# 1. Giới thiệu:

CSS processor: công cụ để tạo ra các tập tin css nhanh hơn thông qua một ngôn ngữ khác (scss, less, compass,).

SASS có hai định dạng file là `*.sass` và `*.scss`. Và cách viết của hai định dạng này cũng là khác nhau (nhưng các control directive, function thì có cùng một ý nghĩa). Và nếu bạn biết một cái, thì bạn cũng có thể viết được cái thứ hai. Điểm qua một số sự khác biệt trong cách viết (mà mình biết) của hai định dạng file này nhé:

- `*.sass`:
  - Sử dụng indent để thể hiện quy tắc xếp chồng (nested rules)
  - Không cần sử dụng `;` khi kết thúc một property
  - Khai báo mixins bằng ký tự `=`
  - Sử dụng mixins bằng ký tự `+`
- `*.scss`:
  - Sử dụng dấu `{` và `}` để thể hiện quy tắc xếp chồng (nested rules)
  - Sử dụng `;` để kết thúc một property
  - Khai báo mixins bằng directive `@mixin`
  - Sử dụng mixins bằng directive `@include`

```
$vendorPrefixes: (-webkit-, -moz-, -khtml-, -o-, -ms-)

=css3-prefix($property, $value...)
    @each $vendorPrefix in $vendorPrefixes
        #{$vendorPrefix}#{$property}: unquote($value)
    #{$property}: unquote($value)

body
    .box
        +css3-prefix(border-radius, 3px)
```

```
$vendorPrefixes: (-webkit-, -moz-, -khtml-, -o-, -ms-);

@mixin css3-prefix($property, $value...) {
    @each $vendorPrefix in $vendorPrefixes {
        #{$vendorPrefix}#{$property}: unquote($value);
    }
    #{$property}: unquote($value);
}

body {
    .box {
        @include css3-prefix(border-radius, 4px);
    }
}
```

## 2. Lợi ích:

- Sử dụng các biến như các ngôn ngữ lập trình, dễ chỉnh sửa. Ví dụ như cài đặt các biến chứa màu sắc để sử dụng, khi chỉnh sửa màu chỉ cần chỉnh sửa giá trị biến này và biên dịch lại
- Tiết kiệm thời gian
- Các đoạn CSS giống nhau (Code Block) có thể được gom nhóm và quản lý, tái sử dụng
- Hỗ trợ cách biểu diễn màu, thuộc tính cho tất cả trình duyệt
- Xây dựng các Mixin có thể truyền tham số tương tự như hàm trong ngôn ngữ lập trình

## 3. Chuẩn bị:

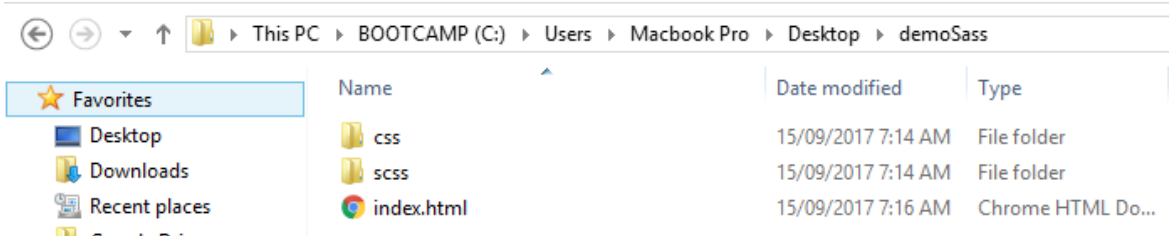
- Cài đặt Koala, download tại: http://koala-app.com/
- Cài đặt Sublime text

Tạo project và tổ chức project

- Tạo thư mục chứa Website
- Tạo các tập tin và thư mục với cấu trúc hợp lý
    - Thư mục chứa các tập tin HTML dựa trên chức năng, đối tượng sử dụng, … (Ví dụ như thư mục administrator chứa các tập tin HTML sử dụng cho người quản trị, products chức các tập tin HTML xử lí liên quan đến sản phẩm, …)
    - Thư mục chứa các tài nguyên của website như hình ảnh, audio, video, tập tin khác. Có thể phân chia thư mục loại này theo ý nghĩa, mục đích sử dụng, …
    - Thư mục chứa tập tin javascript
    - Thư mục chứa tập tin css
- Khởi động Koala và chọn các tập tin scss và css tương ứng

