

# 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

<b>SCSS (.scss)</b>	CSS 互換、波括弧とセミコロンあり
<b>Sass (.sass)</b>	インデントベース、波括弧なし
<b>出力</b>	どちらも標準 CSS にコンパイルされる
<b>推奨</b>	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

<b>@mixin name(\$args)</b>	再利用可能なスタイルブロックを定義
<b>@include name()</b>	ミックスインを使用
<b>デフォルトパラメータ</b>	<code>\$arg: value</code> でオプション引数を設定
<b>\$args...</b>	可変長引数 (残余パラメータ)
<b>@content</b>	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

<b>darken(\$color, 10%)</b>	色を暗くする
<b>lighten(\$color, 10%)</b>	色を明るくする
<b>mix(\$c1, \$c2, 50%)</b>	2つの色を混合する
<b>rgba(\$color, 0.5)</b>	アルファチャンネルを設定
<b>math.div(\$a, \$b)</b>	除算 ( '/' の代替)
<b>math.round(\$n)</b>	数値を四捨五入
<b>string.quote(\$s)</b>	文字列にクォートを追加
<b>if(\$cond, \$t, \$f)</b>	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

<b>@extend</b>	セレクタをグループ化 - CSS 出力が小さくなる
<b>@mixin</b>	宣言をコピー - 引数をサポート
<b>% プレースホルダー</b>	継承専用 (未使用の場合は出力されない)
<b>推奨</b>	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

<b>@use 'file'</b>	名前空間付きでモジュールを読み込む
<b>@use 'file' as *</b>	名前空間なしで読み込む
<b>@use 'file' as alias</b>	カスタム名前空間を設定
<b>@forward 'file'</b>	モジュールメンバーを再エクスポート
<b>_partial.scss</b>	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

<b>@if / @else if / @else</b>	条件ロジック
<b>@for \$i from a through b</b>	数値ループ (終端を含む)
<b>@for \$i from a to b</b>	数値ループ (終端を含まない)
<b>@each \$item in \$list</b>	リストまたはマップを反復処理
<b>@while</b>	条件が真の間ループ
<b>#{ \$var }</b>	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
```

### マップとリストの関数

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

### よく使うパターン

#### レスポンスブレイクポイント

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

@mixin bp($name) {
  @media (min-width: map.get($breakpoints, $name)) {
  }
}

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

#### ユーティリティジェネレーター

```
$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; }
}
```