

# Sass(Syntactically Awesome StyleSheets)

SonicGarden maedana

# Agenda

Sassの特徴と簡単な使い方

基本文法

定数

Mixins

その他Tips

まとめ

# Sassの特徴と簡単な使い方

# Sassとは

Syntactically Awesome StyleSheets

=> イケてるスタイルシート(構文的な意味で

平たくいうとcssを生成するためのメタ言語

具体的には以下のようなもの

```
% cat sample1.sass
```

```
.link  
  color: #0080DD
```

```
.link:hover  
  color: blue
```

↑がsass形式で書いたファイル。

```
% sass sample1.sass -t compressed
```

```
.link{color:#0080DD}.link:hover{color:blue}
```

↑sassをインストールすると使えるコマンドでcssを生成。

# 特徴

- シンプルな記法でcssを構造化して表現できる
- 通常のcss無い「定数」や、「Mixins」等の非常に強力な機能を使える。
- 出力形式が柔軟(nested, expanded, compact, compressed)
- RailsやMerbとの連携が容易に出来る。

# インストール

```
gem install haml
```

インストールするとsassコマンドが使えるようになる。

# sassコマンドの使い方

```
% sass -h
```

```
Usage: sass [options] [INPUT] [OUTPUT]
```

## Description:

Uses the Sass engine to parse the specified template and outputs the result to the specified file.

## Options:

`--rails RAILS_DIR` Install Haml and Sass from the Gem to a

Rails project

`-c, --check` Just check syntax, don't evaluate.

`-s, --stdin` Read input from standard input instead of an

input file

`--trace` Show a full traceback on error

`-t, --style NAME` Output style. Can be nested (default),

compact, compressed, or expanded.

`-?, -h, --help` Show this message

`-v, --version` Print version

とっても簡単。-tオプションで出力形式を切り替え可能。

# 選べる出力形式

nested(default)

```
% sass sample1.sass -t nested
```

```
.link {  
  color: #0080DD; }
```

```
.link:hover {  
  color: blue; }
```

expanded

```
% sass sample1.sass -t expanded
```

```
.link {  
  color: #0080DD;  
}
```

```
.link:hover {  
  color: blue;  
}
```

compact

```
% sass sample1.sass -t compact
```

```
.link { color: #0080DD; }
```

```
.link:hover { color: blue; }
```

compressed

```
% sass sample1.sass -t compressed
```

```
.link{color:#0080DD}.link:hover{color:blue}
```



# 基本文法

# cssと違うところ

- {}がなく、インデントでブロックを表現。
- セミコロン不要。というか書いたらsyntax errorになる。
- プロパティの後にコロンではなく、頭にコロン。

sass ... セレクタに対してプロパティと値をネストさせる。

```
body  
  :color #fff
```

css

```
body { color: #fff; }
```

# ネストしたルール

```
% cat nested.sass
```

```
#main p
```

```
  :color #00ff00
```

```
  :width 97%
```

```
  .redbox
```

```
    :background-color #ff0000
```

```
    :color #000000
```

```
% sass nested.sass -t compact
```

```
#main p { color: #00ff00; width: 97%; }
```

```
#main p .redbox { background-color: #ff0000; color: #000000; }
```

# 親ルールの参照

&で親のセレクタを展開できる

```
% cat referencing_parent_rule.sass
```

```
.redbox
```

```
  :background-color #ff0000
```

```
  div&
```

```
    :color #f6f6f6
```

```
% sass referencing_parent_rule.sass -t compact
```

```
.redbox { background-color: #ff0000; }
```

```
div.redbox { color: #f6f6f6; }
```

↑&が展開されて.redbox|になっている

# 属性ネームスペース

同一ネームスペースのプロパティ(font-size, font-weight等)をネストして記述できる。

```
% cat attribute_namespaces.sass
```

```
.funky
```

```
  :font
```

```
    :family fantasy
```

```
    :size 30em
```

```
% sass attribute_namespaces.sass -t compact
```

```
.funky { font-family: fantasy; font-size: 30em; }
```

