





Variables

`$, !default`



변수(Variabes, \$)

PHP 변수처럼 \$ 이름 값을 설정한 후, 값을 참조하여 적용할 수 있습니다.

Yahoo9_example.html

```
1
$set-width: 940px;
$set-color: #ce4dd6;

#designer {
    width: $set-width;
    color: $set-color;
}

#planner {
    border: 1px {
        style: solid;
        color: $set-color;
    }
}
```

선택된 변수 값 참조

\$set-width: value;
#designer {
 width: \$set_width;
}

대쉬(-), 언더스코어(_) 동일하게 처리

The image shows a code editor interface with two tabs: "variables.sass" and "variables.css".

variables.sass:

```
1 /*  
2  * Color, Size, Percentage ...  
3  * 컬러, 사이즈, % ... 등등  
4 */  
5  
6 $padding: 10px  
7 $page-max-width: 1200px  
8  
9 #page  
10 // SASS 변수는 -, _ 구분하지 않고 같은 데이터로 인식합니다.  
11 max-width: $page_max_width  
12 padding: $padding
```

variables.css:

```
1 /* Color, Size, Percentage ...  
2  * 컬러, 사이즈, % ... 등등  
3 */  
4 #page {  
5   max-width: 1200px;  
6   padding: 10px;  
7 }
```



RWD, SASS 변수 + MQ 활용(SASS 3.2+)

반응형 웹 디자인에서 SASS 변수와 CSS3 미디어쿼리 조합은 매우 유용합니다.

yahoo9_example.html

```
1 $break-small: 320px;
  $break-large: 1200px; 브레이크 포인트 추후 수정!

.profile-pic {
  float: left;
  width: 250px;
  @media screen and (max-width: $break-small) {
    width: 100px;
    float: none;
  }
  @media screen and (min-width: $break-large) {
    float: right;
  }
}
```

320px

1200px

A screenshot of a code editor showing SASS code for responsive design. The code defines two variables: \$break-small at 320px and \$break-large at 1200px. It then uses these variables in a .profile-pic selector. For screens narrower than 320px, the width is set to 100px and float is set to none. For screens wider than or equal to 1200px, the float is set to right. A yellow box highlights the comment '브레이크 포인트 추후 수정!' (Breakpoint to be modified later). Purple arrows point from the variable definitions to their respective uses in the media queries. A red double-headed arrow indicates the range between the two breakpoints. A small icon of a book with 'Rt' on it is visible in the bottom right corner of the editor window.



RWD, SASS 변수 + MQ 활용(SASS 3.2+)

반응형 웹 디자인에서 SASS 변수와 CSS3 미디어쿼리 조합은 매우 유용합니다.

```
yahoo9_example.html
```

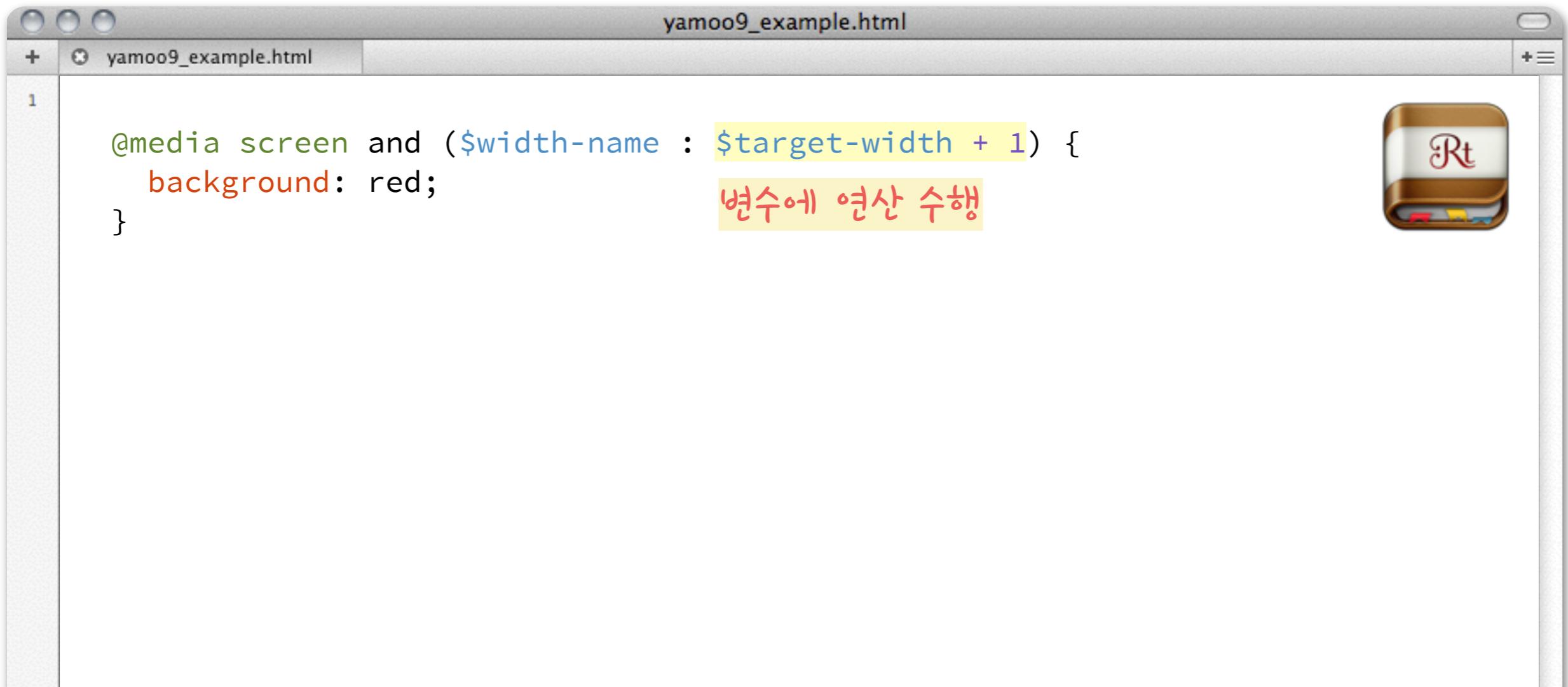
```
+ yahoo9_example.html +≡  
1  
$width-name: max-device-width; max-device-width  
$target-width: 320px;  
  
@media screen and ($width-name : $target-width) {  
    background: red;  
}  
  
속성(Property)도 변수 처리가 가능!
```

The screenshot shows a code editor window titled "yahoo9_example.html". The code demonstrates the use of SASS variables and media queries. It defines two variables: "\$width-name" set to "max-device-width" and "\$target-width" set to "320px". A media query "@media screen and (\$width-name : \$target-width) {" is used to apply a red background when the device width is 320px. An annotation with a purple bracket highlights the variable assignments, and another with a purple arrow points from the variable names to the media query condition. A yellow box labeled "속성(Property)도 변수 처리가 가능!" (Properties can also handle variables) is overlaid at the bottom right. A small icon of a book with "Rt" on it is visible in the top right corner of the editor window.



RWD, SASS 변수 + MQ 활용(SASS 3.2+)

반응형 웹 디자인에서 SASS 변수와 CSS3 미디어쿼리 조합은 매우 유용합니다.



The screenshot shows a code editor window titled "yamoo9_example.html". The file content is as follows:

```
1
@media screen and ($width-name : $target-width + 1) {
  background: red;
}
```

A yellow callout box highlights the expression `$target-width + 1` with the text "변수에 연산 수행" (Performing operations on variables). A small icon of a book with "Rt" on it is positioned next to the callout.

The image shows a code editor with two tabs open: "variables.sass" and "variables.css".

variables.sass:

```
1 /*  
2  * Color, Size, Percentage ...  
3  * 컬러, 사이즈, % ... 등등  
4 */  
5  
6 $padding: 10px  
7 $page-max-width: 1200px  
8 $width-name: max-device-width  
9 $target-width: 480px  
10  
11 #page  
12 // SASS 변수는 -, _ 구분하지 않고 같은 데이터로 인식합니다.  
13 max-width: $page_max_width  
14 padding: $padding  
15 background: #9ab6f5  
16  
17 @media screen and ($width-name : $target-width + 1)  
18 padding: $padding / 2  
19 background: #4bb6ef
```

variables.css:

```
1 /* Color, Size, Percentage ...  
2  * 컬러, 사이즈, % ... 등등  
3 */  
4 #page {  
5  max-width: 1200px;  
6  padding: 10px;  
7  background: #9ab6f5;  
8 }  
9 @media screen and (max-device-width: 481px) {  
10 #page {  
11  padding: 5px;  
12  background: #4bb6ef;  
13 }  
14 }  
15
```



변수 기본 값 설정(\$, !default)

변수를 설정한 후, 초기 값을 지정할 수 있습니다.

yahoo9_example.html

```
1 $set-width: 940px;  
2 $set-width: 1000px !default;  
3 $set-color: #ce4dd6;  
  
#designer {  
    width: $set-width; ●.....▶ 940px  
    color: $set-color;  
}
```

\$set-width 값이 설정된 경우 기본 값 (!default)보다 우선 적용됩니다.





변수 기본 값 설정(\$, !default)

변수를 설정한 후, 초기 값을 지정할 수 있습니다.

yahoo9_example.html

```
1 $set-width: 940px;  
2 $set-width: 1000px !default;  
3 $set-color: #ce4dd6;  
  
#designer {  
    width: $set-width; ●.....► 1000px  
    color: $set-color;  
}
```

\$set-width 값이 별도로 설정되지 않을 경우는 기본 값(!default)이 설정됩니다.