## 4. Ví dụ đầu tiên

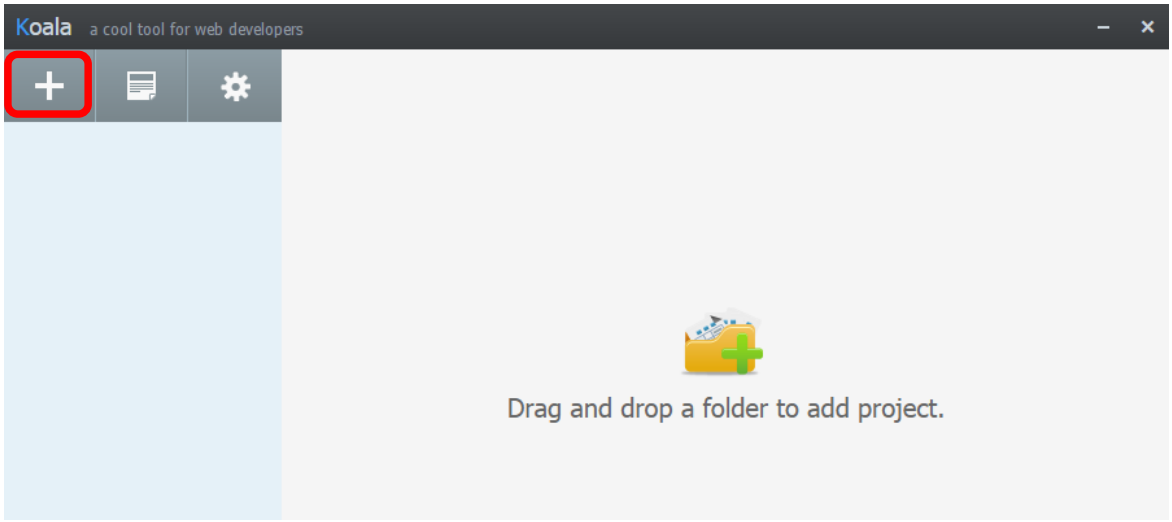
Tạo cấu trúc thư mục như sau

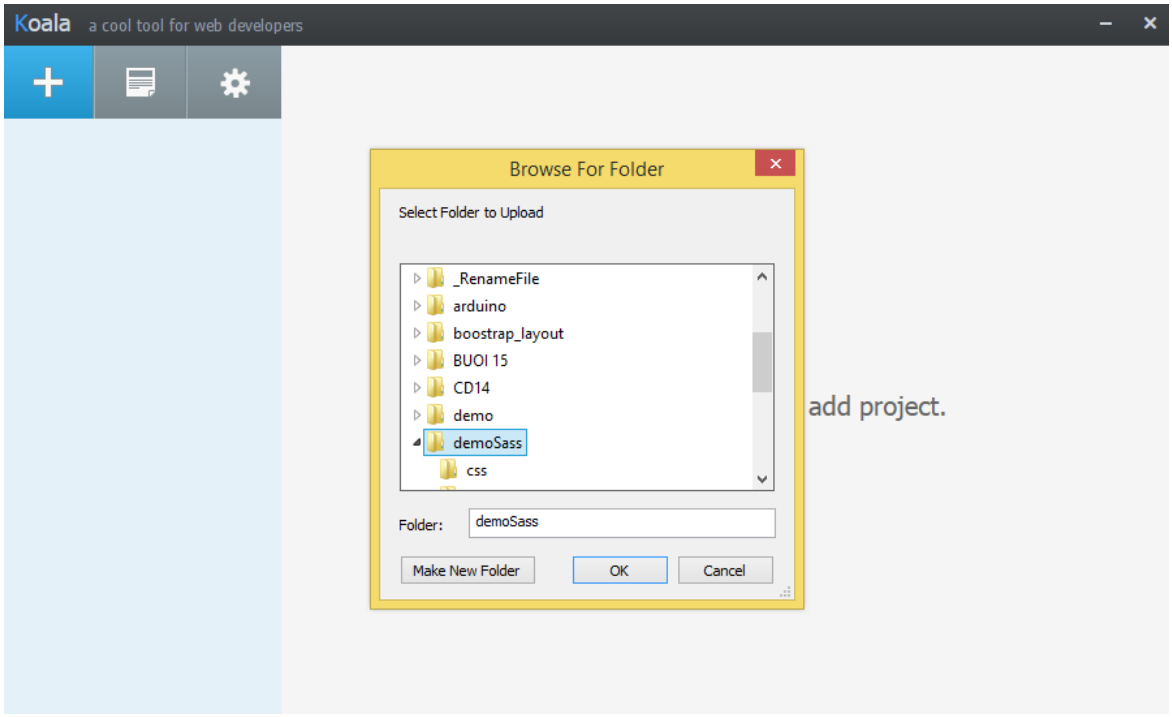


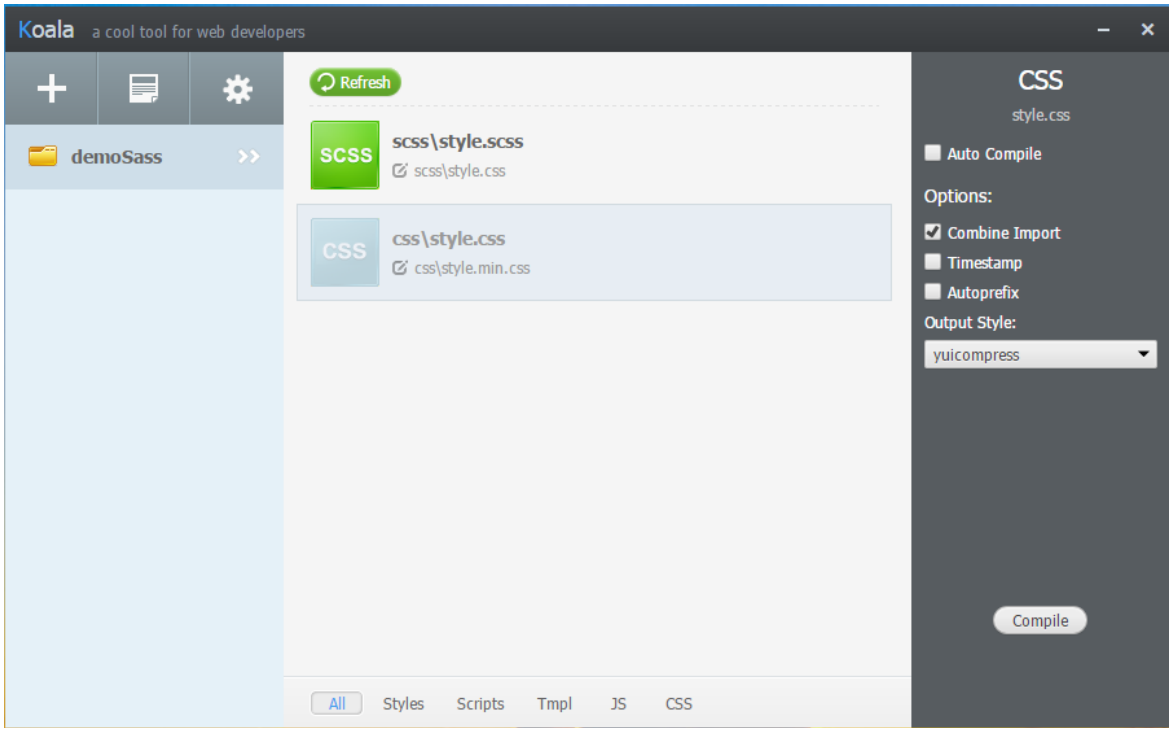

Thư mục scss chứa tập tin scss sử dụng cho việc biên dịch thành tập tin css đặt trong thư mục css.

