

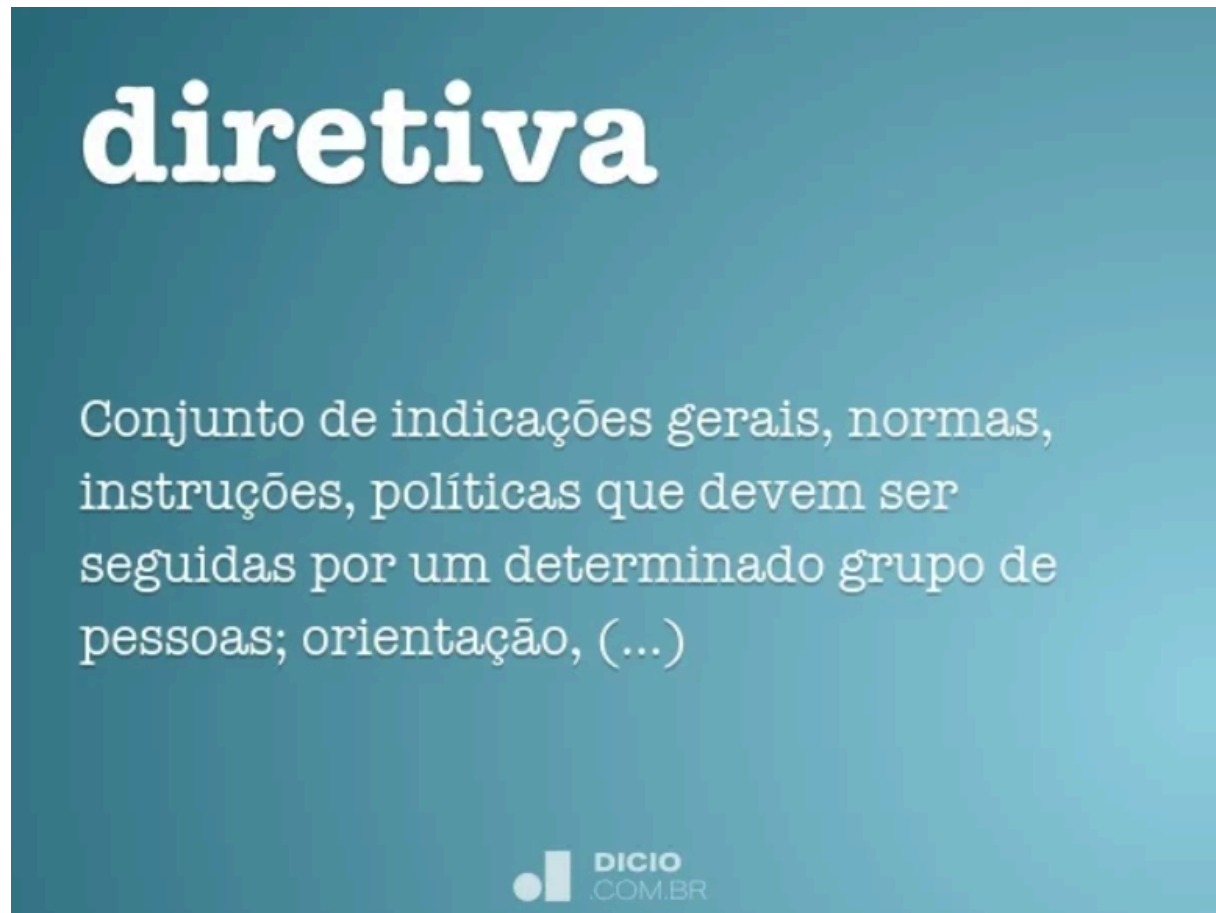
Módulo 1: Introdução ao Vue.js

- Diretivas
- v-bind
- v-model
- v-if
- v-else-if
- v-else
- v-for

Diretivas do Vue.js 2.7

As diretivas no Vue.js são atributos especiais usados em elementos do HTML para adicionar comportamentos reativos a esses elementos. Todas as diretivas do Vue.js são prefixadas com **v-**.

