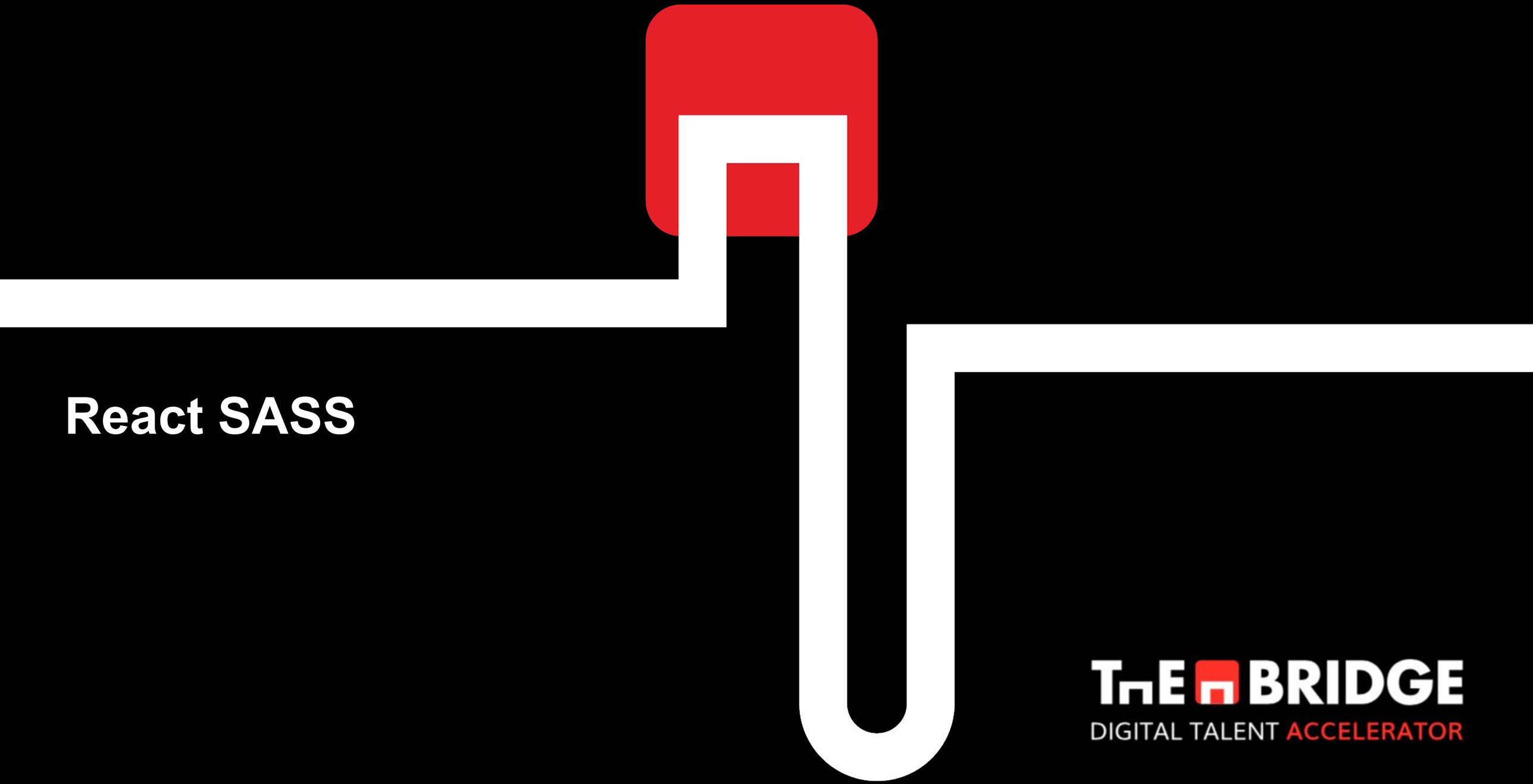
React SASS



Índice

¿Que es SASS?

