Documentação: Configuração e Uso de Arquivos de Estilo SCSS no Angular

No Angular, é possível configurar e utilizar arquivos SCSS tanto em nível global quanto em nível local para componentes. Essa abordagem permite criar um estilo centralizado e reutilizável, simplificando a manutenção e organização do código.

---

1. Configuração Global de Estilos no Angular

O Angular permite configurar arquivos de estilo globais no arquivo `angular.json`. Isso facilita o acesso a estilos e variáveis SCSS em qualquer componente, sem a necessidade de usar caminhos relativos.

1.1 Configurando no `angular.json`

No objeto `options` do seu projeto, adicione o caminho do arquivo SCSS global em `styles`:

```
"styles": [
  "src/styles.scss"
]
```

Além disso, para evitar o uso de caminhos relativos nos arquivos SCSS, configure `stylePreprocessorOptions`:

```
"stylePreprocessorOptions": {
  "includePaths": ["src"]
}
```

Exemplo Completo no `angular.json`:

```
"projects": {
  "componentes-estilizacoes": {
    "architect": {
      "build": {
        "options": {
          "styles": [
            "src/styles.scss"
          ],
          "stylePreprocessorOptions": {
            "includePaths": ["src"]
          }
        }
```

```
    }
  }
 }
}
```

---

2. Uso de Arquivos de Estilo SCSS no Angular

2.1 Importação Global

Com a configuração acima, você pode importar o arquivo `styles.scss` diretamente em qualquer componente usando um caminho simplificado:

```
@import "styles.scss";

.card-cancel-button {
  background-color: $bgColor; // Variável definida no styles.scss
}
```

2.2 Uso Local com Caminhos Relativos

Se preferir não configurar `includePaths`, use caminhos relativos para importar o arquivo SCSS global:

```
@import "../../styles.scss"; // Caminho relativo ao componente

.card-cancel-button {
  background-color: $bgColor; // Variável definida no styles.scss
}
```

---

3. Exemplo de Estilo Global e Local

3.1 Arquivo `styles.scss`

```
// src/styles.scss
$primary-color: #3498db;
$bgColor: #ffcccc;

@mixin center {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

3.2 Estilo de um Componente

```
// src/app/components/button/button.component.scss
@import "styles.scss";

.button {
  @include center; // Usando o mixin definido no styles.scss
  background-color: $primary-color;
  color: white;
  padding: 10px 20px;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}

.button:hover {
  background-color: darken($primary-color, 10%);
}
```

---

4. Benefícios da Configuração Global

1. **Redução de Código Repetido**: Variáveis e mixins podem ser definidos em um único local e reutilizados.
2. **Manutenção Simplificada**: Alterações em um estilo global são automaticamente refletidas em todos os componentes que o utilizam.
3. **Organização do Código**: Ajuda a manter uma estrutura limpa e organizada, especialmente em projetos grandes.

---

5. Testando a Configuração

Após as alterações, reinicie o servidor de desenvolvimento para garantir que as configurações sejam aplicadas:

```
ng serve
```

---

6. Casos de Uso e Exemplos

6.1 Usando Variáveis Globais

```
@import "styles.scss";

.alert {
  color: $primary-color;
  background-color: lighten($primary-color, 40%);
```

```
  padding: 10px;
  border: 1px solid $primary-color;
}
```

6.2 Usando Mixins para Estilização Reutilizável

```
@mixin shadow($color: rgba(0, 0, 0, 0.2), $size: 2px) {
  box-shadow: 0 0 $size $color;
}

