

Sass Basics

작성자 : 배영

소속팀 / 상위부서 : UI개발1팀 / UIT개발실

대외비

1. Sass 시작하기

1.1 Sass 소개

- Sass = CSS의 확장판
- 일반 CSS 파일로 컴파일 되는 CSS 확장 언어 = transcompiler 언어

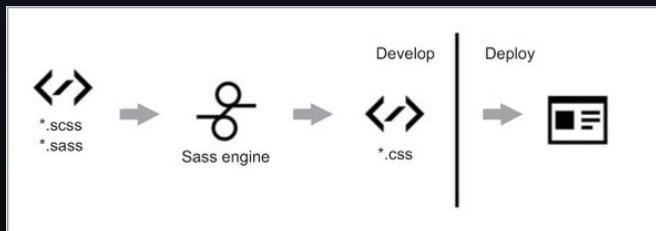
1. transcompiler란?

특정 프로그래밍 언어로 쓰여진 소스 코드를 입력 받아서 이와 다른 프로그래밍 언어를 사용하여

입력된 코드와 등가적인 소스 코드를 만들어내는 컴파일러 (https://en.wikipedia.org/wiki/Source-to-source_compiler)

2. 왜 Sass를 컴파일해야 하나? 브라우저는 CSS 밖에 몰라요!

3. 다른 transcompiler 언어 예: CoffeeScript, LESS



- Sass는 약어가 아니라서 다 대문자로 표기하면 안됨: SASS (x)
- http://sass-lang.com/documentation/file.SASS_REFERENCE.html#features

1.2 Sass 설치하기

- OS: 맥, 윈도우, 리눅스에서 사용 가능
- Windows: Ruby가 먼저 설치되어 있어야 함
 1. Sass는 Ruby로 쓰여짐
 2. 커맨드창에서 `gem install sass` 입력하여 Sass 설치
 3. 설치 후 버전 확인: 커맨드창에 `sass --version` (혹은 `-v`) 입력
- ```
NAVER@AD00030654 MINGW64 ~/Downloads/treehouse-sass (master)
$ sass --version
Sass 3.4.21 (Selective Steve)

NAVER@AD00030654 MINGW64 ~/Downloads/treehouse-sass (master)
$ sass -v
Sass 3.4.21 (Selective Steve)
```
- 더 자세한 설치 관련 정보: <http://sass-lang.com/install>

## 1.3 Sass 테스트

- Sass 설치가 제대로 되었는지 간단 테스트

### 1. 아무 경로에 test.scss 이름으로 파일 생성

```
test.scss

1 √ p {
2 color: blue;
3 }
```

### 2. 커맨드창에 sass test.scss 입력

```
NAVER@AD00030654 MINGW64 ~/Desktop (master)
$ sass test.scss
p {
 color: blue; }
```

### 3. 파일에 입력한 내용이 커맨드창에 제대로 출력되는지 확인

- 커맨드 창 사용이 불편하다면?
  - Scout app (<http://mhs.github.io/scout-app/>)

- 변경사항 자동탐지: sass --watch . (.은 현재 위치를 나타냄)

```
NAVER@AD00030654 MINGW64 ~/Desktop (master)
$ sass --watch .
>>> Sass is watching for changes. Press Ctrl-C to stop.
write ./css-to-sass/1-installing-sass-and-setting-up-
nnind-with-sass/lake-tahoe final/scss/style.css
```

- 빠져나오려면 Ctrl + C
- test.scss의 내용 변경 후 파일 저장하면:

```
write ./css-to-sass/1-installing-sass-and-setting-up-
>>> Change detected to: test.scss
write ./test.css
write ./test.css.map
```

- Sass 버전은 2가지가 있음
  - Lib Sass: 루비 뿐만 아니라 다른 언어들과도 호환성이 있음  
<http://sass-lang.com/libsass>
  - Sass 오리지널 버전
- 기본적인 Sass 문법은 동일

## 1.4 셀렉터 중첩시키기(Nesting Selectors)

```
<!DOCTYPE html>
<html>
 <head>
 <link href="main.css" rel="stylesheet"/>
 <script src="/modernizr-custom.js"
 type="text/javascript"></script>
 </head>
 <body class="blog">
 <h1>My Blog</h1>
 <div class="entry">
 <h1>My Blog Post</h1>
 <p>
 Text
 Link
 </p>
 </div>
 </body>
</html>
```

```
/* .blog .entry h1 {
 font-size: 20px;
 color: blue;
}
/* .blog .entry p {
 font-size: 12px;
}
```