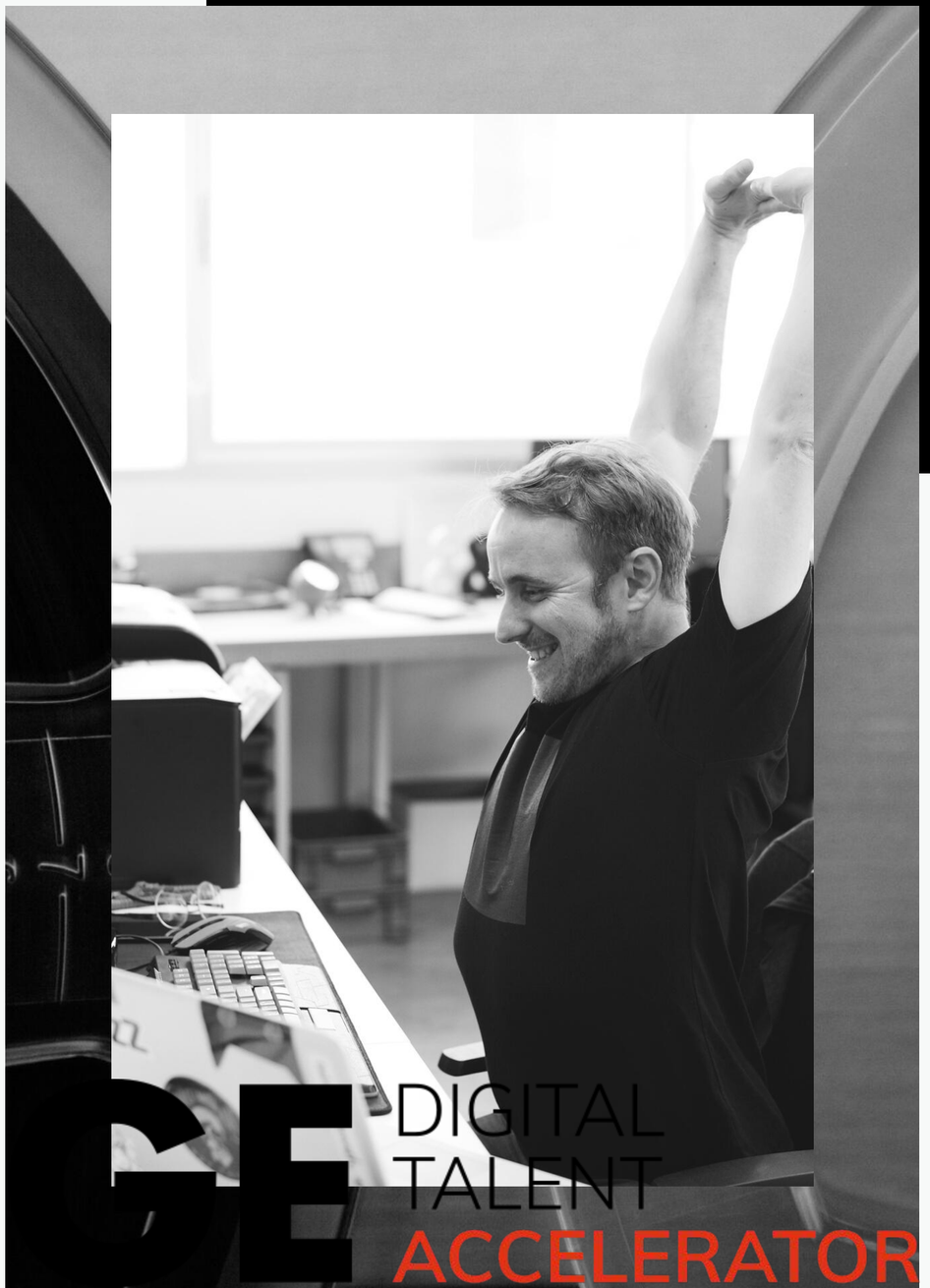
Nesting

Variables

Arquitectura 7 - 1

Imports / Mixins

Extends / Inheritance



THE BRIDGE DIGITAL TALENT ACCELERATOR

¿Que es SASS?

