ゼロからはじめる Sass

2020 年 11 月 29 日

合同会社 Studio-INT

前畑　咲奈

目　次

1. Sass とは

Sass とは「Syntactically Awesome StyleSheet」の略。

CSS のメタ言語で、CSS をさらに拡張してより効率的に書けるようにしたものです。

メリット

・作業効率がアップする
　①変数で値を使いまわすことができる
　②四則演算ができる
　③関数が使える
　④セレクタやプロパティのネストで記述量が減る
　⑤一度定義したスタイルのセットを簡単に呼び出せる

デメリット

①コンパイルが必要で扱いづらい
②Sass、Ruby（Windows）をインストールしないと
　使えない
③学習コストがかかる
④チーム単位で使用しないと意味がない

2. 環境構築

1. Sass、Ruby（Windows）のインストール

Mac 環境に Sass をインストールする | Web 制作者のための Sass の教科書 - 公式サポートサイト
https://book.scss.jp/code/c2/02.html

Windows に Sass を導入する手順【インストール】と【コンパイル】 | B-side Journal
https://bsj-k.com/sass-installation-windows/

2. コンパイラを用意する

■エディタ

・Visual Studio Code

（拡張機能：Live Sass Compiler、Watch Sass）

・Dreamweaver

■タスクランナー

・Gulp

コンパイル形式：

■expanded：

```
ul {
    display: flex;
}
ul li {
    flex-basis: 20%;
}
```

■compressed：

```
ul{display:flex}ul
li{flex-basis:20%}
```

ファイルサイズを減らして
パフォーマンスを
向上させることができる！

3. Sass の書き方

1. 2 つの記法

SCSS 記法が一般的

SASS 記法（拡張子：○○.sass）

■コンパイル前

```
ul
        display: flex;
        li
                flex-basis: 20%;
```

■コンパイル後

```
ul {
    display: flex;
}
ul li {
    flex-basis: 20%;
}
```

SCSS 記法（拡張子：○○.scss）

■コンパイル前

```
ul{
        display: flex;
        li{
                flex-basis: 20%;
        }
}
```

■コンパイル後

```
ul {
    display: flex;
}
ul li {
    flex-basis: 20%;
}
```

4. 実際に書いてみよう！

変数の使用

■Sass

```
$main-color: #214e38;

.notes {
        color: $main-color;
}
.notesBox {
        border: 2px double $main-color;
}
```

■CSS

```
.notes {
    color: #214e38;
}

.notesBox {
    border: 2px double #214e38;
}
```

4. 実際に書いてみよう！

四則演算

■Sass

```
@for $i from 0 through 2 {
	.mt#{$i * 5} {
		margin-top:#{$i * 5}px;
	}
}
```

■CSS

```
.mt0 {
    margin-top: 0px;
}
.mt5 {
    margin-top: 5px;
}
.mt10 {
    margin-top: 10px;
}
```

4. 実際に書いてみよう！

関数の使用

■Sass

```
$width: 100%;

.item-boxA {
        width: $width / 3;
}
.item-boxB {
        width: round($width / 3);
}
```

■CSS

```
.item-boxA {
    width: 33.3333333333%;
}

.item-boxB {
    width: 33%;
}
```

4. 実際に書いてみよう！

一度定義したスタイルのセットを呼び出す（mixin）

■Sass

```
// mixin を定義
@mixin kadomaru($value) {
        border-radius: $value;
        color: #555555;
}

.box {
        @include kadomaru(3px);
        background-color: #efefef;
}
.item {
        @include kadomaru(20px);
        width: 33%;
}
```

■CSS

```
.box {
    border-radius: 3px;
    color: #555555;
    background-color: #efefef;
}

.item {
    border-radius: 20px;
    color: #555555;
    width: 33%;
}
```

5. おわりに

1. 実際の使用頻度

それほど多くない印象。

ただし、コーディングガイドラインに必須条件とされている案件もありました。

2. まずは CSS の形で書けるように

Sass の習得に時間を取られるようでは本末転倒だと思います。

まずは CSS の形式でコーディングできるようになることが先決。

6. 参考サイト

【CSS】Sass は絶対使った方が良いよ！使い方入門編 | WEBDESIGNDAY
https://webdesignday.jp/inspiration/technique/css/5819/

Web 制作者のための Sass の教科書 - 公式サポートサイト
https://book.scss.jp/