06

Lapidando nosso componente

Transcrição

Vamos fazer com que nosso componente que representa nosso botão receba como parâmetro `confirmacao`. Se receber `true`, a confirmação será efetuada, se receber false, nenhum confirmação será feita:

```
<!-- alurapic/src/components/shared/botao/Botao.vue -->

<template>
    <button class="botao botao-perigo" :type="tipo"
@click="disparaAcao()">{{rotulo}}</button>
</template>
<script>
export default {

   props: ['tipo', 'rotulo', 'confirmacao'],

   methods: {

       disparaAcao() {

           if(this.confirmacao) {

               if(confirm('Confirma operacao?')) {
                    this.$emit('botaoAtivado');
               }
               return;
           }
```

```
            this.$emit('botaoAtivado');
        }
    }
}
</script>

<style scoped>
    /* código omitido */

</style>
```

COPIAR CÓDIGO

Agora, em `Home`, podemos adicionar `confirmacao="false"` em noss componente para que nenhuma confirmação seja realizada:

```
<!-- alurapic/src/components/home/Home.vue -->

<template>
    <div>
        <h1 class="titulo">Alurapic</h1>
        <input type="search" class="filtro" @input="filtro =
$event.target.value" placeholder="filtre pelo título da foto">
        <ul class="lista-fotos">
          <li class="lista-fotos-item" v-for="foto in
fotosComFiltro">
              <meu-painel :titulo="foto.titulo">
                <imagem-responsiva :url="foto.url"
:titulo="foto.titulo"/>
                <meu-botao
                  rotulo="remover"
                  tipo="button"
                  confirmacao="false"
                  @botaoAtivado="remove(foto)"/>
              </meu-painel>
```

```
            </li>
          </ul>
      </div>
</template>

<script>

import Painel from '../shared/painel/Painel.vue';
import ImagemResponsiva from '../shared/imagem-
responsiva/ImagemResponsiva.vue'
import Botao from '../shared/botao/Botao.vue';

export default {

  // código omitido
}
</script>
<style>

  /* código omitido */
</style>
```

COPIAR CÓDIGO

Não funciona! Veja que mesmo com `confirmacao="false"` a confirmação é exibida. Qual a razão disso? O problema é que não fizemos um data binding entre com a propriedade `confirmacao` , para tal, precisamos adicionar `v-bind:` ou simplesmente `:` antes do nome da propriedade. Sem o binding, o valor é passado uma única vez para dentro do componente como texto e não como referência. Em JavaScript qualquer texto é considerado `true` e por isso o alerta é sempre exibido.

Podemos verificar o tipo dentro do componente `Botao` da seguinte maneira:

```
<!-- alurapic/src/components/shared/botao/Botao.vue -->

<template>
    <button class="botao botao-perigo" :type="tipo"
@click="disparaAcao()">{{rotulo}}</button>
</template>
<script>
export default {

   props: ['tipo', 'rotulo', 'confirmacao'],
   methods: {

       disparaAcao() {

           / exibindo o tipo da propriedade. Sem o : será
string, com : será boolean
           console.log(typeof(this.confirmacao));

           if(this.confirmacao) {
               if(confirm('Confirma operacao?')) {
                    this.$emit('botaoAtivado');
                }
                return;
           }
           this.$emit('botaoAtivado');
       }
   }
}
</script>

<style scoped>
  /* código omitido */
</style>
```

COPIAR CÓDIGO

Agora, quando usamos `:` o valor passado deixa de ser uma string e passa a ser a expressão, no caso `false`, um booleano!

```
<!-- alurapic/src/components/home/Home.vue -->

<template>
    <div>
        <h1 class="titulo">Alurapic</h1>

        <input type="search" class="filtro" @input="filtro =
$event.target.value" placeholder="filtre pelo título da foto">

        <ul class="lista-fotos">

          <li class="lista-fotos-item" v-for="foto in
fotosComFiltro">

              <meu-painel :titulo="foto.titulo">

                <imagem-responsiva :url="foto.url"
:titulo="foto.titulo"/>

                <meu-botao
                  rotulo="remover"
                  tipo="button"
                  :confirmacao="false"
                  @botaoAtivado="remove(foto)"/>

              </meu-painel>

          </li>

        </ul>
    </div>
</template>
```

```
// código posterior omitido
```

COPIAR CÓDIGO

Faça um teste e veja que agora a confirmação não é exibida. Só não esqueça de colocar `confirmacao="true"` novamente, pois nesse cenário faz sentido pedirmos a confirmação do usuário.