Segundo dicionário significado de diretiva é:



Vamos explorar as principais diretivas do Vue.js 2.7 com exemplos práticos:

v-bind

A diretiva **v-bind** é usada para vincular atributos do HTML a expressões dinâmicas. Isso é útil quando você precisa definir atributos dinamicamente com base nos dados da instância Vue.

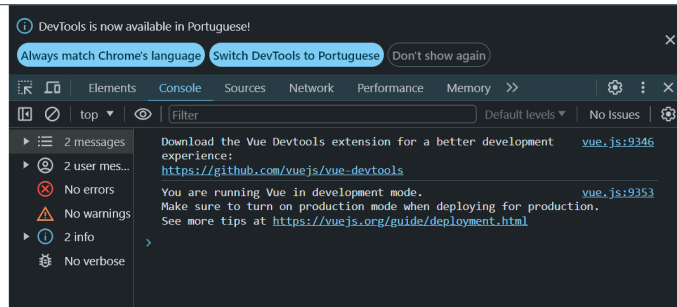
```
JS script.js  <> index.html X
ot02 > <> index.html > html
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <title>Aprendendo Vue</title>
7    <link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10
11    <div id="app">
12      <a v-bind:href="url">Visite o site do VueJS</a>
13    </div>
14
15    <!-- link de desenvolvimento vuejs -->
16    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
17    <!-- link do script que tem o vuejs estanciado -->
18    <script src="script.js"></script>
19  </body>
20 </html>
```

```
JS script.js  X  <> index.html
ot02 > JS script.js > [e] app
1  var app = new Vue({
2    el: '#app',
3    data: {
4      url: 'https://vuejs.org'
5    }
6  })
```

Neste exemplo, o atributo **href** do elemento **<a>** é vinculado ao valor da propriedade **url** na instância Vue.

Por fim teremos um link clicável:

[Visite o site do VueJS](#)



v-model

A diretiva **v-model** cria uma ligação bidirecional entre os dados da instância Vue e os elementos de formulário (como **<input>**, **<textarea>** e **<select>**). Isso significa que quando o valor do elemento do formulário muda, o valor correspondente nos dados do Vue também muda, e vice-versa.

Exemplo:

```
<div id="app">
  <input type="text" v-model="message" placeholder="Digite uma mensagem">
  <p>{{ message }}</p>
</div>
```

```
JS script.js  X  <> vmodel.html
ot02 > JS script.js > [app]
1   var app = new Vue({
2     |   el: '#app',
3     |   data: {
4     |     |   message: ''
5     |   }
6     | })
```

Neste exemplo, o valor do campo de entrada está vinculado à propriedade **message** nos dados da instância Vue. Qualquer alteração no campo de entrada atualizará automaticamente a propriedade **message** e refletirá no parágrafo.

Digite uma mensagem

Carlos Júnior Uchôa

Carlos Júnior Uchôa

v-if, v-else-if, v-else

As diretivas **v-if**, **v-else-if** e **v-else** são usadas para renderizar elementos condicionalmente. O elemento será renderizado apenas se a expressão associada for avaliada como verdadeira.

Exemplo:

```
<div id="app">
  <p v-if="seen">Agora você vê</p>
  <p v-else>Agora você não vê</p>
</div>
```