Rt

The image shows a code editor interface with two tabs: "variables.sass" and "variables.css".

variables.sass:

```
1 /*  
2 * Color, Size, Percentage ...  
3 * 컬러, 사이즈, % ... 등등  
4 */  
5  
6 $padding: 10px !default  
7 $page-max-width: 1200px  
8 $width-name: max-device-width  
9 $target-width: 480px  
10  
11 #page  
12   $padding: 20px  
13   // SASS 변수는 -, _ 구분하지 않고 같은 데이터로 인식합니다.  
14   max-width: $page_max_width  
15   padding: $padding  
16   background: #9ab6f5  
17  
18 @media screen and ($width-name : $target-width + 1)  
19   padding: $padding / 2  
20   background: #4bb6ef
```

variables.css:

```
1 /* Color, Size, Percentage ...  
2 * 컬러, 사이즈, % ... 등등  
3 */  
4 #page {  
5   max-width: 1200px;  
6   padding: 20px;  
7   background: #9ab6f5;  
8 }  
9 @media screen and (max-device-width: 481px) {  
10   #page {  
11     padding: 10px;  
12     background: #4bb6ef;  
13   }  
14 }  
15
```



디폴트(!default) 플래그 스니펫 만들기

Sublime Text에서 활용 가능한 디폴트 플래그 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[ !default ]]></content>  
  <tabTrigger>default</tabTrigger>  
  <description>SASS - !default</description>  
  <scope>source.scss, source.sass</scope>  
</snippet>
```



Data Type



데이터 유형

SASS는 6가지(Null, Number, String & Color, Boolean, list, map) 데이터 형을 지원합니다.

yahoo9_example.html

1

Numbers 숫자형
1.2, 3, 14px

Nulls 비어있음
null

Strings & Colors 문자형
“./images/icon.jpg” | ‘Times New Roman’ | Verdana | lightblue | #fe4940

Booleans 논리형 (참, 거짓)
true, false

Lists 공백, 콤마(,)로 구분되는 목록 (Javascript 배열과 유사), 관련 함수
1.5em 1em 0 2em | 2px solid gray | Helvetica, Sans-Serif

Maps 키:값으로 구성된 그룹 (Javascript 객체와 유사), 관련 함수로 값을 얻을 수 있습니다.
\$map: (key1: value1, key2: value2)



Operations

+,-,*,/,%,
==,!>,<,>=,<=



연산(Operations)

덧셈(+), 뺄셈(-), 곱셈(*), 나눗셈(/), 나머지(%) 등 수학의 연산 결과를 수행할 수 있습니다.

yahoo9_example.html

```
// 변수 설정
$global-padding: 10px;
$page-max-width: 1420px;

// 컨테이너 클래스 설정
.container {
    max-width: $page-max-width - $global-padding * 2;
    padding: 0 $global-padding;
}
```

변수 치환 후, 연산 과정이 이루어집니다.

1420px
10px

The image shows a code editor interface with two tabs: "operations.sass" and "operations.css".

operations.sass:

```
1 /*  
2  * Operation, 연산  
3 */  
4 // 변수 설정  
5 $global-padding: 14px  
6 $page-max-width: 1320px  
7  
8 // 컨테이너 클래스 설정  
9 .container  
10    max-width: $page-max-width - $global-padding * 2  
11    padding: 0 $global-padding
```

operations.css:

```
1 /* Operation, 연산  
2 */  
3 .container {  
4   max-width: 1292px;  
5   padding: 0 14px;  
6 }  
7
```

```
/*
 * SASS 사칙연산
 * + - ÷ × %
 */

$width: 10px
$double_width: $width * 2 // 10 × 2 = 20px
$half_width: $width / 2 // 10 ÷ 2 = 5px
$width_plus_2: $width + 2 // 10 + 2 = 12px
$width_minus_2: $width - 2 // 10 - 2 = 8px
```



operations-minus.sass

```
1 $base: 13px
2 body
3   font: $base/1.5 Dotum
4   font: $base / 1.5 Dotum
5   font: ($base / 1.5) Dotum
6   // 변수 재정의
7   $base: 13
8   font: $base+px/1.5 Dotum
9   font: $base+px / 1.5 Dotum
10  font: ($base/13 + $base/1.3)+px / 1.5 Dotum
```

operations-minus.css

```
1 body {
2   font: 8.66667px Dotum;
3   font: 8.66667px Dotum;
4   font: 8.66667px Dotum;
5   font: 13px/1.5 Dotum;
6   font: 13px/1.5 Dotum;
7   font: 11px/1.5 Dotum;
8 }
9
```

```

1  /*
2   * RWD & Media Queries & @content
3   */
4
5 // 반응형 웹 - 브레이크 포인트 변수
6 $small-screen-width: 320px
7 $wide-screen-width: 1200px
8
9 // 믹스인 respond-to
10 =respond-to($response)
11   @if $response == mobile
12     @media only screen and (max-device-width: $small-screen-width)
13       @content
14   @else if $response == tablet
15     @media only screen and (min-device-width: $small-screen-width + 1)
16       and (max-device-width: $wide-screen-width - 1)
17       @content
18   @else if $response == wide-screen
19     @media only screen and (min-device-width: $wide-screen-width)
20       @content
21
22 // 믹스인 호출
23 +respond-to(mobile)
24   body
25     padding: 0 10%
26
27 +respond-to(tablet)
28   body
29     padding: 5%
30
31 +respond-to(wide-screen)
32   body
33     padding: 20px

```

```

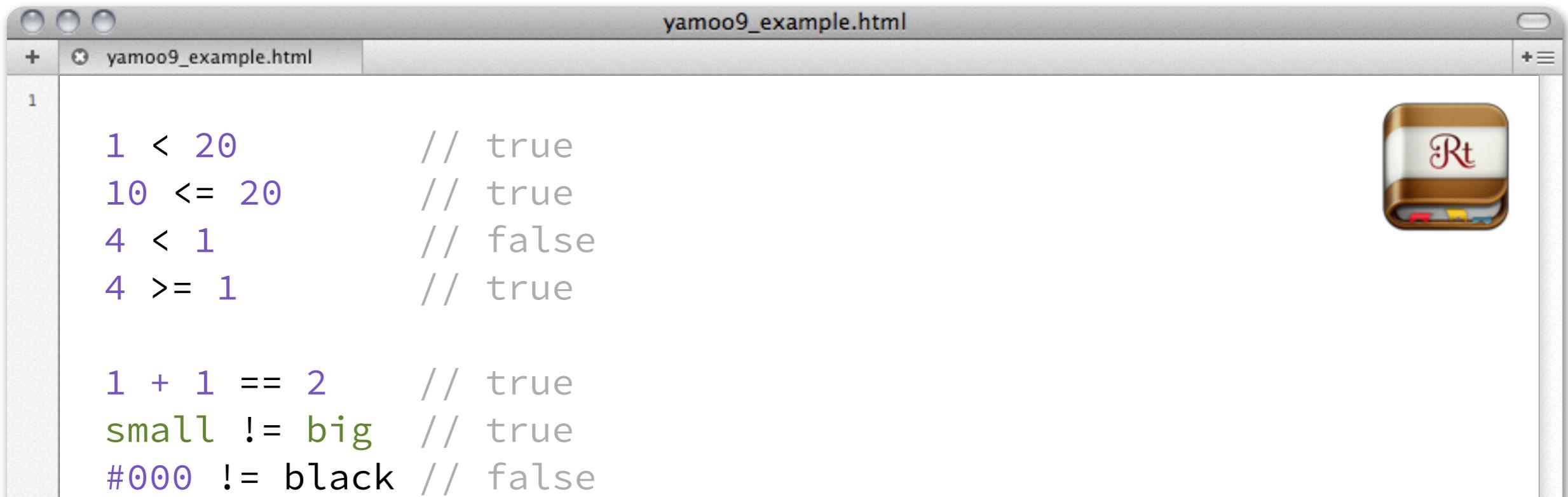
1  /* RWD & Media Queries & @content
2   */
3   @media only screen and (max-device-width: 320px) {
4     body {
5       padding: 0 10%;
6     }
7   }
8   @media only screen and (min-device-width: 321px) {
9     body {
10       padding: 5%;
11     }
12   }
13  @media only screen and (min-device-width: 1201px) {
14    body {
15      padding: 20px;
16    }
17  }
18

```



비교 연산(Comparison Operations)

크다(>), 작다(<), 크거나 같다(>=), 작거나 같다(<=), 같다(==), 다르다(!=) 등 비교 연산 결과를 제공합니다.



The screenshot shows a code editor window titled "yamoo9_example.html". The code contains several comparison operations:

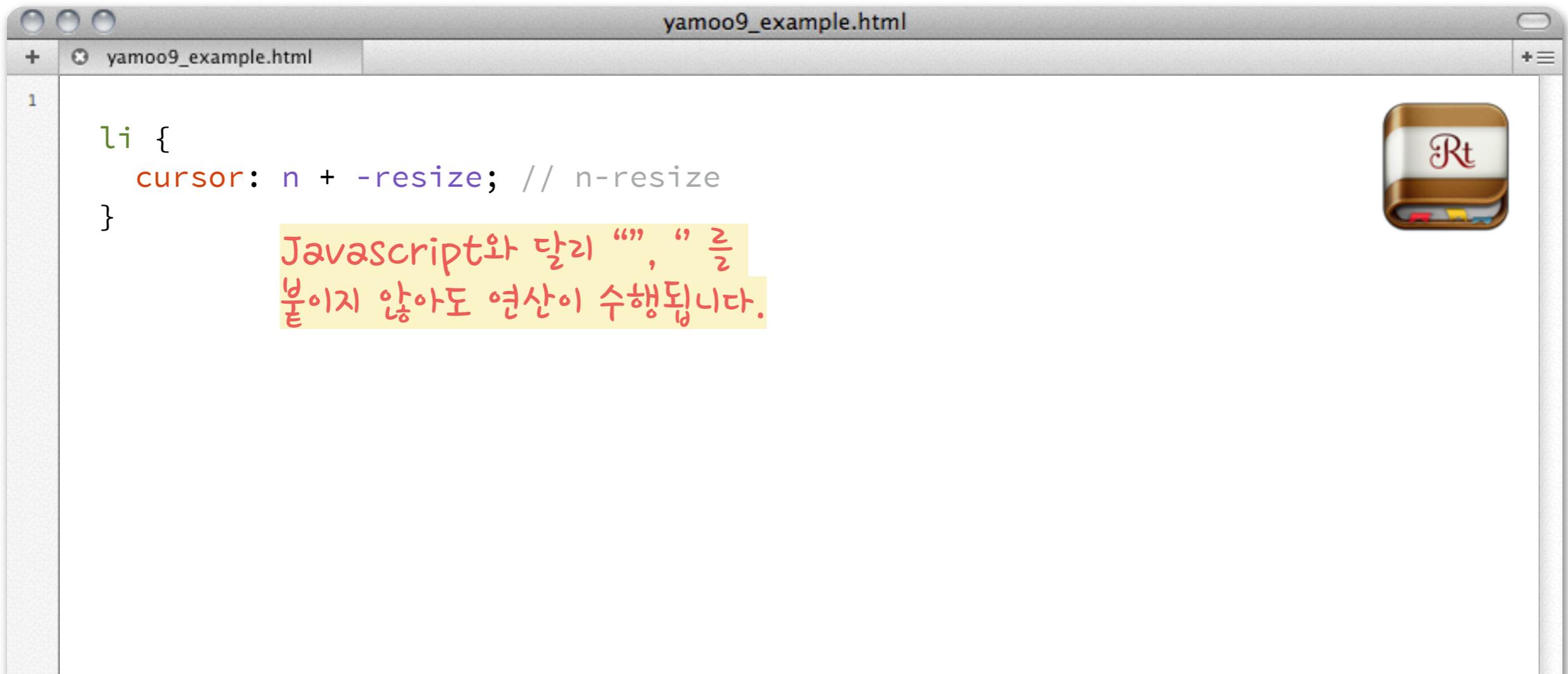
```
1 < 20          // true
10 <= 20        // true
4 < 1           // false
4 >= 1          // true

1 + 1 == 2      // true
small != big    // true
#000 != black   // false
```



문자 연산(String Operations)

문자와 문자를 접합하려는 경우 + 연산자를 사용할 수 있습니다.



```
1
li {
  cursor: n + -resize; // n-resize
}
```

Javascript와 달리 "", " 를
붙이지 않아도 연산이 수행됩니다.



보간법(#{}, Interpolation)

SASS는 변수를 “ ” 내부에서 처리할 수 있는 보간 법을 지원합니다.

