SOFTEX
PERNAMBUCO

Softex

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO

GOVERNO FEDERAL
BRASIL
UNIÃO E RECONSTRUÇÃO

Aula 14 | Módulo: HTML + CSS / SASS (continuação)

- ➔ Introdução ao Tailwind CSS: conceitos e utilitários principais
- ➔ Projeto prático: criação de uma landing page responsiva com SASS ou Tailwind

Abrindo editor de código

Vamos agora abrir o VSCode e criar os arquivos

- Com o programa aberto, clique em File > Open Folder... (Arquivo > Abrir Pasta...).
- Escolha um local para criar a sua pasta, crie uma nova pasta e dê o nome de **seunome_aula_15**. Depois dê dois clique nessa pasta criada e clique em **Selecionar pasta**. O VSCode reabrirá dentro dessa pasta que foi criada.
- Agora vamos criar os arquivos HTML, JS e CSS:
- Dê o nome de **index.html** , **script.js** e **style.css**

HTML + CSS / SASS (continuação)

index.html > ...

```html
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Aula 15</title>
</head>
<body>
    <h1>Meu site</h1>
    <script src="script.js"></script>
</body>
</html>
```

O que é o Tailwind CSS?

- Framework CSS baseado em classes utilitárias.
- Foco em velocidade de desenvolvimento e consistência.
- Permite escrever menos CSS customizado.
- Muito usado em projetos modernos.

HTML + CSS / SASS (continuação)

Vantagens do Tailwind

- Montar interfaces muito rápido.
- Mantém um padrão de espaçamento, cores e fontes.
- Responsividade fácil usando classes como **sm**, **md**, **lg**, **xl**
- Configuração no arquivo tailwind.config.js.
- Plugins e comunidade ativa.

HTML + CSS / SASS (continuação)

Configuração Inicial

- Certifique que o NodeJS está instalado:
  - **node -v**
  - **npm -v**
- Caso não esteja instalado, baixe e instale:
  - **https://www.nodejs.tech/pt-br/download**
- Instale os 3 pacotes abaixo:
  - **npm init -y** (criar o package.json, que controla as dependências do projeto.)
  - **npm install -D tailwindcss@3 postcss autoprefixer**

HTML + CSS / SASS (continuação)

tailwindcss@3

- Esse é o framework principal que iremos utilizar.
- Vamos usar a versão 3 porque a versão 4.x ainda está muito nova e pode dar incompatibilidade, principalmente no Windows com npx.

HTML + CSS / SASS (continuação)

postcss

- É uma ferramenta que processa CSS com plugins.
- O Tailwind precisa do PostCSS para processar o CSS de entrada e então gerar o CSS final.
- Ele funciona como um compilador de CSS: lê seus arquivos e aplica as transformações.
- Sem PostCSS, o Tailwind não conseguiria gerar o output.css.

HTML + CSS / SASS (continuação)

autoprefixer

- É um plugin do PostCSS.
- Ele adiciona automaticamente os prefixos necessários para compatibilidade entre navegadores (como -webkit-, -moz-).
- Exemplo: se você usa display: flex, ele gera também -webkit-flex quando necessário.
- Isso evita problemas em navegadores mais antigos.
- Sem Autoprefixer, seu site pode quebrar em alguns navegadores.

HTML + CSS / SASS (continuação)

Configuração Inicial

- Criando arquivo de configuração: **npx tailwindcss init**
- Se apresentar algum erro, rode estes comandos:
  - **del package-lock.json**
  - Instale novamente: **npm install -D tailwindcss@3 postcss autoprefixer**
- Tente novamente criar o arquivo tailwind.config.js: **npx tailwindcss init**
- Se ainda apresentar erro, tente: **.\node_modules\.bin\tailwindcss.cmd init**

HTML + CSS / SASS (continuação)

Editando o arquivo de configuração

- No arquivo gerado (tailwind.config.js), precisamos indicar onde o Tailwind deve procurar os arquivos HTML/JS para remover CSS não utilizado:
- Adicione o conteúdo abaixo no **content** do arquivo **tailwind.config.js**:
  - **"./*.html", "./src/**/*.{js,jsx}"**

HTML + CSS / SASS (continuação)

Explicação sobre "./*.html", "./src//*.{js,jsx}"**

- **"./*.html"**
  - **./** significa a raiz do projeto (a mesma pasta onde está o tailwind.config.js).
  - ***.html** significa todos os arquivos que terminam com .html.
  - Ou seja: pega qualquer HTML que esteja diretamente na raiz do projeto.
  - Exemplo: Se você tiver index.html e contato.html na raiz, o Tailwind vai escanear esses arquivos.