Tạo tập tin style.scss trong thư mục scss và style.css trong thư mục css. Mở các tập tin html, css, scss sử dụng sublime text.

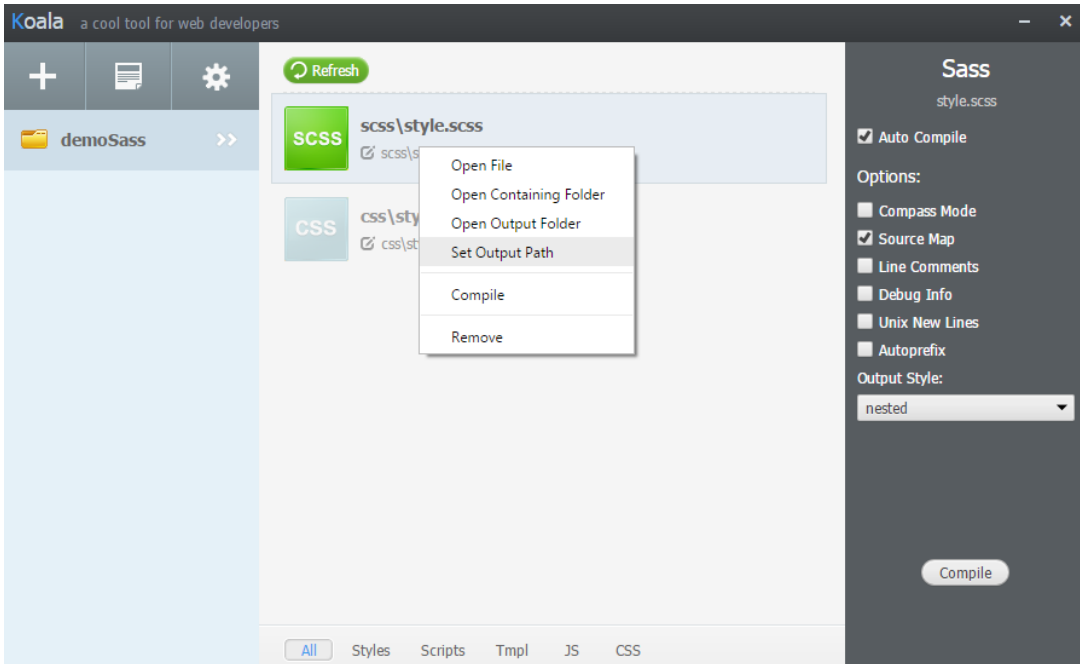
Mở chương trình Koala và thêm vào 1 project mới (chọn thư mục chứa Website, ở đây là demoscss)

