**SESSÃO 3 – Renderização condicionais e listas**

Links: <https://br.vuejs.org/v2/guide/conditional.html>

<https://br.vuejs.org/v2/guide/list.html>

**Aula 47 – Condicional v-if / v-else.**

1. Vamos criar um novo arquivo chamado condicional\_v\_if\_e\_V\_else.
2. Neste arquivo iremos configurar o arquivo.
3. <meta *charset*="UTF-8">
4. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
5. <div *id*="app">
6. <p *v-if*="logado">Usuário Logado: {{nome}}</p>
7. <p *v-if*="!logado">Nenhum usuário logado</p>
8. <button *v-on:click*="logado = !logado">{{ logado ? 'Sair' : 'Entrar'}}</button>
9. </div>
10. <script>
11. **new** Vue({
12. el:'#app' ,
13. data:{
14. nome: 'Maria',
15. logado: false
16. }
17. })
18. </script>
19. A lógica desta aplicação é que quando o usuário clicar sobre o botão o primeiro parágrafo será mostrado somente se a variável logado for true.
20. O segundo parágrafo será mostrado somente se a variável logado for false.
21. No botão temos a diretiva v-on:click que toda vez que o botão é clicado a variável logado muda de true para false ou vice-versa.
22. A outra opção para não mostrar o segundo parágrafo é utilizando a condicional v-else, porém temos que ter um v-if associado, senão não irá funcionar.
23. <div *id*="app">
24. <p *v-if*="logado">Usuário Logado: {{nome}}</p>
25. <p *v-else*>Nenhum usuário logado</p>
26. <button *v-on:click*="logado = !logado">{{ logado ? 'Sair' : 'Entrar'}}</button>
27. </div>

**Aula 48 – Seleção múltipla com v-else-if.**

1. Iremos ver como utilizar uma seleção múltipla para isso vamos copiar o arquivo anterior e iremos criar uma nova variável chamada anomino que irá receber o valor inicial de true. Vamos também adicionar um parágrafo intermediário e vamos utilizar condicional v-else-if passando para ela a variável anônimo, que em caso de true, o parágrafo será mostrado. Podemos ter quantos v-else-if quisermos.
2. Porém a execução do código sempre irá parar na primeira sentença verdadeira que encontrar.
3. <meta *charset*="UTF-8">
4. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
5. <div *id*="app">
6. <p *v-if*="logado">Usuário Logado: {{nome}}</p>
7. <p *v-else-if*="anonimo">Navegando como anônimo</p>
8. <p *v-else*>Nenhum usuário logado</p>
9. <button *v-on:click*="logado = !logado">{{ logado ? 'Sair' : 'Entrar'}}</button>
10. </div>
11. <script>
12. **new** Vue({
13. el:'#app' ,
14. data:{
15. nome: 'Maria',
16. logado: false,
17. anonimo: true,
18. }
19. })
20. </script>
21. Para utilizarmos a variável anônimo iremos utilizar um input do tipo checkbox e iremos utilizar um v-model. Desta forma quando o checkbox for selecionado a variável anônimo será alterada para true ou false.
22. <input *type*="checkbox" *v-model*="anonimo">

**Aula 49 – Usando v-if com template.**

1. Iremos utilizar o exemplo anterior, porém vamos adicionar uma div no inicio html. Dentro desta div iremos mover o parágrafo de usuário logado e criar outro parágrafo que terá o perfil do usuário.
2. <div *v-if*="logado">
3. <p>Usuário Logado: {{nome}}</p>
4. <p>Perfil: Admin</p>
5. </div>
6. Para não mostrarmos a div com os parágrafos dentro, temos que utilizar a tag template. Desta forma os parágrafos irão aparecer na página html, mas o template não.
7. <template *v-if*="logado">
8. <p>Usuário Logado: {{nome}}</p>
9. <p>Perfil: Admin</p>
10. </template>

