F 11

Adequando o template gerado e mais data binding

Transcrição

Vamos adequar a nossa aplicação para que ela fique mais próxima de um sistema que gerencia fotos. Antes disto, vamos recapitular como subimos nosso projeto:

- 1. Abrimos o Prompt de Comando (ou Terminal, no Linux);
- acessamos nossa área de trabalho (cd Desktop) e depois a pasta "alurapic"
 (cd alurapic);
- 3. iniciamos a aplicação com ng serve --open, que a abrirá no browser automaticamente.

Deste modo, a aplicação passa a rodar em um servidor local disponibilizado pelo Angular CLI. Então, poderemos abrir o Visual Studio Code, que estará com a pasta do projeto aberta. A ideia é que o primeiro componente a ser carregado pela nossa aplicação seja o <app-root>, que equivale ao app.component.ts, e sua apresentação está em templateUrl, que é ./app.component.html, pasta em que se encontra o arquivo .ts.

Vamos abrir app.component.html, deletar todo seu conteúdo, e salvar. Ao voltarmos ao navegador, teremos uma página em branco, já que não há nada no template do componente. Adicionaremos uma tag cujo source pode ser uma imagem qualquer:

<img src="https://upload.wikimedia.org/wikipedia/commons/thumb,</pre>

COPIAR CÓDIGO

Com isso, a página do navegador passa a exibir uma imagem de leão, como gostaríamos. Por ora, não há nada de sofisticado no Angular, e este curso ficará, claro, cada vez mais complexo conforme avançamos.

Se queremos fazer um *Data binding*, e que o componente no qual este template está associado forneça os dados para o template, criaremos em app.component.ts uma propriedade com descrição Leão, e outra, url, cujo valor será o link da imagem que acabamos de incluir:

```
export class AppComponent {
    title = 'alurapic';
    description = 'Leão';
    url = 'https://upload.wikimedia.org/wikipedia/commons/thuml
}
COPIAR CÓDIGO
```

Feito isso, precisaremos remover os valores de app.component.html para de fato realizarmos o *Data binding*:

Ao voltarmos para o navegador, nada estará sendo exibido, justamente porque ainda não passamos o src nem o alt. A primeira reação é querermos usar uma *Angular Expression*, porém, não faremos isso. Este comportamento é utilizado quando temos o conteúdo de uma tag e queremos exibir o valor de uma propriedade do componente dentro da tag de conteúdo.

Sendo assim, salvaremos o código abaixo e voltaremos ao navegador, em que teremos apenas o texto "alurapic", o title da página:

```
<h1>{{ title }}</h1></mg src="" alt="">
```

COPIAR CÓDIGO

Mas quando queremos realizar o *Data binding* de um atributo HTML, isto ocorrerá de outra maneira — teremos que envolvê-lo em colchetes. E então, setaremos seus valores como o nome do componente correspondente:

```
<h1>{{ title }}</h1>
<img [src]="url" [alt]="title">
```

COPIAR CÓDIGO

Assim, ao salvarmos e voltarmos ao navegador, teremos o título e a imagem sendo exibidos na página. O que acabamos de fazer é denominado *One way data binding* ("data binding unidirecional"), isto porque os dados saem do componente, da fonte de dados, e vão para o template, no entanto não fazem o caminho contrário, e é importante frisarmos isto.

Reparem que, ao usarmos os colchetes, url e title se tornam expressões que o Angular sabe que precisa buscar no componente ao qual o template está associado. Se deixarmos sem os colchetes, uma vez que não fazemos *Data binding*, isto é, não avaliamos uma expressão, o valor de src será a *string* url , enquanto o de alt será o texto title , e não o valor title .

Assim, quando formos realizar *Data binding* com atributos HTML do nosso component, usaremos os colchetes. E quando formos exibir um valor em uma tag, ou isoladamente, usaremos uma *Angular Expression*. Agora, somos capazes de evidenciar algumas convenções do Angular, assunto que veremos mais adiante.