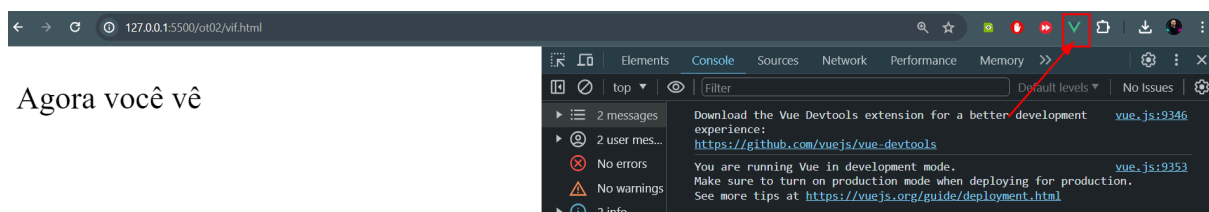
Neste exemplo, o parágrafo com **v-if** será renderizado se a propriedade **seen** for verdadeira. Caso contrário, o parágrafo com **v-else** será renderizado.

```
JS script.js  X  <> vif.html

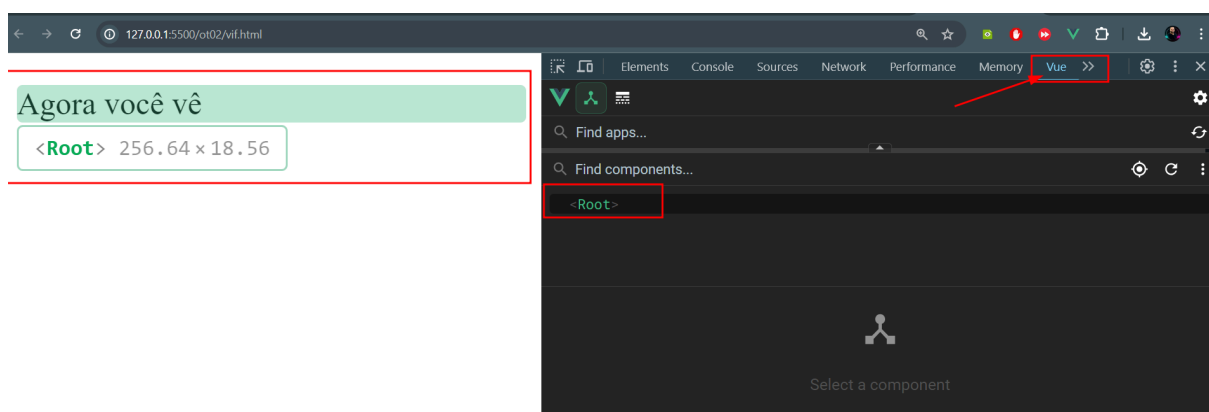
ot02 > JS script.js > [🔗] app
1  var app = new Vue({
2    |   el: '#app',
3    |   data: {
4    |     seen: true
5    |   }
6    | })
```

Podemos testar a mudança se vemos o site vai aparecer apenas o que tiver a nossa variável **seen**, pois **está como true**, mas podemos alterar para false pela extensão **vue-devtools** conforme imagem abaixo.

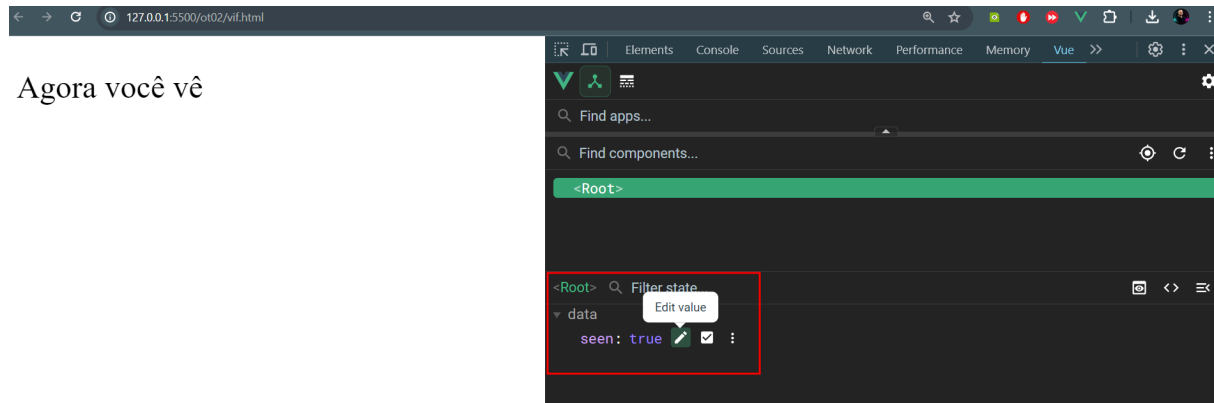
<https://devtools.vuejs.org/> – LINK PARA DOWNLOAD E INSTALAÇÃO