**Aula 50 – v-show**

1. É utilizado para ocultar elementos da página.
2. Mas por motivos de segurança não deve ser utilizado para ocultar elementos que envolva segurança no acesso à página, por exemplo permissões de acesso.
3. Vamos utilizar o exemplo anterior.
4. Neste arquivo iremos criar um footer.
5. Dentro do footer iremos adicionar a condicionalde renderização v-show passando para ela a variável logado. Se for verdadeira o footer será mostrado caso contrário será oculto.
6. <meta *charset*="UTF-8">
7. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
8. <div *id*="app">
9. <template *v-if*="logado">
10. <p>Usuário Logado: {{nome}}</p>
11. <p>Perfil: Admin</p>
12. </template>
13. <p *v-else-if*="anonimo">Navegando como anônimo</p>
14. <p *v-else*>Nenhum usuário logado</p>
15. <button *v-on:click*="logado = !logado">{{ logado ? 'Sair' : 'Entrar'}}</button>
16. <input *type*="checkbox" *v-model*="anonimo">
17. <hr>
18. <footer *v-show*="logado">
19. <p>Desenvolvido por você.</p>
20. </footer>
21. </div>
22. <script>
23. **new** Vue({
24. el:'#app' ,
25. data:{
26. nome: 'Maria',
27. logado: false,
28. anonimo: true,
29. }
30. })
31. </script>

**Aula 51 – Renderização de listas – v-for**

1. Vamos configurar um exemplo vazio, porém iremos criar duas listas ou seja dois arrays dentro de data(). Neste exemplo iremos utilizar a princípio a lista de cores.
2. Dentro do template iremos criar uma ul e um li.
3. Iremos acrescentar um v-for na tag que será repetida.
4. E dentro de v-for nós criamos uma variável ‘cor’ que será utilizada para renderizar as cores recebendo do array ‘cores’. Dentro da tag li nós iremos interpolar a variável ‘cor’ que foi criada, desta forma será renderizado na tela todas cores.
5. <meta *charset*="UTF-8">
6. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
7. <div *id*="app">
8. <ul>
9. <li *v-for*="cor in cores">{{cor}}</li>
10. </ul>
11. </div>
12. <script>
13. **new** Vue({
14. el:'#app' ,
15. data:{
16. cores:['vermelho','verde','amarelo','azul'],
17. pessoas:[
18. {nome:'Ana', idade: 26, peso: 59},
19. {nome:'Guilherme', idade: 22, peso: 89}
20. ]
21. }
22. })
23. </script>

**Aula 52 – Acessando o índice atual**

1. Para termos acesso ao índice basta adicionarmos no v-for um parênteses que terá a variável cor e outra variável que será o índice.
2. <meta *charset*="UTF-8">
3. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
4. <div *id*="app">
5. <ul>
6. <li *v-for*="(cor, i) in cores">índice: {{i}} - {{cor}}</li>
7. </ul>
8. </div>
9. <script>
10. **new** Vue({
11. el:'#app' ,
12. data:{
13. cores:['vermelho','verde','amarelo','azul'],
14. pessoas:[
15. {nome:'Ana', idade: 26, peso: 59},
16. {nome:'Guilherme', idade: 22, peso: 89}
17. ]
18. }
19. })
20. </script>
21. <style>
22. ul{
23. list-style: none;
24. }
25. </style>

**Aula 53 – Usando o v-for com template**

1. A vantagem de utilizarmos template é que ao realizarmos o v-for, as tags não serão repetidas na página. Vamos criar um template com um v-for criando nele uma variável cor que terá um índice. Dentro da tag template vamos colocar um h1 e um parágrafo. Dentro do h1 nós vamos interpolar a variável cor e dentro do parágrafo iremos interpolar o índice.
2. <meta *charset*="UTF-8">
3. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
4. <div *id*="app">
5. <ul>
6. <li *v-for*="(cor, i) in cores">índice: {{i}} - {{cor}}</li>
7. </ul>
8. <hr>
9. <template *v-for*="(cor, i) in cores">
10. <h1>{{cor}}</h1>
11. <p>{{i}}</p>
12. </template>
13. </div>
14. <script>
15. **new** Vue({
16. el:'#app' ,
17. data:{
18. cores:['vermelho','verde','amarelo','azul'],
19. pessoas:[
20. {nome:'Ana', idade: 26, peso: 59},
21. {nome:'Guilherme', idade: 22, peso: 89}
22. ]
23. }
24. })
25. </script>
26. <style>
27. ul{
28. list-style: none;
29. }
30. </style>

