CSS 强化装备！Sass 安装与使用

来源：https://segmentfault.com/a/1190000003912703

Sass 是什么？

Sass 是 *Syntactically Awesome Style Sheets* 的缩写，它是 CSS 的 一个开发工具，提供了很多便利和简单的语法，让 CSS 看起来更像是一门语言，这种特性也被称为“CSS 预编译”。它的主要涉及思想是让我们可以按照编程的思路编写自己的样式，然后通过“编译器”生成我们所需要的 CSS 文件。

官网

Sass 和 SCSS 有什么区别？


```scss
$blue: #3bbfce;
$margin: 16px;

.content-navigation {
  border-color: $blue;
  color:
    darken($blue, 9%);
}

.border {
  padding: $margin / 2;
  margin: $margin / 2;
  border-color: $blue;
}
```

Sass 和 Scss 其实是同一种东西，我们平时都称之为 Sass，两者不同之处有以下两点：

1. **文件扩展名不同**。
   Sass 是以“.sass”后缀为扩展名，而 SCSS 是以“.scss”后缀为扩展名；
2. **语法书写方式不同**。
   Sass 是以严格的缩进式语法规则来书写，不带大括号“{}”和分号“；”，而 SCSS 的语法书写和我们的 CSS 语法书写方式非常类似。

PS：本文只讨论 SCSS。

SCSS 与 Sass 异同 - Sass 中文文档

为什么选择 Sass？

“LESS 要靠 JavaScript 解析，我不喜欢这种做法。另外，LESS 的变量用 @ 表示，我也不太习惯。” by 阮一峰

SASS 用法指南 - 阮一峰

Sass、LESS 和 Stylus 简单对比

- 三者都是开源项目；
- Sass 诞生是最早也是最成熟的 CSS 预处理器，有 Ruby 社区和 Compass 支持；Stylus 是早期服务器 NodeJS 项目，在该社区得到一定支持者；Less 出现于 2009 年，支持者远超远于 Ruby 和 NodeJS 社区；
- Sass 和 LESS 语法较为严谨、严密，而 Stylus 语法相对散漫，其中 LESS 学习起来更快一些，因为他更像 CSS 的标准；
- Sass 和 LESS 相互影响较大，其中 Sass 受 LESS 影响，已经进化到了全面兼容 CSS 的 SCSS；
- Sass 和 LESS 都有第三方工具提供转译，特别是 Sass 和 Compass 是绝配；
- Sass、LESS 和 Style 都具有变量、作用域、混合、嵌套、继承、运算符、颜色函数、导入和注释等基本特性，而且以“变量”、“混合”、“嵌套”、“继承”和“颜色函数”成为五大基本特性，各自特性实现功能基本形似，只是使用规则上有所不同；
- Sass 和 Stylus 具有类似于语言处理的能力，比如说条件语句、循环语句，而 LESS 需要通过 When 等关键词模拟这些功能，在这一方面略逊一筹。

CSS 预处理的缺点

- 个人感觉 CSS 预处理器语言那是程序员的玩具，想通过编程的方式跨界解决 CSS 的问题。可以说 CSS 应该面临的问题一个也少不了，只是增加了一个编译过程而已，简单来说 CSS 预处理器语言较 CSS 玩法变得更高级了，但同时降低了自己对最终代码的控制力。更致命的是提高了门槛，首先是上手门槛，其次是维护门槛，再来是团队整体水平和规范的门槛。这也造成了初学学习成本的昂贵。
  CSS 预处理器——Sass、LESS 和 Stylus 实践【未删减版】 - W3CPlus - 2013-03-13

我选择 Sass 的原因：

1. Sass 也是成熟的 CSS 预处理器之一，而且又有一个稳定，强大的团队在维护；
2. Sass 对于我来说参考的教程多；
3. Sass 有一些成熟稳定的框架，特别是 Compass，新秀还有 Foundation 之类；
4. 还有一个原因是国外讨论 Sass 的同行要多于 LESS。