- 중복되는 코드들이 많음 (e.g. 클래스 선택자)
- DRY: Don't Repeat Yourself
  - code small: 같은 부분을 복사 & 붙여넣기 하고 있을 때는 작업이 비효율적으로 되어가고 있다는 것.
- 따라서, 셀렉터를 중첩시켜서 반복적인 코드가 생기는 것을 방지한다.

```
4
5 ✓ .blog {
6 ✓ .entry {
7 ✓ h1 {
8 font-size: 20px;
9 color: blue;
10 }
11
12 ✓ p {
13 font-size: 12px;
14 }
15 }
16
```

## 1.4 셀렉터 중첩시키기(Nesting Selectors)

---

- 주의할 점: 중첩의 깊이가 너무 깊어지면 안 된다!
- 4~5레벨 보다 더 중첩이 발생하면:
  1. 마크업에서 class 혹은 id 부여를 더 해야 하거나
  2. 스타일시트가 알아보기 쉬운, 명확한 방식으로 작성되지 않고 있다는 신호
- 참고 문서: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#nested\\_rules](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#nested_rules)



## 1.5 셀렉터 중첩시키기 조금 더 알아보기

- `&`: Sass 코드 블록 상의 부모 요소 셀렉터 (마크업 태그 부모가 아니라)
- 셀렉터가 오는 순서를 바꿀 수도 있음
- `>`: html 태그 상의 직계 자손 셀렉터



- 코드의 가독성이 높아짐
- 컴파일된 CSS 파일에서도 Sass 코드 블록처럼 들여쓰기가 된 것을 확인할 수 있음
- 참고: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#parent-selector](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#parent-selector)

2.

---

변수 선언, 믹스인(mixin), @extend

## 2.1 변수 선언

- [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#variables\\_](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#variables_)
- 변수를 사용해야 하는 예
  - 같은 색상 코드가 여러 곳에서 반복되어 나타날 때: DRY 법칙을 어기는 것!
- Sass에서 변수의 역할: 어떤 특정한 값을 가지고 있는 placeholder
- 표기법
  1. 달러 기호(\$)가 맨 앞에 와야 함
  2. 하이픈(-)과 밑줄(\_)는 같은 기호로 취급됨 (e.g. \$primary-color = \$primary\_color)

```
1 $primary-color: #369;
2 $margin: 20px;
3 $padding: $margin * 1.5;
4
```

## 2.1 변수 선언

- 더하기, 빼기, 곱하기, 나누기 가능!
- 변수가 아니라 직접 단위를 입력해도 (e.g. 5px, 5em 등등) 계산 가능

```
1 $primary-color: #369;
2 $margin: 20px;
3 $padding: $margin * 1.5;
4
5 √ body {
6 background: $primary-color;
7 font-family: Futura, "Trebuchet MS", sans-serif;
8 √ > h1 {
9 font-size: 3em;
10 color: #eee;
11 margin: $margin 0 0 $margin;
12 }
13 }
14
15 √ .entry {
16 background: white;
17 margin: $margin;
18 padding: $padding;
19 √ h1 {
20 font-size: 1.2em;
21 margin: 0px;
22 }
23 √ .date {
24 font-size: 0.3em;
25 color: $primary-color;
26 }
27 √ a {
28 color: $primary-color;
29 }
30 }
```

```
1 √ body {
2 background: #369;
3 font-family: Futura, "Trebuchet MS", sans-serif; }
4 √ body > h1 {
5 font-size: 3em;
6 color: #eee;
7 margin: 20px 0 0 20px; }
8
9 √ .entry {
10 background: white;
11 margin: 20px;
12 padding: 30px; }
13 √ .entry h1 {
14 font-size: 1.2em;
15 margin: 0px; }
16 √ .entry .date {
17 font-size: 0.3em;
18 color: #369; }
19 .entry a {
20 color: #369; }
21
22 /** sourceMappingURL=main.css.map */
```