Agora, precisamos lidar com o estilo do botão. Vamos adicionar a propriedade `estilo`. Se o seu valor for `padrão`, usaremos a classe `botao botao-padrao`, no entanto, se for `perigo`, usaremos a classe `botao botao-perigo`. Primeiro, vamos adicionar a propriedade em `Botao` para que ele aceite recebê-la:

```
<!-- alurapic/src/components/shared/botao/Botao.vue -->

<template>
    <button :class="estiloDoBotao" :type="tipo"
@click="disparaAcao()">{{rotulo}}</button>
</template>
<script>
export default {

   props: {
            tipo: {
                required: true,
                type: String
            },

            rotulo: {
                required: true,
                type: String
            },
```

```
            confirmacao: {
                required: false,
                default: false,
                type: Boolean
            },

            estilo: {
                required: false,
                default: 'padrao',
                type: String
            }
        },

  // código posterior omitido
```

COPIAR CÓDIGO

Agora, em `Home`, vamos passar a propridade `estilo="padrao"` para que nosso botão seja apresentado com o estilo padrão, ou seja, numa cor azul clara:

```
<!-- alurapic/src/components/home/Home.vue -->

<template>
    <div>
        <h1 class="titulo">Alurapic</h1>
        <input type="search" class="filtro" @input="filtro =
$event.target.value" placeholder="filtre pelo título da foto">
        <ul class="lista-fotos">
          <li class="lista-fotos-item" v-for="foto in
fotosComFiltro">
            <meu-painel :titulo="foto.titulo">
              <imagem-responsiva :url="foto.url"
:titulo="foto.titulo"/>
              <meu-botao
                rotulo="remover"
```

```
                tipo="button"
                :confirmacao="true"
                @botaoAtivado="remove(foto)"
                estilo="padrao"/>
            </meu-painel>
        </li>
    </ul>
</div>
</template>
```

COPIAR CÓDIGO

Agora, em `Botao`, a classe da tag `button` deve receber um ou outro estilo de acordo com o parâmetro passado para `estilo`. Vamos usar uma computed property para isso. Nosso `Botao` final fica assim:

```
<!-- alurapic/src/components/shared/botao/Bota.vue -->

<template>
    <button :class="estiloDoBotao" :type="tipo"
@click="disparaAcao()">{{rotulo}}</button>
</template>
<script>
export default {

   props: {
            tipo: {
                required: true,
                type: String
            },

            rotulo: {
                required: true,
                type: String
            },
```

```
            confirmacao: {
                required: false,
                default: false,
                type: Boolean
            },

            estilo: {
                required: false,
                default: 'padrao',
                type: String
            }
        },
    methods: {

        disparaAcao() {
            console.log(typeof(this.confirmacao));
            if(this.confirmacao) {
                if(confirm('Confirma operacao?')) {
                    this.$emit('botaoAtivado');
                }
                return;
            }
            this.$emit('botaoAtivado');
        }
    },

    computed: {

        estiloDoBotao() {

            // se o valor é padrão ou não passou nada para
estilo
            if(this.estilo == 'padrao') return 'botao botao-
padrao';
```

```
            if(this.estilo == 'perigo') return 'botao botao-
perigo';
        }
    }
}
</script>

<style scoped>
    .botao {
        display: inline-block;
        padding: 10px;
        border-radius: 3px;
        margin: 10px;
        font-size: 1.2em;
    }

    .botao-perigo {
        background: firebrick;
        color: white;
    }

    .botao-padrao {
        background: darkcyan;
        color: white;
    }

</style>
```

COPIAR CÓDIGO

Recarregando a página, vamos que nosso botão esta na cor azul. Sabemos que isso foi apenas um teste, pois seu estilo deve ser `perigo` . Alterando:

```
<!-- alurapic/src/components/home/Home.vue -->
```

```
<!-- código anterior omitido -->

<meu-botao
  rotulo="remover"
  tipo="button"
  :confirmacao="true"
  @botaoAtivado="remove(foto)"
  estilo="perigo"/>


<!-- código posterior omitido -->
```

COPIAR CÓDIGO

Excelente, temos mais um componente reutilizável que nos será útil ao longo do treinamento.