Innominatum CONSULTORIA

Programação em JavaScript

ServiceNow - Deloitte

Rodrigo Costa

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL

Innominatum CONSULTORIA

Fundamentos Angular

Módulo 5

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL

Innominatum CONSULTORIA

Templates, Data Binding, Diretivas e Pipes

Sessão 26

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL

Objetivo geral

Capacitar os formandos na construção de templates dinâmicos em Angular, compreendendo os mecanismos de ligação entre o modelo e a interface através de data binding, diretivas estruturais e pipes. A sessão estabelece a base para a criação de interfaces reativas, permitindo controlar a renderização condicional, responder a eventos do utilizador e transformar dados diretamente no template.

Objetivos específicos

Ao final da sessão, os formandos deverão ser capazes de:

- Aplicar diferentes tipos de Data Binding, incluindo interpolation, property binding, event binding e two-way data binding;
- Controlar a renderização condicional e iterativa, utilizando diretivas estruturais como @if, @for, @switch e variantes;
- Responder a eventos do utilizador, conectando ações da interface à lógica do componente;
- Utilizar Pipes para transformar dados no template, incluindo pipes nativos e customizados;
- Construir templates dinâmicos e reativos, conectando dados e interface de forma declarativa.

Fundamentos do Data Binding

O que é Data Binding?

É o mecanismo que liga o TS (lógica) ao HTML (interface) ele permite:

- Mostrar dados no template;
- Atualizar a interface automaticamente;
- Responder a eventos do utilizador.

Tipos de Data Binding

Angular possui 4 tipos principais:

1. Interpolation: Exibição dos dados;
2. Property Binding: Envio de dados para propriedades do HTML;
3. Event Binding: Capturar eventos do utilizador;
4. Two-way Data Binding: Comunicação em duas direções.

Interpolation {{}}

Com a interpolação conseguimos enviar os dados do nosso TS, diretamente no HTML, sem a dificuldade de manipularmos as queries, como fazíamos no JS.

```ts
titulo: string = 'Título';
```

TS

```html
<section>
    <h1>{{ titulo }}</h1>
</section>
```

Chamamos a variável interpolada pelas {{}} no HTML

Título

Texto sendo exibido no navegador

Property Binding []

Permite enviar dados do TS para propriedades de um elemento HTML. A propriedade é algo que configura o comportamento ou aparência de um elemento HTML, como:

- src;
- disabled;
- value;
- checked.

```
<section>
    <img [src]="logotipo" alt="Logo IPCA" class="logo">
    <h1>{{ titulo }}</h1>
</section>
```

```
export class Header {
  titulo: string = 'Título';
  logotipo: string = 'logo.png';
  nome: string = "";
```

Event Binding ()

Permite capturar eventos do HTML e executar métodos no TS, que irão ditar o comportamento do evento. Temos também uma série de eventos comuns:

- (click);
- (input);
- (change);
- (keyup).

```
<section>
    <img [src]="logotipo" alt="Logo IPCA" class="logo">
    <h1>{{ titulo }}</h1>

    <button (click)="clicar()">Clique aqui</button>
    <input type="text" placeholder="Digite algo..." [(ngModel)]="nome">
</section>
```

```
clicar() {
  alert('Botão clicado!');
}
```

Two-way data binding [()]

Cria uma comunicação em duas direções entre o TypeScript e o HTML, ele permite enviar dados do TS para o HTML e atualizar o TS quando o utilizador altera o valor.

```
<section>
    <img [src]="logotipo" alt="Logo IPCA" class="logo">
    <h1>{{ titulo }}</h1>

    <button (click)="clicar()">Clique aqui</button>
    <input type="text" placeholder="Digite algo..." [(ngModel)]="nome">
</section>
```

```
export class Header {
  titulo: string = 'Título';
  logotipo: string = 'logo.png';
  nome: string = "";

  clicar() {
    alert('Botão clicado!');
  }
```

Diretivas Estruturais

O que são diretivas estruturais?

As diretivas estruturais controlam a criação e remoção de elementos no DOM, elas nos permitem:

- Mostrar ou esconder elementos;
- Renderizar listas;
- Controlar fluxo de exibição.

Diretiva @if

Permite mostrar ou esconder elementos com base em uma condição. Se a condição for verdadeira o elemento aparece, se for falso o elemento não aparece.

```
@if (isLogado) {
    <p>Você está logado!</p>
}
<p *ngIf="isLogado">Você  está logado!</p>
```

```
isLogado: boolean = true;
```

```
imports: [FormsModule, CommonModule],
```

*ngIf implica na importação do CommonModule para ser utilizado.

Diretiva @else

Umas das vantagens da nova sintaxe de condicionais, é a possibilidade de termos o @else, tornando o fluxo lógico igual ao ficheiro de TS, algo que não existia na sintaxe antiga.

```
@if (isLogado) {
    <p>Você está logado!</p>
} @else {
    <p>Você não está logado!</p>
}
```

```
<p *ngIf="!isLogado">Você não está logado!</p>
```

```
<p *ngIf="isLogado" else="naoLogado">Você  está logado!</p>
<p *ngIf="!isLogado">Você não está logado!</p>

<ng-template #naoLogado>
    <p>Você não está logado!</p>
</ng-template>
```

Anteriormente era necessário invertermos o valor lógico da variável. Ou adicionarmos o else com uma variável adicional junto a um template.