```
yamoo9_example.html
```

```
+ ① yamoo9_example.html +≡  
1  
$family: "Droid+Sans";  
@import url("http://fonts.googleapis.com/css?family=#{$family}");  
  
@mixin do($selector, $message) {  
  body #{$selector}::before {  
    content: $message;  
  }  
}  
  
@include do(".demo", "this is demo.");
```

웹 폰트 CDN을 사용할 경우,
보간법이 유용하게 활용될 수 있어요.

```
operations-string.sass
```

```
1 /*  
2  * 문자 연산  
3 */  
4 li  
5   $strong-font: 12  
6   cursor: poi + nter // pointer  
7  
8   a::before  
9     content: ' hey! ' + ' 너 어디서 왔어?'  
10  
11 .selected &::before  
12   content: '네 나이는 20 + 3 아니냐?'  
13  
14 .focused & .strong::first-line  
15   font-size: '$strong-font px'  
16   // 보간법  
17   font-size: '#{\$strong-font}px'  
18  
19 .active & span::after  
20   // 보간법  
21   content: '네 나이는 #{20 + 3}이 맞아'
```

```
operations-string.css
```

```
1 /* 문자 연산  
2 */  
3 li {  
4   cursor: pointer;  
5 }  
6 li a::before {  
7   content: " hey! 너 어디서 왔어?";  
8 }  
9 .selected li::before {  
10  content: "네 나이는 20 + 3 아니냐?";  
11 }  
12 .focused li .strong::first-line {  
13  font-size: "$strong-font px";  
14  font-size: "12px";  
15 }  
16 .active li span::after {  
17  content: "네 나이는 23이 맞아";  
18 }  
19
```



RWD, 쿼리문 보간법 활용(#{ })

SASS는 변수를 “ ” 내부에서 처리할 수 있는 보간 법을 지원합니다.



The screenshot shows a browser window with the title "yamoo9_example.html". The code editor contains the following SASS code:

```
1 $information-phone: "only screen and (max-width : 320px)";

@media #{$information-phone} {
  background: red;
}
```

A small icon of a book with the letters "Rt" is visible on the right side of the code editor.



보간법(#{}) 스니펫 만들기

Sublime Text에서 활용 가능한 보간법 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[#${0}]]></content>  
  <tabTrigger>`</tabTrigger>  
  <description>SASS - #{ }</description>  
  <scope>source.scss, source.sass</scope>  
</snippet>  
  
' 단축 키워드 말고 직접 'interpolation'으로  
등록하셔도 좋겠죠? ^ - ^
```



컬러 연산(Color Operations)

덧셈(+), 뺄셈(-), 곱셈(*), 나눗셈(/), 나머지(%) 등 수학의 연산 결과를 수행할 수 있습니다.

yahoo9_example.html

```
1
p {
  color: #010203 + #040506;
/*
  01 + 04 = 05
  02 + 05 = 07
  03 + 06 = 09
  -----
#050709
*/}
```

RRGGBB 값 중, 짝이 맞는 유형끼리
연산 과정이 일어나게 됩니다.



컬러 연산(Color Operations)

덧셈(+), 뺄셈(-), 곱셈(*), 나눗셈(/), 나머지(%) 등 수학의 연산 결과를 수행할 수 있습니다.

yahoo9_example.html

```
1
p {
  color: #010203 * 2;
/*
  01 * 2 = 02
  02 * 2 = 04
  03 * 2 = 06
-----
#020406
*/}
```

RRGGBB 값의 각 짝에
곱하기 연산 과정이 수행됩니다.

operations-color.sass

```
1  /*  
2   * Color Operation  
3   */  
4 p  
  color: #010203 + #040506  
5  
6  
7 div  
  color: #010203 * 2  
8  
9  
10
```

operations-color.css

```
1  /* Color Operation  
2   */  
3 p {  
  color: #050709;  
5 }  
6  
7 div {  
  color: #020406;  
9 }  
10
```



기타 연산(Etc. Operations)

불린 데이터 연산을 사용할 수 있습니다.

yahoo9_example.html

+ yahoo9_example.html +≡

1

Boolean Operations

SassScript supports `and`, `or`, and `not` operators for boolean values.

불린(Boolean) 연산 지원: `and`, `or`, `not`

List Operations

Lists don't support any special operations. Instead, they're manipulated using the [list functions](#).

리스트(List) 연산 미지원. 배열 객체의 `length`, `join` 등 유사 함수 지원!



Mixins

@mixin, @include

=, +



믹스인(@mixin)

JS 함수와 흡사한 믹스인은 @mixin으로 모듈을 정의한 후, @include로 호출할 수 있어 재 사용이 가능합니다.

```
yamoo9_example.html
```

```
+ ① yamoo9_example.html +≡  
1  
① @mixin box-sizing {  
    -webkit-box-sizing: border-box;  
    -moz-box-sizing: border-box;  
    box-sizing: border-box; }  
  
#app {  
    @include box-sizing; }  
  
#design {  
    @include box-sizing; }  
  
@mixin으로 믹스인을 정의하고,  
@include으로 믹스인 호출합니다.
```



reset 믹스인 만들기

초기화와 관련된 믹스인을 만들어 봅시다.

```
yahoo9_example.html
```

```
1 @ mixin reset-box-model {  
    margin: 0;  
    border: 0;  
    padding: 0;  
} @include reset-box-model;  
  
@ mixin reset-table {  
    border-collapse: collapse;  
    border-spacing: 0;  
} @include reset-table;  
  
@ mixin reset-table-cell {  
    text-align: left;  
    font-weight: normal;  
    vertical-align: middle;  
} @include reset-table-cell;  
  
@ mixin reset-font {  
    font: inherit;  
    font-size: 100%;  
    vertical-align: baseline;  
} @include reset-font;
```



yahoo9_example.html

```
1 @mixin reset-focus {
    outline: 0;
} @include reset-focus;

@mixin reset-body {
    line-height: 1;
} @include reset-body;

@mixin reset-quotation {
    quotes: none;
    &::before, &::after {
        content: '';
    }
} @include reset-quotation;

@mixin reset-list-style {
    list-style: none;
} @include reset-list-style;

@mixin reset-a-img-border {
    border: 0;
} @include reset-a-img-border;

@mixin reset-html5-block {
    #{$html5-block} {
        display: block;
    }
} @include reset-html5-block;
```



```
+ yamoo9_example.html
1
@mixin reset-focus {
    outline: 0;
} @include reset-focus;

@mixin reset-body {
    line-height: 1.5;
} @include reset-body;

@mixin reset-quotation {
    quotes: none;
    &::before, &::after {
        content: '';
    }
} @include reset-quotation;

@mixin reset-list-style {
    list-style: none;
} @include reset-list-style;

$html5-block: "article, aside, details,
figcaption, figure, footer, header, hgroup,
main, nav, section, summary" !default;
header {
    border: 1px solid #ccc;
    padding: 5px;
}

@mixin reset-html5-block {
    #{$html5-block} {
        display: block;
    }
} @include reset-html5-block;
```



yahoo9_example.html

```
1 @mixin nested-reset {
  #{$html-elements} {
    @include reset-box-model;
    @include reset-font;
  }
  table {
    @include reset-table;
  }
  caption, th, td {
    @include reset-table-cell;
  }
  q, blockquote {
    @include reset-quotation;
  }
  a img {
    @include reset-a-img-border;
  }
}
@include nested-reset;
```

yahoo9_example.html

```
@mixin global-reset {
  html { @include reset-body; }
  @include nested-reset;
  @include reset-html5;
}
@include global-reset;
```

\$html-elements: "div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre, a, abbr, acronym, address, big, cite, code, del, dfn, em, img, ins, kbd, q, s, samp, small, strike, strong, sub, sup, tt, var, b, u, i, center, dl, dt, dd, ol, ul, li, fieldset, form, label, legend, table, caption, tbody, tfoot, thead, tr, th, td, article, aside, canvas, details, embed, figure, figcaption, footer, header, hgroup, menu, nav, output, ruby, section, summary, time, mark, audio, video" !default;



전달 인자(Arguments)

JS 함수 확장처럼 @ mixin으로 정의된 모듈에 인자를 전달하여 믹스인을 확장할 수 있습니다.

The screenshot shows a code editor window titled "yamoo9_example.html". The code defines a mixin "border-radius" that takes a variable argument \$radius and applies it to various browser-specific border-radius properties. It also shows examples of how to use this mixin in CSS rules for "#app" and "#design". A red callout box highlights the \$radius variable and its definition in the mixin, explaining it as a passed-in variable. An icon of a book with "Rt" on it is visible in the sidebar.

```
@mixin border-radius( $radius ) {
  -webkit-border-radius: $radius;
  -moz-border-radius: $radius;
  border-radius: $radius; }

#app {
  @include border-radius( 10px ); }

#design {
  @include border-radius( 3px 3px 0 0 ); }
```

\$radius는 일종의 변수로
믹스인 내부에 전달된 값을
받기에 전달인자라고 부릅니다.



활용성이 떨어지는 ‘정적 믹스인’

정적인 믹스인은 매번 동일한 스타일만 설정이 가능하기 때문에 활용성이 떨어집니다.

```
yahoo9_example.html  
+ yahoo9_example.html +  
1  
@ mixin hover-link {  
    text-decoration: none;  
    &:hover, &:focus {  
        padding-bottom: 0.02em;  
        border-bottom: 1px solid #404040;  
    }  
}  
  
@include hover-link;
```



활용성이 향상된 ‘동적 믹스인’

동적인 믹스인은 전달된 인자에 따라 스타일 변경이 가능하기 때문에 활용성이 향상됩니다.

```
1  yahoo9_example.html
+ yahoo9_example.html +≡
  @mixin hover-link($padding-bottom, $border-bottom) {
    text-decoration: none;
    &:hover, &:focus {
      padding-bottom: $padding-bottom;
      border-bottom: $border-bottom;
    }
  }
  @include hover-link(0.3em, 1px dashed #6b6b6b);
```

The code block shows a Sass file named 'yahoo9_example.html'. It contains a mixin definition for 'hover-link' that takes two arguments: '\$padding-bottom' and '\$border-bottom'. Inside the mixin, there is a rule for the '&' selector (representing the element itself) that applies 'text-decoration: none' and changes the padding and border for ':hover' and ':focus' states. A purple oval highlights the entire mixin definition. A blue curly brace highlights the argument list at the bottom. Two small blue squares with arrows point from the bottom of the purple oval to the start of the argument list and the start of the curly brace respectively. The file is currently open in a browser window titled 'yahoo9_example.html'.



전달 인자 기본 값 설정(Args, Default Value)

믹스인 호출시 값을 전달하지 않아 오류가 발생하는 것을 방지하기 위해 특정 값을 기본으로 설정할 수 있습니다.