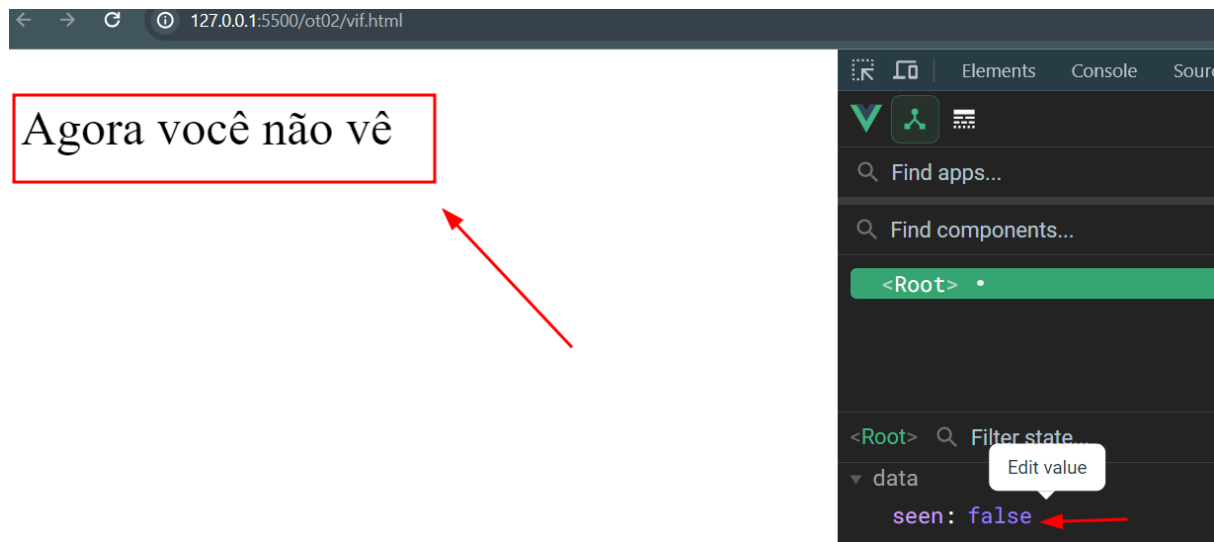
E como podem ver temos uma aba nova chamada **Vue** ao acessarmos teremos **acesso a todas os componentes , variáveis, funcionalidades iremos utilizar muito.**



Com isso podemos alterar o valor da variável em tempo real para testarmos e ao fazer isso colocamos ela como **“false”** e vamos ver o que acontece.



Com a alteração agora podemos ver que ele entrou no v-else:



Outro exemplo utilizando v-if, v-else, v-show

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Curso Vue JS - Diretivas v-if v-else intro</title>
7 </head>
8 <body>
9   <div id="app">
10    <h1 v-if="status">{{ titulo }}</h1>
11    <h1 v-else="status">Não existe título!</h1>
12    <h2 v-show="status">{{ titulo }}</h2>
13  </div>
14
15
16  <script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>
17  <script>
18    var app = new Vue({
19      el: '#app',
20      data: {
21        status: true, //so apresenta esse titulo se o status for true
22        titulo: 'Aprendendo Vue JS'
23      }
24    })
25  </script>
26 </body>
27 </html>
```

