

Eventos customizados

Transcrição

Antes de continuarmos precisamos entender o seguinte. O evento `clique` do nosso botão chama um método no componente pai `Home`. Agora, precisamos que o código de confirmação seja executado pelo nosso botão, mas a lógica de execução deve ser feita no elemento pai. Em suma, nosso elemento filho precisa de alguma maneira chamar um método do seu elemento pai.

Para isso, precisamos trabalhar com eventos customizados. Por exemplo, não implementamos ainda esse evento, mas quando clicamos no botão e o evento `botaoAtivado` for disparado, o método `remove` recebendo a foto deve ser chamado. Não preciso dizer que `botaoAtivado` será o evento que criaremos. Como todo evento, podemos associá-lo a um elemento usando o atalho `@`:

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

<!-- código anterior omitido -->

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

<!-- código anterior omitido -->
```

A ideia é o seguinte. Quando nosso componente `Botao` for clicado, ele disparará um o evento customizado `botaoAtivado`, mas apenas se o usuário confirmar. Quando esse evento for disparado, o método associado ao nome do nosso evento será executado.

Agora, precisamos alterar `Botao` para que dispare nosso evento toda vez que for clicado. Para isso, no seu template, vamos adicionar o evento `click` que chamará um método do componente. É nele que realizamos a lógica de confirmação disparando o evento ou não:

```
<!-- 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'],
  methods: {

    disparaAcao() {

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

É através de `this.$emit` que disparamos um evento customizado passando como nome do parâmetro o evento.

Com o CLI rodando, nosso projeto com certeza foi recarregado pelo seu navegador, inclusive já podemos realizar nosso teste. Quando clicamos em `remove`, o diálogo de confirmação é exibido. Se não confirmamos, nada é exibido na tela, quando confirmamos, o alerta com o título da foto é exibido.

Apesar de funcionar nosso botão ainda esta incompleto. Precisamos fazer com que o componente possa confirmar ou não uma ação, inclusive aplicar a classe correta de acordo com seu estilo. é isso que veremos no próximo vídeo.