## 2.1 변수 선언

- Sass가 지원하는 데이터 타입
  1. 숫자 (e.g. 1.2, 13, 10px)
  2. 따옴표를 포함하거나 제외한 스트링 문자열 (e.g. "foo", 'bar', baz)
  3. 색깔 (e.g. blue, #04a3f9, rgba(255, 0, 0, .5))
  4. 불리언 값 (e.g. true, false)
  5. 널 값 (e.g., null)
  6. 공백 혹은 콤마로 값을 구분한 배열 (e.g. 1.5em 1em 0 2em / Helvetica, Arial, sans-serif)
  7. 키-값을 가지고 있는 맵(map) (e.g. (key1: value1, key2: value2))
  8. 그 외 CSS 속성 값 (e.g. 유니코드 지원범위, !important 선언): 따옴표 없는 스트링 문자열 값으로 취급

## 2.2 믹스인(Mixins)

- 이런 식으로 코드 작성이 계속되면 DRY 원칙 위배

```
1 √ .box {
2 border-radius: 4px;
3 border-top-right-radius: 8px;
4 border-bottom-left-radius: 8px;
5 }
6
7 √ .button {
8 border-radius: 4px;
9 border-top-right-radius: 8px;
10 border-bottom-left-radius: 8px;
11 background: #345;
12 }
```



```
1 √ @mixin roundy($radius) {
2 border-radius: $radius;
3 border-top-right-radius: $radius * 2;
4 border-bottom-left-radius: $radius * 2;
5 }
6
7 √ .box {
8 @include roundy(4px);
9 }
10
11
12 √ .button {
13 @include roundy(2px);
14 background: #345;
15 }
```

- Mixin 정의: 재사용 가능한 코드를 만들자

### 1. @mixin: mixin 정의하는 directive

- 인자는 옵션으로 줄 수 있음
- 코드 블록에 mixin 콘텐츠가 담김
- mixin 이름 역시 하이픈(-)과 밑줄(\_)이 같은 기호로 여겨짐 (e.g. add-column = add\_column)

### 2. @include: 정의된 mixin을 사용하는 directive

- Mixin 호출: 함수 호출과 동일한 방식. 믹스인 이름 뒤에 인자를 써서 넘겨준다.



- ```

1  V @mixin roundy($radius, $color) {
2      border-radius: $radius;
3      border-top-right-radius: $radius * 2;
4      border-bottom-left-radius: $radius * 2;
5      a {
6          color: $color;
7      }
8  }

```

```
1 V .box {
2     border-radius: 4px;
3     border-top-right-radius: 8px;
4     border-bottom-left-radius: 8px; }
5 .box a {
6     color: blue; }
7
8 V .button {
9     border-radius: 2px;
10    border-top-right-radius: 4px;
11    border-bottom-left-radius: 4px;
12    background: #345; }
13 .button a {
14     color: red; }
15
```

- ```

7 @mixin green_links {
8 a {
9 color: green;
10 &:hover {
11 color: blue;
12 }
13 }
14 }
15
16 .box {
17 @include roundy(4px);
18 @include green_links
19 }

```

```
1 V .box {
2 border-radius: 4px;
3 border-top-right-radius: 8px;
4 border-bottom-left-radius: 8px; }
5 .box a {
6 color: green; }
7 .box a:hover {
8 color: blue; }
9
```

## 2.2 믹스인(Mixins)

- Mixin 작성하는 것
  - = 팀 협력이 좀 더 원활 하게 이루어질 수 있도록 도와주는 도구를 제작하는 것
- Mixin은 언제 작성하면 좋을까?
  1. 비슷한 패턴이 코드에서 2~3번 반복 시: mixin 후보
  2. 6번 이상: 전역 mixin 후보
- Mixin 장점
  1. 전달해야 할 인자가 있을 때 사용하기 좋음
  2. 정교한 로직 구현에 사용하기 좋음
- Mixin 더 알아보기
  - [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#defining\\_a\\_mixin](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#defining_a_mixin)
  - [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#mixin-arguments](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#mixin-arguments)

## 2.3 @extend

- @extend directive
  - 클래스나 아이디의 속성값을 다른 셀렉터로 확장할 수 있게 해줌
- 반복되는 코드가 생길 때 해결방법은?

```
1 √ .bar {
2 height: 14px;
3 font-size: 10px;
4 √ > div {
5 float: left;
6 clear: none;
7 }
8 }
9
10 √ .menu {
11 height: 14px;
12 font-size: 10px;
13 √ > div {
14 float: left;
15 clear: none;
16 }
17 }
```

- Mixin 사용
- @extend 사용

```
9
10 √ .menu {
11 @extend .bar;
12 }
13
```



컴파일된 CSS 코드도 더 깔끔하게 나온다

```
1 √ .bar, .menu {
2 height: 14px;
3 font-size: 10px; }
4 √ .bar > div, .menu > div {
5 float: left;
6 clear: none; }
7
```

