Template Variables em Angular

As *Template Variables* são usadas para criar referências locais a elementos ou diretivas em um template. Elas permitem acessar e manipular propriedades desses elementos diretamente no HTML, sem precisar criar vínculos no componente TypeScript.

Passagem por Parâmetro e Escopo

1. Passagem por parâmetro em funções

Você pode passar variáveis de template como argumento para métodos no componente.

```
<input #meuInput type="text" placeholder="Digite algo" />
<button (click)="mostrarValor(meuInput)">Exibir Valor</button>

<!-- Output na tela -->
<p>Você digitou: {{ valor }}</p>
```

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
})
export class AppComponent {
  valor = '';

  mostrarValor(input: HTMLInputElement) {
    // Pega o valor atual do campo de entrada e atribui à variável "valor"
    this.valor = input.value;
    console.log('Valor do input:', input.value);
  }
}
```

Explicação

- O `#meuInput` cria uma variável de template associada ao campo `<input>`.
- O valor de `meuInput` é passado para a função `mostrarValor` ao clicar no botão.
- O componente exibe e registra o valor digitado no console.

---

2. Comunicação entre elementos usando Template Variables

Você pode sincronizar o valor de um elemento com outro diretamente no template.

<input #inputOrigem type="text" placeholder="Digite algo" />
Origem: {{ inputOrigem.value }}

<input type="text" [value]="inputOrigem.value" placeholder="Sincronizado automaticamente" />

Explicação

- O `#inputOrigem` cria uma variável de template associada ao primeiro campo de entrada.
- O valor do primeiro `<input>` é exibido no parágrafo e sincronizado automaticamente com o segundo `<input>` usando `[value]`.

---

3. Passagem condicional no escopo

O escopo das variáveis de template é limitado ao elemento onde são criadas e seus filhos. Isso significa que variáveis criadas em um "pai" podem ser acessadas em "filhos", mas não em "irmãos".

  <input #inputPai type="text" placeholder="Valor do Pai" />
  Pai: {{ inputPai.value }}

    Filho acessa: {{ inputPai.value }}

```
</div>

<div class="primo">
  <!-- Não funciona, pois inputPai está fora do escopo -->
  <p>Primo tenta acessar: {{ inputPai?.value }}</p>
</div>
```

Explicação

- O `inputPai` está no escopo do `<div class="pai">` e seus descendentes diretos.
- O "filho" pode acessar `inputPai`, mas o "primo" não pode.

O que pode ser feito com Template Variables

1. Interação com diretivas

Você pode usar variáveis de template para interagir com diretivas Angular.

```
<div *ngIf="true; then blocoTrue; else blocoFalse"></div>

<ng-template #blocoTrue>
  <p>Bloco Verdadeiro</p>
</ng-template>

<ng-template #blocoFalse>
  <p>Bloco Falso</p>
</ng-template>
```

Explicação

- As variáveis `#blocoTrue` e `#blocoFalse` referenciam os templates definidos no HTML.
- A diretiva `*ngIf` decide qual bloco exibir com base na condição.

---

2. Controle de estado de elementos do DOM

Você pode referenciar elementos DOM para manipulação direta.

```
<input #meuInput type="text" placeholder="Digite algo" />
<button (click)="limpar(meuInput)">Limpar</button>

<p>Valor Atual: {{ meuInput.value }}</p>

import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
})
export class AppComponent {
  limpar(input: HTMLInputElement) {
    input.value = ''; // Limpa o valor do campo diretamente
  }
}
```

---

3. Uso com Componentes Filhos

Template Variables podem referenciar componentes filhos para acessar seus métodos e propriedades.

```
<app-filho #meuFilho></app-filho>
<button (click)="meuFilho.metodoDoFilho()">Chamar método do
filho</button>

import { Component } from '@angular/core';

@Component({
  selector: 'app-filho',
  template: `<p>Componente Filho</p>`,
})
export class FilhoComponent {
  metodoDoFilho() {
    console.log('Método do filho foi chamado!');
  }
}
```

Explicação

- O `#meuFilho` cria uma variável de template referenciando o componente `FilhoComponent`.
- O método `metodoDoFilho` pode ser chamado diretamente a partir do template do componente pai.

---

4. Controle de Formulários

Template Variables são úteis para manipular formulários.

```
<form #meuForm="ngForm" (ngSubmit)="onSubmit(meuForm)">
  <input name="nome" ngModel placeholder="Nome" required />
  <button type="submit">Enviar</button>
</form>

<p>Formulário é válido? {{ meuForm.valid }}</p>

onSubmit(form: any) {
  console.log('Dados do formulário:', form.value);
}
```

Explicação

- `#meuForm` referencia o formulário e permite verificar seu estado e acessar seus valores.
- O `ngForm` facilita a vinculação de dados e validação automática.

Principais pontos sobre Template Variables

1. **Escopo**
    - Variáveis de template só podem ser acessadas no mesmo elemento e nos seus descendentes diretos.
2. **Passagem como Parâmetro**

    - Podem ser passadas para métodos no componente.
3. **Interação com Componentes Filhos**
    - Podem acessar propriedades e métodos de componentes filhos diretamente.
4. **Manipulação de Elementos DOM**
    - Útil para acessar propriedades e manipular elementos diretamente no template.
5. **Validação de Formulários**
    - Facilitam o acesso e controle do estado de formulários.

Esses exemplos mostram como as Template Variables podem ser usadas para resolver problemas reais e melhorar a interatividade no Angular, sempre respeitando os limites do escopo.