05

Inbound properties: passando dados para o componente

Transcrição

Vamos voltar ao problema de termos duas imagens (componentes `PhotoCompoment`), sendo que queremos que cada uma tenha informações independentes e específicas. Em `app.component.html`, então, queremos passar título e descrição diferentes para cada um dos `<ap-photo>`. Atualmente, este dado está em `photo.component.ts`, e queremos algo assim:

```
<ap-photo url="https://upload.wikimedia.org/wikipedia/commons/
<ap-photo url="https://upload.wikimedia.org/wikipedia/commons/
```

COPIAR CÓDIGO

O segundo componente está com uma imagem qualquer, e o primeiro está com a mesma imagem de antes — assim, podemos deletar tanto o valor de `description` quanto de `url` em `photo.component.ts`, sem esquecermos de salvar os arquivos. Pela lógica, `PhotoComponent` desconhece os valores de `description` e `url`, então, se acessarmos `app.component.html`, indicaremos quais são estes valores.

Entretanto, se voltarmos ao navegador, nada é exibido. Isso ocorre porque em nenhum momento incluímos estes valores nas propriedades de `photo.component.ts` ao colocarmos `url` e `description` de componentes distintos. Assim, quando o `PhotoComponent` for realizar o *Data binding*, ele lerá valores de `url` e `description` que estão em branco.

O que acontece é que precisamos **explicitar** que tais propriedades podem receber um parâmetro externo, passando um valor por meio da forma declarativa do componente. Caso isto não ocorra, estes valores nunca cairão em suas respectivas propriedades. Para isso, incluiremos `@Input()`, que faz parte do `angular/core`, e o programa apontará um erro porque precisamos importá-lo. Basta clicar nele e no ícone de lupa, e então em "Add 'Input' to existing import declaration from "@angular/core"". Teremos:

```
import { Component, Input } from "@angular/core";

@Component({
    selector: 'ap-photo',
    templateUrl: 'photo.component.html'
})
export class PhotoComponent {

    @Input() description='';
    @Input() url='';
}
```

COPIAR CÓDIGO

Assim, indicamos que `description` e `url` são ***Inbound properties***, ou seja, aceitam receber um valor por meio de sua forma declarativa. Ao salvarmos e retornarmos ao navegador, teremos a exibição das duas imagens desejadas. Antes de continuarmos, uma dúvida: em `app.component.html`, por que não colocamos `url` e `description` entre colchetes? Já não estamos acostumados a fazer *Data binding*?

Se fizéssemos assim, os links seriam buscados como se fossem propriedades, uma vez que, ao realizarmos *Data binding*, avalia-se a expressão como propriedade do componente, e não é isso que queremos. Queremos passar a *string* diretamente. Porém, se estes valores estão caindo em `photo.component.ts`, precisamos usar a expressão no momento de suas leituras no template, como em `photo.component.html`.

Se removermos o *Data binding* deste arquivo, o valor de `src` será a *string* `url`. Portanto, é importante identificarmos e entendermos bem o momento de passar e interpretar os parâmetros para os componentes. Isto é, quando usamos um componente A em um componente B, e queremos passar valores para as suas propriedades, apenas aquelas decoradas com `@Input()` se tornarão *inbound properties*, e poderão recebê-los.

Com isso, finalizamos nosso primeiro componente, o `PhotoComponent`, mas há muito mais para ser visto em termos de organização do projeto, o que faremos em breve.