Data Binding
com o Framework Angular

UNISANTA

Índice

Estrutura de um app Angular

Antes de partirmos para falar sobre Data Binding, é importante entender um pouco da estrutura definida pelo Framework Angular.

Um App Angular é formado por blocos de construção chamados de "Componentes". Um componente é constituído principalmente de:

- Lógica (arquivo com extensão **".ts"** também chamado de class)
- View (arquivo com extensão **".html"** também chamado de template)
- Estilos (arquivo com extensão **".css"** também chamado de style)

Ao criar uma nova aplicação Angular, automaticamente é criado um componente principal chamado de "app". É comum chamarmos esse componente de AppComponent.

Para cada componente existirá uma pasta com seu próprio nome. Dessa forma é possível encontrarmos os seguintes arquivos dentro da pasta "app":

app.component.ts

Contém a lógica do AppComponent. Considerando que ele é o componente principal do app, então este código define a lógica da página principal do App (class).

app.component.html

Conteúdo HTML do AppComponent que será inicialmente aberto no browser (template).

app.component.css

Contém o estilo do AppComponent (style)

Considerando que o componente "app" é o componente principal da aplicação, vamos encontrar também um outro aquivo chamado de "**app.module.ts**" que especifica todos os componentes usados pelo App.

Sempre que você cria um novo componente com o Angular CLI, automaticamente serão criados os arquivos conforme essa estrutura.

Exemplo:

Considere que você deseja criar um componente chamado de "**produto**". Para isso você poderá utilizar o seguinte comando do Angular CLI:

```
ng generate component produto
```

Nesse caso é criada uma pasta chamada "**produto**" dentro da pasta "**app"**, com os seguintes arquivos:

- produto.component.ts
- produto.component.html
- produto.component.css

Um arquivo adicional com o nome "produto.component.spec.ts" também é criado para o propósito de realização de testes.

Componentes Pai e Filho

Sempre que um componente utiliza outros componentes, é estabelecida uma relação de Pai para Filho entre eles, ou seja, o componente que **utiliza** outro componente é considerado o componente **Pai**, enquanto o componente utilizado é considerado o componente **Filho.**

No exemplo que estamos utilizando em nossas aulas, o componente do tipo "**app**" faz uso de componentes do tipo "**item**". Esse relacionamento é evidenciado no momento em que o componente "**app**" apresenta em sua view, uma referência a componentes do tipo "**item**".

Abaixo podemos ver o arquivo "**app.component.html**", destacando em verde a utilização do componente "**item**":

```
<div class="main">
    <h1>Mozilla Dev - My To Do List</h1>
    <input class="lg-text-input" #campoNovoItem />
    <button class="btn-primary" (click)="CREATE_tarefa(campoNovoItem.value)">
        Adicionar Tarefa</button>
    <ul>
        <li *ngFor="let tarefaDoLoop of arrayDeTarefas">
            <app-item></app-item>
        </li>
    </ul>
</div>
```

Neste caso, podemos dizer que o componente "**app**" é pai do componente "**item**".

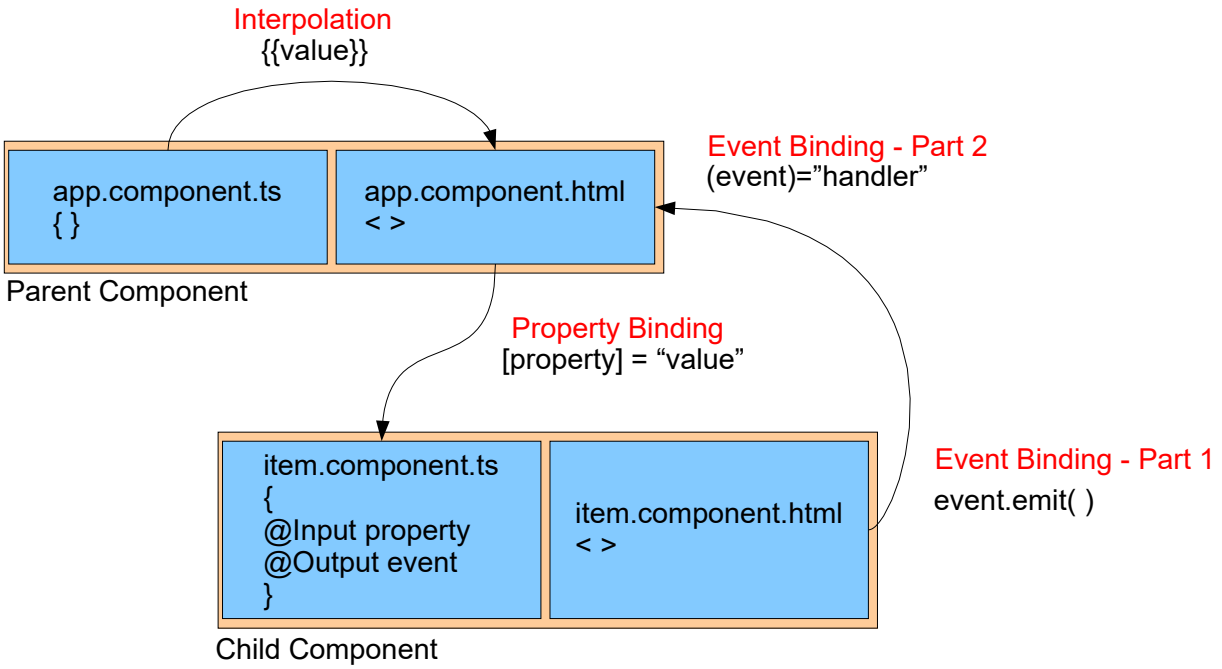
Esse relacionamento é muito importante para entender como funciona o Data Binding.

Data Binding

O termo "Data Binding" em Angular representa os mecanismos utilizados para transferir informações entre diferentes partes do aplicativo. Os três principais métodos de Data Binding do Angular são:

- Interpolation
- Property Binding
- Event Binding

O diagrama abaixo resume uma boa explicação sobre o propósito de cada método:



Interpolation

Este método permite transferir informações da classe para a view de um componente.

São utilizadas chaves duplas {{}} para referenciar a informação.

Esse tipo de construção aparece sempre declarada dentro do arquivo "**.html**" do componente.

Property Binding

Este método permite transferir informações de um componente Pai para a uma propriedade de um componente Filho.

São utilizados colchetes [] para referenciar a propriedade do componente Filho. Para fazer uso desse recurso, é importante lembrar que a propriedade do componente Filho precisa ser declarada utilizando o Decorator @Input.

Esse tipo de construção aparece sempre declarada dentro do arquivo "**.html**" do componente Pai.

Event Binding

Este método permite transferir informações de um componente Filho para um componente Pai.

A transferência de informações é feita através da emissão de um evento do componente Filho para o componente Pai.

Para que um filho possa emitir um evento, ele precisa declarar uma propriedade utilizando o Decorator @Output.

A emissão do evento pelo Filho ocorre quando o método "emit()" é executado a partir de um objeto do tipo EventEmitter.

Já dentro da view do componente Pai, são utilizados parêntesis para capturar o evento e executar um código que funcionará como manipulador do mesmo.

Histórico de revisões

Revisão: 00
Data: 19/07/2022
Descrição das alterações:
Documento original