## 2.3 @extend

- .bar 클래스는 CSS 코드에 포함시키고 싶지 않을 때: %를 bar앞에 써준다.
- %: placeholder selector

```
1 √ %bar {
2 height: 14px;
3 font-size: 10px;
4 √ > div {
5 float: left;
6 clear: none;
7 }
8 }
9
10 √ .menu {
11 @extend %bar;
12 }
13
14 √ .nav {
15 @extend %bar;
16 }
17
```



```
1 √ .menu, .nav {
2 height: 14px;
3 font-size: 10px; }
4 √ .menu > div, .nav > div {
5 float: left;
6 clear: none; }
7
```

## 2.3 @extend

---

- @extend는 언제 사용하는 것이 좋을까?  
: @extend는 실제 CSS 코드에는 포함시킬 필요가 없는,  
보여지기 위한 (presentational) 클래스에 사용하는 것이 좋음

3.

---

색상 관련 함수, @import,  
Sass 함수 작성, Sass 라이브러리



## 3.1 색상 관련 함수

- Sass 함수를 이용해서 하나의 색깔에서 여러 가지 색상을 추출할 수 있음
- Lighten 함수

```
1 $background: black;
2 $text-color: #3ccd48;
3
4 ∨ body {
5 background: $background;
6 font-family: sans-serif;
7 color: $text-color;
8 }
9
10 ∨ a {
11 color: lighten($text-color, 20%);
12 }
```



```
1 ∨ body {
2 background: black;
3 font-family: sans-serif;
4 color: #3ccd48; }
5
6 a {
7 color: #8de294; }
8
```

16진수 색상 코드 값을 자동으로 계산해줌

You should totally [click here!](#)

## 3.1 색상 관련 함수

- Darken 함수

```
9
10 √ a {
11 color: darken($text-color, 30%);
12 }
```

You should totally click here!

- Complement 함수: 색상환에서 반대편에 위치해 있는 색상(보색)을 골라줌

```
√ body {
 background: $background;
 font-family: sans-serif;
 color: $text-color;
}
```

```
4
5 √ body {
6 background: complement($background);
7 font-family: sans-serif;
8 color: $text-color;
9 }
6
```

You should totally click here!



You should totally click here!

## 3.1 색상 관련 함수

- Desaturate 함수: 채도를 원하는 만큼 낮게 해줌

```
$background: desaturate(#687, 30%);
```

You should totally click here!



You should totally click here!

- 더 알아보기
  - [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#color\\_operations](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#color_operations)
  - <http://sass-lang.com/documentation/Sass/Script/Functions.html>

## 3.2 @import

---

- DRY 원칙을 지키다 하더라도 코드가 하나의 파일로 합쳐져 있는 것은 아님
  1. 성능 문제 때문에 html 파일에는 css 파일 하나만 링크하고 싶음
  2. 그러나, 개발자 입장에서는 css 파일을 기능별로 모듈화 시키는 것이 효율적
- 따라서, @import directive를 써서 모듈화된 sass 파일들을 하나의 파일에 합친 후, html 파일에는 @import 문을 모아 놓은 파일만 링크 시킨다.

## 3.2 @import

- Scss 나눠보기
  - \_variables.scss 파일: 변수 선언만을 따로 모아놓음

```
_variables.scss

1 $background: desaturate(#687, 30%);
2 $text-color: #3ccd48;
3 $highlighted-text-color: lighten($text-color, 30%);
```

```
1 @import "_variables.scss";
2
3 √ body {
4 background: complement($background);
5 font-family: sans-serif;
6 color: $text-color;
7 }
8
9 √ a {
10 color: $highlighted-text-color;
11 }
```

- 파일명 앞에 \_을 붙이면 CSS 파일로 컴파일 하지 않고 남겨 놓는다.
- 참고: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#partials](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#partials)