```
<!-- v-show esconde o elemento no código html -->
<!-- v-if retira o elemento do código html -->
<!-- São códigos para realizar uma função especifica no componente
v-if, v-show, v-for, v-text, v-html, v-else
-->
```

Explicação do código:

`<div id="app">`: Div onde a instância do Vue será montada.

`<h1 v-if="status">{{ titulo }}</h1>`: Diretiva `v-if` do Vue que renderiza este elemento `<h1>` apenas se a condição `status` for verdadeira. Se `status` for `true`, exibe o valor de `titulo`.

`<h1 v-else="status">Não existe título!</h1>`: Diretiva `v-else` do Vue que renderiza este elemento `<h1>` se a condição `v-if` for falsa. Se `status` for `false`, exibe "Não existe título!".

`<h2 v-show="status">{{ titulo }}</h2>`: Diretiva `v-show` do Vue que renderiza este elemento `<h2>` se a condição `status` for verdadeira. Diferente de `v-if`, o `v-show` mantém o elemento no DOM, mas o esconde se a condição for falsa.

- `var app = new Vue({ ... })`: Cria uma nova instância Vue.
- `el: '#app'`: Monta a instância Vue no elemento com id "app".

- `data: { ... }`: Define os dados que serão utilizados na instância Vue. Aqui, são definidos:
 - `status`: Um booleano inicializado como `true`.
 - `titulo`: Uma string com o valor 'Aprendendo Vue JS'.

Funcionamento

- Quando a página é carregada, a instância Vue é criada e vinculada ao elemento `#app`.
- A diretiva `v-if` verifica se `status` é verdadeiro:
 - Se `status` for `true`, exibe o `<h1>` com o valor de `titulo`.
 - Se `status` for `false`, exibe o `<h1>` com a mensagem "Não existe título!".
- A diretiva `v-show` exibe o `<h2>` com o valor de `titulo` apenas se `status` for `true`. Se `status` for `false`, o `<h2>` ainda estará no DOM, mas estará oculto.

`v-show`: Esconde o elemento no HTML, mas ele ainda está presente no DOM.

`v-if`: Remove o elemento completamente do DOM se a condição for falsa.

Comentários listando algumas diretivas Vue úteis para manipulação de elementos na interface.

Exemplo um pouco mais elaborado:

Estrutura:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Curso Vue JS - Diretivas v-if v-else</title>
</head>
<body>
  <div id="app">
    <div v-if="usuario.id == 1">
      ID: {{ usuario.id }},<br>
      Nome: {{ usuario.nome }},<br>
      Email: {{ usuario.email }},<br>
      Profissão: {{ usuario.profissao }}
    </div>
    <div v-else>
      Não existe usuário com o ID: 1
    </div>
  </div>
```

continuação da estrutura:

```

<script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>
<script>
  var app = new Vue({
    el: '#app',
    data: {
      usuario: {
        id: 1,
        nome: 'João',
        email: 'j@j.com',
        profissao: 'Programador'
      }
    }
  })
</script>
</body>
</html>
<!-- v-show esconde o elemento no código html -->
<!-- v-if retira o elemento do código html -->
<!-- São códigos para realizar uma função específica no componente
v-if, v-show, v-for, v-text, v-html, v-else
-->

```

Explicação do código:

`<div id="app">`: Div onde a instância do Vue será montada.

`<div v-if="usuario.id == 1">`: Diretiva `v-if` do Vue que renderiza este bloco apenas se a condição `usuario.id == 1` for verdadeira.

- Exibe as informações do usuário se o `id` for igual a 1.

`<div v-else>`: Diretiva `v-else` do Vue que renderiza este bloco se a condição `v-if` for falsa.

- Exibe a mensagem "Não existe usuário com o ID: 1" se o `id` do usuário não for 1.
- `var app = new Vue({ ... })`: Cria uma nova instância Vue.
- `el: '#app'`: Monta a instância Vue no elemento com id "app".
- `data: { ... }`: Define os dados que serão utilizados na instância Vue. Aqui, é definido um objeto `usuario` com as propriedades `id`, `nome`, `email` e `profissao`.

Funcionamento

- Quando a página é carregada, a instância Vue é criada e vinculada ao elemento `#app`.
- A diretiva `v-if` verifica se o `id` do usuário é igual a 1:
 - Se for verdadeiro (`id == 1`), exibe as informações do usuário.
 - Se for falso, exibe a mensagem "Não existe usuário com o ID: 1".

`v-show`: Esconde o elemento no HTML, mas ele ainda está presente no DOM.

v-if: Remove o elemento completamente do DOM se a condição for falsa.

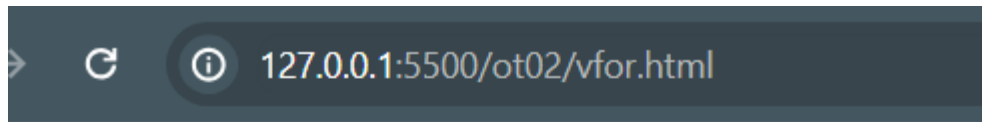
v-for

A diretiva **v-for** é usada para renderizar uma lista de itens usando um loop. É semelhante a um loop **for** em linguagens de programação.

