

## 一. 安装说明

在安装之前先检查是否已经安装成功，在 cmd 中输入 `sass -v` 或 `sass --version` 有版本号就不需要以下操作了

- 1.在 cmd 中输入 `ruby -v`，查看是否有版本号
- 2.如果有版本号，如 `ruby 2.5.1` 等，就是用 `gem install sass` 安装。如果是 mac 系统一般情况下自带 ruby，使用 `sudo gem install sass`，安装过程因为版本不同可能需要选择，有的话可以选 1
- 3.有 ruby 的情况下，安装完成在 cmd 中 `sass -v` 出现版本号就是安装成功
4. 如没有 ruby 版本号，则使用 npm 安装。先检查是否有 `npm -v`
5. 使用 npm 安装在 cmd 中 `npm install -g sass` 安装过程因为版本不同可能需要选择，有的话可以选 1
- 6.如果使用 npm 安装，在 cmd 中输入 `sass --version` 检查是否有版本号，有就是安装成功

## 二. sass 介绍

sass 它是一种“预编译”语言，或者叫做“预处理器”，不是直接的 css，它是一门专门的 css 编程语言。他增加了函数，变量，嵌套关系等等内容，可以让 css 编写的更加清晰，快捷，但也存在问题。

预编译语言不止 sass 一种，还有 less，stylus。这种预编译语言，不能直接当做 css 放到 html 中，需要进行一步“编译”变成 css 后才能使用。

## 5.编译过程

编译命令，注意结构目录

### 新建其他文件



scss文件夹要和css编译好的文件夹同级

```
sass scss目录/文件.scss:css目录/文件.css
```

监听整个文件夹进行整体编译,直接监听目录

```
sass -w scss目录:css目录
```

## scss 基本规则

### 1. 嵌套规则

```
2.scss 1.scss 2.css index.html
按照标签的嵌套关系书写
1 @charset "utf-8";
2 div {
3   h2 {
4     color: red;
5   }
6   ol {
7     li:nth-child(1){
8       font-weight: 900;
9     }
10  }
11 }

8 <body>
9   <div>
10     <h2>这是老师排名 红字</h2>
11     <ol>
12       <li>我 加粗</li>
13       <li>明明</li>
14       <li>涛桑</li>
15       <li>亮亮</li>
16     </ol>
17   </div>
18 </body>
19 </html>
```

scss文件可以按照html的包裹和嵌套规则书写，直接在{}写内部都是选择器，之后会编译成css文件，css会自动生成编译后合理的关系型选择器。重要体现是权重值的体现。会按照结构生成后代或子代选择器。