Sass es un preprocesador CSS. Un **preprocesador CSS** es una herramienta que nos permite generar, de manera automática, hojas de estilo, **añadiéndoles características** que no tiene CSS, y que son **propias de los lenguajes de programación**, como pueden ser **variables**, **funciones**, **selectores anidados**, **herencia**, etcétera.

Estas características de los preprocesadores nos permiten, además, que **el CSS que se genera** sea **más fácil de mantener y más reutilizable**.

https://sass-lang.com/

Sintaxis

scss (Sassy CSS): utiliza la extensión de archivo **.scss** y es totalmente compatible con la sintaxis CSS.

Utilizando la extensión de archivo **.sass** no es totalmente compatible con la sintaxis CSS, pero es más rápido de escribir.

SCSS VS SASS

SCSS SYNTAX

```scss
$font-stack: Helvetica,sans-serif;
$primary-color:#333;

body{
font:100% $font-stack;
color:$primary-color;
}
```

Extensión *.scss
Requiere el uso de {}
Requiere el uso de ;

SASS SYNTAX

```sass
$font-stack: Helvetica,sans-serif
$primary-color:#333

body
font:100% $font-stack
color:$primary-color
```

Extensión *.sass
No requiere el uso de {}
No requiere el uso de ;

Nesting

Al escribir **HTML**, probablemente has visto que tiene una **jerarquía visual y anidada** clara. **CSS**, por otro lado, **no lo hace**.

Sass permite anidar sus selectores CSS de una manera que siga la **misma jerarquía visual de su HTML**.

EXAMPLE

SCSS SYNTAX

```scss
nav {
  ul {
    margin: 0;
    padding: 0;
    list-style: none;
  }
  li {
    display: inline-block;
  }
  a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
  }
}
```

CSS SYNTAX

```css
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}
nav li {
  display: inline-block;
}
nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none;
}
```

Instalación

DOCKER

```
$ docker-compose run app npm i sass
```

NPM

```
$ npm install sass
```

Variables

Pueden almacenar información como colores, pilas de fuentes o cualquier valor de CSS que queramos reutilizar. **Sass** usa el símbolo **$** para convertir algo en una variable.

