**Comunicação entre componentes**

**Aula 119 – Comunicação entre componentes**

1. Baixe o conteúdo desta aula para podermos realizar as tarefas.
2. Instale as dependências digitando : npm i
3. Vemos que a aplicação possui três componentes Usuário, UsuárioEditar e UsuárioInfo, nós iremos realizar a comunicação entre estes componentes.
4. No componente Usuario vamos criar um botão abaixo do primeiro parágrafo chamando a função click que irá receber a função ‘Alterar Nome’.
5. Vamos dentro de export Default utilizar a propriedade data() criando dentro dela a variável nome inicialmento com o valor ‘Pedro’.
6. Vamos ainda adicionar a propriedade methods criando o método ‘alterarNome’ dentro do método iremos passar o valor ‘Ana’ para a variável ‘nome’.
7. <template>
8. <div *class*="container">
9. <h1>Componente Usuário</h1>
10. <p>Esse é um componente muito legal!</p>
11. <button @*click*='alterarNome'>Alterar Nonme</button>
12. <hr>
13. <div *class*="componentes">
14. <app-usuario-info />
15. <app-usuario-editar />
16. </div>
17. </div>
18. </template>
19. <script>
20. import AppUsuarioInfo from './UsuarioInfo'
21. import AppUsuarioEditar from './UsuarioEditar'
22. export default {
23. components: { AppUsuarioInfo, AppUsuarioEditar },
24. data() {
25. return {
26. nome: 'Pedro'
27. }
28. },
29. methods: {
30. alterarNome(){
31. *this*.nome = 'Ana'
32. }
33. },
34. }
35. </script>
36. <style *scoped*>
37. *.container* {
38. background-color: #27363b;
39. color: #fff;
40. padding: 10px;
41. }
42. *.container* hr {
43. margin: 20px 10px;
44. }
45. *.componentes* {
46. display: flex;
47. }
48. *.componentes* > \* {
49. margin: 10px;
50. }
51. </style>
52. No componente UsuarioInfo iremos adicionar outro parágrafo e interpolar a variável nome que foi declarada no componente Usuário.
53. Porém para passarmos valores para outros componentes temos que definir uma propriedade chamada props onde iremos indicar quais são as variáveis que serão recebidas pelo componente passadas para o template.
54. <template>
55. <div *class*="componente">
56. <h2>As Informações de Usuário</h2>
57. <p>Vários detalhes...</p>
58. <p>Nome do Usuário: <strong>{{nome}}</strong></p>
59. </div>
60. </template>
61. <script>
62. export default {
63. props:[
64. 'nome'
65. ]
66. }
67. </script>
68. <style *scoped*>
69. *.componente* {
70. flex: 1;
71. background-color: #ec485f;
72. color: #fff;
73. }
74. </style>
75. Agora no componente Usuario vamos adicionar a variável nome no carregamento do componente no template.
76. <div *class*="componentes">
77. <app-usuario-info *nome*='Pedro' />
78. <app-usuario-editar />
79. </div>
80. A comunicação entre componentes ‘pai’ e ‘filho’ se dá através de props que são criadas para passarmos valores entre componentes.
81. No componente filho nós dizemos quais são as props que esperamos receber e no componente pai nós passamos essa propriedade.
82. Ainda no componente Usuário nós iremos utilizar a diretiva v-bind:nome para passarmos o valor da ‘variavel nome’ para o template, pois da forma que estava a variável era estática.
83. <div *class*="componentes">
84. <app-usuario-info *v-bind*:*nome*='nome' />
85. <app-usuario-editar />
86. </div>
87. Desta forma conseguimos também alterar o nome através do botão no template.