A screenshot of a code editor window titled "yamoo9_example.html". The code editor shows the following Sass code:

```
1 @mixin border-radius( $radius: 4px ) {  
  -webkit-border-radius: $radius;  
  -moz-border-radius: $radius;  
  border-radius: $radius; }  
  
#app {  
  @include border-radius; } ←  
  
#design {  
  @include border-radius( 5px ); }
```

The value `$radius: 4px` in the mixin definition is highlighted with a purple box and a purple bracket points from it to the corresponding value in the `#app` selector's `@include` statement. A red annotation box with a red arrow points to the `#design` selector's `@include` statement, containing the following text:

border-radius 믹스인 호출 시,
전달인자가 없으면 기본 값인 4px이
대입됩니다.



오류를 방지하는 ‘기본값 설정 믹스인’

기본값 설정 믹스인은 인자가 전달되지 않아도 기본값이 사용되기 때문에 오류를 방지할 수 있습니다.

```
yahoo9_example.html  
+ yahoo9_example.html +  
1  
@Mixin hover-link($padding-bottom: 2px, $border-bottom: 1px solid #767676) {  
    text-decoration: none;  
    &:hover, &:focus {  
        padding-bottom: $padding-bottom;  
        border-bottom: $border-bottom;  
    }  
}  
  
@include hover-link();  
  
전달인자 없이 호출해도  
오류가 발생하지 않습니다.
```



키워드 전달 인자(Args, Keyword)

믹스인 호출시 값을 전달하지 않아 오류가 발생하는 것을 방지하기 위해 특정 값을 기본으로 설정할 수 있습니다.

1

\$text는 '기본 값' 사용

```
@mixin link-colors($text:blue, $hover:red) {  
  color: $text;  
  &:hover { color: $hover; }  
}  
  
a {  
  @include link-colors($hover:green);  
}
```

전달인자 중, 특정인자만 선택하여 값을 전달할 수 있어 매우 유연합니다.

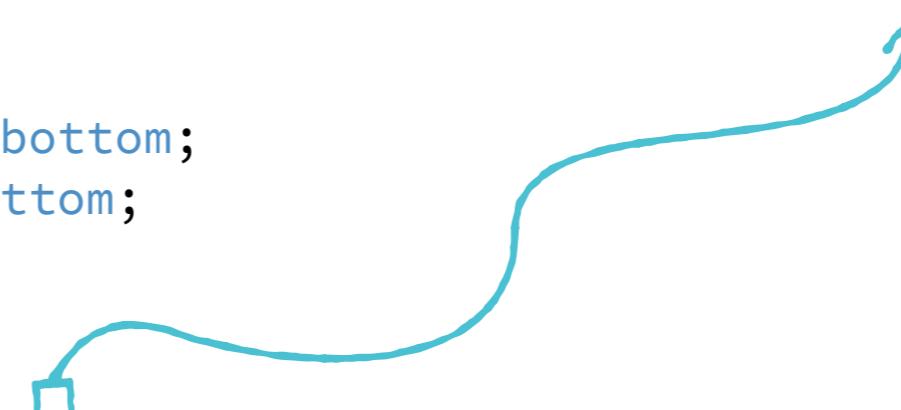


특정 인자만 전달/수정하는 것이 가능!

특정 키워드를 지정해 믹스인에 인자 값을 전달할 수 있어 수정이 용이합니다.

```
yahoo9_example.html
```

```
1 @mixin hover-link($padding-bottom: 2px, $border-bottom: 1px solid #767676) {  
  text-decoration: none;  
  &,:hover, &,:focus {  
    padding-bottom: $padding-bottom;  
    border-bottom: $border-bottom;  
  }  
}  
  
@include hover-link($border-bottom: 1px dashed #6b6b6b);
```





멀티 전달 인자, 유지보수 문제 (Maintenance)

믹스인에 전달할 인자의 개수가 고정된 경우, 응용하여 활용하기 불편합니다.

yahoo9_example.html

```
1 @mixin box-shadow( $args1, $args2 ) {  
  -webkit-box-shadow: $args1, $args2;  
  -moz-box-shadow: $args1, $args2;  
  box-shadow: $args1, $args2; }  
  
.box-double-shadow {  
  @include box-shadow(2px 6px 10px #999, 0px 4px 5px #666); }
```

2개 이상 박스 그림자를 설정해야 할 경우,
믹스인을 수정해야 합니다.



가변 전달 인자(Args, Variable)

믹스인 호출시 전달 받을 인자 List를 … 로 처리하면 복수의 인자를 처리하기 쉽습니다.



The screenshot shows a code editor window titled "yamoo9_example.html". The file contains the following Sass code:

```
1 @mixin box-shadow( $args... ) {  
  -webkit-box-shadow: $args;  
  -moz-box-shadow: $args;  
  box-shadow: $args; }  
  
.box-double-shadow {  
  @include box-shadow(2px 6px 10px #999, 0px 4px 5px #666, ...); }
```

A small icon of a book with the letters "Rt" is visible on the right side of the editor window.

Create Configurable CSS3 Shapes with Mixins

CSS3로 구현된 도형(원, 삼각형) 코드를 빠르게 처리할 수 있는 믹스인을 만들어 봅니다.

```
@ mixin border-radius ($radius: 4px) {  
    -webkit-border-radius: $radius;  
    -moz-border-radius: $radius;  
    border-radius: $radius;  
}  
  
@ mixin background-origin ($origin: padding-box) {  
    -webkit-background-origin: $origin;  
    -moz-background-origin: $origin;  
    background-origin: $origin;  
}  
  
@ mixin background-clip ($content: padding-box) {  
    -webkit-background-clip: $content;  
    -moz-background-clip: $content;  
    background-clip: $content;  
}
```

Create Configurable CSS3 Shapes with Mixins

CSS3로 구현된 도형(원, 삼각형) 코드를 빠르게 처리할 수 있는 믹스인을 만들어 봅니다.

```
// 사용자 정의 믹스인: 원 도형 만들기
@mixin circle ($diameter: 10px, $bgcolor: black) {
  width: $diameter;
  height: $diameter;
  background: $bgcolor;
  @include border-radius($diameter/2);
  @include background-clip;
}
```

Create Configurable CSS3 Shapes with Mixins

CSS3로 구현된 도형(원, 삼각형) 코드를 빠르게 처리할 수 있는 믹스인을 만들어 봅니다.

```
// 사용자 정의 믹스인: 삼각형 도형 만들기
// http://css-tricks.com/snippets/css/css-triangle/
@mixin triangle ($base, $direction, $bgcolor) {
    width: 0;
    height: 0;
    $half-base: $base/2;
    border: {
        left: $half-base solid transparent;
        right: $half-base solid transparent;
        bottom: $half-base solid $bgcolor;
    }
}
```

Create Configurable CSS3 Shapes with Mixins

CSS3로 구현된 도형(원, 삼각형) 코드를 빠르게 처리할 수 있는 믹스인을 만들어 봅니다.

```
@if $direction == up {  
    border: {...}  
}  
@else if $direction == bottom {  
    border: {...}  
}  
@else if $direction == left {  
    border: {...}  
}  
@else {  
    border: {...}  
}  
  
@if $direction == up {  
    border: {...}  
}  
@if $direction == bottom {  
    border: {...}  
}  
@if $direction == left {  
    border: {...}  
}  
@if $direction == right {  
    border: {...}  
}
```

Create Configurable CSS3 Shapes with Mixins

CSS3로 구현된 도형(원, 삼각형) 코드를 빠르게 처리할 수 있는 믹스인을 만들어 봅니다.

```
// 사용자 정의 믹스인(단축): 빠른 삼각형 도형 만들기
@mixin triangle-top ($base, $bgcolor) {
    @include triangle($base, top, $bgcolor);
}

@mixin triangle-right ($base, $bgcolor) {
    @include triangle($base, right, $bgcolor);
}

@mixin triangle-bottom ($base, $bgcolor) {
    @include triangle($base, bottom, $bgcolor);
}

@mixin triangle-left ($base, $bgcolor) {
    @include triangle($base, left, $bgcolor);
}
```



yamoo9 / _shapes.scss

Last active just now

<https://gist.github.com/yamoo9/85029ec79b80292e06aa>

CSS3 도형(원, 삼각형)을 빠르게 그릴 수 있는 SASS 믹스인: Create Configurable CSS3 Shapes with SASS Mixins

Gist Detail

Revisions 4

Download Gist

Clone this gist

<https://gist.github.com/yamoo9/85029ec79b80292e06aa>

Embed this gist

<script src="https://gist.github.com/yamoo9/85029ec79b80292e06aa">

Link to this gist

<https://gist.github.com/yamoo9/85029ec79b80292e06aa>

_shapes.scss

Raw

```
1 // CSS3 둥근 테두리
2 // https://developer.mozilla.org/ko/docs/CSS/border-radius
3 @ mixin border-radius ($radius) {
4     -webkit-border-radius: $radius;
5     -moz-border-radius: $radius;
6     border-radius: $radius;
7 }
8
9 // CSS3 배경 위치 설정
10 // http://www.css3.info/preview/background-origin-and-background-clip/
11 // http://caniuse.com/#search=background-origin
12 // 배경 위치 설정의 기준은 background-image 이미지 설정의 원점
13 // background-attachment: fixed; 일 경우 background-origin은 무시됨.
14 // https://developer.mozilla.org/en-US/docs/Web/CSS/background-attachment
15 @ mixin background-origin ($origin: padding-box) {
16     -webkit-background-origin: $origin;
17     -moz-background-origin: $origin;
18     background-origin: $origin;
19 }
20
21 // CSS3 배경 다듬기
22 // http://tumble.sneak.co.nz/post/928998513/fixing-the-background-bleed
23 // https://developer.mozilla.org/en-US/docs/Web/CSS/background-clip
24 @ mixin background-clip ($content: padding-box) {
25     -webkit-background-clip: $content;
26     -moz-background-clip: $content;
27     background-clip: $content;
28 }
29
30 // 사용자 정의 믹스인: 원 도형 만들기
31 //
32 @ mixin circle ($diameter: 10px, $bgcolor: black) {
33     width: $diameter;
34     height: $diameter;
35     background: $bgcolor;
36     @include border-radius($diameter/2);
37     @include background-clip;
```



변수 범위(Scope)와 콘텐츠 블록(@content)

변수 범위는 JS의 전역/지역 변수 개념과 유사하며, 콘텐츠 블록은 믹스인 호출 시 {} 코드문입니다.

yahoo9_example.html

```
// 전역 변수  
$color: #3fb5c8;  
  
@mixin colors($color: darkred) {  
    // 믹스인 내부에서 $color 값은 지역 변수인 '전달인자'를 가리킵니다.  
    background-color: $color; // darkred  
    // @content 값은 @include 믹스인 {}에서 {} 블록 영역을 말합니다.  
    @content; ←  
    border-color: $color; // darkred  
}  
  
.bg-border-colors {  
    // $color는 전역 변수인 #3fb5c8이 대입됩니다.  
    @include colors { color: $color; } ←  
}
```



@content 스니펫 만들기

Sublime Text에서 활용 가능한 @content 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[@content;]]></content>  
  <tabTrigger>content</tabTrigger>  
  <description>SASS - @content</description>  
  <scope>source.scss, source.sass</scope>  
</snippet>
```



믹스인 *.sass 문법

@ mixin은 '='으로, @include는 '+'로 선언/호출이 가능합니다.

yamoo9_example.html