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

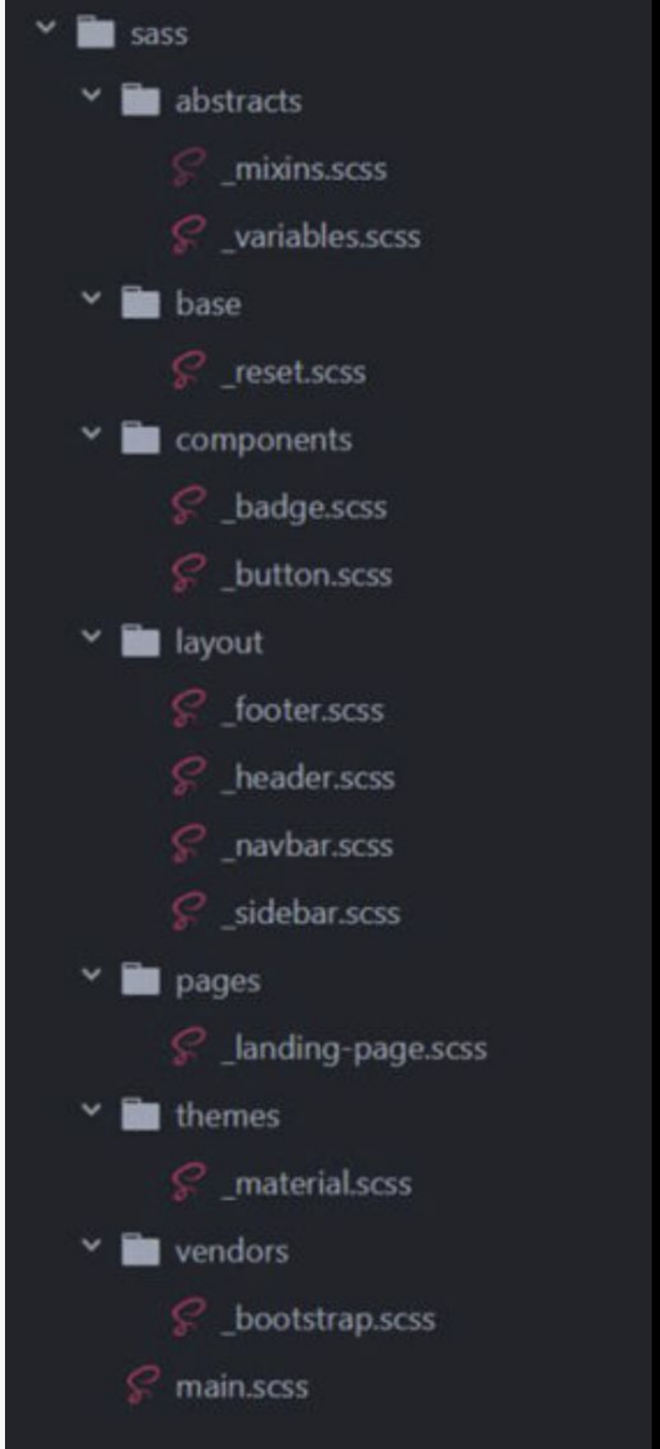
declaramos las variables

utilizamos las variables

Arquitectura 7 - 1

El **patrón de SASS 7-1** significa **7 carpetas** y **1 archivo**. 7 son las carpetas que tienen sus etiquetas significativas que incluyen fragmentos de sass o parciales. Fuera de las 7 carpetas, 1 archivo incluirá todos los parciales o fragmentos de 7 carpetas y lo compilará en 1 archivo CSS.

[7-1 pattern]



main.scss

En primer lugar, crearemos un archivo **main.scss** dentro de la carpeta **/assets/styles**.
Lo importaremos en **main.jsx** para que se apliquen estos estilos de manera global.

```
import React from 'react'
import ReactDOM from 'react-dom/client'

import './assets/styles/main.scss'
import App from './App/App.jsx'

ReactDOM.createRoot(document.getElementById('root')
).render(<App />)
```

colors.scss

Creamos un archivo **colors.scss** en la carpeta /**assets/styles/abstracts** que irá enlazada al archivo **main.scss** y definimos los colores principales de nuestra aplicación:

```
$green: #60A491;
$white: #FFFFFF;
$gray: #F5F5F4;
```

Import

La directiva **@import** permite incluir el contenido de un archivo en otro. De esta forma, cualquier variable o combinación definida se puede usar en el principal.

```scss
@import "./abstracts/colors";

body {
  font-family: Helvetica, sans-serif;
  font-size: 18px;
  background-color: $green;
  color: $gray;
}
```

importamos colors.scss

utilizamos las variables

Mixins

Un **mixin** permite crear grupos de declaraciones CSS reutilizables en todo el proyecto. Ayuda a mantener el principio **DRY** (Dont Repeat Yourself)

```scss
@mixin color-invertido($color: #111) {
   background-color: $color;
   color: #eee;
}
h1.invertido {
   font-size: 1.3rem;
   padding: 1rem;
   @include color-invertido(red);
}
div.invertido {
   padding: 5px;
   @include color-invertido;
}
blockquote {
   margin: 2rem 3eem;
   padding: 1.5rem;
   text-align: center;
   @include color-invertido;
}
```

Mixins

La **interpolación** se puede utilizar casi en cualquier parte de una hoja de estilo Sass para incrustar el resultado de una expresión SassScript en un fragmento de CSS. Simplemente ajusta una expresión en **#{}**:

```
@mixin corner-icon(
   $name, $top-or-bottom, $left-or-right) {
   .icon-#{$name} {
      background-image: url("/icons/#{$name}.svg"
      position: absolute;
      #{$top-or-bottom}: 0;
      #{$left-or-right}: 0;
   }
}

@include corner-icon("mail", top, left);
```

extend / herencia

El uso de **@extend** le permite compartir un conjunto de propiedades CSS de un selector a otro.

```
%message-shared {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}
.message {
  @extend %message-shared;
}
.success {
  @extend %message-shared;
  border-color: green;
}
```