HTML + CSS / SASS (continuação)

Explicação sobre "./*.html", "./src//*.{js,jsx}"**

- **"./src/**/*.{js,jsx}"**
  - **./src/** indica a pasta chamada src (onde geralmente ficam os arquivos de código).
  - **/ significa qualquer subpasta dentro de src (pode ser components, pages, etc.).
  - * significa qualquer nome de arquivo.
  - **{js,jsx}** significa extensões permitidas: **.js** e **.jsx**
  - Em outras palavras: “Escaneie todos os arquivos JavaScript e React (JSX) dentro da pasta src e de todas as suas subpastas”.

Exemplos práticos

Se você tem essa estrutura:

```
src/
├─ index.js
├─ App.jsx
├─ components/
│    └─ Navbar.jsx
└─ pages/
     └─ Home.js
```

O Tailwind vai procurar classes em:

- index.js
- App.jsx
- components/Navbar.jsx
- pages/Home.js

HTML + CSS / SASS (continuação)

Como funcionam as classes do Tailwind

- Diferente do CSS tradicional, no Tailwind cada classe faz uma única coisa.
- Você combina várias classes para montar o estilo.
- Imagine um blocos de Lego: simples individualmente, mas poderosos/complexos quando combinados.

Principais categorias do Tailwind

- Spacing (espaçamento): m-4 (margin), p-2 (padding).
- Cores: bg-red-500 (fundo vermelho), text-green-700 (texto verde).
- Tipografia: text-lg, font-bold, uppercase.
- Bordas: border, border-2, rounded-lg.
- Sombras e efeitos: shadow, shadow-lg, opacity-50.

HTML + CSS / SASS (continuação)

Exemplo prático

Sem Tailwind (CSS tradicional):

```
<style>
    .botao {
        background-color: #3498db; color: white;
        padding: 8px 16px; border-radius: 6px;
    }
</style>
<button class="botao">Clique aqui</button>
```

Com Tailwind (classes utilitárias):

```
<button class="bg-blue-500 text-white px-4 py-2 rounded-md">
    Clique aqui
</button>
```

Resultado visual: o mesmo botão.

Diferença: no Tailwind não criamos um arquivo CSS separado, usamos classes já prontas.

HTML + CSS / SASS (continuação)

Como compilar o Tailwind

Dentro do arquivo **style.css** adicionamos as diretivas:

@tailwind base;
@tailwind components;
@tailwind utilities;

style.css

```
1   @tailwind base;
2   @tailwind components;
3   @tailwind utilities;
```

Adicione agora o código abaixo no seu **index.html**:

```
<body>
    <button class="bg-blue-500 text-white px-4 py-2 rounded-md">Clique aqui</button>
</body>
```

E no HEAD do index.html, importamos o output.css que será compilado:

<link rel="stylesheet" href="output.css">

HTML + CSS / SASS (continuação)

Como compilar o Tailwind

Para compilar, rode o comando abaixo no terminal:

npx tailwindcss -i ./style.css -o ./output.css --watch

Explicação:

- **-i** arquivo de entrada (style.css)
- **-o** arquivo de saída (output.css)
- **--watch** recompila sempre que houver mudanças

Criando layouts responsivos com Tailwind

- Por que responsividade importa?
  - Os sites precisam se adaptar a diferentes telas (celular, tablet, desktop).
  - Antes, era comum criar vários arquivos CSS (ex: mobile.css, desktop.css).
  - O Tailwind simplifica isso usando prefixos de breakpoints.

HTML + CSS / SASS (continuação)

Breakpoints do Tailwind

- O Tailwind vem com breakpoints pré-configurados (que podem ser alterados no tailwind.config.js):
  - sm: Small (≥ 640px)
  - md: Medium (≥ 768px)
  - lg: Large (≥ 1024px)
  - xl: Extra Large (≥ 1280px)
  - 2xl: 2x Extra Large (≥ 1536px)
- Funcionam como condições: a classe só é aplicada se a tela tiver pelo menos aquela largura.