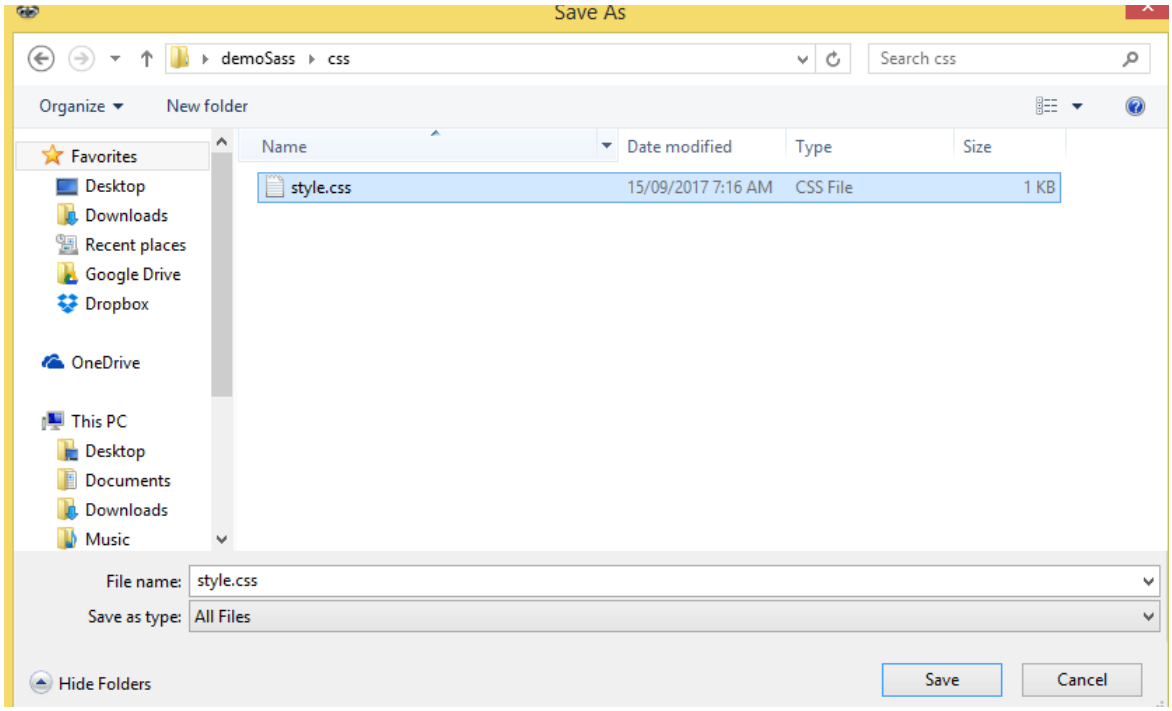





Thay đổi đường dẫn file biên dịch kết quả




Soạn thảo tài liệu HTML như sau:

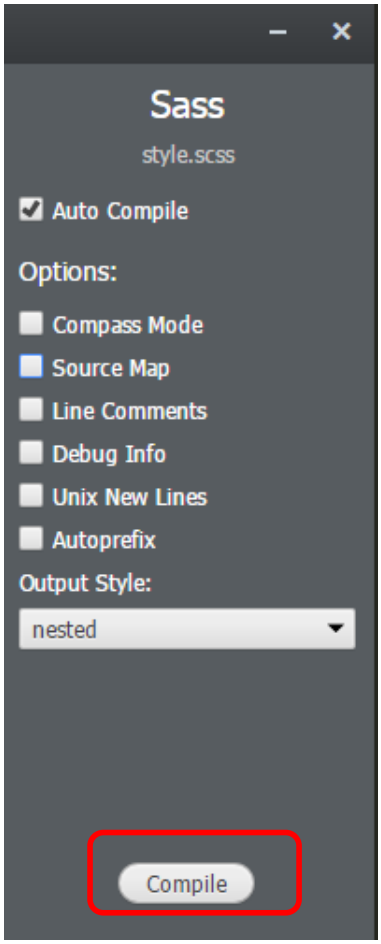
```
<div class="header">
     <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit, sed do eiusmod
```

```
                tempor incididunt ut labore et dolore magna
aliqua. Ut enim ad minim veniam,
                quis nostrud exercitation ullamco laboris nisi ut
aliquip ex ea commodo
                consequat. Duis aute irure dolor in reprehenderit
in voluptate velit esse
                cillum dolore eu fugiat nulla pariatur. Excepteur
sint occaecat cupidatat non
                proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.
     </p>
</div>
```

