Guia de Estilos – Montando Páginas Com Componentes

1. Aula 1 – Figma e Guia de Estilos:

1.1. Baixei o app desktop do figma;

1.2. Recolhendo recursos:

1.2.1. O primeiro passo é recolher todas as informações que temos disponíveis da página no modelo do figma, tais como imagens, logos e afins.

1.2.2. Para fazer isso basta clicar 2 vezes na imagem que deseja baixar e na barra lateral direita clicar em exportar, escolhendo o tipo do arquivo.

1.2.3. Vamos saltar todas as imagens e logos na pasta img dentro da pasta assets dentro do projeto.

1.2.4. Sempre bom deixar tudo organizado, portanto, todos os ícones separados em uma pasta de ícones, logos em logos e assim por diante.

1.2.5. Todo projeto tem uma página de guia de estilos onde tem todas as cores, fontes, espaçamentos e afins que utilizaremos para executar o projeto.

1.2.6. Site para fontes gratuitas: https://fonts.google.com.

1.3. Criando um Componente:

1.3.1. Nada novo, só criamos um botão.

1.4. O que aprendemos:

1.4.1. Importar o projeto no Figma;

1.4.2. Recolher recursos para o projeto;

1.4.3. Preparar a estrutura de arquivos;

1.4.4. Criar o componente de botão.

2. Aula 2 – Arquitetura e Ferramentas:

2.1. EMMET e Intellisense:

2.1.1. Nada novo

2.2. Variáveis CSS:

2.2.1. Podemos criar variáveis que guardem valores de cores e nomear elas com a cor correspondente, bem mais simples que lembrar o código dela. Mas nada novo ainda.

2.3. O que aprendemos:

2.3.1. Como baixar e instalar plugins no Visual Studio Code;

2.3.2. O uso de variáveis;

2.3.3. Padronização;

2.3.4. Import CSS;

2.3.5. Criar uma folha de estilos única que contém todas as informações de estilos mínimas para uma página.

3. Aula 3 – Os Primeiros Componentes:

3.1. O input:

3.1.1. Podemos import outros arquivos css dentro de um único usando o @import url(_caminho_do_arquivo);. Note que antes do nome do arquivo importado tem um '_' isso é sempre uma boa prática.

3.1.2. Outra boa prática é criar um arquivo css 'base.css' onde ficará a escrita de todo o css base da nossa página e assim vamos importando sempre os outros arquivos -como o de variáveis ou o reset) para que sempre que tivermos uma página com os mesmos componentes que a primeira -sendo esse um caso super comum, como botão e afins- basta importar somente 1 arquivo html para ele e colocar as classes gerais, criando um arquivo separado depois somente para fazer as alterações específicas de cada componente do site que seja necessário fazer.

3.1.3. Valor inherit das propriedades diz respeito a herdar aquele valor do que foi colocado no corpo página, tais como a font-family/size e color do texto.

3.1.4. Em um input existe o placeholder que indica para que serve aquele input, mas quando escrevemos nele, o placeholder se perde e o leitor de tela não consegue mais saber para que serve aquele input, perdendo a acessibilidade dele. Para corrigir isso podemos colocar um aria-label no input com o mesmo texto do placeholder.

3.2. Para saber mais:

3.2.1. https://medium.com/trainingcenter/bem-em-5min-f5c80fd23439

3.2.2. http://getbem.com

3.3. O cartão:

3.3.1. Antes de criar um componente precisamos ver no nosso modelo em quais outros lugares esse componente pode ser utilizado e, se ele não for utilizado em mais nenhum outro lugar, mas partes dele sim, então nós

dividimos o que seria apenas 1 componente em vários separados para que dessa forma possamos reutilizar esses componentes o máximo possível.

3.3.2. No caso do nosso cartão, não iriamos utilizar o cartão todo, com o título composto e a lista com ícones em nenhum outro lugar, mas precisaremos utilizar eles separados em vários lugares, tais como somente o título composto, ou somente o card ou somente a lista com ícones. Nesse caso o melhor é criar vários componentes separados e reutilizá-los durante o projeto.

3.4. O título:

3.4.1. Como o título ocorre em muitos pontos de várias paginas e não só isso, mas toda a tipografia vai ser a mesma em todos os textos diferenciando somente nos destaques dos títulos, podemos colocar direto no documento base.css. Como o nosso título tem apenas 1 parte que se destaca, podemos criar um span em uma h2 e colocar a classe de destaque do título junto com o escrito dele dentro dessa tag, dando o diferencial.

3.4.2. Display: block; força a quebra de linha quando usamos em textos. Bom pra títulos com uma parte com destaque e tag span.

3.5. Para saber mais – Componentes:

3.5.1. Durante o processo de desenvolvimento de uma página e quando vamos componentizar algo, precisamos entender e diferenciar componentes.

3.5.2. Alguns dos passos para diferenciar componentes é isolar partes diferentes do que queremos componentizar e ver se são usadas em outros lugares. Se sim, quer dizer que dentro do componente existem outros possíveis componentes, e o que queremos extrair na verdade é apenas a "casca". Se não, então o conjunto inteiro forma um componente, tanto o conteúdo quanto o que contém essas informações.

3.6. O que aprendemos:

3.6.1. Como baixar e instalar plugins no Visual Studio Code;

3.6.2. O uso de variáveis;

3.6.3. Padronização;

3.6.4. Import CSS;

3.6.5. Criar uma folha de estilos única que contém todas as informações de estilos mínimas para uma página.

4. **Aula 4 – Outros Componentes:**
    4.1. O item:
        4.1.1. Além da técnica de usar o ::before para colocar ícones, também podemos utilizar a tag spam para isso.
    4.2. O produto:
    4.3. Styleguide - O vídeo:
    4.4. O cabeçalho:
    4.5. O que aprendemos:
        4.5.1. Separar componentes;
        4.5.2. Lidar com estilizações idênticas com conteúdos diferentes.
5. **Aula 5 – Iniciando a Página:**
    5.1. O rodapé:
        5.1.1. Colocar title=”nome_da_media_social” para que o leitor de tela saiba para onde o link vai.
    5.2. O que aprendemos:
        5.2.1. Como planejar o desenvolvimento de uma página com as ferramentas que criamos;
        5.2.2. Maneiras de implementar componentes;
        5.2.3. Importância de uma documentação.

6. **Aula 6 – Encerrando a Página:**
    6.1. As primeiras seções:
    6.2. Seção de produtos:
        6.2.1. Ao colocar {$} quando usando o Emmet e o seu multiplicador ele faz a contagem do elemento:

```
ul>li*6{$}
<ul>
      <li>1</li>
      <li>2</li>
      <li>3</li>
      <li>4</li>
      <li>5</li>
      <li>6</li>
    </ul>
```

6.2.2. Para colocar o $ como símbolo e não contador, usamos ‘\’ antes dele.

6.3. Seção Vídeos:

6.4. O que aprendemos:

6.4.1. Como implementar tudo o que fizemos até agora no curso;

6.4.2. Estruturas mais complexas com EMMET.