01

Estilizando um componente

Transcrição

Começando deste ponto? Você pode fazer o DOWNLOAD (https://s3.amazonaws.com/caelum-online-public/vue/stages/04-alurapic.zip) completo do projeto do capítulo anterior e continuar seus estudos a partir deste capítulo. Será necessário abrir seu terminal, entrar dentro da pasta alurapic e executar o comando npm install para baixar novamente todas as dependências da aplicação.

Até agora não lidamos com a parte estética da nossa aplicação e sabemos que no mundo web, o CSS é aquela tecnologia responsável pela aplicação de estilos. Nela, através da tag `link` importamos folhas de estilos que podem estar em um ou mais arquivos. No entanto, nossa aplicação utilizará uma outra abordagem. Vejamos.

Como Vue trabalha com o conceito de componente, a ideia é definirmos os estilos no próprio componente. Sendo assim, qualquer desenvolvedor que olhe o código do componente compreenderá os estilos que operam sobre ele. Vamos olhar o arquivo `alurapic/src/App.vue` .

Pode parecer estranho em um primeiro momento, mas `App.vue` é um componente. Dessa forma, páginas em uma aplicação com vue são todas componentes. Nesse sentido, precisamos realizar os seguintes ajustes em nosso componente:

- Ajustes:
  - centralizar o título
  - adicionar uma margem direita e esquerda de mesmo tamanho na página inteira
  - utilizar a fonte Helvetica
  - remover o bullet que aparece antes de cada item da lista
  - cada foto deve ir ao lado da outra

Primeiro, vamos adicionar as classes "corpo", "centralizado" e "lista-fotos", 'lista-fotos-item" nas tags `div` , `h1` , `ul` e `li` respectivamente.

```
<!-- alurapic/src/App.vue -->

<template>

  <div class="corpo">
    <h1 class="centralizado">{{ titulo }}</h1>

    <ul class="lista-fotos">
      <li class="lista-fotos-item" v-for="foto of fotos">
        <img :src="foto.url" :alt="foto.titulo">
      </li>
    </ul>

  </div>
</template>

<script>

    // código omitido

</script>
```

```
<style>
</style>
```

COPIAR CÓDIGO

Agora, dentro da tag `style` vamos adicionar os estilos que atendem cada um dos nosso requisitos de design:

```
<!-- alurapic/src/App.vue -->
<template>

  <div class="corpo">
    <h1 class="centralizado">{{ titulo }}</h1>

    <ul class="lista-fotos">
      <li class="lista-fotos-item" v-for="foto of fotos">
        <img :src="foto.url" :alt="foto.titulo">
      </li>
    </ul>

  </div>
</template>

<script>

    // código omitido

</script>
<style>
  .centralizado {
    text-align: center;
  }

  .corpo {
    font-family: Helvetica, sans-serif;
```

```
    margin: 0 auto;
    width: 96%;
  }

  .lista-fotos {
    list-style: none;
  }

  .lista-fotos .lista-fotos-item {
    display: inline-block;
  }
</style>
```

COPIAR CÓDIGO

Usamos tags para não ficarmos amarrados à determinada estrutura do HTML. Por exemplo, se mais tarde alguém trocar nosso título de `h1` para `h2`, como estamos usando um seletor de classe, nosso estilo ainda continuará a ser aplicado.

Como livereloading, a alteração do nosso componente `App.vue` é refletida instantaneamente em nosso navegador. Temos o título centralizado entre outros estilos aplicados. Mas como o estilo definido em `App.vue` funciona, se estilos devem ser declarados ou definidos na tag `<head>`. Aliás, nosso componente nem possui essa tag!

Isso funciona, os estilo definido na tag `style` do componente é inserida em `index.html` assim que abrimos nossa aplicação no navegador. Não precisamos nos preocupar com isso, a própria infra do Vue se encarregará de fazer isso para nós.