Desenvolvimento Web III

Vue.JS

Condicional

Para controlar a exibição de elementos dentro do nosso componente, podemos usar as diretivas v-if / v-else

<p v-if=“mostrar">O atributo mostrar é verdadeiro</p>

<p v-else>O atributo mostrar não é verdadeiro</p>

Condicional

Podemos precisar de realizar mais de um teste para controlar a apresentação, assim como no PHP, podemos usar a combinação do else com o if.

<p v-if=“mostrar">O atributo mostrar é verdadeiro</p>

<p v-else-if=“anonimo">Navegando como anônimo</p>

<p v-else>O atributo mostrar não é verdadeiro</p>

Praticando

Faça uma página que tenha um campo para o usuário digitar o seu nome, um checkbox para entrar como anônimo e um botão para entrar, que após o click deve mudar o rótulo(escrito) para sair.

Antes de clicar no botão entrar devemos exibir a mensagem “Nenhum usuário logado”.

Após clicar, se o campo anônimo não foi marcado, mostrar o nome do usuário que foi digitado na caixa, caso contrário, mostrar “Usuário anônimo logado”.

Percorrendo uma lista

Para percorrer um vetor com diversos elementos podemos usar a diretiva v-for

```
<ul>
   <li v-for="cor in cores">
      <label>{{ cor }}</label>
   </li>
</ul>
```

Praticando

Crie uma lista de tarefas, onde o usuário poderá preencher um campo com a tarefa e clicar em um botão adicionar, logo essa tarefa irá aparecer na lista de tarefas.

Quando a página é aberta, ainda não existe nenhuma tarefa na lista, portanto devemos exibir a mensagem: “Nenhuma tarefa cadastrada”.

Parâmetros personalizados

Passando parâmetros personalizados no framework.

Como vimos por padrão ao disparar um evento ele envia o parâmetro event, mas pode ser necessário enviar outro parâmetro.

<button v-on:click="atualizar(5, $event)">Somar</button>

Modificadores de evento

- Para parar a propagação do evento podemos utilizar o modificador stop

```
<p v-on:click="atualizar">
        Clique aqui
        <span v-on:click.stop="">Aqui não</span>
</p>
```

Modificadores de evento

- Prevenindo o comportamento default do elemento. Por exemplo, podemos cancelar o reload que ocorre ao clicar em um botão submit.

```
<input type="submit" v-on:click.prevent="" value="Gravar">
```

Modificadores de evento

- Também podemos utilizar um modificador para os eventos do teclado.

```
<input type="text" v-on:keyup.enter="exibirAlerta">
```

- Os modificadores podem ser combinados para exigir a utilização de mais de uma tecla para disparar o evento.

```
<input type="text" v-on:keyup.enter.alt="exibirAlerta">
```

V-model

- O v-model garante o two-way data binding sem ter a necessidade de usar o v-bind e o v-on.

<input type="text" v-model="nome">

- Nesse caso se houver atualização no HTML, automaticamente o atributo é atualizado na Vue Instance e vice-versa.

Método computado

O método computado só será disparado caso aconteça alguma atualização em um dos seus valores.

```
new Vue({
      el: "#app",
      data: {  contador : 0  },
      computed: {
         resultado() {
            return this.contador >= 5 ? "Maior ou igual a 5" :   "Menor do que 5"
         }
      }
  })
```

Monitorando mudanças

O watch monitora uma variável, ou seja, assim que ela sofre alguma alteração o método é chamado.

A diferença entre o método computado é que o watch não necessita de retorno.

O método tem que ter o mesmo nome do atributo.

```
...
    watch: {
        contador(novo, antigo) {
            this.contador = novo*2;    }
    }
...
```

Sintaxe reduzida

Podemos reduzir a escrita de diretivas do Vue.

V-bind:value = :value

V-on:click = @click

Praticando

Crie uma lista de compras, semelhante ao exercício anterior. Nesse caso, o usuário deve informar o nome do produto e a quantidade, se o usuário pressionar alt+enter o item deve ser cadastrado em um array de objetos.

Quando a página é aberta, ainda não existe nenhum item na lista, portanto devemos exibir a mensagem: “Nenhuma item para ser comprado”.