## 2. 基本语法

### 1.1 计算功能{不是重点}

### 1.2 变量

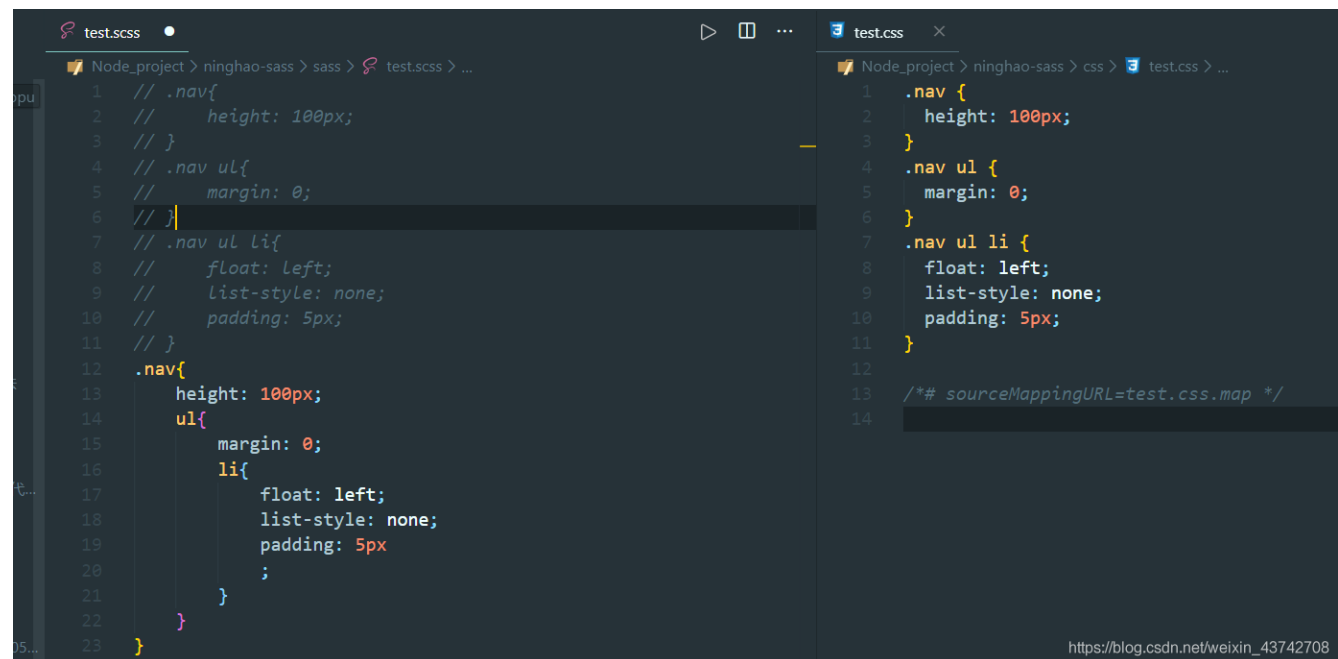
- 使用 \$符号+名称 可以定义变量

变量里可以使用另外一个变量

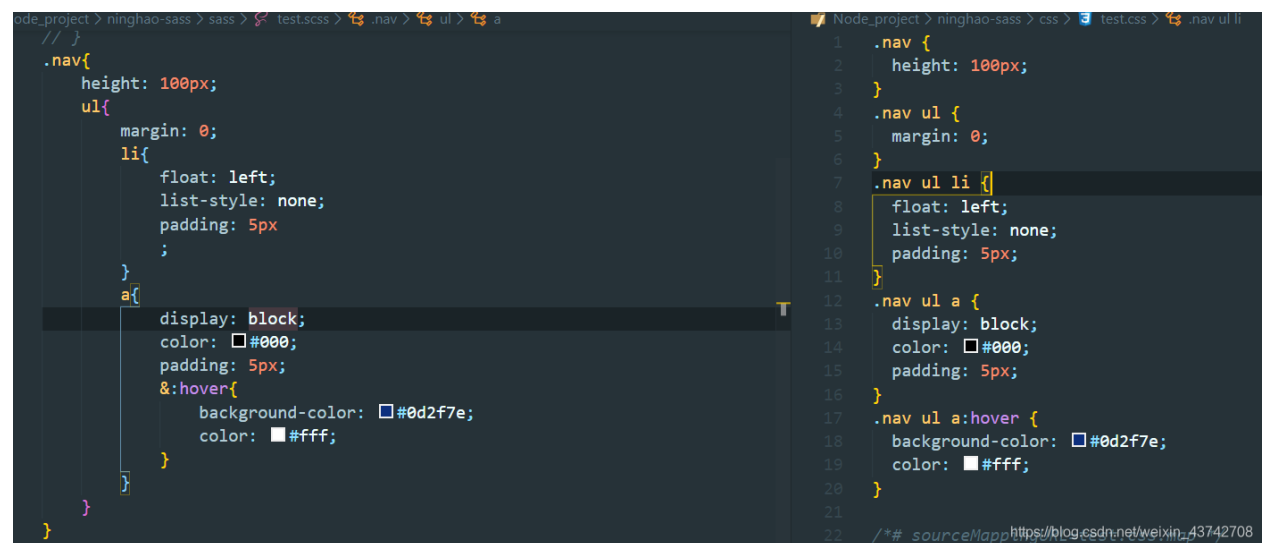
```
style.scss X style.css X
Node_project > ninghao-sass > sass > style.scss > h1.page-header
1 $primary-color: #1269b5;
2 $primary-border: 1px solid $primary-color;
3
4 div.box{
5   background-color: $primary-color;
6 }
7 h1.page-header{
8   border: $primary-border;
9 }
10

Node_project > ninghao-sass > sass > style.scss > div.box
1 div.box {
2   background-color: #1269b5;
3 }
4
5 h1.page-header {
6   border: 1px solid #1269b5;
7 }
8
9 /*# sourceMappingURL=style.css.map */
10 https://blog.csdn.net/weixin_43742708
```

## 1.3 嵌套语法



伪类选择器则需要&



## 混合指令 mixin

### 1. 定义混合指令

```
@mixin 名字(参数 1, 参数 2){  
  ...  
}
```

```
//定义混合指令
@mixin fk100 {
  width: 100px;
  height: 100px;
}

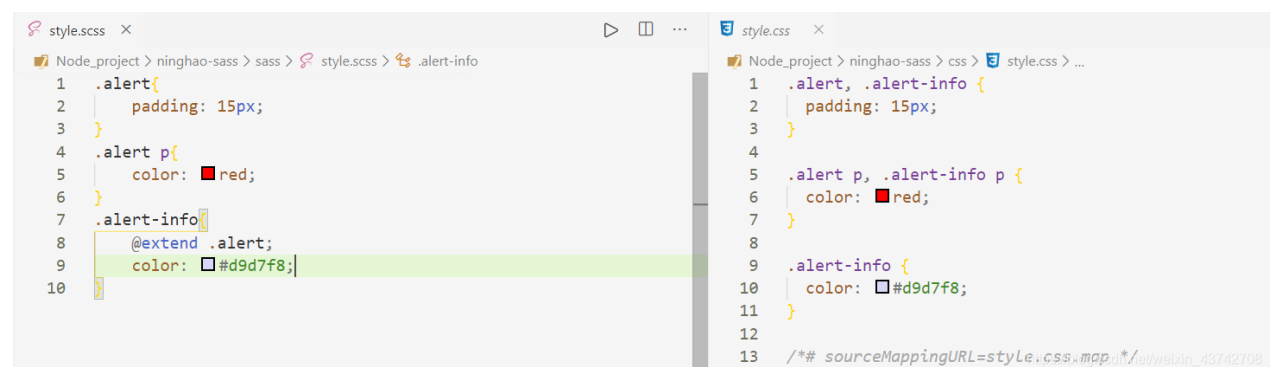
.box1 {
  background-color: red;
  // 引用混合指令
  @include fk100;
}
```