```
+ yahoo9_example.html
```

```
1
@mixin my-btn($color) {
  color: $color;
}
@include my-btn(red);
```

VS

= 믹스인

+ 인클루드

VS

=my-btn(\$color)
color: \$color

+my-btn(red)





믹스인/인클루드(@mixin, @include) 스니펫 만들기

Sublime Text에서 활용 가능한 믹스인/인클루드 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[ @mixin ${1:믹스인 이름} ${2:($3)} {  
    $4  
  }  
  $5 ]></content>  
  <tabTrigger>= </tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - 믹스인 정의</description>  
</snippet>  
  
'=' 단축 키워드 말고 직접 'mixin'으로  
등록하셔도 좋겠죠? ^ - ^
```



믹스인/인클루드(@mixin, @include) 스니펫 만들기

Sublime Text에서 활용 가능한 믹스인/인클루드 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[@include ${1:믹스인 이름}${2:($3)}]]></content>  
  <tabTrigger>+</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - 믹스인 호출</description>  
</snippet>
```

‘+’ 단축 키워드 말고 직접 ‘include’로
등록하셔도 좋겠죠? ^ - ^



function

@function, @return



컬러 함수 (Color Functions)

RGBA 함수



```
// SET 함수
rgb($red, $green, $blue)
rgba($red, $green, $blue, $alpha)
rgba($color, $alpha)

// GET 함수
red($color)
green($color)
blue($color)

// 혼색 함수
mix($color-1, $color-2, [$weight])
```



RGB 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 RGB 컬러 함수 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[rgb(${1:$red}, ${2:$green}, ${3:$blue})]]></content>  
  <tabTrigger>rgb</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - rgb() 컬러 함수</description>  
</snippet>
```



RGBA 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 RGBA 컬러 함수 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[rgba(${1:$red}, ${2:$green}, ${3:$blue},  
 ${4:$alpha})]]></content>  
  <tabTrigger>rgba</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - rgba() 컬러 함수</description>  
</snippet>
```



RED 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 RED 컬러 함수 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[ red(${1:$color}) ]]></content>  
  <tabTrigger>red</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - red() 컬러 함수</description>  
</snippet>
```



GREEN 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 GREEN 컬러 함수 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yamoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[green(${1:$color})]]></content>  
  <tabTrigger>green</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - green() 컬러 함수</description>  
</snippet>
```



BLUE 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 BLUE 컬러 함수 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yamoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[blue(${1:$color})]]></content>  
  <tabTrigger>blue</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - blue() 컬러 함수</description>  
</snippet>
```



MIX 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 MIX 컬러 함수 스니펫을 만들어 볼까요?,

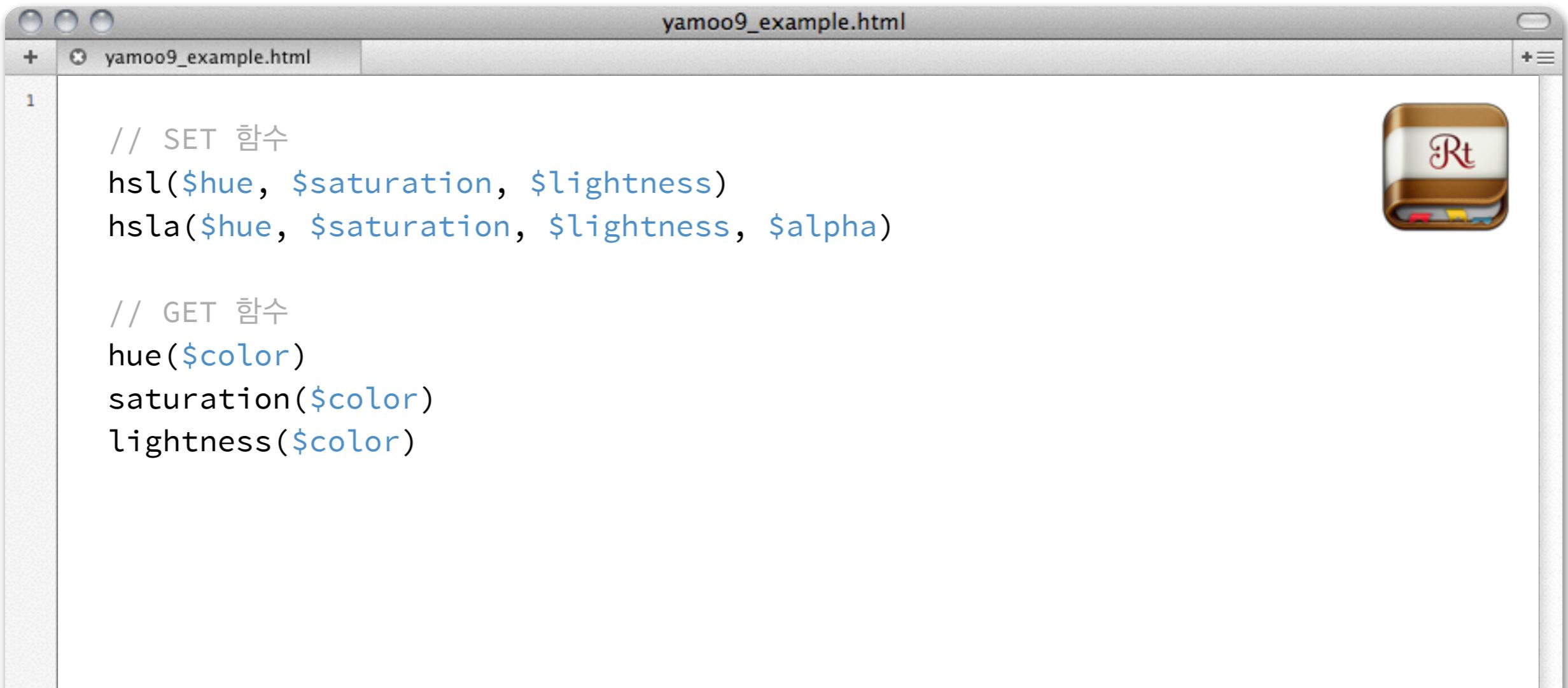
```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[mix(${1:$color1}, ${2:$color2}, ${3:$weight:50%})  
 ]]></content>  
  <tabTrigger>mix</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - mix 컬러 함수</description>  
</snippet>
```



컬러 함수 (Color Functions)

HSLA 함수



The screenshot shows a code editor window titled "yamoo9_example.html". The code is written in Sass and defines HSLA color functions:

```
// SET 함수
hsl($hue, $saturation, $lightness)
hsla($hue, $saturation, $lightness, $alpha)

// GET 함수
hue($color)
saturation($color)
lightness($color)
```

A small icon of a book with the letters "Rt" is visible in the top right corner of the editor window.



HSL 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 HSL 컬러 함수 스니펫을 만들어 볼까요?,

```
1 <snippet>
  <content><! [CDATA[hsl(${1:$hue}, ${2:$saturation}, ${3:$lightness})]></content>
  <tabTrigger>hsl</tabTrigger>
  <scope>source.scss, source.sass</scope>
  <description>SASS - hsl() 컬러 함수</description>
</snippet>
```



HSLA 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 HSLA 컬러 함수 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[hsla(${1:$hue}, ${2:$saturation}, ${3:$lightness},  
 ${4:$alpha})]]></content>  
  <tabTrigger>hsla</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - hsla() 컬러 함수</description>  
</snippet>
```



HUE 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 HUE 컬러 함수 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[hue(${1:$color})]]></content>  
  <tabTrigger>hue</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - hue 컬러 함수</description>  
</snippet>
```



SATURATION 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 SATURATION 컬러 함수 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[saturation(${1:$color})]]></content>  
  <tabTrigger>saturation</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - saturation 컬러 함수</description>  
</snippet>
```



LIGHTNESS 컬러 함수 스니펫 만들기

Sublime Text에서 활용 가능한 LIGHTNESS 컬러 함수 스니펫을 만들어 볼까요?,

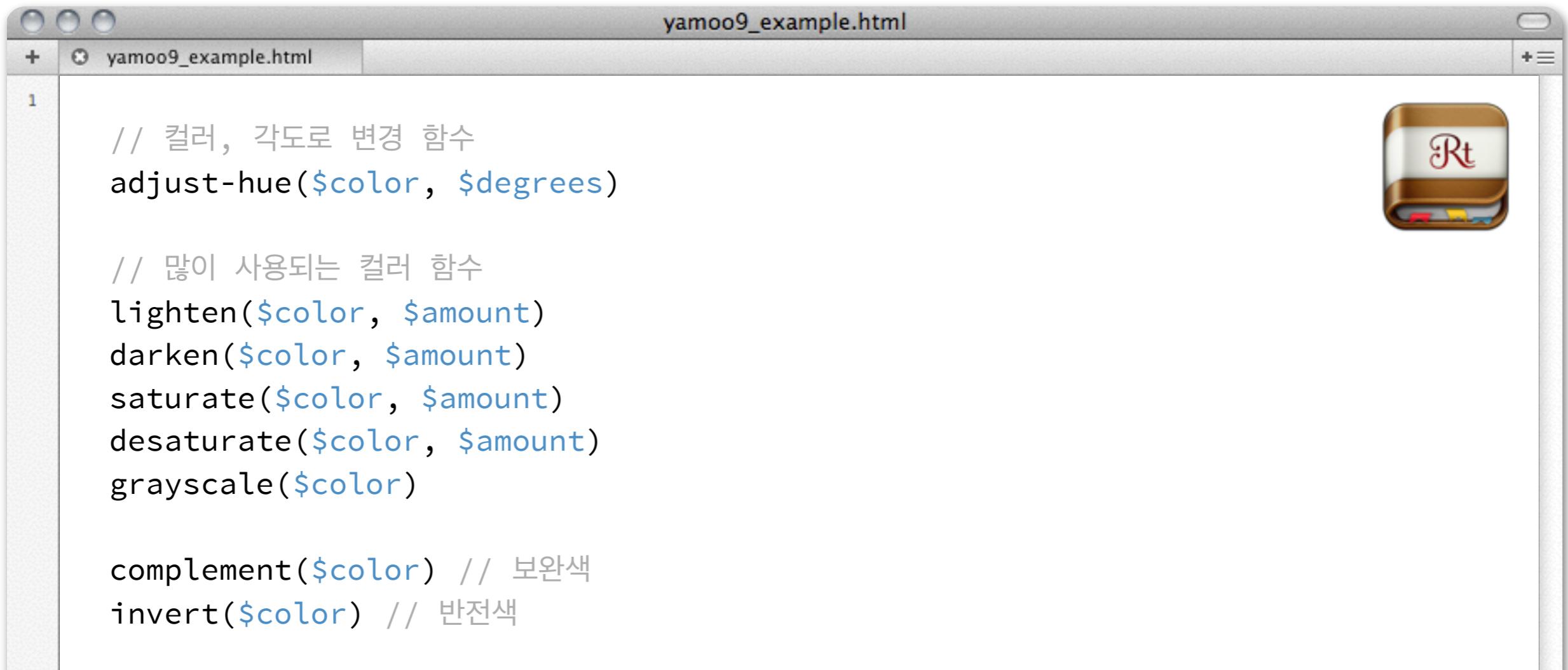
```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[lightness(${1:$color})]]></content>  
  <tabTrigger>lightness</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - lightness 컬러 함수</description>  
</snippet>
```



컬러 함수 (Color Functions)

HSLA 함수



The screenshot shows a web browser window with a code editor. The title bar says "yamoo9_example.html". The code editor contains the following Sass code:

