

REFERÊNCIA RÁPIDA DE SASS/SCSS

Variáveis, aninhamento, mixins, funções, controle de fluxo

Sintaxe

SCSS vs Sass

```
// SCSS (superset of CSS – uses braces)
.nav { display: flex; }

// Sass (indented – no braces or semicolons)
.nav
  display: flex
```

SCSS é a sintaxe mais utilizada

Comparação

SCSS (.scss) Compatível com CSS, chaves e ponto-e-vírgula
Sass (.sass) Baseado em indentação, sem chaves
Output Ambos compilam para CSS padrão
Recommended SCSS (adoção mais ampla, migração mais fácil)

Variáveis

Definindo e Usando

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

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

Escopo de Variável

```
$color: red; // global
.card {
  $color: blue; // local to .card
  color: $color; // blue
}
.other { color: $color; } // red
```

Flags

!default Definir apenas se ainda não estiver definido
!global Promover variável local para escopo global

Aninhamento

Aninhamento de Seletores

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

Seletor Pai (&)

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

Aninhamento de Propriedades

```
._box {
  border: { width: 1px; style: solid; color: #ccc; }
} // compiles to: border-width, border-style, border-color
```

Mixins

Definir e Incluir

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

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

Blocos de Conteúdo

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

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

Recursos de Mixin

@mixin name(\$args) Definir bloco de estilos reutilizável
@include name() Usar o mixin
Default params *Sarg: value* para parâmetros opcionais
\$args... Argumentos variáveis (rest params)
@content Injetar bloco de conteúdo do chamador

Funções

Funções Personalizadas

```
@function rem($px, $base: 16) {
  @return math.div($px, $base) * 1rem;
}

.title { font-size: rem(24); } // 1.5rem
```

Funções Embutidas

darken(\$color, 10%) Escurecer uma cor
lighten(\$color, 10%) Clarear uma cor
mix(\$c1, \$c2, 50%) Misturar duas cores
rgba(\$color, 0.5) Definir canal alfa
math.div(\$a, \$b) Divisão (substitui '/')
math.round(\$n) Arredondar número
string.quote(\$s) Adicionar aspas à string
if(\$cond, \$t, \$f) Condicional inline

Extends

Extend e Placeholder

```
%flex-center { // placeholder – not emitted
  display: flex;
  justify-content: center;
  align-items: center;
}

.hero { @extend %flex-center; }
.modal { @extend %flex-center; }
// Both share one CSS rule via selector grouping
```

Extend vs Mixin

@extend Agrupa seletores — saída CSS menor
@mixin Copia declarações — suporta argumentos
% placeholder Somente para extend (não emitido se não usado)
Recommendation Prefira mixins para estilos parametrizados

Parciais e Import

Organização de Arquivos

```
// variables.scss (partial – not compiled alone)
$primary: #3498db;

// main.scss
@use "variables"; // modern: namespaced
.btn { color: variables.$primary; }

@use "variables" as v; // alias
.btn { color: v.$primary; }
```

Sistema de Módulos

@use 'file' Carregar módulo com namespace
@use 'file' as * Carregar sem namespace
@use 'file' as alias Namespace personalizado
@forward 'file' Re-exportar membros do módulo
_partial.scss Arquivo não compilado independentemente

@import está obsoleto — use @use e @forward

Controle de Fluxo

Condicionais

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

Laços

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

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

Diretivas

@if / @else if / @else Lógica condicional
@for \$i from a through b Laço numérico (inclusivo)
@for \$i from a to b Laço numérico (fim exclusivo)
@each \$item in \$list Iterar lista ou mapa
@while Laço enquanto condição for verdadeira
#{ \$var } Interpolação em seletores/propriedades

Mapas e Listas

Mapas

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

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

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

Listas

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

Funções de Mapa e Lista

map.get(\$map, \$key) Obter valor por chave
map.merge(\$m1, \$m2) Mesclar dois mapas
map.keys(\$map) Lista de todas as chaves
map.has-key(\$map, \$key) Verificar se chave existe
list.nth(\$list, \$n) Obter item no índice (base 1)
list.length(\$list) Número de itens
list.append(\$list, \$val) Adicionar item à lista

Padrões Comuns

Breakpoints Responsivos

```
$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; } }
```

Gerador de Utilitários

```
$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; }
}
```

Modo Escuro

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