Diretiva @else if

Permite verificar

múltiplas condições,

assim como no TS.

```
@if (controleParental >= 18) {
    <p>Conteúdo para maiores de 18 anos.</p>
} @else if (controleParental >= 15) {
    <p>Conteúdo recomendado para maiores de 14 anos.</p>
} @else {
    <p>Conteúdo livre para toda as idades.</p>
}
```

```
controleParental: number = 18;
```

Diretiva @for

O @for é é usado para renderizar listas no HTML de forma dinâmica, o array será percorrido, criando um elemento a cada item iterado.

```
nomes = ["Ana", "João", "Maria"];
```

```
@for (nome of nomes; track nome) {
    <p>{{ nome }}</p>
}
```

```
<p *ngFor="let nome of nomes">{{ nome }}</p>
```

Diretiva @switch

Permite renderizar diferentes elementos com base no valor de uma variável, funciona como o switch do TS.

```
@switch (tipo) {
    @case ('admin'){
        <p>Bem-vindo, administrador!</p>
    }
    @case ('user') {
        <p>Bem-vindo, usuário!</p>
    }
    @default {
        <p>Bem-vindo, visitante!</p>
    }
}
```

```
<div [ngSwitch]="tipo">
    <p *ngSwitchCase="'admin'">Administrador</p>
    <p *ngSwitchCase="'user'">Usuário</p>
    <p *ngSwitchDefault>Visitante</p>
</div>
```

Pipes

Definição

Pipes são utilizados para transformar os dados diretamente no template HTML, sem alterar o valor original no TypeScript. Eles permitem por exemplo:

- Transformar texto em maiúsculas;
- Formatar datas;
- Formatar números como moeda;
- Formatar percentagens.

Pipes vs lógica no TS

```
distrito: string = "Braga";
cidade: string = "Barcelos";
```

```
distritoUpperCase: string = this.distrito.toUpperCase();
```

```
<p>Distrito: {{ distritoUpperCase }}</p>
<p>Cidade: {{ cidade | uppercase }}</p>
```

O Pipe permite realizar a mesma ação diretamente no HTML, sem necessidade de criar uma lógica no TS.

Tipos comuns de Pipes

date:

```
<p>Data de hoje: {{ hoje | date }}</p>
```

```
hoje = new Date();
```

Date com parâmetros:

```
<p>Hoje com parâmetros: {{ hoje | date:'dd/MM/yyyy' }}</p>
```

Currency:

```
<p>Preço: {{ preco | currency:'EUR' }}</p>
```

```
preco: number = 19.99;
```

Percent:

```
<p>Progresso: {{ progresso | percent }}</p>
```

```
progresso: number = 0.75;
```

Innominatum CONSULTORIA

Pipes Customizados

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL

Definição

Pipes customizados são pipes criados pelo desenvolvedor para transformar dados de forma personalizada, quando os pipes fornecidos pelo Angular não atendem à necessidade. Eles permitem criar regras específicas de formatação diretamente no template.

Pipe para saudação baseado no horário

Vamos construir um pipe que baseado no horário do dispositivo do utilizador, teremos a saudação como bom dia, boa tarde ou boa noite.

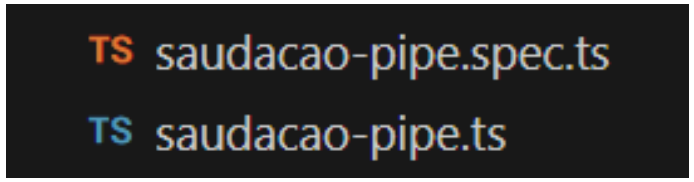
```
<p>{{ "João" | saudacao }}</p>
```

Antes precisamos criar esse pipe utilizando o Angular CLI:

ng generate pipe saudacao

Ficheiros criados

Ao criarmos o pipe no Angular CLI, nos é gerado dois ficheiros:



Como não definimos na linha de comando em qual diretório seria criado esse ficheiro, os mesmos foram criados dentro da pasta app.

Estrutura de um Pipe

```
@Pipe({
  name: 'saudacao',
  standalone: true
})
```

Diretiva Pipe registra uma classe como um pipe no Angular, é responsável pela configuração do Pipe, como a definição do nome e formato de importação.

```
transform(nome: string): string {
  const hora = new Date().getHours();

  if(hora < 12) {
    return `Bom dia, ${nome}`;
  }

  if (hora < 18) {
    return `Boa tarde, ${nome}`;
  }

  return `Boa noite, ${nome}`;
}
```

É o coração do Pipe, é ele quem recebe um valor e retorna o valor transformado que será exibido no template, em outras palavras, ele define o que um pipe faz.

Síntese

- **Data Binding**: Aplicação dos diferentes tipos de binding para estabelecer ligação entre o estado do componente e o template;
- **Diretivas Estruturais de controlo do DOM**: Nova sintaxe moderna (@if, @for, @switch, @else, @case) para controlar a renderização condicional e iterativa;
- **Pipes para transformação declarativa de dados**: Aplicação de pipes nativos para formatação de valores diretamente no template;
- **Criação de Pipes Customizados**: Implementação de pipes personalizados utilizando o decorator @Pipe e o método transform().

Innominatum CONSULTORIA

Conclusão

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL

Innominatum CONSULTORIA

Programação em JavaScript

ServiceNow - Deloitte

Rodrigo Costa

IPCA POLITÉCNICO DO CÁVADO E DO AVE
INSTITUTO DO EMPREGO E FORMAÇÃO PROFISSIONAL
PRR Plano de Recuperação e Resiliência
REPÚBLICA PORTUGUESA
Financiado pela União Europeia NextGenerationEU
IAPMEI Parcerias para o Crescimento
PORTUGAL DIGITAL