元素中写样式，要么从头选下来，要么另外给这个元素加上类名或id名，但在Less中我们不需要这样操作了，我们只要使用他的嵌套规则就可以完成，我们来看一个简单的实例：

Html Markup:

```
<div id="header">
    <h1><a href="">W3cplus</a></h1>
    <p>记述前端那些事—引领Web前沿</p>
</div>
```

Less Code:

```
#header {
    display: inline;
    float: left;
    h1 {
        font-size: 26px;
        font-weight: bold;
        a {
            text-decoration: none;
            color: #f36;
            &:hover {
                text-decoration: underline;
```

```
                color: #63f;
            }
        }
    }
    p {
        font-size: 12px;
    }
}
```

Compiled Css Code:

```
#header {
    display: inline;
    float: left;
}
#header h1 {
    font-size: 26px;
    font-weight: bold;
}
#header h1 a {
    color: #FF3366;
    text-decoration: none;
}
#header h1 a:hover {
    color: #6633FF;
    text-decoration: underline;
}
#header p {
    font-size: 12px;
}
```

使用Less的嵌套规则让你的CSS代码更简洁，因为他的写法就是模仿HTML的DOM结构来写的。

从上在的实例代码中，我都很清楚的了解到，嵌套规则可以让我们写样式时能像DOM树形那样有结构的去写代码，从而减了选择器的层级关系，更主要的是这样使用我们的代码更简洁，更具有阅读生，这种嵌套规则对我们操作伪元素更为方便和重要，如:hover，:link,:focus等，他的写法是：

Less Code:

```
a {
    color: red;
    text-decoration: none;
    &:hover {
        color: blue;
        text-decoration: underline;
    }
}
```

Compiled Css Code:

```
a {
    color: red;
    text-decoration: none;
}
a:hover {
    color: blue;
    text-decoration: underline;
}
```

大家注意了，这里的&很重要，在Less中嵌套书写中有没有&区别是完全不一样的效果，有&时解析的是同一个元素或此元素的伪类，没有&解析是后代元素，我们一起来看一段代码：

Less Code:

```
#header {
    &.fl{
        float: left;
    }
    .mln {
        margin-left: 0;
```

```
    }
}
```

Compiled Css Code:

```
#header.fl{float: left;}
#header .mln {margin-left: 0;}
```

4、Functions & Operations

这两个功能很有意思的。在我们平时的样式中，有很多元素的属性都具有一定的比例或倍数。那么这两个刚好可以帮我们实现这方面的功能，首先来看Operations（直译“动作”）他可以让你对元素的属性值，颜色进行四则运算：加、减、乘、除。而Function就像javascript中的function一样可以让你进行你想要的值的操作。下面我们先来看一个简单的实例：

Less Code:

```
@the-border: 1px;
@base-color: #111;
@red: #842210;
#header {
    color: @base-color *3;
    border: 1px solid desaturate(@red,100%);
    border-width: @the-border @the-border*2 @the-border*3 @the-border;
    border-color:desaturate(@red,100%) @red lighten(@red, 10%)  darken(@red, 30%);
}
```

Compiled Css Code:

```
#header {
    color: #333;
    border: 1px solid #4a4a4a;
    border-width: 1px 2px 3px 1px;
    border-color: #4A4A4A #842210 #B12E16 #000000;
}
```

这里提出一点，Less中的Operations主要是针对任何数字、颜色、变量的操作，可以对其是行加、减、、乘、除或者更复杂的综合运算；而Functions主要是针对Color funtions，Less提供了多种变换颜色的功能，下面多们来俱体看一下这两个功能的使用。

先来看Operation的使用

Less Code:

```
@base: 5%;
@filler: @base*2;
@other: @base + @filler;
#header {
    color: #888 / 4;
    height: 100% / 2 + @filler;
}
```

Compiled Css Code:

```
#header {
    color: #222222;
    height: 60%;
}
```

上面是一些简单的四则运算，他们都是在同一单位下进行操作，现在我们一起来看一个不同单位的操作

Less Code:

```
@var: 1px + 5;
#header {
    border: @var solid red;
}
```

Compiled Css Code:

```
#header {
    border: 6px solid red;
}
```

上面的代码直接反应出了，“@var: 1px + 5” ，Less最终解析的值是“6px” 。在Less中我们同样可以像做小学算术一样，使用括号 “ ( ) ” 来改变其运算的先后顺序，如：

Less Code:

```
@var: 20px;
#header {
    width: @var + 5 * 2;
    height: (@var + 5 ) * 2;
}
```

Compiled Css Code:

```
#header {
    height: 50px;
    width: 30px;
}
```

从结果中我们很明显的得出他们的区别

```
@var: 20px;
#header {
    width: @var + 5 * 2;/* 先计算了5 * 2 = 10 然后在计算了 @var + 10 = 30px,其实就是"@var+(5*2)"*/
    height: (@var + 5 ) * 2;/*先计算了(@var + 5) = 25px,然后在计算了25*2=50px，因为括号更具有优先权，小学数学题*/
}
```