Exemplo:

```
<div id="app">
  <ul id="example-1">
    <li v-for="item in items" :key="item.id">
      {{ item.text }}
    </li>
  </ul>
</div>
```

```
JS script.js  X  <> vfor.html
ot02 > JS script.js > [ ] app
1   var app = new Vue({
2     el: '#app',
3     data: {
4       items: [
5         {id: 1 , text: 'Item 1'},
6         {id: 2 , text: 'Item 2'},
7         {id: 3 , text: 'Item 3'}
8       ]
9     }
10  })
```



- Item 1
- Item 2
- Item 3

Neste exemplo, a diretiva **v-for** itera sobre a lista **items** e renderiza um item **** para cada objeto na lista. A propriedade **key** é usada para fornecer uma identificação única para cada item, o que é uma prática recomendada para ajudar o Vue a otimizar a renderização.

Outro exemplo utilizando v-for:

Estrutura:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Curso Vue JS - Diretivas v-for</title>
</head>
<body>
  <div id="app">
    <h1>{{ titulo }}</h1>
    <ul>
      <li v-for="item in linguagens">{{ item }}</li><br>
      <li v-for="item in linguagens">{{ item.nome }}</li><br>
      <li v-for="item in linguagens">{{ item.nome }} - {{ item.status }}</li><br>
      <li v-for="(item, index) in linguagens">{{ index + " - " }}{{ item }}</li><br>
      <li v-for="(item, index) in linguagens">{{ index + " - " }}{{ item.nome }}</li>
    </ul>
  </div>
```

Continuando a estrutura:

```
<script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>
<script>
  var app = new Vue({
    el: '#app',
    data: {
      titulo: 'Aprendendo Vue JS',
      linguagens: [
        { nome: 'HTML', status: true },
        { nome: 'CSS', status: true },
        { nome: 'JavaScript', status: false },
        { nome: 'PHP', status: false },
        { nome: 'Python', status: false },
      ]
    }
  })
</script>
</body>
</html>
<!-- v-show esconde o elemento no código html -->
<!-- v-if retira o elemento do código html -->
<!-- São códigos para realizar uma função específica no componente
v-if, v-show, v-for, v-text, v-html, v-else
-->
```

Explicação do código:

- `var app = new Vue({ ... })`: Cria uma nova instância Vue.
- `el: '#app'`: Monta a instância Vue no elemento com id "app".
- `data: { ... }`: Define os dados que serão utilizados na instância Vue. Aqui, são definidos:
 - `titulo`: Uma string com o valor 'Aprendendo Vue JS'.
 - `linguagens`: Uma lista de objetos, onde cada objeto representa uma linguagem de programação com as propriedades `nome` e `status`.

Funcionamento

- Quando a página é carregada, a instância Vue é criada e vinculada ao elemento `#app`.
- O título `titulo` é exibido no elemento `<h1>`.
- A diretiva `v-for` é usada para iterar sobre a lista `linguagens` e renderizar cada item de diferentes formas nos elementos ``:

- O primeiro `li` tenta exibir o objeto `item` diretamente, o que não é uma prática comum e pode resultar na conversão do objeto em uma string "[object Object]".
- O segundo `li` exibe a propriedade `nome` de cada objeto `item`.
- O terceiro `li` exibe as propriedades `nome` e `status` de cada objeto `item`, separados por " - ".
- O quarto `li` exibe o índice (`index`) e o objeto `item`, semelhante ao primeiro `li`.
- O quinto `li` exibe o índice (`index`) e a propriedade `nome` de cada objeto `item`.
- `v-show`: Esconde o elemento no HTML, mas ele ainda está presente no DOM.
- `v-if`: Remove o elemento completamente do DOM se a condição for falsa.

Conclusão

As diretivas são uma parte fundamental do Vue.js, pois permitem que você adicione comportamento reativo aos seus elementos de forma declarativa. Com essas diretivas, você pode criar interfaces dinâmicas e interativas de maneira simples e eficiente. Pratique usando essas diretivas em diferentes cenários para se familiarizar com seu funcionamento e potencial. Se precisar de mais exemplos ou esclarecimentos, estarei aqui para ajudar!

Vamos praticar?

Estrutura:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Atividade Vue JS - Diretivas v-if, v-else, v-show</title>
</head>
<body>
  <div id="app">
    <h1>{{ titulo }}</h1>

    <!-- Se o usuário estiver autenticado, exibir mensagem de boas-vindas -->
    <div v-if="autenticado">
      <h2>Bem-vindo, {{ usuario.nome }}!</h2>
      <button @click="logout">Sair</button>
    </div>

    <!-- Se o usuário não estiver autenticado, exibir formulário de login -->
    <div v-else>
      <h2>Faça login</h2>
      <form @submit.prevent="login">
        <label for="username">Usuário:</label>
        <input type="text" id="username" v-model="username" required><br><br>

        <label for="password">Senha:</label>
        <input type="password" id="password" v-model="password" required><br><br>

        <button type="submit">Entrar</button>
      </form>
    </div>
  </div>

