CSS niTe LP32

2014.2.15

お持ち帰りできる

コピペで使える！変数とmixin

PixelGrid Inc.
@tacamy & @geckotang

セッションの目的

Compass is perfect ?

- Compassにはないけれど、
  あると捗るmixinや変数は結構ある
- ないものは自分で追加する必要がある
- でも自分でイチから書くのは敷居が高い

まずはコピペでOK

- 初めから自分でイチから書けなくてもOK
- コピペなら今日からでもつかえる
- まずは使いドコロだけ理解しよう

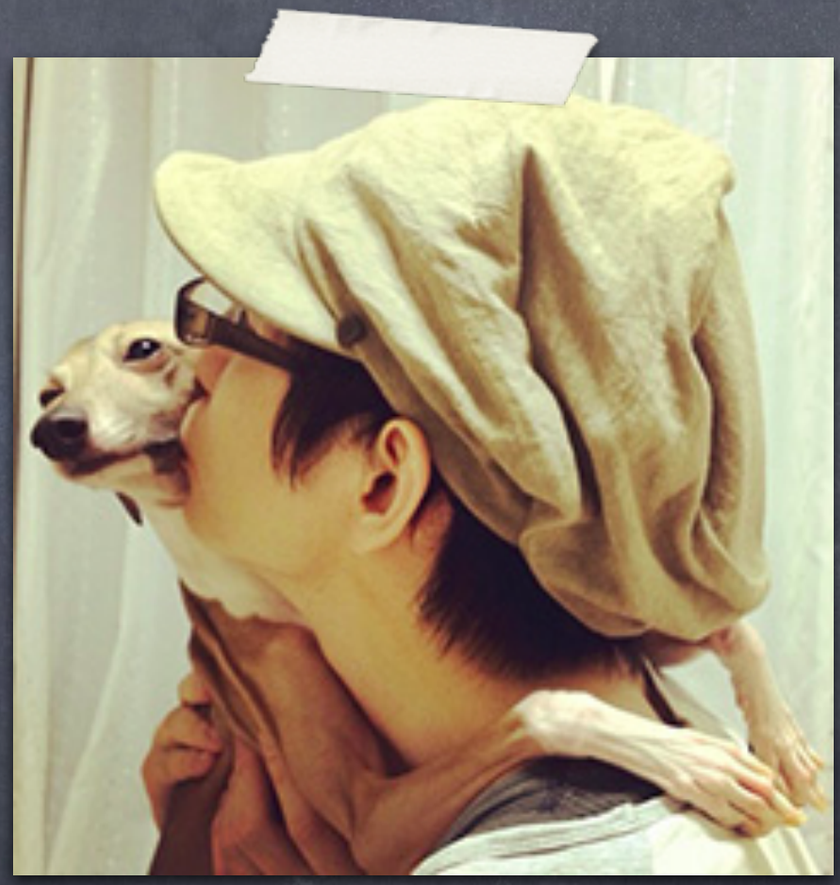
はじめに

※ 本セッションは、
先生と生徒の掛け合いで
進行していきます

登場人物

生徒げこたん

- CSS学園を主席で卒業したエリート
- Sass学園の新入生
- インターン先での、提出日ギリギリの仕様変更に悩む



たかみー先生

- Sass学園の教師
- げこたんの担任
- CSSerな生徒の悩みを、Sassで解決する方法を教える

時間割表

1時限目	変数の使いドコロ
2時限目	mixinの使いドコロ
3時限目	まとめ
放課後	おまけ

はじまりはじまり

1時限目

変数の使いドコロ

フォントサイズ

CSS

```
body  { font-size: 12px; }
h1    { font-size: 18px; }
h2    { font-size: 16px; }
h3    { font-size: 14px; }
small { font-size: 10px; }
── 省略 ──
.entry-title  { font-size: 18px; }
.entry-body   { font-size: 12px; }
.entry-date   { font-size: 10px; }
.entry-author { font-size: 10px; }
── 省略 ──
.copyright { font-size: 10px; }
```

CSS

```
body   { font-size: 12px; }
h1     { font-size: 18px; }
h2     { font-size: 16px; }
h3     { font-size: 14px; }
small { font-size: 10px; }
── 省略 ──
.entry-title  { font-size: 18px; }
.entry-body   { font-size: 12px; }
.entry-date   { font-size: 10px; }
.entry-author { font-size: 10px; }
── 省略 ──
.copyright { font-size: 10px; }
```