Viết mã cho trang scss:

```
$fontColor:red;
p{
     color:$fontColor;
}
```

Tiến hành biên dịch (compile):




Kết quả biên dịch:

```
p {
  color: red; }
```

## 5. Biến – Kiểu dữ liệu

Biến là cách mà chúng ta có thể khai báo một giá trị nào đó mà chúng ta đã xác định được sẽ dùng nó nhiều lần, chẳng hạn như các mã màu, giá trị border, shadows,….Để khai báo một biến, chúng ta sẽ viết dấu **$** đằng trước tên biến như thế này.

Khai báo biến và gán giá trị:

| |
|---|
| **$<*Tên biến*>:giá trị;** |

Lưu ý: Các biến được hiểu trong một block và các block con.

Các kiểu dữ liệu tương ứng các độ đo của css:

| |
|---|
| Số thể thiện kích thước tỉ lệ (2.0,15,12px) |
| Chuỗi ("foo",bar) |
| Màu sắc (#f00, rgb(255,255,150)) |
| Luận lý (true, false) |
| Chuỗi kết hợp các độ đo (10px 20px, 'Tahoma','Arial',san-serif) |

# BÍ QUYẾT: CÁCH THỨC ĐẶT TÊN BIẾN

* Tên biến phù hợp với ngữ cảnh (section, tiêu đề, box,…)

* Gạch ngang ở giữa

* Vị trí áp dụng : bottom, right, left,…

* thuộc tính áp dụng hoặc là màu…

```
$title-font: normal 24px/1.5 'Open Sans', sans-serif;
$cool-red: #F44336;
$box-shadow-bottom-only: 0 2px 1px 0 rgba(0, 0, 0, 0.2);

h1.title {
  font: $title-font;
  color: $cool-red;
}

div.container {
  color: $cool-red;
  background: #fff;
  width: 100%;
  box-shadow: $box-shadow-bottom-only;
}
```

```
h1.title {
  font: normal 24px/1.5 "Open Sans", sans-serif;
  color: #F44336;
}

div.container {
  color: #F44336;
  background: #fff;
  width: 100%;
  box-shadow: 0 2px 1px 0 rgba(0, 0, 0, 0.2);
}
```

## 6. Gom nhóm

**Ý nghĩa**: Gom nhóm định dạng các thành phần con bên trong 1 thành phần chứa

Cho đoạn HTML:

```
<section class="header">
     <h1>Lorem ipsum dolor sit amet</h1>
     <img src="xxx.png" alt="Mô tả">
```

```
    <p>
    Lorem ipsum dolor sit amet, consectetur adipisicing elit,
sed do eiusmod tempor incididunt ut labore et dolore magna
aliqua. Ut enim ad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo     consequat.
Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.
    </p>
</section>
```

**Mã scss:**

```
.header{
    h1{
        font-weight:bold;
    }
    p{
        line-height:1em;
    }
    img{
        float:left;
    }
}
```

**Mã css sau biên dịch:**

```
.header h1 {
  font-weight: bold;
}
.header p {
  line-height: 1em;
}
.header img {
  float: left;
}
```

# 7. Parent selector

**Ý nghĩa**: Định dạng cho các pseudo css

Thêm dấu **&** ở đằng trước một phần tử con nào, thì nó sẽ mang toàn bộ vùng chọn của một phần tử mẹ đứng trước nó một bậc

**Mã scss:**

```
.header{
```

```
     h1{
          font-weight:bold;
     }
     p{
          line-height:1em;
          &:hover {
               background-color:yellow
          }
     }
     img{
          float:left;
     }
}
```

**CSS sau biên dịch:**

```
.header h1 {
  font-weight: bold;
}

.header p {
  line-height: 1em;
}

.header p:hover {
    background-color: yellow;
}

.header img {
  float: left;
}
```

## 8. Gom nhóm thuộc tính

**Ý nghĩa**: gom nhóm các thuộc tính cũng loại (tiền tố)

```
.header{
     font:{
          family:Arial;
          style:italic;
     }
     h1{
          font-weight:bold;
     }
     p{
          line-height:1em;
```

```
            &:hover{
                background-color:yellow
            }
        }
        img{
            float:left;
        }
}
```

Kết quả:

```
.header {
  font-family: Arial;
  font-style: italic;
}

.header h1 {
    font-weight: bold;
}
.header p {
    line-height: 1em;
}
.header p:hover {
      background-color: yellow;
}
.header img {
    float: left;
}
```

## 9. Quy tắc Mixin – @mixin *tên_mix*

**Mixin** là một cơ chế khá phổ biến trong **SASS** mà nếu bạn biết cách áp dụng thì sẽ rất có lợi khi viết CSS. Công dụng của nó là mang nhiều thuộc tính mà bạn đã quy ước trong một **mix** nào đó bỏ vào một thành phần bất kỳ mà không cần phải viết lại các thuộc tính đó. Thuận tiện chỉnh sửa và tái sử dụng.

