Sass/SCSS クイックリファレンス

変数、ネスト、ミックスイン、関数、制御フロー

構文

SCSS と Sass

```
// SCSS（CSSのスーパーセット — 波括弧を使用）
.nav { display: flex; }

// Sass（インデント形式 — 波括弧・セミコロン不要）
.nav
  display: flex
```

SCSS が最も広く使われる構文

比較

SCSS（.scss）	CSS 互換、波括弧とセミコロンあり
Sass（.sass）	インデントベース、波括弧なし
出力	どちらも標準 CSS にコンパイルされる
推奨	SCSS（採用例が多く移行が容易）

変数

定義と使用

```
$primary: #3498db;
$spacing: 16px;
$font-stack: "Helvetica", Arial, sans-serif;

.btn {
  color: $primary;
  padding: $spacing;
  font-family: $font-stack;
}
```

変数スコープ

```
$color: red;        // グローバル
.card {
  $color: blue;     // .card ローカル
  color: $color;    // blue
}
.other { color: $color; } // red
```

フラグ

!default	未定義の場合のみ設定する
!global	ローカル変数をグローバルスコープに昇格

ネスト

セレクタのネスト

```
.nav {
  ul { list-style: none; }
  li { display: inline-block; }
  a { text-decoration: none;
    &:hover { color: blue; }   // & = 親セレクタ
  }
}
```

親セレクタ (&)

```
.btn {
  &--primary { background: blue; }  // BEM: .btn--primary
  &__icon { margin-right: 4px; }    // BEM: .btn__icon
  .dark & { color: white; }         // .dark .btn
}
```

プロパティのネスト

```
.box {
  border: { width: 1px; style: solid; color: #ccc; }
  // コンパイル後: border-width, border-style, border-color
}
```

ミックスイン

定義と使用

```
@mixin flex-center($direction: row) {
  display: flex;
  justify-content: center;
  align-items: center;
  flex-direction: $direction;
}

.hero { @include flex-center(column); }
```

コンテンツブロック

```
@mixin responsive($breakpoint) {
  @media (min-width: $breakpoint) { @content; }
}

.sidebar {
  width: 100%;
  @include responsive(768px) { width: 300px; }
}
```

ミックスイン機能

@mixin name($args)	再利用可能なスタイルブロックを定義
@include name()	ミックスインを使用
デフォルトパラメータ	**$arg: value** でオプション引数を設定
$args...	可変長引数（残余パラメータ）
@content	呼び出し元のコンテンツブロックを挿入

関数

カスタム関数

```
@function rem($px, $base: 16) {
  @return math.div($px, $base) * 1rem;
}
.title { font-size: rem(24); } // 1.5rem
```

組み込み関数

darken($color, 10%)	色を暗くする
lighten($color, 10%)	色を明るくする
mix($c1, $c2, 50%)	2 つの色を混合する
rgba($color, 0.5)	アルファチャンネルを設定
math.div($a, $b)	除算（/ の代替）
math.round($n)	数値を四捨五入
string.quote($s)	文字列にクォートを追加
if($cond, $t, $f)	インライン条件式

継承

継承とプレースホルダー

```
%flex-center {          // プレースホルダー — 単独では出力されない
  display: flex;
  justify-content: center;
  align-items: center;
}

.hero { @extend %flex-center; }
.modal { @extend %flex-center; }
// セレクタグループ化により1つの CSS ルールを共有
```

継承とミックスインの比較

@extend	セレクタをグループ化 — CSS 出力が小さくなる
@mixin	宣言をコピー — 引数をサポート
% プレースホルダー	継承専用（未使用の場合は出力されない）
推奨	パラメータ付きスタイルにはミックスインを優先

パーシャルとインポート

ファイル整理

```
// _variables.scss（パーシャル — 単独ではコンパイルされない）
$primary: #3498db;

// main.scss
@use "variables";          // モダン: 名前空間付き
.btn { color: variables.$primary; }

@use "variables" as v;     // エイリアス
.btn { color: v.$primary; }
```

モジュールシステム

@use 'file'	名前空間付きでモジュールを読み込む
@use 'file' as *	名前空間なしで読み込む
@use 'file' as alias	カスタム名前空間を設定
@forward 'file'	モジュールメンバーを再エクスポート
_partial.scss	単独ではコンパイルされないファイル

@import は非推奨 — @use と @forward を使用すること

制御フロー

条件分岐

```
@mixin theme($mode) {
  @if $mode == dark {
    background: #333; color: #fff;
  } @else {
    background: #fff; color: #333;
  }
}
```

ループ

```
@for $i from 1 through 4 {
  .col-#{$i} { width: 25% * $i; }
}

@each $name, $color in (primary: blue, danger: red) {
  .text-#{$name} { color: $color; }
}
```

ディレクティブ

@if / @else if / @else	条件ロジック
@for $i from a through b	数値ループ（終端を含む）
@for $i from a to b	数値ループ（終端を含まない）
@each $item in $list	リストまたはマップを反復処理
@while	条件が真の間ループ
#{$var}	セレクタ・プロパティ内での変数展開

マップとリスト

マップ

```
$colors: (primary: #3498db, danger: #e74c3c, success: #2ecc71);

.alert { color: map.get($colors, danger); }

@each $name, $color in $colors {
  .bg-#{$name} { background: $color; }
}
```

リスト

```
$sizes: 8px 16px 24px 32px;
.box { padding: list.nth($sizes, 2); } // 16px
```

Sass/SCSS クイックリファレンス

マップとリストの関数

map.get($map, $key)	キーで値を取得
map.merge($m1, $m2)	2 つのマップを結合
map.keys($map)	すべてのキーのリスト
map.has-key($map, $key)	キーの存在を確認
list.nth($list, $n)	インデックスで要素を取得（1 始まり）
list.length($list)	要素数
list.append($list, $val)	リストに要素を追加

よく使うパターン

レスポンシブブレークポイント

```
$breakpoints: (sm: 576px, md: 768px, lg: 992px, xl: 1200px);

@mixin bp($name) {
  @media (min-width: map.get($breakpoints, $name)) {
    @content;
  }
}
.sidebar { width: 100%; @include bp(md) { width: 300px; } }
```

ユーティリティジェネレータ

```
$spaces: (0: 0, 1: 4px, 2: 8px, 3: 16px, 4: 32px);
@each $key, $val in $spaces {
  .mt-#{$key} { margin-top: $val; }
  .mb-#{$key} { margin-bottom: $val; }
  .p-#{$key} { padding: $val; }
}
```

ダークモード

```
@mixin dark { @media (prefers-color-scheme: dark) { @content; } }
body {
  background: #fff;
  @include dark { background: #1a1a1a; color: #eee; }
}
```