やっぱり、
10pxの箇所を
11pxに変更して！

エッ！
エディタで
置換しよう…

関係ないところまで
置換されてしまった
／(^o^)＼

SCSS - 変数の定義

```
$fs-base:    12px;
$fs-large:   14px;
$fs-xlarge:  16px;
$fs-xxlarge: 18px;
$fs-small:   10px;
```

SCSS - 変数の利用

```
$fs-base: 12px;
$fs-large: 14px;
$fs-xlarge: 16px;
$fs-xxlarge: 18px;
$fs-small: 10px;
```

```
body  { font-size: $fs-base; }
h1    { font-size: $fs-xxlarge; }
h2    { font-size: $fs-xlarge; }
h3    { font-size: $fs-large; }
small { font-size: $fs-small }
── 省略 ──
.entry-title  { font-size: $fs-xxlarge; }
.entry-body   { font-size: $fs-large; }
.entry-date   { font-size: $fs-small; }
.entry-author { font-size: $fs-small; }
── 省略 ──
.copyright { font-size: $fs-small; }
```

SCSS - 変数の利用

```
$fs-base: 12px;
$fs-large: 14px;
$fs-xlarge: 16px;
$fs-xxlarge: 18px;
$fs-small: 10px;
```

```
body   { font-size: $fs-base; }
h1     { font-size: $fs-xxlarge; }
h2     { font-size: $fs-xlarge; }
h3     { font-size: $fs-large; }
small { font-size: $fs-small }
── 省略 ──
.entry-title  { font-size: $fs-x
.entry-body   { font-size: $fs-l
.entry-date   { font-size: $fs-sm
.entry-author { font-size: $fs-smal
── 省略 ──
.copyright { font-size: $fs-small; }
```

変数の値を変更すれば
一括で置き換えできる

U(_A_U)

フォントファミリー

CSS

```
body {
  font-family: Verdana, Roboto, 'Droid
Sans', 'Hiragino Kaku Gothic ProN', Meiryo,
sans-serif;
}

h1 {
  font-family: Georgia, 'Hiragino Mincho
ProN’, 'MS PMincho', serif;
}

code {
  font-family: Consolas, Courier, monospace;
}
```

CSS

```
body {
  font-family: Verdana, Roboto
Sans', 'Hiragino Kaku Gothic Pr
sans-serif;
}

h1 {
  font-family: Georgia, 'Hiragino
ProN’, 'MS PMincho', serif;
}

code {
  font-family: Consolas, Courier,
}
```

CSSのあちこちで
これを何度も書くのが面倒
ヽ(^o^)ノ

フォントの順番が
変更になっても
一括変更できない！

SCSS - 変数の定義

```
$sans-serif: Verdana, Roboto, 'Droid Sans',
'Hiragino Kaku Gothic ProN', Meiryo, sans-
serif;

$serif: Georgia, 'Hiragino Mincho ProN’, 'MS
PMincho', serif;

$monospace: Consolas, Courier, monospace;
```

SCSS - 変数の利用

```scss
body {
  font-family: $sans-serif; }

h1 {
  font-family: $serif;
}

code {
  font-family: $monospace;
}
```

コードの見通し++

変更があっても
一括で置き換えできる

```scss
$sans-serif: Verdana, Roboto, 'Droid
Sans', 'Hiragino Kaku Gothic ProN',
Meiryo, sans-serif;

$serif: Georgia, 'Hiragino Mincho ProN’,
'MS PMincho', serif;

$monospace: Consolas, Courier, monospace;
```

٩(๑•́ᴗ•́๑)۶

2時限目

mixinの使いドコロ

画像の中央配置

CSS

```
.box {
  position: relative;
  width: 200px;
  height: 200px;
}
img {
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;
  margin: auto;
}
```

この中央配置用の
スタイル6点セットを
毎回書くのが面倒！

SCSS - mixinの利用

```scss
@import trbl;

.box {
  position: relative;
  width: 200px;
  height: 200px;
}
img {
  @include trbl;
}
```