```

Continuação:

```

    <button type="submit">Entrar</button>
  </form>
</div>

<!-- Mensagem de erro exibida com v-show -->
<p v-show="erro" style="color: red;">Nome de usuário ou senha incorretos.</p>
</div>

```

Continuação:

```

<script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>
<script>
  var app = new Vue({
    el: '#app',
    data: {
      titulo: 'Aplicação de Login',
      autenticado: false,
      usuario: {
        nome: 'João'
      },
      username: '',
      password: '',
      erro: false
    },
    methods: {
      login() {
        // Simulação de login
        if (this.username === 'usuario' && this.password === 'senha') {
          this.autenticado = true;
          this.erro = false;
        } else {
          this.erro = true;
        }
      },
      logout() {

```

Continuação:

```

      },
      logout() {
        this.autenticado = false;
        this.username = '';
        this.password = '';
      }
    }
  })
</script>
</body>
</html>

```

Explicação:

Estrutura do HTML:

- O documento HTML é configurado com a codificação UTF-8 e é responsivo.
- O título da página é definido como "Atividade Vue JS - Diretivas v-if, v-else, v-show".

Div com ID `app`:

- O elemento principal onde a instância Vue será montada.

Título da Aplicação:

- Exibe o título da aplicação.

Verificação de Autenticação com **v-if** e **v-else**:

- Se o usuário estiver autenticado (**autenticado** é **true**), exibe uma mensagem de boas-vindas e um botão para sair.
- Se o usuário não estiver autenticado (**autenticado** é **false**), exibe um formulário de login.

Formulário de Login:

- Contém campos para nome de usuário e senha, que são vinculados aos dados **username** e **password**.
- Quando o formulário é submetido, a função **login** é chamada para simular o processo de login.

```
<p v-show="erro" style="color: red;">Nome de usuário ou senha incorretos.</p>
```

- Exibe uma mensagem de erro em vermelho se a variável **erro** for **true**.

Definição da Instância Vue:

- Os dados incluem **titulo**, **autenticado**, **usuario**, **username**, **password** e **erro**.
- Métodos **login** e **logout** são definidos para simular o processo de autenticação e logout.

Testando a aplicação:



127.0.0.1:5500/aula03/praticaDiretivas.html

Aplicação de Login

Faça login

Usuário:



Senha:




Entrar

Ao tentar colocar um usuario e senha incorretos:

Aplicação de Login

Faça login

Usuário: 

Senha: 

Nome de usuário ou senha incorretos.

Ao colocarmos o usuário e senha que está configurado para serem iguais ao que pedimos:
vide exemplo:


```


methods: {
  login() {
    // Simulação de login
    if (this.username === 'usuario' && this.password === 'senha') {
      this.autenticado = true;
      this.erro = false;
    } else {
      this.erro = true;
    }
  },
  logout() {
    this.autenticado = false;
    this.username = '';
    this.password = '';
  }
}


```

Aplicação de Login

Faça login

Usuário: 

Senha: 

 senha

Ao clicarmos no botão “Entrar”:

Temos esta visão:

Aplicação de Login

Bem-vindo, João!

Sair

Apenas simulando uma aplicação com autenticação.

Tarefas para os Alunos

1. Implemente o código em um arquivo HTML.
2. Estude o funcionamento das diretivas `v-if`, `v-else` e `v-show`.
3. Experimente alterar os valores de `username` e `password` para testar a funcionalidade de login e exibição de erros.
4. Adicione mais funcionalidades, como um botão para redefinir o formulário de login