## 3.2 @import

- \_globals.scss: footer, nav 등 전역적으로 쓰여지는 CSS 코드를 모아놓음

\_globals.scss

```
1 √ body {
2 background: complement($background);
3 font-family: sans-serif;
4 color: $text-color;
5 }
6
7 √ a {
8 color: $highlighted-text-color;
9 }
```

main.scss

```
1 @import "_variables.scss";
2
3 @import "_globals.scss";
```

- main.scss 파일
  1. @import를 써서 모듈화된 scss 파일들을 한 군데에 모아놓는 용도
  2. CSS의 특성 때문에 파일 import하는 순서가 중요
    - 정의(변수, 믹스인) ► 글로벌 스타일 ► 페이지별 스타일

```
1 // Definings
2 @import "_variables.scss";
3 @import "_mixins.scss";
4
5 // Global Styles
6 @import "_globals.scss";
7
8 // Page Specifics
9 @import "pages/_about_us.scss";
10
```



## 3.2 @import

- 다른 scss 파일에서도 main.scss 파일을 import하여 스타일 상속 및 오버라이드 가능
- 어떤 방식으로 파일 import를 하는가에 대해서는 의견이 분분함
- CSS reset 파일을 맨 앞에 import 하는 것도 좋은 테크닉!

```
1 @import "_reset.scss";
2
3 // Definings
4 @import "_variables.scss";
5 @import "_mixins.scss";
6
7 // Global Styles
8 @import "_globals.scss";
9
10 // Page Specifics
11 @import "pages/_about_us.scss";
12
```

## 3.3 Sass 함수 작성하기

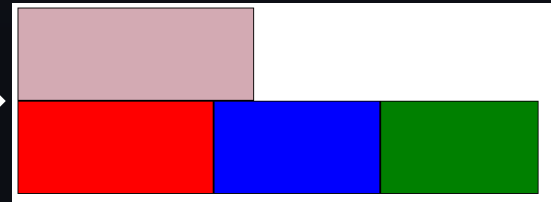
- 나만의 함수도 작성 가능
- 예제:

색상 값(변수 \$color)을 넣으면 RGB 값을 분리하여 각 직사각형의 세로 길이를 그 값에 따라 변하게 할 것임.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <link href="main.css" rel="stylesheet"/>
5 </head>
6 <body>
7 <div class="color"> </div>
8 <div class="red"> </div>
9 <div class="blue"> </div>
10 <div class="green"> </div>
11 </body>
12 </html>
```



함수 적용 전



완성!

## 3.3 Sass 함수 작성하기

- red(), blue(), green()에 \$color값을 바로 넘기면?

```
26 √ .red {
27 background: red;
28 width: red($color);
29 }
```

```
12
13 √ .red {
14 background: red;
15 width: 211;
16 }
```

px 단위가 붙지 않아서 유효한 값이 아니기 때문에 무시되고 부모 길이를 그냥 상속받음!

- px 단위를 붙여주는 pixify 함수 만들기
- 그냥 "px"를 더해주면 안되나요?

```
26 √ .red {
27 background: red;
28 width: red($color) + "px";
29 }
```

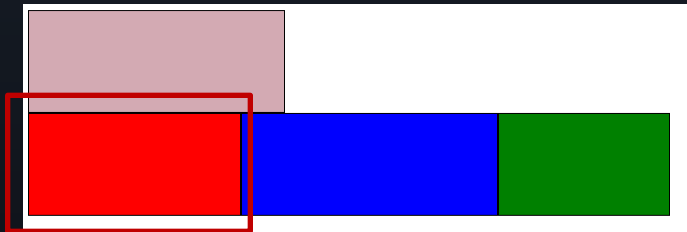
```
12
13 √ .red {
14 background: red;
15 width: "211px";
16 }
```

숫자값까지 스트링으로 전환됨!

## 3.3 Sass 함수 작성하기

- 그렇다면, unquote 함수로 따옴표를 없애서 문자열에서 일반 CSS 값으로 되돌리기

```
25
26 √ .red {
27 background: red;
28 width: unquote(red($color) + "px");
29 }
```



- blue와 green의 경우에도 적용할 것이므로 (재사용하므로) 함수로 추출하기
  - @function, @return directive 사용
  - 함수 내에서는 사칙연산도 가능