※親要素に
position指定必須

この1行だけで
さっきの6点セットを
呼び出せる

SCSS - mixinの利用（画像以外の場合）

```
@import trbl;

.box {
  position: relative;
  width: 200px;
  height: 200px;
}
span {
  @include trbl(100px, 50px);
}
```

画像以外の場合は、
widthとheightの
指定が必須

CSS - 出力結果（画像以外の場合）

```
.box {
  position: relative;
  width: 200px;
  height: 200px;
}
span {
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;
  width: 100px;
  height: 50px;
  margin: auto;
}
```

引数に渡した値が
出力される

SCSS - mixinの中身

```scss
@mixin trbl($width: null, $height: null) {
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;
  width: $width;    // nullの場合出力されない
  height: $height; // nullの場合出力されない
  margin: auto;
}
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_trbl.scss

ヾ(｡A ｡)ﾒ(｡A ｡)ﾒ(｡A ｡)ﾉ

リンクアンダーライン

CSS

```
a:link,
a:visited { text-decoration: underline; }
a:hover,
a:active,
a:focus { text-decoration: none; }

.footer a:link,
.footer a:visited,
.footer a:hover,
.footer a:active,
.footer a:focus { text-decoration: none; }

.entry a:link,
.entry a:visited,
.entry a:hover,
.entry a:active,
.entry a:focus { text-decoration: underline; }
```

CSS

```
a:link,
a:visited { text-decoration: underline; }
a:hover,
a:active,
a:focus { text-decoration: none; }

.footer a:link,
.footer a:visited,
.footer a:hover,
.footer a:active,
.footer a:focus { text-decoratio

.entry a:link,
.entry a:visited,
.entry a:hover,
.entry a:active,
.entry a:focus { text-decoration: underline; }
```

擬似クラスすべてを
毎回指定するのが
面倒すぎる！

SCSS - mixinの利用

```
@import link-underline;

a {
  @include link-underline(line-to-none);
}

.footer a {
  @include link-underline(none);
}

.entry a {
  @include link-underline(line);
}
```

擬似クラスの出力を
mixinに任せることで
こんなにスッキリ！

SCSS - mixinの中身

```
@mixin link-underline($type, $important: false) {
── 省略 ──
   @if $type == line-to-none {
      &:link,
      &:visited { text-decoration: underline#{$important}; }
      &:hover,
      &:active,
      &:focus {
         text-decoration: none#{$important};
      }
   }
── 省略 ──
}
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_link-underline.scss

SCSS - mixinの利用（第一引数がnoneの場合）

```
.foo a {
  @include link-underline(none);
}
```

```
.foo a:link,
.foo a:visited,
.foo a:hover,
.foo a:active,
.foo a:focus {
  text-decoration: none;
}
```

SCSS - mixinの利用（第一引数がlineの場合）

```
.foo a {
  @include link-underline(line);
}
```

```
.foo a:link,
.foo a:visited,
.foo a:hover,
.foo a:active,
.foo a:focus {
  text-decoration: underline;
}
```

SCSS - mixinの利用（第一引数がline-to-noneの場合）

```
.foo a {
  @include link-underline(line-to-none);
}
```

```
.foo a:link,
.foo a:visited {
  text-decoration: underline;
}
.foo a:hover,
.foo a:active,
.foo a:focus {
  text-decoration: none;
}
```

SCSS - mixinの利用（第一引数がnone-to-lineの場合）

```scss
.foo a {
  @include link-underline(none-to-line);
}
```

```scss
.foo a:link,
.foo a:visited {
  text-decoration: none;
}
.foo a:hover,
.foo a:active,
.foo a:focus {
  text-decoration: underline;
}
```

SCSS - mixinの利用（!importantをつける場合）

```
.foo a {
  @include link-underline(none, important);
}
```

引数に important

```
.foo a:link,
.foo a:visited,
.foo a:hover,
.foo a:active,
.foo a:focus {
  text-decoration: none !important;
}
```

