Sass

世界上最成熟、最稳定、最强大的专业级CSS扩展语言！

15.1.3 sass语法

特别说明

因为sass的环境安装比较复杂，而在我们日后的开发过程中，是不需要这样一个环境的，所以本套课程跳过了安装步骤，那大家可能奇怪如何编译sass，这里我们可以使用其他方式编译sass，例如：考拉软件、构建工具和编辑器都可以做到。如果大家还是希望安装一个sass环境，可以参考官网

使用变量

sass让人们受益的一个重要特性就是它为css引入了变量。你可以把反复使用的css属性值 定义成变量，然后通过变量名来引用它们，而无需重复书写这一属性值。或者，对于仅使用过一次的属性值，你可以赋予其一个易懂的变量名，让人一眼就知道这个属性值的用途。

sass使用$符号来标识变量(老版本的sass使用!来标识变量。改成$是多半因为!highlight-color看起来太丑了。)，比如$highlight-color和$sidebar-width。为什么选择$ 符号呢？因为它好认、更具美感，且在CSS中并无他用，不会导致与现存或未来的css语法冲突。

```
$nav-color: #F90;
nav {
  $width: 100px;
  width: $width;
  color: $nav-color;
}

//编译后

nav {
  width: 100px;
  color: #F90;
}
```

嵌套CSS 规则

css中重复写选择器是非常恼人的。如果要写一大串指向页面中同一块的样式时，往往需要 一遍又一遍地写同一个ID：

```
#content article h1 { color: #333 }
#content article p { margin-bottom: 1.4em }
#content aside { background-color: #EEE }
```

像这种情况，sass可以让你只写一遍，且使样式可读性更高。在Sass中，你可以像俄罗斯套娃那样在规则块中嵌套规则块。

sass在输出css时会帮你把这些嵌套规则处理好，避免你的重复书写。

```
#content {
  article {
    h1 { color: #333 }
    p { margin-bottom: 1.4em }
  }
  aside { background-color: #EEE }
}
```

```
article a {
  color: blue;
  &:hover { color: red }
}

// 编译后

article a { color: blue }
article a:hover { color: red }
```

导入SASS文件

css有一个特别不常用的特性，即@import规则，它允许在一个css文件中导入其他css文件。然而，后果是只有执行到@import时，浏览器才会去下载其他css文件，这导致页面加载起来特别慢。

sass也有一个@import规则，但不同的是，sass的@import规则在生成css文件时就把相关文件导入进来。这意味着所有相关的样式被归纳到了同一个css文件中，而无需发起额外的下载请求。

```
@import "./init.scss"
```

静默注释

css中注释的作用包括帮助你组织样式、以后你看自己的代码时明白为什么这样写，以及简单的样式说明。但是，你并不希望每个浏览网站源码的人都能看到所有注释。

sass另外提供了一种不同于css标准注释格式/* ... */的注释语法，即静默注释，其内容不会出现在生成的css文件中。静默注释的语法跟JavaScriptJava等类C的语言中单行注释的语法相同，它们以//开头，注释内容直到行末。

```
body {
  color: #333; // 这种注释内容不会出现在生成的css文件中
```

```
  padding: 0; /* 这种注释内容会出现在生成的css文件中 */
}
```

混合器

如果你的整个网站中有几处小小的样式类似（例如一致的颜色和字体），那么使用变量来统一处理这种情况是非常不错的选择。但是当你的样式变得越来越复杂，你需要大段大段的重用样式的代码，独立的变量就没办法应付这种情况了。你可以通过sass的混合器实现大段样式的重用。

混合器使用@mixin标识符定义。看上去很像其他的CSS @标识符，比如说@media或者@font-face。这个标识符给一大段样式赋予一个名字，这样你就可以轻易地通过引用这个名字重用这段样式。下边的这段sass代码，定义了一个非常简单的混合器，目的是添加跨浏览器的圆角边框。

```
@mixin rounded-corners {
  -moz-border-radius: 5px;
  -webkit-border-radius: 5px;
  border-radius: 5px;
}
```

然后就可以在你的样式表中通过@include来使用这个混合器，放在你希望的任何地方。@include调用会把混合器中的所有样式提取出来放在@include被调用的地方。如果像下边这样写：

```
notice {
  background-color: green;
  border: 2px solid #00aa00;
  @include rounded-corners;
}

// 编译后

.notice {
  background-color: green;
  border: 2px solid #00aa00;
  -moz-border-radius: 5px;
  -webkit-border-radius: 5px;
  border-radius: 5px;
}
```

使用选择器继承来精简CSS

使用sass的时候，最后一个减少重复的主要特性就是选择器继承。基于Nicole Sullivan面向对象的css的理念，选择器继承是说一个选择器可以继承为另一个选择器定义的所有样式。这个通过@extend语法实现，如下代码:

```
//通过选择器继承继承样式
.error {
```

```
  border: 1px solid red;
  background-color: #fdd;
}
.seriousError {
  @extend .error;
  border-width: 3px;
}
```