```
3 √ @function pxify($value) {
4 @return unquote($value + "px");
5 }
6
```

```
6 √ .red {
7 background: red;
8 width: pxify(red($color));
9 }
10
11 √ .blue {
12 background: blue;
13 width: pxify(blue($color));
14 }
15
16 √ .green {
17 background: green;
18 width: pxify(green($color));
19 }
20
```

## 3.3 Sass 함수 작성하기

### 전체 코드

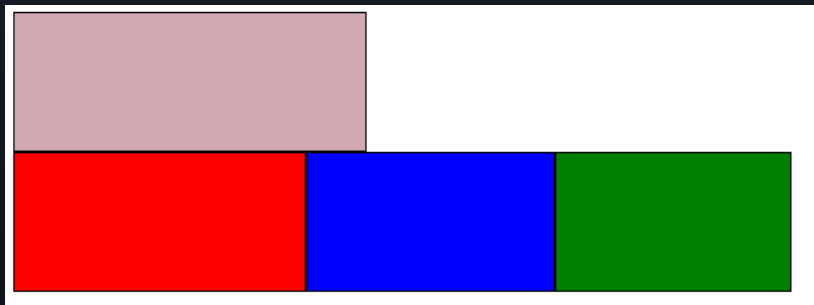
```
1 /* color 변수 생성하여 rgb 색상값을 분리하고자 하는 색상 코드값 할당 */
2 $color: #A69;
3
4 /* pxify 함수: 인자로 받은 값을 px단위를 가지는 plain CSS로 변환시켜 반환 */
5 ∇ @function pxify($value) {
6 @return unquote($value + "px")
7 }
8
9 /* div 요소의 초기값 설정 */
10 ∇ div {
11 height: 100px;
12 width: 255px;
13 border: 1px solid black;
14 }
15
16 /* color변수로 받은 색상 코드를 첫번째 div의 배경색으로 지정*/
17 ∇ .color {
18 background: $color;
19 }
20
21 /* color변수의 r값, g값, b값을 width로 표현하는 각각의 div를 일렬로 배치*/
22 ∇ .red, .blue, .green {
23 float: left;
24 }
```

```
1 <!DOCTYPE html>
2 ∇ <html>
3 ∇ <head>
4 <link href="main.css" rel="stylesheet"/>
5 ∇ </head>
6 ∇ <body>
7 <div class="color"> </div>
8 <div class="red"> </div>
9 <div class="blue"> </div>
10 <div class="green"> </div>
11 ∇ </body>
12 ∇ </html>
```

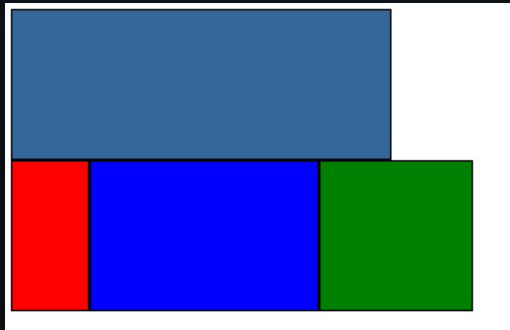
```
26 ∇ /*****
27 color변수의 rgb값이 각각 어느 정도 비율로 존재하는지
28 pixify함수를 이용하여 각 div의 width로 표현한다.
29 *****/
30
31 /* color변수로 받은 색상 코드의 R값만 추출하여 pxify함수를 이용해
32 px단위값으로 전환한 후, 그 값을 red클래스를 가지는 div의 width값으로 사용*/
33 ∇ .red {
34 background: red;
35 width: pxify(red($color));
36 }
37
38 /* 위와 동일한 로직 (G값 전달) */
39 ∇ .green {
40 background: green;
41 width: pxify(green($color));
42 }
43
44 /* 위와 동일한 로직 (B값 전달) */
45 ∇ .blue {
46 background: blue;
47 width: pxify(blue($color));
48 }
```

## 3.3 Sass 함수 작성하기

- 완성!



- 다른 색상 값을 넣어보면? (#336699)





## 3.4 Sass 라이브러리

---

- Bootstrap Sass 버전: <https://github.com/twbs/bootstrap-sass>
- Foundation Sass 버전: <http://foundation.zurb.com/sites/docs/v/5.5.3/sass.html>
- Compass: <http://compass-style.org/>
- Bourbon: <http://bourbon.io/>
  - Compass보다 경량화된 프레임워크
  - 브라우저 호환성에 잘 대비한 mixin 및 폰트 스택을 갖추
    - <http://bourbon.io/docs/#linear-gradient>
    - <http://bourbon.io/docs/#font-stacks>

4.

---

Sass 디버깅,  
Interpolation,  
@if/@else, @for/@each  
mixin 인자 더 알아보기

- Sass 컴파일 오류 화면
  - 컴파일 오류시, html 문서 <body> 태그 앞에 error 구문을 삽입해줌
  - 세미콜론 빠트리기는 오류는 비교적 찾기 쉬우나,  
괄호 등을 빠트리면 어느 부분에서 놓쳤는지 정확한 줄수를 알려주지 않음

