02

Validando no próprio template

Transcrição

A primeira coisa que precisamos ter em mente é que toda a validação do VeeValidade é feita no próprio template. Sendo assim, o primeiro passo é adicionar a diretiva `v-validate` no elemento de entrada que desejamos validar. Mas isso não é suficiente.

Precisamos indicar que tipo de validação queremos. Isso é feito através da diretiva `data-vv-rules` . No caso, usaremos o valor `required` para esta propriedade indicando que o campo é obrigatório. Mas se você acha que vai funcionar, ainda não é o caso. Somos obrigados a adicionar a propriedade `name` no input, no caso, usaremos `titulo` .

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código anterior omitido -->
      <div class="controle">
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required" >
      </div>
<!-- código posterior omitido -->
```

Apesar no nosso campo já estar sendo validado, não temos uma indicação visual de erros de validação. Para isso, vamos adicionar uma tag `span` imediatamente abaixo do tag `input` do título. Ela só será exibida caso haja algum erro de validação com o campo:

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código anterior omitido -->
      <div class="controle">
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required" >
         <span v-show="errors.has('titulo')">ERRO</span>
      </div>
<!-- código posterior omitido -->
```

Para que a nosso `span` seja exibido apenas quando houver um erro, usamos a diretiva `v-show` . Contudo, veja que o valor da diretiva é um valor especial, no casso `erros.has('titulo')` . O object `erros` é criado automaticamente pelo VeeValidate e através do método `has` podemos consultar se um input falhou na validação. É por isso que o método recebe o `name` do input, para poder identificá-lo.

Quando deixamos o campo em branco a mensagem "ERRO" é exibida. Queremos uma mensagem mais profissional, por isso podemos solicitar ao próprio módulo de validação que exiba sua mensagem de erro padrão para `required` :

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código anterior omitido -->
      <div class="controle">
```

```
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required" >
         <span v-show="errors.has('titulo')">{{ errors.first('titulo') }}</span>
      </div>
<!-- código posterior omitido -->
```

Veja que usamos uma interpolação com `{{ erros.first('titulo')}}` . Isso fará com que uma mensagem padrão em inglês seja exibida para o usuário. Gostaria que fosse me português? Não se preocupe, veremos isso ainda nesse capítulo, mas para não perdermos o foco do mecanismo de validação continuaremos com as mensagens em inglês.

Para ficar ainda melhor, vamos adicionar a classe `erro` na tag `span` e declarar a classe `erro` em nosso componente para que a mensagem fique na cor vermelha:

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código anterior omitido -->
      <div class="controle">
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required" >
         <span class="erro"  v-show="errors.has('titulo')">{{ errors.first('titulo') }}</span>
      </div>
<!-- código posterior omitido -->
<style>
/* código anterior omitido */

  .erro {
    color: red;
  }

</style>
```

Mas se quisermos outros tipos de validação? No próprio site do VeeValidate há uma série de validadores já prontos que você pode consultar em `http://vee-validate.logaretm.com/#available-rules` .

Por exemplo, queremos que o título tenha no mínimo três caracteres e no máximo 30. Para isso vamos adicionar os validadores `min` e `max` , cada um separado por um *pipe* :

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código anterior omitido -->
      <div class="controle">
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required|min:3|max:30" >
         <span class="erro" v-show="errors.has('titulo')">{{ errors.first('titulo') }}</span>
      </div>
<!-- código posterior omitido -->
```

Veja que para cada validator definimos a quantidade de caracteres logo após os dois pontos. Apenas isso é suficiente. Podemos testar o resultado.

No entanto, mesmo com a mensagem em inglês, um olhar atento percebe que o a palavra título deve ter acento, mas na mensagem de validação ela aparece sem. Isso é assim, porque é o `name` do input que é considerado internamente pelo VeeValidate na mensagem que ele gera automaticamente. Isso não precisa ser assim.

Podemos usar a diretiva `data-vv-as` para que o nosso validador considere este nome ao invés do `name` na mensagem exibida para o usuário:

```
<!-- código anterior omitido -->
      <div class="controle">
        <label for="titulo">TÍTULO</label>
        <input name="titulo" v-model="foto.titulo" id="titulo" autocomplete="off"
        v-validate data-vv-rules="required|min:3|max:30" data-vv-as="título">
         <span class="erro" v-show="errors.has('titulo')">{{ errors.first('titulo') }}</span>
      </div>
<!-- código posterior omitido -->
```

Como o valor de `data-vv-as` é título com acento, a mensagem será exibida levando em consideração o valor de `data-vv-as` .

Por fim, podemos definir que o campo `url` é obrigatório:

```
<!-- alurapic/src/components/cadastro/Cadastro.vue -->

<!-- código posterior omitido -->
      <div class="controle">
        <label for="url">URL</label>
        <input name="url" v-model="foto.url" id="url" autocomplete="off"
        v-validate data-vv-rules="required">
        <span class="erro" v-show="errors.has('url')">{{ errors.first('url') }}</span>
        <imagem-responsiva v-show="foto.url" :url="foto.url" :titulo="foto.titulo"/>
      </div>
<!-- código posterior omitido -->
```

Apesar de estarmos recebendo as mensagem de erros, ainda somos capazes de gravar os dados inválidos. Precisamos testar em nosso método `grava` se há algum campo inválido. Alterando:

```
// alurapic/src/components/cadastro/Cadastro.vue

// código anterior omitido
  methods: {

    grava() {

        this.$validator
          .validateAll()
          .then(success => {
            if(success) {

              this.service
                .cadastra(this.foto)
                .then(() => {
                  if(this.id) this.$router.push({ name: 'home'});
```

```
                    this.foto = new Foto()
                },
                err => console.log(err));
            }
        });
    }
  },
// código posterior omitido
```

Quando usamos `VueValidate` temos acesso ao objeto `$validator` em nosso componente. Podemos invocar o método `validadeAll` que faz um procedimento como se o usuário tivesse entrado em cada campo disparando a validação. Seu retorno é uma promise que devolve um boolean. Se for verdadeiro, é porque todos os campos passaram na validação.

Não seria muito melhor que nossas mensagens fossem exibidas em português? Com certeza! É exatamente isso que veremos no próximo vídeo.