## 第二种传参



## 继承指令

在 Sass 中我们可以使用 `@extend` 来减少重复的动作



## 占位符选择器%

占位符选择器可以被继承，但不会被编译

```

1 .box4 {
2   // 继承
3   @extend .box3;
4   font-size: 14px;
5 }
6
7 .box5 {
8   @extend .box4,%boxcolor;
9 }
10
11 %boxcolor {
12   color: red;
13 }
14
15
16
17
18
19 height: 40px; }
20
21
22
23
24
25
26
27
28
29
30
31 .box4, .box5 {
32   font-size: 14px; }
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

占位符选择器，会被继承但不会被编译

```

1 .box5 {
2   @extend .box4,%boxcolor;
3 }
4
5 %boxcolor {
6   color: red;
7 }
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

## 高级语句

### @if 判断

```

1 style.scss
2
3 Node_project > ninghao-sass > sass > style.scss > $use-prefixes
4
5 1 $use-prefixes: 'true';
6
7 2
8
9 3 body{
10
11 4   @if $use-prefixes == 'true' {
12
13 5     background-color: green;
14
15 6   }@else if $use-prefixes == 'false'{
16
17 7     background-color: red;
18
19 8   }@else{
20
21 9     background-color: black;
22
23 10
24 11 }
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

```

1 style.css
2
3 Node_project > ninghao-sass > sass > style.scss > $use-prefixes
4
5 1 body {
6
7 2   background-color: green;
8
9 3 }
10
11 4
12
13 5 /*# sourceMappingURL=style.css.map */
14
15 6
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

### @for 循环

在 Sass 中，我们可以使用“@for”来实现循环操作。其中，Sass 中的 @for 循环有 2 种方式。

#### 语法：

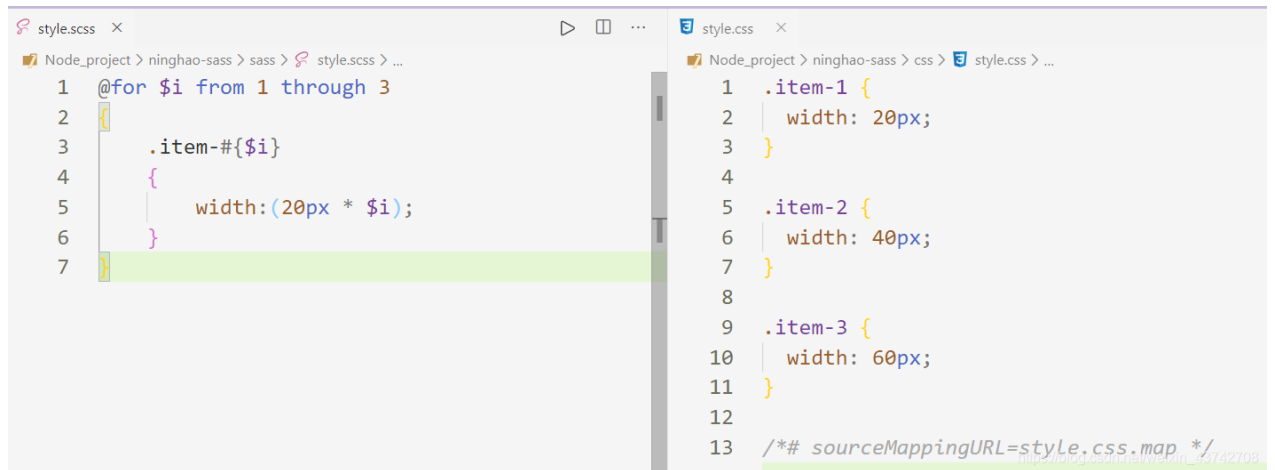
方式 1: @for \$i from 开始值 through 结束值

方式 2: @for \$i from 开始值 to 结束值

说明：

这 2 种方式是相似的，唯一的区别是：方式 1 包括结束值，方式 2 不包括结束值。

其中“开始值”和“结束值”都是正整数。



```
style.scss
Node_project > ninghao-sass > sass > style.scss > ...
1 @for $i from 1 through 3
2 {
3   .item-#{ $i }
4   {
5     width:(20px * $i);
6   }
7 }
```

```
style.css
Node_project > ninghao-sass > css > style.css > ...
1 .item-1 {
2   width: 20px;
3 }
4
5 .item-2 {
6   width: 40px;
7 }
8
9 .item-3 {
10  width: 60px;
11 }
12
13 /*# sourceMappingURL=style.css.map */
```