LESS

**一.简介**

**1**.LessCSS是一种动态样式语言，属于CSS预处理语言的一种，它使用类似CSS的语法，为CSS赋予了动态语言的特性，如变量，继承，运算，函数等，更方便编写和维护

**2**.Less可以在多种语言，环境中使用，包揽浏览器端，桌面客户端，服务端。

lesscss.cn

**二.LESS的使用**

1.直接使用

（1）官网下载less.js

（2）编写less（类似于css），但style标记的**type属性必须写text/less（内部）**，link标记的**rel属性必须写为stylesheet/less**（外部，此时必须通过web形式访问）

（3）在less后链接less.js文件

2.编译成css后使用

（1）安装npm

（2）安装less

**三.LESS变量**

1.LESS中使用@定义变量，然后在需要的时候调用变量

2.变量不仅用于值，也可以用于选择器、URL、样式属性、变量嵌套等

3.变量的加载是用lazy loading模式，无需先定义再使用

4.LESS中对于变量名没有js那样的严格的规范，但是建议按照规范起名

**（1）值变量**

@color: #f40;

@width: 100px;

@height: 100px;

#three {

width: @width;

height: @height;

background-color: @color;

}

**（2）选择器变量**

@sel1: .demo1;

@{sel1} {

color: #f40;

}

**（3）URL变量**

@path :'../images';

#one {

background: url('@{path}/1.webp')no-repeat;

background-size: 100% 100%;

}

**（4）属性变量**

@fontSize: font-size;

#one {

@{fontSize}: 30px;

}

**（5）变量嵌套**

@foo :#9c9c9c;

@too: @foo;

#four {

background-color: @too;

}

**（6）延迟加载lazy loading**

#five {

width: 100px;

height: 100px;

border: 1px solid black;

background-color: @taobao;

@taobao: #3c3c3c;//最终显示

}

@taobao: #f40;

**（7）总结建议**

1.除了值变量，其他变量在调用时都要加{}

2.建议先声明变量再使用变量

3.建议将变量声明放在代码的最前面

**四.混合Mix-in**

**（一）不带参数的简单混合**

1.在LESS中我们可以定义一些通用的属性集为一个class，然后在另一个class中调用这些属性

.bordered {

border-top: dotted 1px black;

border-bottom: solid 1px red;

}

2.在其他class中引入那些通用的属性集

#menu a {

color:#111;

.bordered

}

3.混合可以将一个定义好的class轻松的引入到另一个class中，从而实现样式的复用

**（二）带参数的混合**

1.在LESS中可以定义带一个或多个带参数的属性集合，并制定缺省值，在调用可以传递参数或使用缺省值，就像使用函数一样

**（1）普通的混合，没有参数**

.wbs {

width: 100px;

height: 100px;

background-color: red;

}

#six {

border: 1px solid blue;

.wbs;

}

**（2）带参数 1个参数**

.ww(@wid) {

width: @wid;

}

#seven {

height: 100px;

background-color: burlywood;

.ww(100px);

}

**（3）带默认值**

.box-shadow(@shadow: 0 1px 3px rgba(0, 0, 0, 0.5)) {

-webkit-box-shadow: @shadow; //兼容ie

box-shadow: @shadow;

}

#eight {

width: 100px;

height: 100px;

background-color: cadetblue;

.box-shadow;

}

**（4）多个参数**

.setColumns(@columns, @column-fill, @column-gap, @column-rule) {

-webkit-columns: @columns;

-moz-columns: @columns;

-o-columns: @columns;

columns: @columns;

-webkit-column-fill: @column-fill;

-moz-column-fill: @column-fill;

-o-column-fill: @column-fill;

column-fill: @column-fill;

-webkit-column-gap: @column-gap;

-moz-column-gap: @column-gap;

-o-column-gap: @column-gap;

column-gap: @column-gap;

-webkit-column-rule: @column-rule;

-moz-column-rule: @column-rule;

-o-column-rule: @column-rule;

column-rule: @column-rule;

}

#nine {

width: 400px;

.setColumns(100px 3, balance, 5px, 1px solid red);

}

**五.导引表达式**

**类似于if else**

导引表达式使**用when进行判断**

.hello(@font-size) when(@font-size<20) {

font-size: **unit(@font-size,px);**

color: red;

}

.hello(@font-size) when(@font-size>=20) {

font-size: 30px;

color: blue;

}

.hello(@font-size) {

background-color: #ccc;

}

#ten {

.hello(13);

}

**六.运算**

任何数字、颜色或者变量都可以参与运算

@num1: 20;

@num2: @num1+10;

@num3: @num2+@num1;

@color1:#666/2;

@bgcolor:@color1+#111;

#eleven {

font-size: unit(@num3, px);

color: @color1;

border: unit(@num1/2+3,px) solid red;

background-color: @bgcolor;

}

**七.函数**

LESS内置了多种函数，用于转换颜色，处理字符串，算数运算等，函数的用法非常简单

**1.杂项函数**

**（1）color** 解析颜色，将代表颜色的字符串转换为颜色值

参数：string 代表颜色的字符串

返回值：color

案例：color('#aaa');