余白調整用クラス

CSS

```
.mt00 { margin-top: 0px !important }
.mt01 { margin-top: 1px !important }
.mt02 { margin-top: 2px !important }
.mt03 { margin-top: 3px !important }
.mt04 { margin-top: 4px !important }
.mt05 { margin-top: 5px !important }
.mt06 { margin-top: 6px !important }
.mt07 { margin-top: 7px !important }
.mt08 { margin-top: 8px !important }
.mt09 { margin-top: 9px !important }
.mt10 { margin-top: 10px !important }
.mt11 { margin-top: 11px !important }
.mt12 { margin-top: 12px !important }
.mt13 { margin-top: 13px !important }
.mt14 { margin-top: 14px !important }
.mt15 { margin-top: 15px !important }
```

CSS

```
.mt00 { margin-top: 0px !important }
.mt01 { margin-top: 1px !important }
.mt02 { margin-top: 2px !important }
.mt03 { margin-top: 3px !important }
.mt04 { margin-top: 4px !important }
.mt05 { margin-top: 5px !important }
.mt06 { margin-top: 6px !important }
.mt07 { margin-top: 7px !important }
.mt08 { margin-top: 8px !important }
.mt09 { margin-top: 9px !important }
.mt10 { margin-top: 10px !important }
.mt11 { margin-top: 11px !important }
.mt12 { margin-top: 12px !important }
.mt13 { margin-top: 13px !important }
.mt14 { margin-top: 14px !important }
.mt15 { margin-top: 15px !important }
.mt16 { margin-top: 16px !important }
.mt17 { margin-top: 17px !important }
.mt18 { margin-top: 18px !important }
.mt19 { margin-top: 19px !important }
.mt20 { margin-top: 20px !important }
.mt30 { margin-top: 30px !important }
.mt40 { margin-top: 40px !important }
.mt50 { margin-top: 50px !important }
.mb00 { margin-bottom: 0px !important }
.mb01 { margin-bottom: 1px !important }
.mb02 { margin-bottom: 2px !important }
.mb03 { margin-bottom: 3px !important }
.mb04 { margin-bottom: 4px !important }
.mb05 { margin-bottom: 5px !important }
.mb06 { margin-bottom: 6px !important }
.mb07 { margin-bottom: 7px !important }
.mb08 { margin-bottom: 8px !important }
.mb09 { margin-bottom: 9px !important }
.mb10 { margin-bottom: 10px !important }
.mb11 { margin-bottom: 11px !important }
.mb12 { margin-bottom: 12px !important }
.mb13 { margin-bottom: 13px !important }
.mb14 { margin-bottom: 14px !important }
.mb15 { margin-bottom: 15px !important }
.mb16 { margin-bottom: 16px !important }
.mb17 { margin-bottom: 17px !important }
.mb18 { margin-bottom: 18px !important }
.mb19 { margin-bottom: 19px !important }
.mb20 { margin-bottom: 20px !important }
.mb30 { margin-bottom: 30px !important }
.mb40 { margin-bottom: 40px !important }
.mb50 { margin-bottom: 50px !important }
.mtN01 { margin-top: -1px !important }
.mtN02 { margin-top: -2px !important }
.mtN03 { margin-top: -3px !important }
.mtN04 { margin-top: -4px !important }
.mtN05 { margin-top: -5px !important }
.mtN06 { margin-top: -6px !important }
.mtN07 { margin-top: -7px !important }
.mtN08 { margin-top: -8px !important }
.mtN09 { margin-top: -9px !important }
.mtN10 { margin-top: -10px !important }
.mtN11 { margin-top: -11px !important }
.mtN12 { margin-top: -12px !important }
.mtN13 { margin-top: -13px !important }
.mtN14 { margin-top: -14px !important }
.mtN15 { margin-top: -15px !important }
.mtN16 { margin-top: -16px !important }
.mtN17 { margin-top: -17px !important }
.mtN18 { margin-top: -18px
.mtN19 { margin-top: -19px
.mtN20 { margin-top: -20px
.mtN30 { margin-top: -30px
.mtN40 { margin-top: -40px
.mtN50 { margin-top: -50px !i
.mbN01 { margin-bottom: -1px !i
.mbN02 { margin-bottom: -2px !impor
.mbN03 { margin-bottom: -3px !important }
.mbN04 { margin-bottom: -4px !important }
.mbN05 { margin-bottom: -5px !important }
.mbN06 { margin-bottom: -6px !important }
.mbN07 { margin-bottom: -7px !important }
.mbN08 { margin-bottom: -8px !important }
.mbN09 { margin-bottom: -9px !important }
```