出于这几个原因，我想我学习或者使用 Sass 更容易一些，碰到问题更有参考资料，更有朋友帮忙解决。

该使用 SASS 还是 LESS？ - 大漠 - 知乎 - 2014-07-04

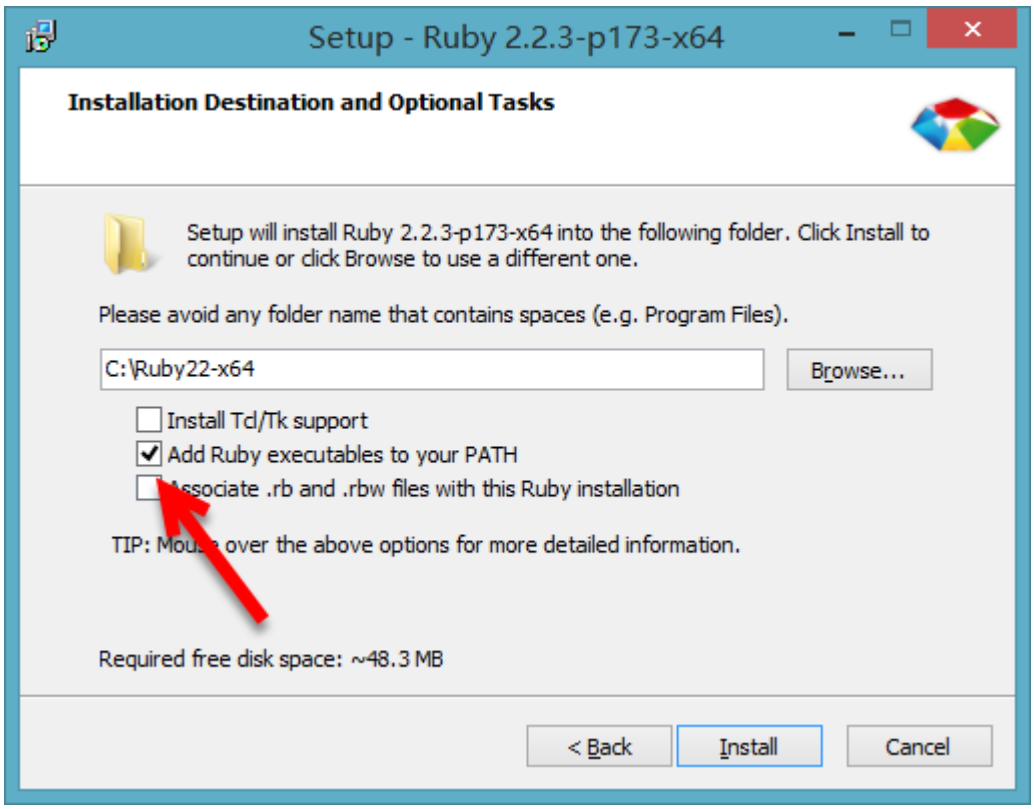
安装

Sass 是 Ruby 语言写的，但是两者的语法没有关系。不懂 Ruby，照样使用。不过必须要先安装 Ruby，然后再安装 Sass。

ruby 安装

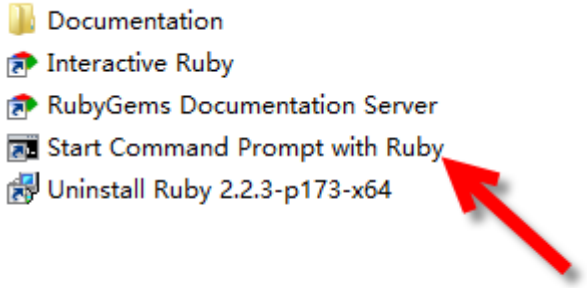
先从官网下载并安装 ruby，在安装的时候，请勾选 Add Ruby executables to your PATH 这个选项，添加环境变量，不然以后使用编译软件的时候会提示找不到 ruby 环境。

Downloads - 官网



sass 安装

安装完 ruby 之后，在开始菜单，找到刚才安装的 ruby，打开 `Start Command Prompt with Ruby`



然后直接在命令行输入：

```
gem install sass
```

按回车键确认，等待一段时间就会提示你 Sass 安装成功（如果因为墙的原因安装失败，请参考下面用淘宝 RubyGems 镜像安装 Sass）。

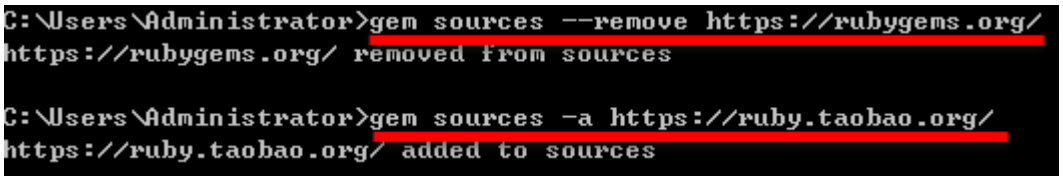
淘宝 RubyGems 镜像安装 Sass

由于国内网络原因（你懂的），导致 rubygems.org 存放在 Amazon S3 上面的资源文件间歇性连接失败。
这时候我们可以通过 `source` 命令来配置源，先移除默认的 `https://rubygems.org` 源：

