Formulários e validação em *template* no Angular (*Template Forms*)
Bruno E. G. Gomes – IFRN – Parnamirim

A seguir detalha-se os passos necessários para se validar um controle em um formulário. Entenda por controle qualquer campo do formulário, como *input*, select ou *button*.

1. Na *tag* **`<form>`** criar uma variável local e atribuir a **`ngForm`**

   No exemplo a seguir a variável local se chama "fm":

```
<form #fm="ngForm">
</form>
```

2. Em cada controle (*input*, *select*, etc.) do *form* sujeito a validação criar uma variável local com a notação **`#nome_da_variável`** e associá-la a **`ngModel`**. Por exemplo, **`#n="ngModel"`**. Deve ser feito também a associação a algum atributo do template com **`[(ngModel)]`** ou **`[ngModel]`**. A variável local é necessária para se referir ao controle nas *tags html* caso se deseje obter o seu estado. Os estados possíveis são exibidos no Quadro 1 e podem ser utilizados, por exemplo, para imprimir mensagens de erro/advertência ou modificar o CSS. O controle também deve ter obrigatoriamente a propriedade **`name`**.

   Suponha que há um controle de **`input`** para receber um valor do usuário.
   Para que esse controle possar estar sujeito à validação e para que se possa obter os estados dessa validação, deve-se criar uma variável local ao controle e fazer a associação a **`ngModel`**. No exemplo a seguir, para um campo em que o usuário deve digitar um *email*, essa variável se chama "*#em*" (**`#em="ngModel"`**). Observe também o atributo *name* da *tag input* (**`name="email"`**) e a ligação a um atributo do componente chamado *_email* (**`[ngModel]="_email"`**).

```
<form #fm="ngForm">
  <label for="email"></label>
  <input id="email" name="email" type="email"
         #em="ngModel" [ngModel]="_email">
</form>
```

3. Inserir os elementos de validação para o controle. Os elementos que podem ser inseridos como atributos (diretivas no *Angular*) de um controle no *template* HTML são:
   - **`required`**: define que o controle é obrigatório. Ou seja, o usuário deve obrigatoriamente fornecer um valor.
   - **`email`**: verifica se o texto fornecido está no padrão de um *email* válido.
   - **`pattern`**`="expressão regular"`. A expressão regular passada é utilizada para validar o controle. Só serão válidas as entradas do usuário que satisfizerem ao padrão de expressão definido.
   - **`minlenght=`**`"tamanho"`: tamanho mínimo de um campo. Por exemplo, número de caracteres em um *input* de texto.
   - **`maxlenght=`**`"tamanho"`: tamanho máximo de um campo. Por exemplo, máximo de caracteres em um *input* de texto.

```
<form #fm="ngForm">
  <label for="email"></label>
  <input id="email" name="email" type="email"
         #em="ngModel" [ngModel]="_email" email required>
</form>
```

Estados possíveis dos controles em um *form*

Os estados a seguir podem ser utilizados para exibir mensagens para o usuário sobre a validade ou não de um controle ou para se alterar o CSS do *template* a partir da classe de estado em que o controle se encontra.

Estado	Valor se verdadeiro	Valor se falso	Classe CSS associada
O estado do controle é válido (**`valid`**) ou não (**`invalid`**) de acordo com as regras de validação estabelecidas.	**`valid`**	**`invalid`**	**`ng-valid / ng-invalid`**
O valor do controle foi modificado. Nesse caso, o usuário modificou o controle (**`dirty`**) ou ainda não modificou nada (**`pristine`**).	**`dirty`**	**`pristine`**	**`ng-dirty / ng-pristine`**
O controle foi visitado. Ou seja, recebeu foco (**`touched`**) ou não recebeu foco (**`untouched`**).	**`touched`**	**`untouched`**	**`ng-touched / ng-untouched`**

Quadro 1: Estados possíveis de um controle em um form sujeito à validação. Fonte: autoria própria.

4. Fazer com que o botão de submissão do form seja habilitado apenas se o estado de todos os seus controles estiver válido.

```
<form #fm="ngForm">
  <label for="email"></label>
  <input id="email" name="email" type="email"
         #em="ngModel" [ngModel]="_email" email required>
  <button type="submit" [disabled]="fm.form.invalid">Enviar
  </button>
</form>
```

5. Exibir alguma mensagem de erro ou advertência ao usuário. No caso abaixo, caso o e-mail esteja inválido (**`em.invalid`**) e o usuário já tenha digitado algo no campo e-mail ou tenha colocado foco no controle a mensagem é exibida (**`em.dirty || em.touched`**) uma classe CSS chamada de erro será aplicada.

```
<form #fm="ngForm">
  <label for="email"></label>
  <input id="email" name="email" type="email"
         #em="ngModel" [ngModel]="_email" email required>
  <button type="submit" [disabled]="fm.form.invalid">
     Enviar
  </button>
</form>
<!--Continua na próxima página -->
```

```
<div class="erro"
     *ngIf="em.invalid && (em.dirty || em.touched)">
     <p>Digite um email válido</p>
</div>
```