*Ví dụ, thường khi sử dụng thuộc tính float trong CSS thì chúng ta phải khai báo luôn margin như thế này*

```
.class_1 {

float: left;

margin: 0px 10px;

}
```

Bây giờ bạn có quá nhiều thành phần mà cần ***float*** trong HTML thì viết lại 2 dòng kia hoài cũng chán. Bây giờ để nhanh, chúng ta sẽ sử dụng cơ chế ***mixin*** trong SASS để giải quyết nó.
**Đầu tiên**: Tạo một mixin tên là ***float-left*** như sau.

```
@mixin float-left {

float: left;

margin: 0px 10px;

}
```

Sau đó, mình muốn cho một **thành phần nào đó** có **thuộc tính float: left** và **margin** như thế kia thì chỉ cần viết như sau. (**@include** *tên_mixin*)

```
.class { @include float-left;}
```

Nhưng bây giờ, bạn không muốn sử dụng **float: left mà là float:right** thì sao? Không lẽ lại tạo thêm một **mixin** nữa sao? Không cần, chúng ta có thể đặt thêm tham số cho cái **mixin** kia để chúng ta có thể thay đổi nó tùy vào thời điểm. Chúng ta sẽ sửa lại code **mixin** như sau **(Chú ý từ left chúng ta có thể đặt một từ khác cho phù hợp.). Nguyên tắc này giống truyền tham số vào hàm.**

```
@mixin float-left($float,$margin) {

float: $float;

margin: $margin;

}
```

Và khi đi include, chúng ta sẽ viết như sau

```
.class_1 {

@include float-left(right,5px 10px)

}
```

Ví dụ 2:

```
@mixin square($size, $color) {
  width: $size;
  height: $size;
  background-color: $color;
}

.small-blue-square {
  @include square(20px, rgb(0,0,255));
}

.big-red-square {
  @include square(300px, rgb(255,0,0));
}
```

```
.small-blue-square {
  width: 20px;
  height: 20px;
  background-color: blue;
}

.big-red-square {
  width: 300px;
  height: 300px;
  background-color: red;
}
```

## Ví dụ 3:

```
@mixin transform-tilt() {
  $tilt: rotate(15deg);

  -webkit-transform: $tilt; /* Ch <36, Saf 5.1+, iOS, An =<4.4.4 */
      -ms-transform: $tilt; /* IE 9 */
          transform: $tilt; /* IE 10, Fx 16+, Op 12.1+ */
}

.frame:hover {
  @include transform-tilt;
}
```

```
.frame:hover {
  -webkit-transform: rotate(15deg);  /* Ch <36, Saf 5.1+, iOS, An =<4.4.4 */
  -ms-transform: rotate(15deg);  /* IE 9 */
  transform: rotate(15deg);  /* IE 10, Fx 16+, Op 12.1+ */
}
```

## 10.Extends – Kế thừa @extend *tên_class*

Đây là một **tính năng quan trọng** mà bạn cần hiểu càng sớm càng tốt vì sau này bạn sẽ dùng rất nhiều, nhất là trong khi làm việc với một CSS Framework. Tính năng kế thừa này nghĩa là bạn chỉ định cho một thành phần nào đó thừa hưởng tất cả các thuộc tính của một class (hoặc vùng chọn nào đó) bất kỳ mà bạn đã khai báo sẵn.

```
.button_1 {
 -moz-box-shadow:inset 0px 1px 0px 0px #ffffff;
 -webkit-box-shadow:inset 0px 1px 0px 0px #ffffff;
 box-shadow:inset 0px 1px 0px 0px #ffffff;
 background:-webkit-gradient( linear, left top, left bottom, color-stop(0.0
 background:-moz-linear-gradient( center top, #ededed 5%, #dfdfdf 100% );
 filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#ededed',
 background-color:#ededed;
 -moz-border-radius:6px;
 -webkit-border-radius:6px;
 border-radius:6px;
 border:1px solid #dcdcdc;
 display:inline-block;
 color:#777777;
 font-family:arial;
 font-size:15px;
 font-weight:bold;
 padding:6px 24px;
 text-decoration:none;
 text-shadow:1px 1px 0px #ffffff;
}
```