```
34 */
35 v body:before {
36 white-space: pre;
37 font-family: monospace;
38 content: "Error: Invalid CSS after \"\${padding}\":
 expected selector or at-rule, was \"\": \${margin} *
 1.5;\"\"A on line 3 of ./main.scss\"\"A 1:
 \${primary-color}: #369;A 2: \${margin}: 5px
 \${padding}: \${margin} * 1.5;A 4: A 5: @mixin old {
 A 6: @warn \"Please use new() instead\";
 7: }A 8: "; }
```

Error: Invalid CSS after "\\${padding}": expected selector or at-rule, was "": \\${margin} \* 1.5;"  
on line 3 of ./main.scss

```
1: \${primary-color}: #369;
2: \${margin}: 5px
3: \${padding}: \${margin} * 1.5;
4:
5: @mixin old {
6: @warn "Please use new() instead";
7: }
8:
```

- Sass 파일 컴파일 시 줄 수 표기하기
- `sass -l main.scss:main.css`

```
1 /* line 5, main.scss */
2 body {
3 background: #369;
4 font-family: Futura, "Trebuchet MS", sans-serif; }
5 /* line 8, main.scss */
6 body > h1 {
7 font-size: 3em;
8 color: #eee;
9 margin: 5px 0 0 5px; }
10
11 /* line 15, main.scss */
12 .entry {
13 background: white;
14 margin: 5px;
15 padding: 7.5px; }
16 /* line 19, main.scss */
17 .entry h1 {
18 font-size: 1.2em;
19 margin: 0px; }
20 /* line 23, main.scss */
21 .entry .date {
22 font-size: 0.3em;
23 color: #369; }
24 /* line 27, main.scss */
25 .entry a {
26 color: #369; }
27
28 /** sourceMappingURL=main.css.map */
```

어떤 파일의 몇 번째 줄인지 주석으로 표기해서  
문제가 되는 부분을 찾기 좀 더 수월하게

## 4.1 Sass 디버깅

- @Warn directive
  - mixin 등을 사용시 유저에게 경고 메시지를 띄울 수 있음 (deprecated된 mixin 표시 등)

```
4
5 ① @mixin old {
6 ② @warn "Please use new() instead";
7 }
8
9 ③ body {
10 background: $primary-color;
11 font-family: Futura, "Trebuchet MS", sans-
12 serif;
13 > h1 {
14 font-size: 3em;
15 color: #eee;
16 margin: $margin 0 0 $margin;
17 }
18 ④ @include old;
19 }
```

```
NAVER@AD00030654 MINGW64 ~/Downloads/treeho
ugging-sass/start (master)
$ sass -l main.scss:main.css
WARNING: Please use new() instead
on line 6 of main.scss, in `old'
from line 17 of main.scss
```

- 참고: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#\\_5](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#_5)

## 4.2 Interpolation / 4.3 루프문

- `#{}구문` 안에 들어가는 것은 어떤 것이던 간에 일반 CSS 구문으로 취급
- url이나 기타 파일 주소에도 interpolation 사용 가능
- `@if/else` 구문에 비교 연산자 사용 가능
  - 예시: color 변수가 red면 테두리 추가

```
1 ∇ @mixin color_class($color) {
2 ∇ .#{$color} {
3 color: $color;
4 background-image: url("images/#
5 {$color}.jpg");
6 @if $color == red {
7 border: 1px solid black;
8 }
9 }
10 }
11
12 @include color_class(blue);
13 @include color_class(red);
14 @include color_class(green);
15
```

```
5
6 ∇ .blue {
7 color: blue;
8 background-image: url("images/blue.jpg"); }
9
10 ∇ .red {
11 color: red;
12 background-image: url("images/red.jpg");
13 border: 1px solid black;
14 }
15 ∇ .green {
16 color: green;
17 background-image: url("images/green.jpg"); }
18
```

## 4.3 @if/@else

- @if/@else 구문
- 입력된 width 값에 따라 padding 결정



```
10
17 √ @mixin box($width) {
18 √ @if $width > 100px {
19 padding: 0;
20 √ } @else if $width == 100px {
21 padding: 5px;
22 √ .big {
23 content: "huge!";
24 }
25 √ } @else {
26 padding: 10px;
27 }
28 }
29
30 √ div {
31 @include box(100px);
32 }
```

