

Projeto de Interfaces WEB

Introdução ao Angular Aula 02

Diretivas

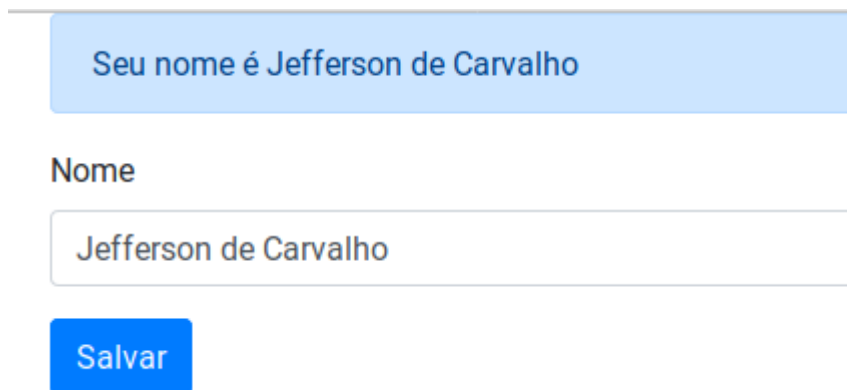
- Instruções passada no template.
 - **Componentes**
 - Código html atrelado ao componente (.ts)
 - `<app-hello></app-hello>`, por exemplo (instrução)
 - **Diretivas Estruturais**
 - Altera o DOM da página
 - `<h1 *ngIf="autenticado"> Olá {{usuario}}!</h1>`
 - **Diretivas de Atributos**
 - Modifica o comportamento ou a aparência do elemento **sem** alterar o DOM
 - `<h2 [style.color]="red"> Olá {{usuario}}</h2>`

Diretivas - DOM

- <https://tableless.com.br/entendendo-o-dom-document-object-model/>
- “O DOM (Document Object Model) é uma interface que representa como os documentos HTML e XML são lidos pelo seu browser. Após o browser ler seu documento HTML, ele cria um objeto que faz uma representação estruturada do seu documento e define meios de como essa estrutura pode ser acessada. Nós podemos acessar e manipular o DOM com JavaScript, é a forma mais fácil e usada.”

Diretivas ngIf e Hidden

- No exemplo anterior, vamos mostrar o alerta apenas quando o usuário clicar no botão salvar.



Seu nome é Jefferson de Carvalho

Nome


Jefferson de Carvalho

Salvar

The image shows a web form with a light blue header bar. Below the header, there is a light blue rectangular box containing the text 'Seu nome é Jefferson de Carvalho'. Below this box, the label 'Nome' is followed by a text input field containing 'Jefferson de Carvalho'. At the bottom of the form is a blue button with the text 'Salvar'.

Diretivas ngIf e Hidden

- No app.component.ts
 - Adicionar uma variável booleana “clicou”

```
export class AppComponent {  
  nome = 'Jefferson de Carvalho';  
  contador = 0;  
  clicou = false;  
  
  salvar(nome:string){  
    console.log(`Salvando ${this.nome}`);  
    this.nome = "Jefferson " + this.contador;  
    this.contador++;  
    this.clicou = true;   
  }  
  
  mudar(event:any){  
    this.nome = event.target.value;  
  }  
}
```

Diretivas ngIf e Hidden

- No app.component.html (diretiva estrutural)
 - Inserir a diretiva *ngIf na div do alerta:

```
<div class="alert alert-primary" role="alert" *ngIf="clique">  
  Seu nome é {{nome}}  
</div>
```

Nome



Seu nome é Jefferson 0

Nome

Diretivas ngIf e Hidden

- <https://angular.io/api/common/NgIf>
- “A structural directive that conditionally **includes** a **template** based on the **value** of an expression coerced to Boolean. When the expression evaluates to true, Angular renders the template provided in a **then** clause, and when false or null, Angular renders the template provided in an optional **else** clause. The default template for the else clause is blank”.