HTML + CSS / SASS (continuação)

Exemplo com texto

```
<body>
    <p class="text-sm md:text-lg lg:text-2xl">
        Texto que cresce conforme a tela aumenta.
    </p>
</body>
```

- Em telas pequenas: **text-sm**
- A partir de 768px (md): **text-lg**
- A partir de 1024px (lg): **text-2xl**

HTML + CSS / SASS (continuação)

Exemplo com cores

```
<body>
    <div class="bg-red-300 sm:bg-green-300 md:bg-blue-300 lg:bg-yellow-300 p-6">
        Teste de cores responsivas
    </div>
</body>
```

- **Mobile:** vermelho
- **Tablet:** verde
- **Desktop médio:** azul
- **Desktop grande:** amarelo

HTML + CSS / SASS (continuação)

Exemplo com layout (grid)

```
<body>
    <section class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-4">
        <div class="bg-gray-200 p-4">Item 1</div>
        <div class="bg-gray-200 p-4">Item 2</div>
        <div class="bg-gray-200 p-4">Item 3</div>
    </section>
</body>
```

- **Mobile:** 1 coluna
- **Tablet (md):** 2 colunas
- **Desktop (lg):** 3 colunas

Como consultar a documentação do Tailwind CSS

- Como estamos usando a versão 3, o site oficial é: **https://v3.tailwindcss.com/docs**
- É atualizado pela própria equipe do Tailwind.
- Contém todos os utilitários disponíveis (cores, espaçamentos, tipografia, grid, etc.).
- Possui exemplos prontos que você pode copiar e adaptar.

HTML + CSS / SASS (continuação)

Como usar a documentação na prática

- Primeiro vamos criar 3 divs no nosso index.html:

```
<body>
    <div class="">DIV 1</div>
    <div class="">DIV 2</div>
    <div class="">DIV 3</div>
</body>
```

HTML + CSS / SASS (continuação)

Como usar a documentação na prática

- Agora vamos começar a procurar pelos estilos na barra de pesquisa do site:
  - Digite **background color**, pegue 3 estilos e coloque dentro das classes de cada uma das divs.
  - Digite **margin**, pegue 3 estilos e faça o mesmo que fez acima.

```
<body>
    <div class="bg-sky-500/100 m-5">DIV 1</div>
    <div class="bg-sky-500/75 m-5">DIV 2</div>
    <div class="bg-sky-500/50 m-5">DIV 3</div>
</body>
```

Deve ficar algo parecido com esse código acima.

HTML + CSS / SASS (continuação)

Como usar a documentação na prática

- Agora sozinhos, procure por esses estilos abaixo e adicione também nas divs:
  - Adicione uma **border radius** nas divs.
  - Adicione um **padding** nas divs.
  - Adicione **font size** nas divs.

HTML + CSS / SASS (continuação)

Entrega para a Próxima Aula

- Criar a estrutura inicial da Landing Page no index.html, com o link para o output.css funcionando.
- Montar pelo menos estas partes:
  - Navbar (com logo e links de navegação).
  - Seção Hero (título, subtítulo e um botão).
- Seguir o planejamento de design apresentado:
  - Usar cores neutras + destaque em azul (bg-blue-500, text-blue-600).
  - Fonte padrão do Tailwind (font-sans).
  - Espaçamentos consistentes (p-6, m-4).
  - Layout mobile-first ajustado com sm:, md:, lg:.

Dica:

- Teste sua página em celular e desktop (redimensionando o navegador).
- Se algo não funcionar, revise o tailwind.config.js e se o output.css está sendo atualizado.

HTML + CSS / SASS (continuação)

index.html > ...

```html
<!DOCTYPE html>
<html lang="pt-BR">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="output.css">
    <title>Minha Landing Page</title>
</head>

<body class="font-sans antialiased">
    <!-- Aqui vamos construir cada seção -->
</body>

</html>
```

ATÉ A PRÓXIMA AULA!

Front-end - Design. Integração. Experiência.

Professor: Hygor Rasec
https://www.linkedin.com/in/hygorrasec
https://github.com/hygorrasec