／(^o^)＼

SCSS - mixinの利用

```
@import output-mamrgin;
@import spacing;

@include output-margin;
```

たった1行
mixinを読み込むだけ！

SCSS - mixinの中身

```
@mixin output-margin {
   // margin-top
   @for $i from 0 through 20 {
      @include spacing(margin, top, $i);
   }
   $i: 30;
   @while $i <= 50 {
      @include spacing(margin, top, $i);
      $i: $i + 10;
   }
—— 省略 ——
}
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_output-margin.scss

SCSS - mixinの中身

```
@mixin spacing($property, $direction, $size, $unit:
px) {
  $property       : $property - $direction;
  $property-value: $size + $unit;
  $class-prefix   : '';
  $class-unit     : $unit;
  $class-base-prefix          : m;
  $class-base-prefix-padding: p;
  $class-prefix-top           : t;
  ―― 省略 ――
}
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_spacing.scss

SCSS - mixinを修正（数値を変更）

```scss
@mixin output-margin {
  // margin-top
  @for $i from 0 through 20 {
    @include spacing(margin, top, $i);
  }
  $i: 30;
  @while $i <= 50 {
    @include spacing(margin, top, $i);
    $i: $i + 10;
  }
── 省略 ──
}
```

mixinの数値を変更すれば
間隔や最大値を調整可能

SCSS - mixinを自分で追加

```
@mixin output-padding {
  // padding-top
  @for $i from 0 through 20 {
    @include spacing(padding, top, $i);
  }
  $i: 30;
  @while $i <= 50 {
    @include spacing(padding, top, $i);
    $i: $i + 10;
  }
── 省略 ──
}
```

padding調整用
クラスも生成できる

ヽ(=´▽`=)ノ

remフォントサイズ

CSS

```
html {
  font-size: 62.5%; /* =10px */
}
body {
  font-size: 1.2rem;
}
p {
  font-size: 1.4rem;
  margin: 1.5rem 0;
}
```

もしかして
IE8ではremが
使えない!?

CSS

```
html {
  font-size: 62.5%; /* =10px */
}
body {
  font-size: 12px;
  font-size: 1.2rem;
}
p {
  font-size: 14px;
  font-size: 1.4rem;
  margin: 15px 0;
  margin: 1.5rem 0;
}
```

もしかして
IE8ではremが
使えない!?

IE8用のpx指定と
rem指定の両方を
書くなんて面倒

SCSS - mixinの利用

```
@import rem;

html {
  font-size: 62.5%;
}
body {
  @include rem(font-size, 12px);
}
p {
  @include rem(font-size, 14px);
  @include rem(margin, 15px 0);
}
```

px指定だけで
自動でremでの指定も
両方書き出してくれる

SCSS - mixinの利用（IE8対応不要時）

```
@import rem;

$rem-legacy-support: false;

html {
  font-size: 62.5%;
}
body {
  @include rem(font-size, 12px);
}
p {
  @include rem(font-size, 14px);
  @include rem(margin, 15px 0);
}
```

IE8用のpx指定が
CSSに出力されなくなる

SCSS - mixinの中身

```
@mixin rem($property, $values) {
  $px : ();
  $rem: ();
  @each $value in $values {
    @if $value == 0 or $value == auto {
      $px : append($px , $value);
      $rem: append($rem, $value);
    }
  ―― 省略 ――
}
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_rem.scss

(●ˆoˆ●)

メディアクエリ

CSS

```
.box {
  /* 共通スタイル */
}
@media screen and (min-width: 801px) {
  .box { /* デスクトップ用スタイル */ }
 }
@media only screen and (min-width: 481px)
and (max-width: 800px) {
  .box { /* タブレット用スタイル */ }
}
@media only screen and (max-width: 480px) {
  .box { /* スマートフォン用スタイル */ }
}
```

CSS

```
.box {
  /* 共通スタイル */
}
@media screen and (min-width: 801p
  .box { /* デスクトップ用スタイル */ }
 }
@media only screen and (min-width: 481px)
and (max-width: 800px) {
  .box { /* タブレット用スタイル */ }
}
@media only screen and (max-wi
  .box { /* スマートフォン用スタ
}
```

メディアクエリが
長すぎて
書くのが面倒

ブレイクポイントを
後から変更するのが大変

SCSS - mixinの利用

