

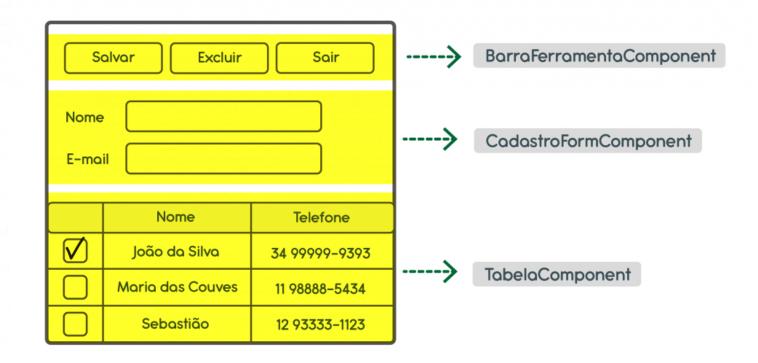
# Formulários

Geucimar Briatore geucimar@up.edu.br

Atualizado em 03/2022

# Criação do componente CadastroPessoaComponent





#### **Formulários**

- Os formulários reativos fornecem acesso direto e explícito ao modelo de objeto de formulários subjacente. Em comparação com os formulários orientados por modelo, eles são mais robustos: são mais escaláveis, reutilizáveis e testáveis. Se os formulários são uma parte importante do seu aplicativo, ou se você já usa padrões reativos para construir seu aplicativo, use os formulários reativos;
- Os formulários orientados a modelos contam com diretivas no modelo para criar e manipular o modelo de objeto subjacente. Eles são úteis para adicionar um formulário simples a um aplicativo, como um formulário de inscrição de lista de e-mail. Eles são fáceis de adicionar a um aplicativo, mas não escalam tão bem quanto as formas reativas. Se você tiver requisitos de formulário e lógica muito básicos que podem ser gerenciados apenas no modelo, os formulários orientados por modelo podem ser uma boa opção.

https://angular.io/guide/forms-overview

## Vinculação de evento (event binding)

```
<a (click)="aoExcluir()" class="excluir">Excluir</a>
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
export class AppComponent {
  aoExcluir() {
     console.log('Excluído!');
```

# Interpolação e transformação de dados

- Interpolação: permite a inclusão de valores dinâmicos nos templates através da inclusão de expressões nas tags. Por padrão a interpolação utiliza as <u>chaves duplas</u> {{ }};
- Transformação de dados (pipes): são utilizados para transformar strings, valores de moeda, datas e outros dados para exibição. São funções usadas em expressões como operador | para aceitar um valor de entrada e retornar um valor transformado.

https://angular.io/api?type=pipe

## Exemplos de interpolação e transformação e dados

```
<h1> {{ algumTexto }} </h1> ou <h1> {{ umaFunc() }} </h1>
 A soma de 1 + 1 é {{ 1 + 1 }}, mas 1 - 1 não é {{ 1 + 1 + getVal() }} 
 A data de Nascimento é {{ dtaNasc | date:"MM/dd/yy" }} 
. . .
export class AppComponent {
   algumTexto = 'Algum texto!';
   umaFunc() { return 'Outro texto!';}
   getVal() { return 1;}
   dtaNasc = new Date(1998, 3 ,15);
```

#### Diretivas: estruturais e de atributos

- As diretivas são classes que adicionam comportamento aos elementos dos aplicações Angular;
- As diretivas estruturais alteram o fluxo de layout adicionando, removendo e substituindo elementos no DOM;
- As diretivas de atributo alteram a aparência ou comportamento de um elemento existente. Nos <u>templates</u> eles se parecem com as tags regulares de HTML.

### **Principais diretivas estruturais**

- nglf: condicionalmente cria ou descarta visualizações do modelo;
- ngFor: repete um elemento para cada item em uma lista;
- ngSwitch: um conjunto de diretivas que alterna entre visualizações alternativas.

https://angular.io/guide/built-in-directives#built-in-structural-directives

### **Exemplo de diretivas estruturais**

```
<span *ngIf="ehVerdade"> {{ msgOK }} </span>
<span *ngIf="ehFalso === 'falso'"> {{ msgFalso }} </span>
*ngFor="let p of pessoas" [ngSwitch]="p.uf">
  *ngSwitchCase="'PR'">{{ p.nome }} é paranaense. 
  {{ p.nome }} é catarinense.) 
  *ngSwitchCase="'RS'">{{ p.nome }} é gaúcho. 
  *ngSwitchDefault> {{ p.nome }} não é sul brasilense.
```

### Principais diretivas de atributos

- ngClass: adiciona e remove um conjunto de classes CSS;
- ngStyle: adiciona e remove um conjunto de estilos HTML;
- ngModel: adiciona vínculo de dados bidirecional a um elemento de formulário HTML.

https://angular.io/guide/built-in-directives#built-in-attribute-directives

### **Exemplos de diretivas de atributos**

```
<div [ngClass]="isEsSpecial ? 'especial' : ''">
    Este cliente é especial!
</div>
<div [ngStyle]="funcEstilos()">
    Este texto está em itálico, negrito e 24px.
</div>
<label for="exemplo-ngModel">[(ngModel)]:</label>
<input [(ngModel)]="item.nome" id="exemplo-ngModel">
```