```
1 div {
2 padding: 5px; }
3 div .big {
4 content: "huge!"; }
5
```

더 알아보기: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#control\\_directives\\_expressions](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#control_directives_expressions)

- Box 1007





## 4.4 @for/@each

- @each문 사용해서 밴드 멤버들 사진 url 변경해보기
- 배열 변수들은 콤마(,)로 구분

```
6 @each $member in tom, jonny, colin, phil {
7 .bandmember.#{ $member } {
8 background: url("images/#{ $member }.jpg");
9 }
10 }
11
```



```
5 .bandmember.tom {
6 background: url("images/tom.jpg"); }
7
8 .bandmember.jonny {
9 background: url("images/jonny.jpg"); }
10
11 .bandmember.colin {
12 background: url("images/colin.jpg"); }
13
14 .bandmember.phil {
15 background: url("images/phil.jpg"); }
16
```

## 4.5 Mixin 인자 더 알아보기

- band() mixin 예제
  - 밴드 멤버 수는 mixin 사용할 때마다 변함!
  - mixin 작성시 두번째 인자에 '...' 추가하면 동적인 mixin 작성 가능

```
12 ✓ @mixin band($name $members...) {
13 ✓ @each $member in $members {
14 ✓ .#{ $name } .#{ $member } {
15 background: url("images/#{ $name }/#{ $member }.jpg");
16 }
17 }
18 }
19
20 @include band(radiohead, thom, jonny, colin, phil);
21 @include band(nin, trent);
22
```



```
1 .radiohead .thom {
2 background: url("images/radiohead/thom.jpg"); }
3
4
5
6
7
8
9
10 .radiohead .jonny {
11 background: url("images/radiohead/jonny.jpg"); }
12
13
14
15
16
17
18
19 .radiohead .colin {
20 background: url("images/radiohead/colin.jpg"); }
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
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

## 4.5 Mixin 인자 더 알아보기

- 혹은 '...' 추가 하지 않고, mixin @include시에  
배열 변수들을 공백으로 구분할 수도 있다


```
12 ∨ @mixin band($name, $members) {
13 ∨ @each $member in $members {
14 ∨ .#{$name} .#{$member} {
15 background: url("images/#{$name}/#{ $member}.jpg");
16 }
17 }
18 }
19
20 @include band(radiohead, thom jonny colin phil);
```

- 그러나, 다른 프로그래밍 언어의 컨셉과 충돌하는 문법이기 때문에 이 표기방식은 추천 X
- 참고: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#variable\\_arguments](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#variable_arguments)

## 4.5 Mixin 인자 더 알아보기

- box() mixin 예제
- 인자 기본값 설정하기 (key: value)

```
1 ∨ @mixin box($size: 10px, $color: black, $display: block) {
2 width: $size;
3 height: $size;
4 background: $color;
5 display: $display;
6 }
7
8 ∨ .box {
9 @include box(20px, blue);
10 }
11
```



```
1 ∨ .box {
2 width: 20px;
3 height: 20px;
4 background: blue;
5 display: block; }
6
```

- @include하여 mixin 호출할 시에 해당 인자를 넘겨주지 않으면 기본값이 적용됨
- 자주 사용하거나 혹시 모를 상태에 대비하고 싶으면 기본값을 설정해 놓는 것이 좋음

## 4.5 Mixin 인자 더 알아보기

- squire() mixin 예제
  - mixin @include시에 인자를 순서대로 잘 넣어줘야 함
  - 순서를 신경 쓰고 싶지 않으면 key: value 형태로 명시적으로 적어주면 됨

```
23 ∨ @mixin square ($size, $color: black) {
24 height: $size;
25 width: $size;
26 border: 1px solid $color;
27 }
28
29 ∨ .box {
30 @include square(10px, red);
31 }
32
33 ∨ .box {
34 @include square($color:red, $size: 10px);
35 }
36
```

- 참고: [http://sass-lang.com/documentation/file.SASS\\_REFERENCE.html#keyword\\_arguments\\_2](http://sass-lang.com/documentation/file.SASS_REFERENCE.html#keyword_arguments_2)

- 본 ppt는 아래 링크의 강의 내용을 요약해 놓은 것입니다.
  - <https://teamtreehouse.com/library/sass-basics>
- 기타 참고 사이트
  - <http://sass-lang.com/guide>