Giả sử, bây giờ minh muốn cho một class nào trở thành một button giống bên trên thì sao? Như thường lệ, chúng ta sẽ tiến hành viết thêm một class kiểu như ***.button_1,.class_1***. Nhưng như thế sẽ khá rắc rối vì mình phải tiến hành đi tìm lại phần này và viết vào, hoặc sẽ mở code HTML ra và viết thêm một class cho nó, nhưng cũng chẳng khá hơn. Nhưng ở SASS, mình sẽ có thể làm nhanh gọn lẹ chỉ với duy nhất vài dòng ngắn ngủi thế này.

```
.button_2 { @extend .button_1; }
```

```
.button_1, .button_2 {
 -moz-box-shadow: inset 0px 1px 0px 0px #ffffff;
 -webkit-box-shadow: inset 0px 1px 0px 0px #ffffff;
 box-shadow: inset 0px 1px 0px 0px #ffffff;
 background: -webkit-gradient(linear, left top, left bottom, color-stop(0.0
 background: -moz-linear-gradient(center top, #ededed 5%, #dfdfdf 100%);
 filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#ededed'
 background-color: #ededed;
 -moz-border-radius: 6px;
 -webkit-border-radius: 6px;
 border-radius: 6px;
 border: 1px solid #dcdcdc;
 display: inline-block;
 color: #777777;
 font-family: arial;
 font-size: 15px;
 font-weight: bold;
 padding: 6px 24px;
 text-decoration: none;
 text-shadow: 1px 1px 0px #ffffff; }
```

**Ví dụ 2:**

```
.dialog-button {
  box-sizing: border-box;
  color: #ffffff;
  box-shadow: 0 1px 1px 0 rgba(0, 0, 0, 0.12);
  padding: 12px 40px;
  cursor: pointer;
}

.confirm {
  @extend .dialog-button;
  background-color: #87bae1;
  float: left;
}

.cancel {
  @extend .dialog-button;
  background-color: #e4749e;
  float: right;
}
```

```
.dialog-button, .confirm, .cancel {
  box-sizing: border-box;
  color: #ffffff;
  box-shadow: 0 1px 1px 0 rgba(0, 0, 0, 0.12);
  padding: 12px 40px;
  cursor: pointer;
}

.confirm {
  background-color: #87bae1;
  float: left;
}

.cancel {
  background-color: #e4749e;
  float: right;
}
```

## 11.Các toán tử

**Ý nghĩa**: Tính toán các độ đo. Sử dụng chia các thành phần con

Cộng (+), trừ (-), nhân (*), chia (/)

Ví dụ:

```
$container:100%;

$contentArea:50%;

$gutter:10%;

.sidebar{

     width: $container - ($contentArea+$gutter);

}
```

## 12.Cấu trúc điều khiển

**Cú pháp:**

```
@if ĐK{

}@else{

}
```

Hoặc

```
@if ĐK{

}@else if ĐK{

}
```

**Ví dụ:**

```
$blockColor: green;
.container {
     @if $blockColor == red {
          background-color: #f00;
     } @else if $blockColor == green {
          background-color: #0f0;
     } @else if $blockColor == blue {
```

```
            background-color: #00f;
        } @else {
            background-color: #000;
        }
}
```

# 13.Cấu trúc lặp

**Cú pháp for:**

**@for** $counter **from** <*start_value*> **through** <*end_value*>{

}

**hay**

**@for** $counter **from** <*start_value*> **to** <*end_value*>{

}

(through: lặp đến cả end_value, to: lặp đến <end_value)

Mã scss:

```
@for $size from 1 through 3 {
   .header-#{$size}
   { font-size: 2em * $size; }
}
```

Kết quả biên dịch:

```
.header-1 {
font-size: 2em;
}
.header-2 {
font-size: 4em;
}
.header-3 {
font-size: 6em;
}
```

**Cú pháp @each:**

@each $<Phần tử> in $<tập hợp>{

}

Mã scss:

```
$socials: twitter linkedin pinterest;
@each $social in $socials {
     .#{$social} {
     background-image: url(/images/icons/#{$social}.png);
     width:16px;height:16px;
     }
}
```

Kết quả dịch:

```
.twitter {
     background-image: url(/images/icons/twitter.png);
     width: 16px;
     height: 16px;
}
.linkedin {
     background-image: url(/images/icons/linkedin.png);
     width: 16px;
     height: 16px;
}
.pinterest {
     background-image: url(/images/icons/pinterest.png);
     width: 16px;
     height: 16px;
}
```

Mã scss:

```
@each $notification, $color in (success, green),(warning,
amber),(error,red) {
     .#{$notification}-icon {
          background-color: $color;
     }
}
```

Kết quả dịch:

```
.success-icon {
     background-color: green;
}
.warning-icon {
     background-color: amber;
}
.error-icon {
     background-color: red;
}
```

