

Referencia Rápida de Sass/SCSS

Variables, anidamiento, mixins, funciones, flujo de control

Sintaxis

SCSS vs Sass

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

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

SCSS es la sintaxis más comúnmente utilizada

Comparación

SCSS (.scss)	Compatible con CSS, usa llaves y punto y coma
Sass (.sass)	Basado en indentación, sin llaves
Output	Ambos compilan a CSS estándar
Recommended	SCSS (mayor adopción, migración más sencilla)

Variables

Definir y Usar

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

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

Alcance de Variables

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

Banderas

!default	Establecer solo si no está ya definida
!global	Promover variable local al alcance global

Anidamiento

Anidamiento de Selectores

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

Selector Padre (&)

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

Anidamiento de Propiedades

```
.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); }
```

Bloques de Contenido

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

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

Características de los Mixins

@mixin name(\$args)	Definir bloque de estilos reutilizable
@include name()	Usar el mixin
Default params	Arg: value para parámetros opcionales
\$args...	Argumentos variables (parámetros rest)
@content	Insertar bloque de contenido del llamador

Funciones

Funciones Personalizadas

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

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

Funciones Integradas

darken(\$color, 10%)	Oscurecer un color
lighten(\$color, 10%)	Aclarar un color
mix(\$c1, \$c2, 50%)	Mezclar dos colores
rgba(\$color, 0.5)	Establecer canal alfa
math.div(\$a, \$b)	División (reemplaza /)
math.round(\$n)	Redondear número
string.quote(\$s)	Agregar comillas a la cadena
if(\$cond, \$t, \$f)	Condiciona en línea

Extends

Extend y 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	Agrupar selectores — menor salida CSS
@mixin	Copia declaraciones — soporta argumentos
% placeholder	Solo para extend (no se emite si no se usa)
Recommendation	Preferir mixins para estilos parametrizados

Parciales e Importación

Organización de Archivos

```
// _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'	Cargar módulo con espacio de nombres
@use 'file' as *	Cargar sin espacio de nombres
@use 'file' as alias	Espacio de nombres personalizado
@forward 'file'	Re-exportar miembros del módulo
_partial.scss	Archivo no compilado independientemente

@import está obsoleto — usar @use y @forward en su lugar

Flujo de Control

Condicionales

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

Bucles

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

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

Directivas

@if / @else if / @else	Lógica condicional
@for \$i from a through b	Bucle numérico (fin inclusivo)
@for \$i from a to b	Bucle numérico (fin exclusivo)
@each \$item in \$list	Iterar lista o mapa
@while	Bucle mientras la condición sea verdadera
#{\$var}	Interpolación en selectores/props

Mapas y 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
```

Referencia Rápida de Sass/SCSS

Funciones de Mapa y Lista

<code>map.get(\$map, \$key)</code>	Obtener valor por clave
<code>map.merge(\$m1, \$m2)</code>	Combinar dos mapas
<code>map.keys(\$map)</code>	Lista de todas las claves
<code>map.has-key(\$map, \$key)</code>	Verificar si la clave existe
<code>list.nth(\$list, \$n)</code>	Obtener elemento en el índice (base 1)
<code>list.length(\$list)</code>	Número de elementos
<code>list.append(\$list, \$val)</code>	Agregar elemento a la lista

Patrones Comunes

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; } }
```

Generador de Utilidades

```
$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 Oscuro

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