```
@import mq;

.box {
  /* 共通スタイル */
  @include min-screen(801px) {
    /* デスクトップ用スタイル */
  };
  @include screen(481px, 800px) {
    /* タブレット用スタイル */
  };
  @include max-screen(480px) {
    /* スマートフォン用スタイル */
  };
}
```

面倒なメディアクエリが
簡単に記述できる

セレクタの中で
メディアクエリの
分岐が書ける

SCSS - mixinの中身

```
@mixin min-screen($min-width) {
  @media screen and (min-width: $min-width) {
    @content;
  }
}
@mixin max-screen($max-width) {
  @media screen and (max-width: $max-width) {
    @content;
  }
}
—— 省略 ——
```

https://github.com/geckotang/cssnite-lp32/blob/master/scss/_mq.scss

SCSS - mixinを自分で追加

```
$mq-mobile-max-width: 480px;
$mq-tablet-max-width: 800px;

@mixin mq-mobile {
  @include max-screen($mq-mobile-max-width) {
    @content;
  };
}
@mixin mq-tablet {
  @include screen($mq-mobile-max-width + 1, $mq-
tablet-max-width) {
    @content;
  };
}
@mixin mq-desktop {
  @include min-screen($mq-mobile-max-width + 1) {
    @content;
  };
}
```

SCSS - mixinを自分で追加

```scss
$mq-mobile-max-width: 480px;
$mq-tablet-max-width: 800px;

@mixin mq-mobile {
  @include max-screen($mq-mobil
    @content;
  };
}
@mixin mq-tablet {
  @include screen($mq-mobile-max-width + 1, $mq-
tablet-max-width) {
    @content;
  };
}
@mixin mq-desktop {
  @include min-screen($mq-mobile-max-width + 1) {
    @content;
  };
}
```

ブレイクポイントを変数にして
案件に合わせたオリジナルの
mixinを追加すると便利

$$i \hbar \frac{\partial}{\partial t} \psi(x,t) = H \psi(x,t)$$

3時限目

まとめ

変数の使いドコロ

- 後々変更になりそうなものや、
  値が長くなるものを変数に
- SCSSの頭に記述するとわかりやすい

変数の使用例

- カラーコード
- フォントサイズ、フォントファミリー
- サイト共通で使う、幅や高さの数値
- メディアクエリ用のブレークポイントの数値
- 画像のファイルパスやdataURI化された画像

mixinの使いドコロ

- 同じような記述を繰り返す
- 微妙な違いを、引数をつかって調整
- レガシーブラウザとモダンブラウザの指定を併記

記述の繰り返し

- あちこちで使われるコードを
  1箇所で一元管理
- mixinだけ修正すれば、すべてに反映
- 画像置換やclearfixなど

引数で調整

- mixinを@includeするときに、
  引数を渡すことで、CSSの生成時の
  出力結果を調整できる
- メディアクエリなど

レガシーブラウザ対応

- レガシーブラウザ対応が不要になった時点で、mixinから削除できる

```
@mixin inline-block {
  display: inline-block;
  *display: inline; //IE6,7
  *zoom: 1; //IE6,7
}
.block {
  @include inline-block;
}
```

放課後

おまけ

お土産にどうぞ

- 本日紹介したmixinのセット

https://github.com/geckotang/cssnite-lp32

出演

坂巻 翔大郎

PixelGrid Inc. 10px目

- @geckotang
- フロントエンドエンジニア &
  CSSプログラマー(自称)
- CSS PANIC作者

  http://jsdo.it/GeckoTang/codes

山田 敬美

PixelGrid Inc. 9px目

- @tacamy
- フロントエンドエンジニア &
  スーパージャバスハカー見習い
- 著書: Sass入門

  http://www.amazon.co.jp/dp/B00GJM3BGS

提供

CodeGrid

https://app.codegrid.net/

30日間
無料でお試し

- SassとCSS設計
- Compassで簡単、CSSスプライト作成
- ここまでできる！HTML+CSS

フロントエンドに
関わる人々のガイド

このmixinで
みなさまのSassライフが
豊かになりますように！

ご清聴ありがとう

ございました！