Diretivas ngIf e Hidden

- Exercício
 - Use um eventBinding (focus) para que quando o input fique em foco, o alerta desapareça!
 - Resposta no próximo slide. Tente não ver...

Diretivas ngIf e Hidden

- Resposta:

```
<input type="text" class="form-control"  
      [(ngModel)]="nome"  
      (focus)="clique=false">
```

Diretivas ngIf e Hidden

- Há também a possibilidade de usar a diretiva de atributo, não alterando assim o DOM.

```
<div class="alert alert-primary" role="alert" [hidden]="!clicou">  
  Seu nome é {{nome}}  
</div>
```

Diretiva ngFor

- Vamos adicionar uma lista de pessoas no nosso HTML.
- Inicialmente, vamos até a página de componentes do bootstrap.
- Procure pelo componente “**cards**” e copie seu exemplo.

```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's
content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

Diretiva ngFor

- No app.component.html

```
...  
<button type="button" class="btn btn-primary" (click)="salvar()"  
[disabled]="nome.length==0">Salvar</button>  
  
<div class="card" style="width: 18rem;">  
    
  <div class="card-body">  
    <h5 class="card-title">Card title</h5>  
    <p class="card-text">Some quick example text to build on the card title and make up the bulk of the  
    card's content.</p>  
    <a href="#" class="btn btn-primary">Go somewhere</a>  
  </div>  
</div>
```

Diretiva ngFor

- Melhorando: (apagando informações desnecessárias)
- Usando também a class="row", do bootstrap.

```
...  
<button type="button" class="btn btn-primary" (click)="salvar()"  
[disabled]="nome.length==0">Salvar</button>  
  
<div class="row" style="margin-top: 20px">  
  <div class="col-2">  
  
    <div class="card" >  
        
      <div class="card-body">  
        Maria  
      </div>  
    </div>  
  
  </div>  
</div>
```

Diretiva ngFor

- Em app.component.ts, criar o array de pessoas.

```
export class AppComponent {  
  nome = 'Jefferson de Carvalho';  
  contador = 0;  
  clicou = false;  
  pessoas = [];  
  
  salvar(nome:string){  
    console.log(`Salvando ${this.nome}`);  
    this.nome = "Jefferson " + this.contador;  
    this.contador++;  
    this.clicou = true;  
    this.pessoas.push(this.nome);  
  }  
}
```

Diretiva ngFor

- Em app.component.html, usar o ngFor.

```
<div class="row" style="margin-top: 20px">
  <div class="col-2" *ngFor="let pessoa of pessoas">
    <div class="card">
      
      <div class="card-body">
        {{pessoa}}
      </div>
    </div>
  </div>
</div>
```

Diretiva ngFor

- Finalizando o código (app.component.ts)

```
export class AppComponent {  
  nome = 'Jefferson de Carvalho';  
  contador = 0;  
  clicou = false;  
  pessoas = [];  
  
  salvar(nome:string){  
    console.log(`Salvando ${this.nome}`)  
    this.contador++;  
    this.clicou = true;  
    this.pessoas.push({nome:this.nome,  
                      id:this.contador});  
  }  
  ...  
}
```


Diretiva ngFor

- Finalizando o código (app.component.html)

```
<div class="row" style="margin-top: 20px">
  <div class="col-2" *ngFor="let pessoa of pessoas">
    <div class="card">
      
      <div class="card-body">
        {{pessoa.nome}}
      </div>
    </div>
  </div>
</div>
```

Binding @Input

- Como passar dados para propriedade customizadas dos nossos componentes.
- Vamos criar um novo componente:
 - Abra o terminal do VS
 - `ng g c pessoa-card`

Binding @Input

- Em pessoa-card html:
 - Cole o código do card

```
<div class="card">  
    
  <div class="card-body">  
    {{pessoa.nome}}  
  </div>  
</div>
```

Binding @Input

- Agora chame o seletor de pessoas dentro de app.component.html