**Aula 54 – Iterando Objetos**

1. Vamos utilizar o exemplo anterior e iremos realizar a iteração do objeto pessoas.
2. Vamos criar uma ul com um li passando para o li um v-for que irá percorrer o array de objetos pessoas.
3. Dentro do li iremos criar outro v-for que irá percorrer a variável pessoa criado na iteração anterior.
4. Neste caso podemos também recuperar a chave e o índice do objeto.
5. Se utilizarmos span ao invés de div teremos os itens renderizados em linha.
6. <meta *charset*="UTF-8">
7. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
8. <div *id*="app">
9. <ul>
10. <li *v-for*="(cor, i) in cores">índice: {{i}} - {{cor}}</li>
11. </ul>
12. <hr>
13. <ul>
14. <li *v-for*="pessoa in pessoas">
15. <span *v-for*="(valor, chave, indice) in pessoa">
16. {{chave}}: {{valor}} ({{indice+1}})
17. </span>
18. </li>
19. </ul>
20. </div>
21. <script>
22. **new** Vue({
23. el:'#app' ,
24. data:{
25. cores:['vermelho','verde','amarelo','azul'],
26. pessoas:[
27. {nome:'Ana', idade: 26, peso: 59},
28. {nome:'Guilherme', idade: 22, peso: 89},
29. {nome:'Maria', idade: 43, peso: 55},
30. {nome:'Pedro', idade: 28, peso: 35}
31. ]
32. }
33. })
34. </script>
35. <style>
36. ul{
37. list-style: none;
38. }
39. </style>

**Aula 55 – Iterando em uma lista de números**

1. Vamos utilizar o exemplo anterior e acrescentar ao final um span que terá um v-for onde criaremos uma variável n em 10 e iremos interpolar a variável n.
2. Será renderizado uma lista de números de 1 a 10.
3. <meta *charset*="UTF-8">
4. <script *src*="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
5. <div *id*="app">
6. <ul>
7. <li *v-for*="(cor, i) in cores">índice: {{i}} - {{cor}}</li>
8. </ul>
9. <hr>
10. <ul>
11. <li *v-for*="pessoa in pessoas">
12. <span *v-for*="(valor, chave, indice) in pessoa">
13. {{chave}}: {{valor}} ({{indice+1}})
14. </span>
15. </li>
16. </ul>
17. <hr>
18. <span>
19. <ul *v-for*="n in 10">
20. <li>{{n}}</li>
21. </ul>
22. </span>
23. </div>
24. <script>
25. **new** Vue({
26. el:'#app' ,
27. data:{
28. cores:['vermelho','verde','amarelo','azul'],
29. pessoas:[
30. {nome:'Ana', idade: 26, peso: 59},
31. {nome:'Guilherme', idade: 22, peso: 89},
32. {nome:'Maria', idade: 43, peso: 55},
33. {nome:'Pedro', idade: 28, peso: 35}
34. ]
35. }
36. })
37. </script>
38. <style>
39. ul{
40. list-style: none;
41. }
42. </style>

**Aula 56 – Identificando os elementos no v-for**

1. Vamos utilizar o exemplo anterior e incluir um botão com uma diretiva v-on:click passando para esse evento um push no array cores.
2. <hr>
3. <button *v-on:click*="cores.push('branca')">Adicionar</button>
4. Desta forma será renderizad a cor branca toda vez que o botão for apertado.
5. Por isso temos que utilizar o item key em uma lista de elementos para evitar repetições de elementos, pois o key será um número de identificação único do elemento.