```
// 컬러, 각도로 변경 함수
adjust-hue($color, $degrees)

// 많이 사용되는 컬러 함수
lighten($color, $amount)
darken($color, $amount)
saturate($color, $amount)
desaturate($color, $amount)
grayscale($color)

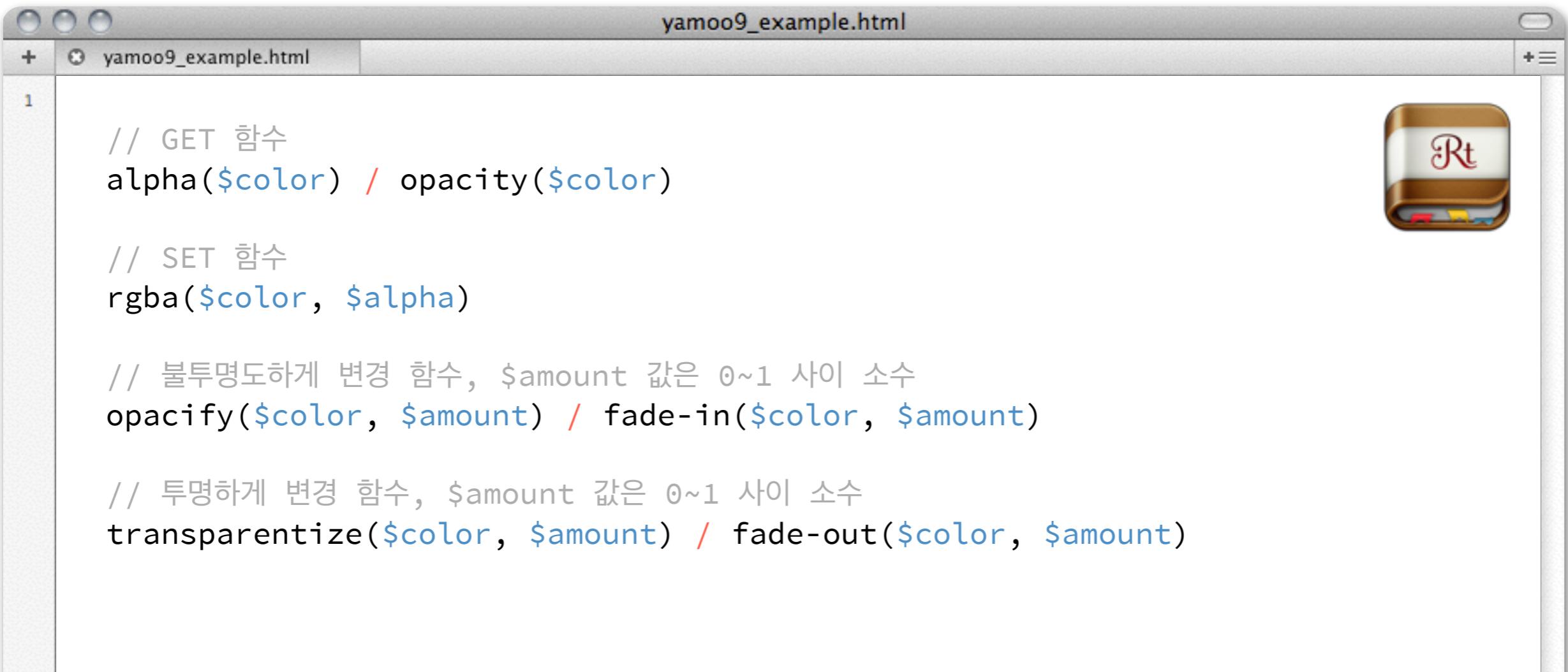
complement($color) // 보완색
invert($color) // 반전색
```

A small icon of a book with the letters "Rt" on it is visible in the bottom right corner of the code editor.



컬러 함수 (Color Functions)

Opacity 함수



A screenshot of a code editor window titled "yamoo9_example.html". The code editor shows the following Sass code:

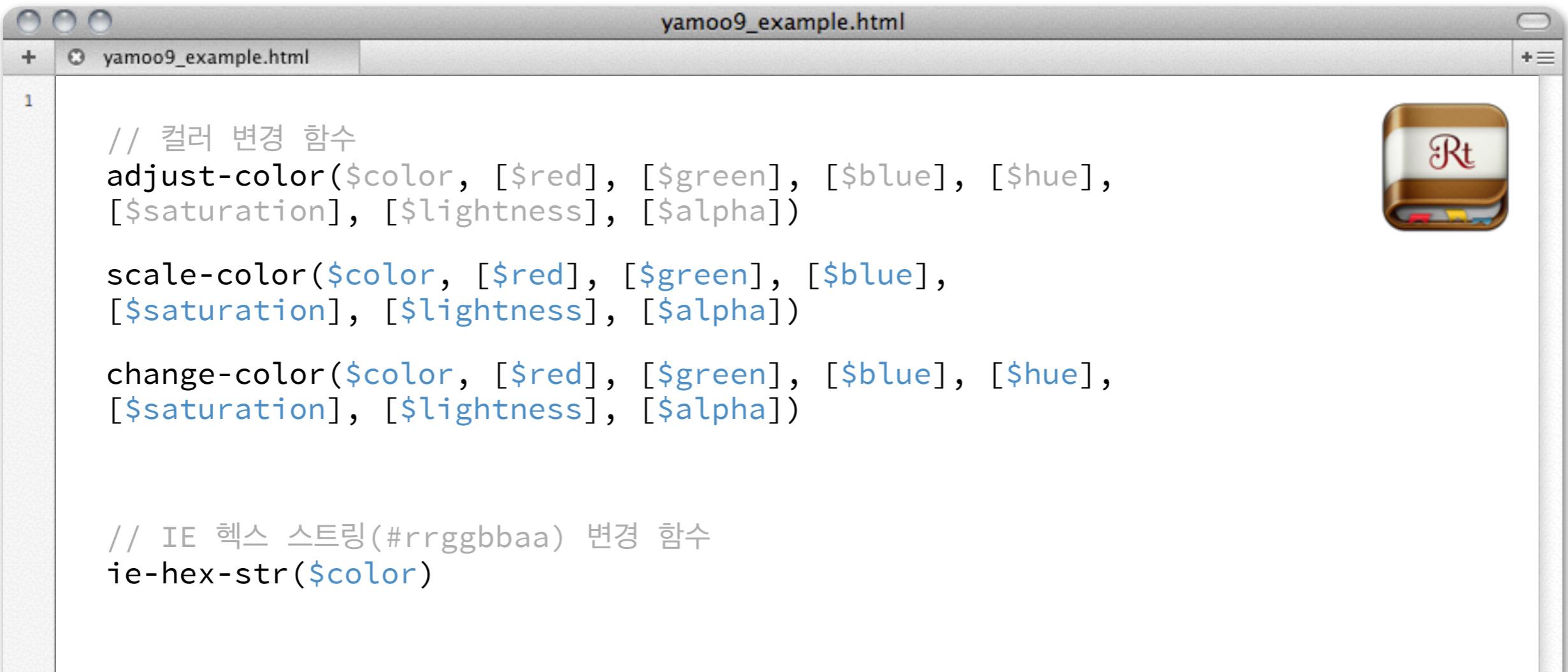
```
// GET 함수  
alpha($color) / opacity($color)  
  
// SET 함수  
rgba($color, $alpha)  
  
// 불투명도하게 변경 함수, $amount 값은 0~1 사이 소수  
opacify($color, $amount) / fade-in($color, $amount)  
  
// 투명하게 변경 함수, $amount 값은 0~1 사이 소수  
transparentize($color, $amount) / fade-out($color, $amount)
```

The code editor interface includes a tab bar with "yamoo9_example.html", a status bar with a file icon, and a small icon in the top right corner.



컬러 함수 (Color Functions)

Other Color 함수



The screenshot shows a browser window with the title "yamoo9_example.html". The code editor pane contains the following Sass code:

```
// 컬러 변경 함수
adjust-color($color, [$red], [$green], [$blue], [$hue],
[$saturation], [$lightness], [$alpha])

scale-color($color, [$red], [$green], [$blue],
[$saturation], [$lightness], [$alpha])

change-color($color, [$red], [$green], [$blue], [$hue],
[$saturation], [$lightness], [$alpha])

// IE 헥스 스트링(#rrggbbaa) 변경 함수
ie-hex-str($color)
```

A small icon of a book with the letters "Rt" is visible on the right side of the code editor.



수학 함수 (Number Functions)

Number 함수

yahoo9_example.html

```
// 퍼센트 변경 함수  
percentage(13/25) // 52%  
  
// 반올림 함수  
round(2.4) // 2  
  
// 올림 함수  
ceil(2.2) // 3  
  
// 내림 함수  
floor(2.6) // 2  
  
// 절대값 함수  
abs(-24) // 24
```

// 비교하여 작은것을 반환하는 함수
min(10px/12px) // 10px

// 비교하여 큰것을 반환하는 함수
max(10px/12px) // 12px

// 난수 함수
random(1) // 0~1



| | |
|---|--------------------------------|
| ► | hsl |
| ► | ie7 |
| ▼ | introspection |
| └ | comparable.sublime-snippet |
| └ | type-of.sublime-snippet |
| └ | unit.sublime-snippet |
| └ | unitless.sublime-snippet |
| ► | list |
| ▼ | miscellaneous |
| └ | if.sublime-snippet |
| ▼ | number |
| └ | abs.sublime-snippet |
| └ | ceil.sublime-snippet |
| └ | floor.sublime-snippet |
| └ | max.sublime-snippet |
| └ | min.sublime-snippet |
| └ | percentage.sublime-snippet |
| └ | round.sublime-snippet |
| ▼ | opacity |
| └ | alpha.sublime-snippet |
| └ | fade-in.sublime-snippet |
| └ | fade-out.sublime-snippet |
| └ | opacify.sublime-snippet |
| └ | rgba-short.sublime-snippet |
| └ | transparentize.sublime-snippet |
| ▼ | other |
| └ | adjust-color.sublime-snippet |
| └ | change-color.sublime-snippet |
| └ | ie-hex-str.sublime-snippet |
| └ | scale-color.sublime-snippet |
| ▼ | rgb |
| └ | blue.sublime-snippet |
| └ | green.sublime-snippet |
| └ | mix.sublime-snippet |
| └ | red.sublime-snippet |
| └ | rgb.sublime-snippet |
| └ | rgba.sublime-snippet |
| ▼ | string |
| └ | quote.sublime-snippet |
| └ | unquote.sublime-snippet |

Create Snippets



사용자 정의 함수(@function)

JS 함수와 거의 흡사한 함수로 @function으로 모듈을 정의한 후, 이름으로 호출할 수 있어 재 사용이 가능합니다.

yahoo9_example.html

```
// 변수 설정
$unit-width: 40px;
$gutter-width: 10px;

// grid-width 사용자 정의 함수(@function)
@function grid-width($n:1) {
    // 연산 결과 반환(@return)
    @return $n * $unit-width + ($n - 1) * $gutter-width;
}

#sidebar {
    // grid-width 함수 호출 결과 값 반환(전달인자 5)
    width: grid-width(5); // 5 * 40 + (5-1) * 10 = 240px
}
```

@return 키워드가 있어야 계산된 결과가 돌려집니다.

A red bracket points from the @return keyword in the function definition to the value being returned in the sidebar's width declaration. A purple arrow points from the sidebar's width declaration down to the value being returned in the function definition.



사용자 정의 함수(@function)

JS 함수와 거의 흡사한 함수로 @function으로 모듈을 정의한 후, 이름으로 호출할 수 있어 재 사용이 가능합니다.