.card {
  @include shadow;
  background-color: white;
  padding: 20px;
  border-radius: 10px;
}
```

6.3 Estilização Condicional com SCSS

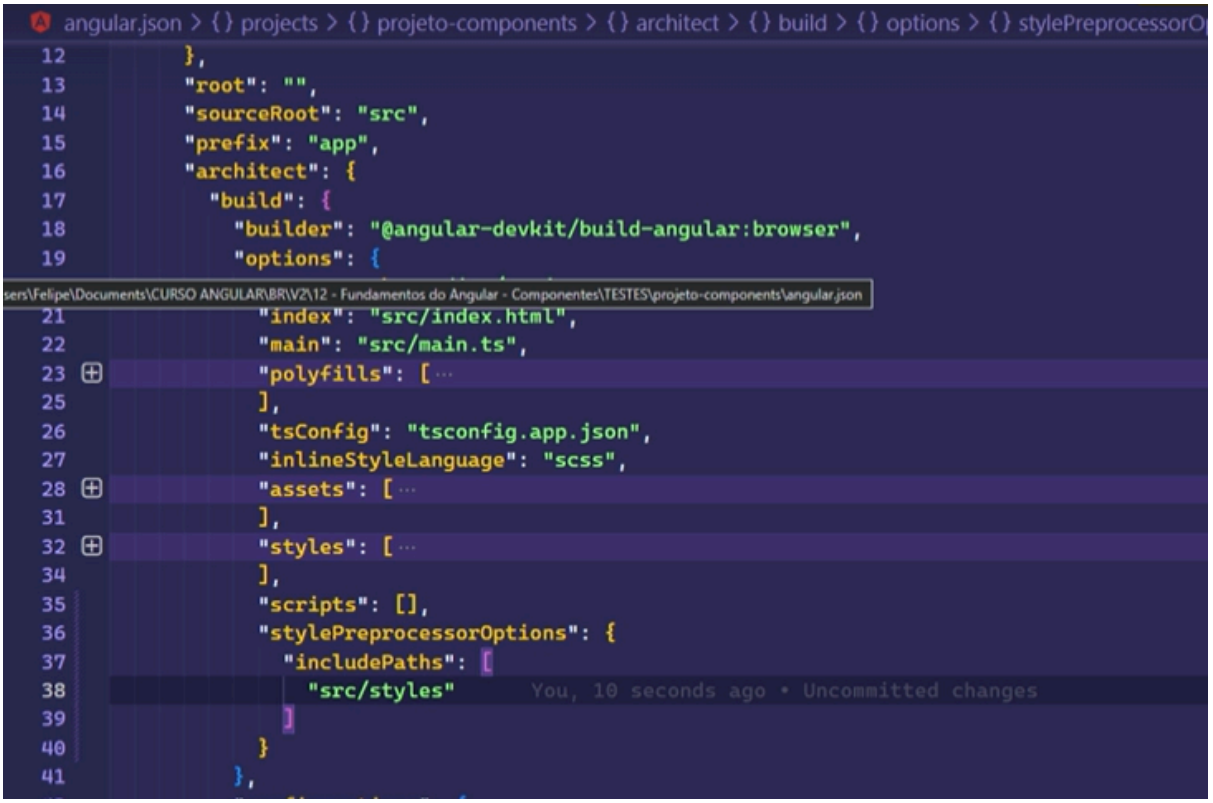
```
.button {
  @if $primary-color == #3498db {
    border: 2px solid $primary-color;
  } @else {
    border: 2px dashed red;
  }
}
```

---

7. Erros Comuns e Soluções

1. **Erro:** `Can't find stylesheet to import`
    - **Causa**: Caminho incorreto ou falta de configuração no `angular.json`.
    - **Solução**: Certifique-se de que o arquivo `styles.scss` está listado em `styles` e que `stylePreprocessorOptions` está configurado corretamente.
2. **Erro: Variável SCSS não definida**
    - **Causa**: O arquivo que define a variável não foi importado.
    - **Solução**: Adicione `@import "styles.scss";` no início do arquivo SCSS do componente.

Com essas práticas, você pode gerenciar os estilos do seu projeto Angular de forma eficiente e reutilizável.



Você pode criar dentro da pasta src uma pasta chamada styles e um arquivo para as variaveis chamado de variables.scss, coloque esse

```
"stylePreprocessorOptions": {
        "includePaths": [
                        "src/styles"
                ]
        }
```

aí no arquivo que você quer importar para usar as variaveis no estilo, lá no arquivo scss você fazer o import
@import "variables.scss";