# Silent Comments

生成されるcssに含めないコメントを記述できる。

```
% cat silent_comments.sass
```

```
// 出力されないよ
```

```
#first.rule
```

```
  // 出力されないってば
```

```
  :width 100%
```

```
#second.rule
```

```
  // 複数行書きたい場合はネストする
```

```
    2行め
```

```
    3行め
```

```
  :width 99%
```

```
% sass silent_comments.sass
```

```
#first.rule {
```

```
  width: 100%; }
```

```
#second.rule {
```

```
  width: 99%; }
```

# Loud Comments

生成されるcssに含めたいコメントを記述できる

```
% cat loud_comments.sass
```

```
/* 出力されるー
```

```
#first.rule
```

```
/* 出力されるよー
```

```
:width 100%
```

```
#second.rule
```

```
/* 複数行書きたい場合はネストする
```

```
  2行め
```

```
  3行め
```

```
:width 99%
```

```
% sass loud_comments.sass
```

```
/* 出力されるー */
```

```
#first.rule {
```

```
/* 出力されるよー */
```

```
width: 100%; }
```

```
#second.rule {
```

```
/* 複数行書きたい場合はネストする
```

```
  * 2行め
```

```
  * 3行め */
```

定数



# 定数の定義と参照

Sassでは本来cssでは使えない定数を利用できる。

利用できる定数の種類

- 数値

- 色

- サイズ(px, em等)

- 文字列

例として色の定数定義と参照は以下ようになる。

```
% cat constants.sass
```

```
!main_color = #00ff00
```

```
#main
```

```
  :background-color = !main_color
```

```
% sass constants.sass -t compact
```

```
#main { background-color: #00ff00; }
```

参照時の注意として = の後に記述する必要がある。

# 定数の演算(数値とサイズ)

Sassの定数は演算することが可能。

↓基本幅に対して、演算する例。

```
% cat arithmetic_number.sass
```

```
!main_width = 10
```

```
!unit1 = em
```

```
!unit2 = px
```

```
#main
```

```
  p.foo
```

```
    :width = !main_width + !unit1
```

```
  p.bar
```

```
    :width = (!main_width + 15) + !unit2
```

```
  p.baz
```

```
    :width = (!main_width / 5) + px
```

```
% sass arithmetic_number.sass -t compact
```

```
#main p.foo { width: 10em; }
```

```
#main p.bar { width: 25px; }
```

# 定数の演算(色)

色に対しての演算も可能

脳内カラーマップなしでも背景よりちょっと濃い罫線とか余裕。

```
% cat arithmetic_color.sass
```

```
!base_color = #ccc
```

```
div.foo
```

```
  :color = !base_color
```

```
  :border = 1px solid (!base_color - #333)
```

```
% sass arithmetic_color.sass -t compact
```

```
div.foo { color: #cccccc; border: 1px solid #999999; }
```

# Mixins

# 定義と使い方

定義は=で行い、+で使う。非常に強力な機能。

```
% cat mixins.sass
```

```
=large-text
```

```
:font
```

```
:family Arial
```

```
:size 20px
```

```
.page-title
```

```
:color #fff
```

```
+large-text
```

```
% sass mixins.sass
```

```
.page-title {
```

```
  color: #fff;
```

```
  font-family: Arial;
```

```
  font-size: 20px; }
```

その他Tips

# 便利スクリプト

Vim

vim-haml(<http://github.com/tpope/vim-haml/tree/master>)

Emacs

<http://github.com/nex3/haml.git/extra/sass-mode.el>

# Railsとの連携

以下のコマンドを実行

```
haml --rails path/to/rails/app
```

#{RAILS\_ROOT/public/stylesheets/sass/foo.sassを作成する。  
/stylesheets/foo.cssにアクセスがあると自動的にコンパイルして  
foo.cssを生成してくれる。

各種カスタマイズSass::Plugin.optionsをenvironment.rb等で設定する。例えば出力をcompactにする場合は以下の通り。

```
Sass::Plugin.options[:style] = :compact
```



# まとめ

Railsとの組み合わせが便利なのは勿論、単純にCSSのジェネレータとして非常に優秀。

これだけのためにruby&rubygemsいれてもいいレベル。

## 合わせて読みたい

- <http://haml.hamptoncatlin.com/docs/rdoc/classes/Sass.html>
- <http://haml.ursm.jp/>
- <http://haml.ursm.jp/getting-started>
- <http://d.hatena.ne.jp/ursm/20080831/1220196477>

おわり