Less中还提供了一个Color Functions，他具有多种变换颜色的功能，先把颜色转换成HSL色，然后在此基础上进行操作，具体包括以下几种：

```
lighten(@color, 10%);     // return a color which is 10% *lighter* than @color
darken(@color, 10%);      // return a color which is 10% *darker* than @color

saturate(@color, 10%);    // return a color 10% *more* saturated than @color
desaturate(@color, 10%);  // return a color 10% *less* saturated than @color

fadein(@color, 10%);      // return a color 10% *less* transparent than @color
fadeout(@color, 10%);     // return a color 10% *more* transparent than @color

spin(@color, 10);         // return a color with a 10 degree larger in hue than @color
spin(@color, -10);        // return a color with a 10 degree smaller hue than @color
```

使用这种functions方法很简单：

Less Code:

```
@base: #f04615;
#header {
    color: @base;
    background-color: fadein(@base, 10%);
    h1 {
        color: lighten(@base,20%);
        background-color: lighten(fadeout(@base,20%),5%);
        a {
            color: darken(@base,50%);
            background-color: spin(@base,10);
            &:hover {
                color: saturate(@base,30%);
                background-color: fadein(spin(@base,-5),20%);
            }
        }
    }
    p {
        color: desaturate(@base,60%);
    }
}
```

Compiled Css Code:

```
#header {
    background-color: #F04615;
    color: #F04615;
}
#header h1 {
    background-color: rgba(242, 89, 45, 0.8);
    color: #F69275;
}
#header h1 a {
    background-color: #F06B15;
    color: #060200;
}
#header h1 a:hover {
    background-color: #F03415;
    color: #FF3E06;
}
#header p {
    color: #A56F60;
}
```

大家还可以通过这样的方式提取颜色值

```
hue(@color);        // returns the `hue` channel of @color
saturation(@color); // returns the `saturation` channel of @color
lightness(@color);  // returns the 'lightness' channel of @color
```

下面我们来看一下如何取得他的颜色

Less Code:

```
@color: #f36;
#header {
    background-color: hsl(hue(@color),45%,90%);
}
```

Compiled Css Code:

```
#header {
    background-color: #F1DAE0;
}
```

5、命名空间——Namespaces

有时候你想把一些变量或mixins组织起来，并将他封装，想用的时候就把要关的一部分取出来，那么我们将在前面的mixins基础上将其功能扩展，比如说我们有一个这样的库：

```
#bundle {
  .button () {
    display: block;
    border: 1px solid black;
    background-color: grey;
    &:hover { background-color: white }
  }
  .tab { ... }
  .citation { ... }
}
```

现在在实际操作中，我们header中的a样式和.button一样，那么我们就可以这样操作：

```
#header a {
  color: orange;
  #bundle > .button;
}
```

换过一种思维来说，如果页面上有几个部分的样是完全一样的，或者只是部分不同，我们就可以这样来写，就如上面的代码，#bundle可是以web页面中已存在的元素，然后#header中的a元素和#bundle中的.button样式是一样的，那么我们就可以把

#bundle中 .button的所有样式引用到#header中的a元素上。

6、变量范围——Scope

Less中的变量和别的程序语言一样，他的变量也有一个范围概念，这个概念就有点像局部变量和全局变量一样，只是在Less中采取的是就近原则，换句话说，元素先找本身有没有这个变量存在，如果本身存在，就取本身中的变量，如果本身不存在，就寻找父元素，依此类推，直到寻找到相对应的变量，我们来看个简单的实例：

```
@var: red;

#page {
  @var: white;
  #header {
    color: @var; // white
  }
}

#footer {
  color: @var; // red
}
```

7、Less的注解——Comments

Less中的注解有两种方式，单行注解很像js中的，如：

```
// Hi, I'm a silent comment, I won't show up in your CSS
.class { color: white }
```

Less中的多行注解和使用css中的一样：

```
/* Hello, I'm a
 CSS-style comment
*/
.class { color: black }
```

当然单行注解也可以使用css的方式注解，本人更强调使用css中的注解方式：

```
/* Hello, I'm a CSS-style comment */
.class { color: black }
```

8、客户端的使用——Client-side usage

客户端的使用其实好简单，我们最开始引用的就是客户端的使用方法，使用这种方法前提条件是需要一个less.js的脚本支持，大家可以先到点击下载less.js然后把他引用到页面的head中，如下所示：

```
<script src="less.js" type="text/javascript"></script>                          }}}
其中src所指定的路径是你项目中的相对路径，当然你也可以把这个js放到你认为安全可用的服务器上，换成绝对路径也是可以的。接着我们就需要把less文件引进到项目中，这个引入的方式和css方式是一样的，只是有一点点不同，css中的是“rel="stylesheet"”而less的却是“rel="stylesheet/less"”,请看下面的代码：
{{{class="prettyprint"
<link rel="stylesheet/less" type="text/css" href="styles.less">
```

特别强调一点，客户端使用Less,一定要注意，“Less样式文件一定要放在less脚本文件之前”。

正确的引入方式：

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

错误的引入方式：

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

来源： http://www.w3cplus.com/css/less