**Cú pháp @while:**

```
@while ĐK {

}
```

Mã scss:

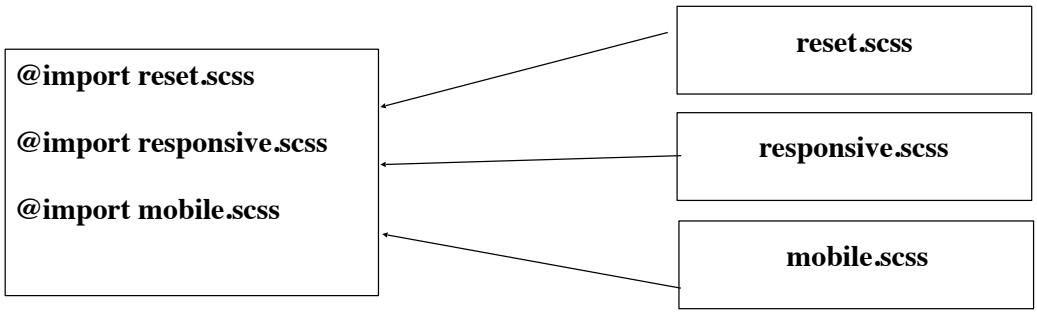
```
$col: 4;
@while $col > 0 {
     .cols-#{$col} {
     width: 100% / $col;
     }
     $col: $col - 1;
}
```

# 14.Liên kết, xây dựng cấu trúc SCSS

## 14.1.@import

**Ý nghĩa:** Hỗ trợ phân chia scss thành các tập tin tiện cho việc quản lý, làm việc nhóm nhưng chỉ tạo ra một file css duy nhất.

Ví dụ:




**Lưu ý:** *Nếu muốn trình biên dịch biên dịch thành tập tin css riêng biệt như scss thì tên tập tin sử dụng dấu _<Tên tập tin>.scss và trong mã scss vẫn để @import <Tên tập tin>.scss.*

**Gom nhóm với @import:**

Ví dụ: Tạo file _parents.scss với đoạn mã sau:

```
.p{
     background-color:#CCC;
}
```

Tạo file child.scss sử dụng phần định dạng trên

```
.child{
     @import "parent";
}
```

Ngoài ra có thể import trực tiếp tập tin css.

## 14.2.@media

**Ý nghĩa:** Chỉnh sửa css cho các kích thước màn hình khác nhau

Ví dụ:

Mã scss:

```
#container{
     width:960px;
     @media screen and (min-width:500px){
          width:100%;
     }
}
```

Kết quả biên dịch:

```
#container {
     width: 960px;
}
@media screen and (min-width: 500px) {
     #container {
          width: 100%;
     }
}
```

Mã scss:

```
@media screen {
     #container {
          @media (min-width:500px) {
               width: 500px;
          }
          @media (min-width:700px) {
               width: 700px;
          }
     }
}
```

Kết quả biên dịch:

```
@media screen and (min-width: 500px) {
     #container {
```

```
          width: 500px;
     }
}
@media screen and (min-width: 700px) {
     #container {
          width: 700px;
     }
}
```

Mixin và media

```
$color1:#CCC;
$sm:320px;
$md:720px;
$lg:1280px;
@mixin setQuery($type){
     @if $type == sm {
          @media only screen and (min-width: $sm) and (max-
width: $md                - 1)
          {
               @content;
          }
     }
     @else if $type == md {
          @media only screen and (min-width: $md) and (max-
width: $lg                - 1)
          {
               @content;
          }
     }
     @else if $type == lg {
          @media only screen and (min-width: $lg){
               @content
          }
     }
}
.container {
     color: $color1;
     @include setQuery(sm) {
          width: 80%;
     }
     @include setQuery(md) {
          width: 70%;
     }
     @include setQuery(lg) {
          width: 60%;
     }
}
```

## 15. Tổ chức dự án - cấu trúc dự án với SASS:

- Một trong những lợi ích khi sử dụng CSS preprocessor là khả năng chia code ra thành nhiều file giúp bạn dễ quản lý vào bảo trì code nhưng không ảnh hưởng nhiều đến hiệu suất của web.
- Nhờ có @import của Sass mà bạn có thể có bao nhiêu file tùy ý cho môi trường phát triển, và sau khi biên dịch tất cả sẽ được gom lại thành một file duy nhất.
- "Mọi thứ cần ở đúng nơi của nó, ở mọi nơi đều có đúng những thứ mà nó thuộc về"

https://scotch.io/tutorials/aesthetic-sass-1-architecture-and-style-organization

http://www.ff-team.com/tao-cau-truc-co-ban-cho-mot-du-an-sass/

http://thesassway.com/beginner/how-to-structure-a-sass-project

https://webstandardssherpa.com/reviews/sass-for-big-sites-part-1/