```
yahoo9_example.html
```

```
+ yahoo9_example.html +≡  
1 // px 값을 em 단위로 변경하는 함수  
@function px2em($font_size, $base_font_size: 16) {  
  @return $font_size / $base_font_size + em;  
}  
  
body {  
  // 함수 호출  
  color: px2em(12, 20); // 12/20 + em = 0.6em  
}
```

2번째 전달인자는 부모요소의 폰트 사이즈를 변수로 처리



사용자 정의 함수 스니펫 만들기

Sublime Text에서 활용 가능한 사용자 정의 함수 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[ @function ${1:함수 이름} (${2:전달인자}) {  
    $3  
  }  
  $0 ]></content>  
  <tabTrigger>function</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - @function 문</description>  
</snippet>
```



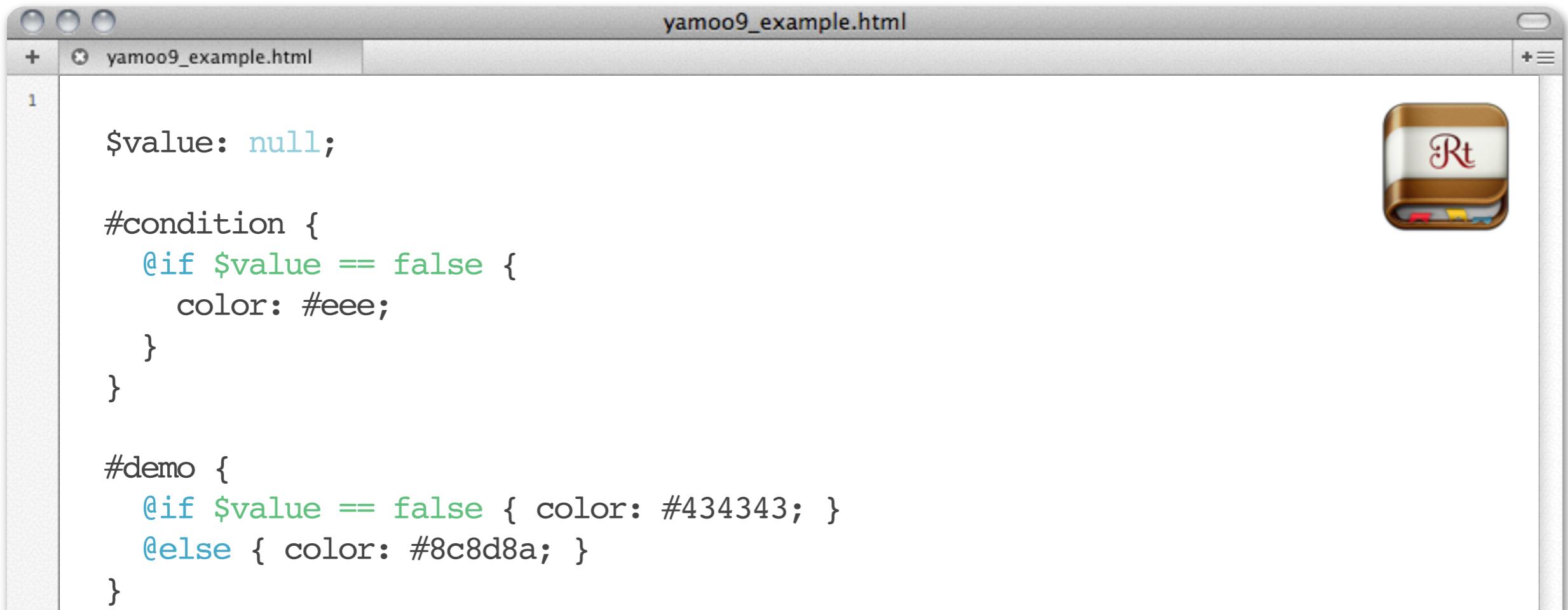
Conditions

`@if, @else if`



@if (Condition) @else if, @else

JS의 if ~ else 문과 유사한 조건문을 처리할 수 있습니다. (JS 코드에서 조건부분의 괄호가 빠집니다)



A screenshot of a Mac OS X-style code editor window titled "yamoo9_example.html". The file content is as follows:

```
1
$value: null;

#condition {
  @if $value == false {
    color: #eee;
  }
}

#demo {
  @if $value == false { color: #434343; }
  @else { color: #8c8d8a; }
}
```

The code defines a variable \$value and two CSS rules. The first rule, under the selector #condition, uses an @if directive to set the color to #eee if \$value is false. The second rule, under the selector #demo, uses an @if directive to set the color to #434343 if \$value is false, and #8c8d8a if it is not. A small icon of a book with the letters "Rt" is visible in the top right corner of the editor window.



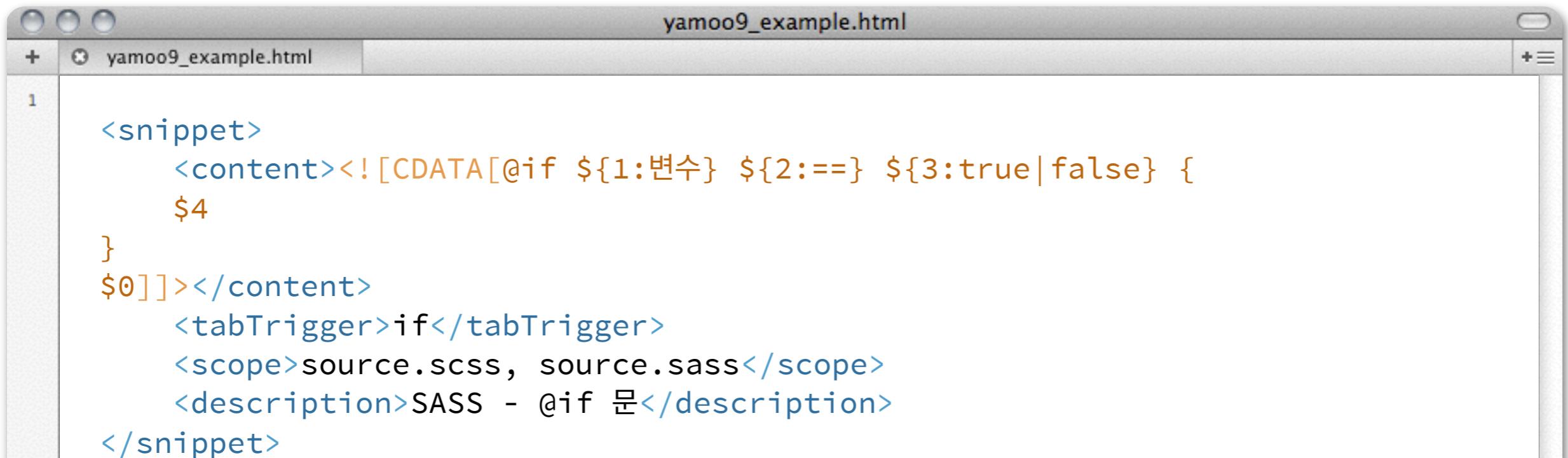
```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 // 커스텀 컬러 설정 변수  
$custom-red: #e44351;  
$custom-green: #3ce1cd;  
$custom-blue: #4524dd;  
$custom-dark: #161515;  
  
// 배경 컬러 설정 변수  
$bg-color: $custom-red;  
  
#page {  
  @if $bg-color == $custom-red {  
    color: invert($custom-red); }  
  @else if $bg-color == $custom-green {  
    color: desaturate(fade-out($custom-green, 20%), 30%); }  
  @else if $bg-color == $custom-blue {  
    color: lighten($custom-blue, 32%); }  
  @else {  
    color: white; }  
}
```



@if 스니펫 만들기

Sublime Text에서 활용 가능한 @if 스니펫을 만들어 볼까요?,



A screenshot of the Sublime Text code editor. The title bar says "yamoo9_example.html". The editor window contains the following XML snippet definition:

```
<snippet>
    <content><! [CDATA[@if ${1:변수} ${2:==} ${3:true|false} {
        $4
    }
    $0]]></content>
    <tabTrigger>if</tabTrigger>
    <scope>source.scss, source.sass</scope>
    <description>SASS - @if 문</description>
</snippet>
```



@if @else 스니펫 만들기

Sublime Text에서 활용 가능한 @if @else if 스니펫을 만들어 볼까요?,

```
yamoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[@if ${1:변수} ${2:==} ${3:true|false} {  
    $4  
  } @else {  
    $5  
  }  
$0]></content>  
  <tabTrigger>ifelse</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - @if @else 문</description>  
</snippet>
```



@if @else if 스니펫 만들기

Sublime Text에서 활용 가능한 @if @else if 스니펫을 만들어 볼까요?,

```
 1 <snippet>
  2   <content><! [CDATA[@if ${1:변수} ${2:==} ${3:true|false} {
  3     $4
  4   } @else if $5 {
  5     $6
  6   }${7: @else {
  7     $8
  8   }}${9:}
  9   $0]></content>
 10   <tabTrigger>ifelseif</tabTrigger>
 11   <scope>source.scss, source.sass</scope>
 12   <description>SASS - @if @else if 문</description>
 13 </snippet>
```



yahoo9_example.html

```
1

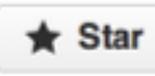
@mixin link-colors(
  $link: #3fb5c8,
  $visited: false,
  $hover: darken($link, 30%),
  $active: false,
  $focus: false
) {
  & { color: $link; }
  @if $visited {
    &:visited { color: $visited; } }
  @if $hover {
    &:hover { color: $hover; } }
  @if $active {
    &:active { color: $active; } }
  @if $focus {
    &:focus { color: $focus; } }
}

@include link-colors();
@include link-colors(#fe4940);
@include link-colors($hover: #fec9a0);
```