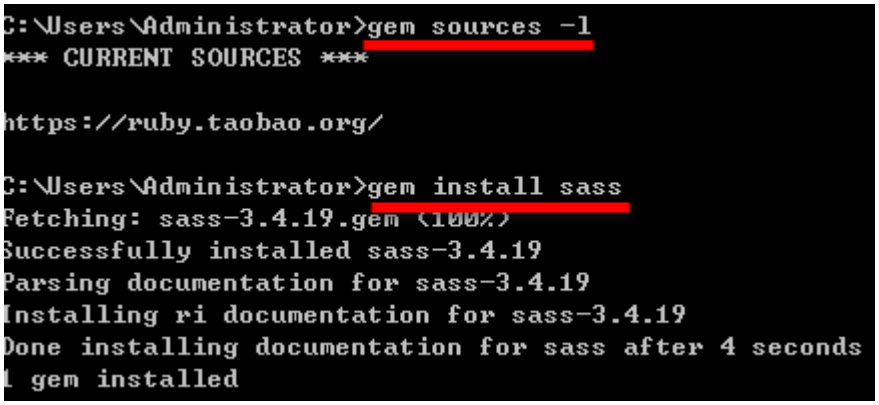
```
gem source --remove https://rubygems.org/
```

然后添加淘宝的源 `https://ruby.taobao.org/`:

```
gem source -a https://ruby.taobao.org/
```



然后查看一下当前使用的源是哪个，如果是淘宝的，则比表示可以输入 Sass 安装命令 `gem install sass` 了。



最后使用版本查看命令确保安装成功：

```
sass -v
```

```
C:\Users\Administrator>sass -v
Sass 3.4.19 (Selective Steve)
```

Sass 安装 - W3CPlus

编译

SASS 文件就是普通的文本文件，里面可以直接使用 CSS 语法。文件后缀名是 .scss，意思为 Sassy CSS。

新建一个 test.scss 文件，内容为：

```
$blue : #1875e7;

div{

    color: $blue;

}
```

然后在命令行输入下面命令，屏幕上便显示 .scss 文件转化的 css 代码：

```
sass test.scss
```

```
E:\sass-demo>sass test.scss
div {
  color: #1875e7; }

E:\sass-demo>
```

如果要将显示结果保存成文件，后面再跟一个 .css 文件名：

```
sass test.scss test.css
```

那么就会默认在当前目录下生成文件。

Sass 提供四个编译风格的选项：

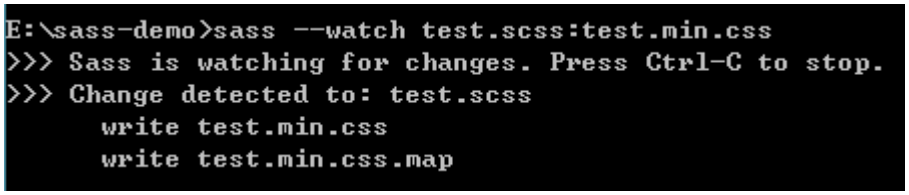
- `nested`: 嵌套缩进的 css 代码，它是默认值；
- `expanded`: 没有缩进的、扩展的 css 代码；
- `compact`: 简洁格式的 css 代码；
- `compressed`：压缩后的 css 代码。

生产环境当中，一般使用最后一个选项。

```
sass --style compressed test.scss test.min.css
```

你还可以让 Sass 监听某个文件或目录，一旦源文件有变动，就自动生成编译后的版本。

```
// 监听文件

// input.scss : scss 文件

// output.css : 编译后的 css 文件

sass --watch input.scss:output.css



// 监听目录

// sassFileDirectory : sass 文件目录

// cssFileDirectory : 编译后的 css 文件目录

sass --watch sassFileDirectory:cssFileDirectory
```



Sass 编译 - W3CPlus

Sass 的官方网站还提供一个在线转换器，方便尝试运行各种栗子：

SassMeister | The Sass Playground !