输出：#aaa

**<1>ligten(@color，10%)**颜色减淡10%

**<2>darken(@color,10%)**颜色加深10%

**<3>saturate(@color,10%)**颜色饱和度加10%

**<4>desaturate(@color,10%)**颜色饱和度降低10%

**<5>fadein(@color,10%)**颜色变得比原来不透明10%

**<6>fadeout(@color,10%)**颜色变得比原来透明10%

**<7>fade(@color,10%)**透明度为10%

**<8>spin(@color,10)**颜色比原来大10个色调（负数就是小10个色调）

**<9>mix(@color1,@color2)**两种颜色混合

**<10>hue(@color)**获取色相

**<11>saturation(@color)**获取饱和度

**<12>lightness(@color)**获取透明度

**（2）convert** 将数字从一个单位转化为另一个单位（单位类型一致才能转换）

**第一个参数**为带单位的数值，**第二个**是参数的单位。如果两个参数的单位是兼容的，则数字的单位被转换。如果不兼容，则原样返回第一参数

兼容的单位包括 1.长度：m、cm、mm、in、pt、pc

2.时间：s、ms

3.角度：rad、deg、grad、turn

参数：number：但单位的浮点数

identifier、string、escaped value：**units**

返回值：number

案例：convert(9s,'ms'); convert(14cm,mm); convert(8,mm)

输出：9000ms 140mm 8

**（3）unit** 改变或移出数据单位

参数：dimension 带单位或不带单位的数字

unit (可选) 目标单位，如果省略此参数，则删除单位

案例： unit(5, px)

输出： 5px

案例： unit(5em)

输出： 5

**（4）data-uri** （将图片数据转换为字节码，数据不超过32k）将资源内联进样式表，如果开启了ieCompat选项并且资源太大，或者此函数的运行环境为浏览器，则会回退到直接使用url( )。如果没有指定mime，则node将使用mime包来决定正确的mime类型

参数：mimetype （可选）mime字符串

url 需要内嵌文件的url

**（5）default**默认函数

.aaa(1) {

font-size: 1rem;

}

.aaa(@size) when(default()){

font-size: unit(@size, px);

}

.a{

.aaa(3);//3px

.aaa(1);//1rem

}

只要不是固定值1 则认为是自定义的默认值

**（6）not取反**

.mixin(1) {

font-size: 1rem;

}

.mixin(@x) when not(default()) {

font-size: unit(@x, px);

}

.c {

.mixin(2);

}

如果是固定值，则上下都能匹配到

如果不是固定值就不取

**2.字符串函数**

**（1）replace**对字符串按照指定的正则进行替换，可以增强替换特性，返回被替换后的字符串

4个参数

string: 用于搜索、替换操作的字符串

pattern: 用于搜索匹配的字符串或正则表达式

replacement: 用于替换匹配项的字符串

flags: (可选) 正则表达式参数

案例：

replace("Hello, Mars?", "Mars\?", "Earth!");

replace("One + one = 4", "one", "2", "gi");全局匹配，不区分大小写

replace('This is a string.', "(string)\.$", "new $1.");

replace(~"bar-1", '1', '2');

输出:"Hello, Earth!";"2 + 2 = 4";'This is a new string.';bar-2;

**（2）escape**对字符串中的特殊字符做 URL-encoding 编码

这些字符不会被编码：, / ? @ & + ' ~ ! $

被编码的字符是：\<space\> # ^ ( ) { } | : > < ; ] [ =

参数：string: 需要转义的字符串。

返回值：转义后不带引号的 string 字符串。

案例：

escape('a=1')

输出：a%3D1

注意：如果参数不是字符串的话，函数行为是不可预知的。目前传入颜色值的话会返回 undefined ，其它的值会原样返回。写代码时不应该依赖这个特性，而且这个特性在未来有可能改变。

**（3）length**获取值的数量（逗号或空格分隔）

@bdr：1px solid red;

@n:length(@bdr)

结果为3

**（4）extract**获取一组值中（逗号或空格分隔）指定位置的数据

@bdr：1px solid red;

@m:extract(@bdr,2); 提取第2个字段

结果为solid

**3.数学函数**

**（1）ceil**向上取整

**（2）floo**r向下取整

**（3）percentage**将浮点数转换为百分比字符串

**（4）round**四舍五入取整

**（5）sqrt**计算一个数的平方根，并原样保持单位

**（6）abs**计算数字的绝对值，并原样保持单位

**（7）sin**正弦函数

**（8）asin**反正弦函数（返回以弧度为单位的角度，区间在 -PI/2 到 PI/2之间）

**（9）cos**余弦函数

**（10）acos**反余弦函数（区间在 0 到 PI之间）

**（11）tan**正切函数

**（12）atan**反正切函数

**（13）pi**返回圆周率 π (pi)

**（14）pow**设第一个参数为A，第二个参数为B，返回A的B次方

**（15）mod**返回第一个参数对第二参数取余的结果

**（16）min**返回一系列值中最小的那个min(3px, 42px, 1px, 16px);

**（17）max**返回一系列值中最大的那个

...

**八.嵌套规则**

LESS可以让我们以嵌套的方式编写层叠样式

#header {

color: black;

>.navigation { //直接子元素

font-size: 12px;

}

.logo {

width: 300px;

:hover{

height: 300px;

}

}

}

还可以使用此方法将伪选择器（pseudo-selectors）与混合（mixins）一同使用。下面是一个经典的 clearfix 技巧，重写为一个混合（mixin） (& 表示当前选择器的父级this,可写可不写）

.clearfix {

display: block;

zoom: 1;

**&:after** {

content: " ";

display: block;

font-size: 0;

height: 0;

clear: both;

visibility: hidden;

}

}

**九.继承extend**

继承被附在选择器后面或者放置在规则集（即具体定于样式处），它看起来就像是一个关键字extend后具有可选参数的伪类

.a{

color:red;

}

.b{

&:extend(.a);//&不能省

}

.c:extend(.a){ }

结果是.a.b.c都是{color:red;}

**可以多继承 :extend(.a,.b) :extend(.a all)继承所有的.a**

**十.导入文件**

@import ‘library’如果是less文件可以省略后缀