Create Adjust Color Contrast Function

텍스트, 배경 색상 대조를 조정해주는 사용자 정의 함수

```
// 텍스트, 배경 색상 대조를 조정해주는 사용자 정의 함수
@function text-contrast($bgcolor, $val: 70%) {
  $value: null;
  // 전달인자 유효성 검사
  @if type-of($bgcolor) != color {
    @warn "전달받은 $bgcolor 인자 값이 올바르지 않습니다.";
    @debug "$bgcolor의 값이 #{\$bgcolor} 입니다."
  // 전달받은 배경색 인자의 명도 값이 50%보다 클 경우,
  @if lightness($bgcolor) > 50% {
    $value: darken($bgcolor, $val); }
  // 전달받은 배경색 인자의 명도 값이 50%보다 작을 경우,
  @else {
    $value: lighten($bgcolor, $val); }
  // 결과 값 반환
  @return $value;
}
```



텍스트/배경 색상 대조를 조정하는 SASS 사용자 정의 함수

Gist Detail

Revisions 1

Download Gist

Clone this gist

<https://gist.github.com/yamoo9/ce4152a2899ff19d7d3f>

Embed this gist

<script src="https://gist.github.com/yamoo9/ce4152a2899ff19d7d3f">

Link to this gist

<https://gist.github.com/yamoo9/ce4152a2899ff19d7d3f>

_text-contrast-function.scss

Raw

```
1 // 텍스트, 배경 색상 대조를 조정해주는 사용자 정의 함수
2 @function text-contrast($bgcolor, $val: 70%) {
3   $value: null;
4   // 전달인자 유효성 검사
5   @if type-of($bgcolor) != color {
6     @warn "전달받은 $bgcolor 인자 값이 올바르지 않습니다.";
7     @debug "$bgcolor의 값이 #{$bgcolor} 입니다."
8   // 전달받은 배경색 인자의 명도 값이 50%보다 클 경우,
9   @if lightness($bgcolor) > 50% {
10     $value: darken($bgcolor, $val); }
11   // 전달받은 배경색 인자의 명도 값이 50%보다 작을 경우,
12   @else {
13     $value: lighten($bgcolor, $val); }
14   // 결과 값 반환
15   @return $value;
16 }
```



if() 함수 (Condition, true, false)

JS의 3항식 조건문과 유사하게 조건문을 처리할 수 있습니다.



A screenshot of a code editor window titled "yamoo9_example.html". The code editor shows the following Sass code:

```
1 // 컬러 변수
$main-bg: #000;

.main {
    // $main-bg 값이 black과 같다면,
    // #fff로 설정
    // 거짓이라면,
    // #000으로 설정
    color: if($main-bg == black, #fff, #000);
}
```

The code defines a variable \$main-bg set to #000. It then creates a .main selector. Inside the .main block, there is a series of comments explaining the logic of the if() function: if \$main-bg is equal to black, set the color to #fff; otherwise, set it to #000. The code editor interface includes a tab bar with "yamoo9_example.html", a toolbar icon resembling a book with "Rt" on it, and a status bar at the bottom.



if() 함수 스니펫 만들기

Sublime Text에서 활용 가능한 if() 함수 스니펫을 만들어 볼까요?,

```
1 <snippet>
  <content><! [CDATA[if(${1:$condition}, ${2:$if-true}, ${3:$if-false})]]>
</content>
  <tabTrigger>if</tabTrigger>
  <scope>source.scss, source.sass</scope>
  <description>SASS - if() 조건 함수</description>
</snippet>
```



Loops

@while, @for, @each



@while (iteration)

JS의 while문과 유사한 반복문을 처리할 수 있습니다. (JS 코드에서 조건부분의 괄호가 빠집니다)

```
yamoo9_example.html
```

```
+ ① yamoo9_example.html +≡  
1  
$i: 1;  
$gutter: 20px;  
  
@while $i <= 12 {  
  .grid-#{$i} {  
    width: 60px * $i + $gutter * ($i-1);  
  }  
  $i: $i + 1;  
}  
$i의 값이 변경됩니다. (조건 변화)  
조건에 변화가 없으면 '무한 반복'에 빠집니다.
```



@while 스니펫 만들기

Sublime Text에서 활용 가능한 @while 스니펫을 만들어 볼까요?,

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[ \${i:\${1:1}};  
@while \${i ${2:<=} ${3:12} {  
  \${4:.\${5:item-}\${6:#{\$i\}}}} {  
    \$7  
    \${8:\$i: \${i ${9: + 1}};}  
  }  
}></content>  
<tabTrigger>while</tabTrigger>  
<scope>source.scss, source.sass</scope>  
<description>SASS - @while 문</description>  
</snippet>
```



@for (iteration, from ~ [to, through])

JS의 for문과 유사한 반복문을 처리할 수 있습니다.

```
1 $total: 12;

@for $i from 1 to $total {
  .grid-#{$i} {
    width: 70px * $i;
  }
}

@for $i from 1 through $total {
  .grid-#{$i} {
    width: 70px * $i;
  }
}
```

Rt

to vs through

to: ~까지 (12 전까지)
through: ~끝까지 (12 끝까지)

```
36
37 // ((960 - 20) - (11*20) ) / 12
38 $i: 1
39 $page-width: 960px
40 $col: 60px
41 $gutter: 20px
42
43 .container
44   width: $page-width
45   margin:
46     left: $gutter/2
47     right: $gutter/2
48
49 .row
50   overflow: hidden
51
52 [class*=grid]
53   float: left
54
55 @for $i from 1 through 12
56   .grid-#${$i}
57     width: $col * $i + $gutter * ($i - 1)
58     @if ($i < 12)
59       margin-right: $gutter
60     @else
61       margin-right: 0
62   $i: $i + 1
63
64
65 .container {
66   width: 960px;
67   margin-left: 10px;
68   margin-right: 10px;
69 }
70
71 .row {
72   overflow: hidden;
73 }
74
75 [class*=grid] {
76   float: left;
77 }
78
79 .grid-1 {
80   width: 60px;
81   margin-right: 20px;
82 }
83
84 .grid-2 {
85   width: 140px;
86   margin-right: 20px;
87 }
88
89 .grid-3 {
90   width: 220px;
91   margin-right: 20px;
92 }
93
94 .grid-4 {
95   width: 300px;
96   margin-right: 20px;
97 }
98
99 .grid-5 {
100   width: 380px;
101   margin-right: 20px;
102 }
```



@for 스니펫 만들기

Sublime Text에서 활용 가능한 @for 스니펫을 만들어 볼까요?

```
yahoo9_example.html
```

```
+ ① yahoo9_example.html +≡  
1 <snippet>  
  <content><! [CDATA[@for \$i from ${1:1} ${2:through} ${3:10} {  
    $4  
  }  
  \$0]]></content>  
  <tabTrigger>for</tabTrigger>  
  <scope>source.scss, source.sass</scope>  
  <description>SASS - @for 문</description>  
</snippet>
```



@each (iteration, in ~ [list, map])

JS의 for~in문, forEach문과 유사한 반복문을 처리할 수 있습니다.

A screenshot of a code editor window titled "yamoo9_example.html". The code shown is:

```
1 @each $obj in phone, tablet, cup, mouse {  
  .item-#{$obj} {  
    background-image: url("img/#{$obj}.jpg");  
  }  
}
```

A yellow callout box highlights the line "@each \$obj in phone, tablet, cup, mouse {". A purple oval surrounds this line and the entire ".item-#{\$obj} {" block. A purple arrow points from the text "활용 가능합니다." (Available for use) to the start of the "@each" directive.

리스트(List) 데이터 유형에서
활용 가능합니다.



@each (iteration, in ~ [list, map])

JS의 for~in문, forEach문과 유사한 반복문을 처리할 수 있습니다.

yahoo9_example.html

```
1 @each $item in (h1: 2em, h2: 1.5em, h3: 1.2em) {  
  #{nth($item, 1)} {  
    font-size: nth($item, 2);  
  }  
}
```

맵(Map) 데이터 유형 사용시에는
nth()를 사용하여 보다 고급스러운
활용이 가능합니다.



@each (iteration, in ~ [list, map])

JS의 for~in문, forEach문과 유사한 반복문을 처리할 수 있습니다.

yahoo9_example.html

```
1 @each $header, $size in (h1: 2em, h2: 1.5em, h3: 1.2em) {  
  #{$header} {  
    font-size: $size;  
  }  
}
```

@each 구문에 변수를 추가하여
제 2항의 데이터를 처리할 수도 있습니다.



@each 스니펫 만들기

Sublime Text에서 활용 가능한 @each 스니펫을 만들어 볼까요?,

```
1 <snippet>
  <content><! [CDATA[@each \$\$1:var} in ${2:item1, item2, item3} {
    .#${\$\$1}\$3 {
      \$4
    }
  }$0]></content>
  <tabTrigger>each</tabTrigger>
  <scope>source.scss, source.sass</scope>
  <description>SASS - @each 문</description>
</snippet>
```

%IR, #{}, @each

이미지 대체기법, 보간법, @each 문을 활용한 아이콘 배치 순환 처리

```
// 이미지 대체기법
// http://nicolasgallagher.com/another-css-image-replacement-technique/
// https://github.com/h5bp/html5-boilerplate/commit/aa0396eae757c9e03dda4e463fb0d4db5a5f82d7
%ir {
    font: 0/0 a;
    color: transparent;
    text-shadow: none;
}

// SASS 리스트(Lists) 데이터 타입 선언
// 소셜 아이콘 이름을 담은 리스트를 담을 변수 $icons
$icons: (twitter, facebook, youtube, rss);

// SASS @each 문을 사용해 $icons 내부를 순환하여
// 클래스 선언 (문자 보간법 및 플레이스홀더 선택자 활용)
@each $icon in $icons {
    .#{$icon} {
        @extend %ir;
        background: url("../images/icons/#{$icons}.png") no-repeat;
    }
}
```

PUBLIC



yamoo9 / _ir+each.scss

Created 5 minutes ago



0

이미지대체기법 + SASS @each문 활용

Gist Detail

Revisions 1

 [Download Gist](#)

Clone this gist

[https://gist.github](https://gist.github.com)

Embed this gist

<script src="https:

Link to this gist

<https://gist.github>[_ir+each.scss](#)

Raw

```
1 // 이미지 대체기법
2 // http://nicolasgallagher.com/another-css-image-replacement-technique/
3 // https://github.com/h5bp/html5-boilerplate/commit/aa0396eae757c9e03dda4e463fb0d4db5a5f82d7
4 %ir {
5     font: 0/0 a;
6     color: transparent;
7     text-shadow: none;
8 }
9
10 // SASS 리스트(Lists) 데이터 타입 선언
11 // 소셜 아이콘 이름을 담은 리스트를 담을 변수 $icons
12 $icons: (twitter, facebook, youtube, rss);
13
14 // SASS @each 문을 사용해 $icons 내부를 순환하여
15 // 클래스 선언 (문자 보간법 및 플레이스홀더 선택자 활용)
16 @each $icon in $icons {
17     .#$icon {
18         @extend %ir;
19         background: url("../images/icons/#{$icons}.png") no-repeat;
20     }
21 }
```