基本语法

1.变量

Sass 中可以定义变量，方便统一修改和维护。

```
//sass style

//-----------------------------------

$gray: #666;

body {

  background-color: $gray;

}



//css style

//-----------------------------------

body {

  background-color: #666;

}
```

2.嵌套

Sass 可以进行选择器的嵌套，表示层级关系。

```
//sass style
//----------------------------------
ul {
    li {
        display: inline-block;
    }
}


//css style
//----------------------------------
ul li {
  display: inline-block;
}
```

3.导入

Sass 中如导入其它 sass 文件，最后编译为一个 css 文件，优于纯 css 的`@import`。

```
//sass style
//----------------------------------
// reset.scss
html,
body,
ul,
```

```
ol {
    margin: 0;
    padding: 0;
}

//sass style
//-----------------------------------
// test.scss
@import 'reset';
body {
    font-size: 100%;
    background-color: #efefef;
}

//css style
//-----------------------------------
html,
body,
ul,
ol {
  margin: 0;
  padding: 0;
}
body {
```

```
  font-size: 100%;

  background-color: #efefef;

}
```

4.mixin

Sass 中可用 mixin 定义一些代码片段，且可传参数，方便日后根据需求调用。从此处理 CSS 3 的前缀兼容轻松便捷。

```
//sass style

//----------------------------------

@mixin box-sizing ($sizing) {

    -webkit-box-sizing: $sizing;

       -moz-box-sizing: $sizing;

           -box-sizing: $sizing;

}

.box-border {

    border: 1px solid #ccc;

    @include box-sizing(border-box);

}


//css style

//----------------------------------

.box-border {

    border: 1px solid #ccc;

    -webkit-box-sizing: border-box;

       -moz-box-sizing: border-box;
```

```
          -box-sizing: border-box;
}
```

5.扩展/继承

Sass 可通过 `@extend` 来实现代码组合声明，使代码更加优越简洁。

```
//sass style
//----------------------------------
.bar-left {
    border: 1px solid #ccc;
}
.bar-right {
    @extend .bar-left;
    color: #999;
}


//css style
//----------------------------------
.bar-left, .bar-right {
    border: 1px solid #ccc;
}
.bar-right {
    color: #999;
}
```

6.运算

Sass 可进行简单的加减乘除运算等。

```
//sass style

//-----------------------------------

$defaultFontSize: 10px;

.msg {

    position: absolute;

    top: (800px/2);

    left: 200px + 200px;

    font-size: $defaultFontSize * 2;

}



//css style

//-----------------------------------

.msg {

    position: absolute;

    top: 400px;

    left: 400px;

    font-size: 20px;

}
```

7.颜色

Sass 中集成了大量的颜色函数，让变换颜色更加简单。

```
//sass style
```

```
//----------------------------------
$linkColor: #08c;
a {
    text-decoration: none;
    color: $linkColor;
    &:hover {
        color: darken($linkColor, 10%);
    }
}

//css style
//----------------------------------
a {
    text-decoration: none;
    color: #08c;
}
a:hover {
    color: #006699;
}
```

8.注释

Sass 共有两种注释风格。

- 标准的 CSS 注释 `/* comment */`，会保留到编译后的文件
- 单行注释 `// comment`，只保留在 SASS 源文件中，编译后会被省略。
  提示：在`/*`后面加一个感叹号，表示这是“重要注释”。即使是压缩模式编译，也会保留这行注释。**通常可以用于声明版权**。

```
/*!

    重要注释！

*/
```

管理 Sass 项目文件结构

CSS 预处理器的特点之一是可以把你的代码分割成很多文件，而且不会影响性能。这都归功于 Sass 的`@import`命令，只要在你的开发环境下，你调用不管多少文件，最终将编译出一个 CSS 样式文件。

管理 Sass 项目文件结构 - 大漠 - W3CPlus

gulp-ruby-sass 与 gulp-sass

- `gulp-ruby-sass`是调用 sass，所以需要 ruby 环境，需要生成临时目录和临时文件；
- `gulp-sass`是调用 node-sass，有 node.js 环境就够了，编译过程不需要临时目录和文件，直接通过 buffer 内容转换。
  gulp-ruby-sass 与 gulp-sass - SegmentFault

More

sass 入门 - sass 教程
sass|博客自由标签|W3CPlus
Sass->十分钟写一个 Sass 组件 - SegmentFault
使用 Sass 来写 OOCSS - SegmentFault

以上，欢迎拍砖斧正。