```
<div class="row" style="margin-top: 20px">  
  <div class="col-2" *ngFor="let pessoa of pessoas">  
    <app-pessoa-card></app-pessoa-card>  
  </div>  
</div>
```

Binding @Input

- Deve-se agora fazer o “binding” entre a variável pessoa de app.component e alguma variável dentro do novo componente.
- No novo componente, criaremos:

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

```
@Component({  
  selector: 'app-pessoa-card',  
  templateUrl: './pessoa-card.component.html',  
  styleUrls: ['./pessoa-card.component.css']  
})
```

```
export class PessoaCardComponent {
```

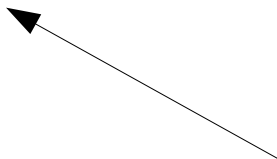
```
  @Input() pessoa: any; //NOTE QUE O INPUT SERVE PARA TORNAR A VARIÁVEL VISÍVEL PARA OUTROS COMPONENTES..
```

```
}
```

Binding @Input

- Voltando ao app.component.html:

```
<div class="row" style="margin-top: 20px">  
  <div class="col-2" *ngFor="let p of pessoas">  
    <app-pessoa-card [pessoa]="p"></app-pessoa-card>  
  </div>  
</div>
```



Objeto sendo passado por parâmetro.

Binding @Input


- Teste novamente a página:

Seu nome é Ingrid


Nome

Ingrid


Salvar



Maria



Chen Ly



Ingrid

Binding @Input

- É possível também criar um alias no input para o nome da propriedade. Só não esqueça de mudar também no seletor.

```
@Input('obj-pessoa') pessoa:any;
```



```
<app-pessoa-card [obj-pessoa]="p"></app-pessoa-card>
```

The diagram consists of a thin black arrow pointing from the text `@Input('obj-pessoa')` in the code block above to the `[obj-pessoa]` attribute in the code block below.

Binding @Output e EventEmitter

- Binding de eventos.
- Comunicação entre componentes, e duas vias.
- Vamos criar um novo componente:
 - `ng g c pessoa-form --spec=false`

Binding @Output e EventEmitter

- Em pessoa-form.html, copie e cole o código do label, input e button do app.component.html. Não esqueça de chamar o seletor de pessoa-form e app.componente.ts

```
<div class="alert alert-primary" role="alert" [hidden]="!clitou">  
  Seu nome é {{nome}}  
</div>
```

```
<div class="form-group">  
  <label>Nome</label>  
  <input type="text" class="form-control" [(ngModel)]="nome" (focus)="clitou=false">  
</div>  
<button type="button" class="btn btn-primary" (click)="salvar()"  
[disabled]="nome.length==0">Salvar</button>
```

Binding @Output e EventEmitter

- pessoa-form.ts

```
import { Component, OnInit, Output, EventEmitter } from '@angular/core';
```

```
..
```

```
export class PessoaFormComponent {
```

```
  nome = 'Jefferson de Carvalho';
```

```
  contador = 0;
```

```
  clicou = false;
```

```
  @Output('criado') pessoaSalva = new EventEmitter();
```

```
  salvar(){
```

```
    this.contador++;
```

```
    this.clicou = true;
```

```
    const pessoa = {nome:this.nome,  
                    id:this.contador};
```

```
    this.pessoaSalva.emit(pessoa);
```

```
  }  
}
```

Binding @Output e EventEmitter

- App.component.html

```
<div class="container">
  <app-pessoa-form (criado) = 'aoSalvar($event)'></app-pessoa-
form>
  <div class="row" style="margin-top: 20px">
    <div class="col-2" *ngFor="let p of pessoas">
      <app-pessoa-card [obj-pessoa]="p"></app-pessoa-card>
    </div>
  </div>
</div>
```

Binding @Output e EventEmitter

- App.component.html

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

```
@Component({  
  selector: 'app-root',  
  templateUrl: './app.component.html',  
  styleUrls: ['./app.component.css']  
})
```

```
export class AppComponent {  
  pessoas = [];
```

```
  aoSalvar(pessoa){  
    this.pessoas.push(pessoa);  
  }  
}
```

Binding @Output e EventEmitter

- Como funciona?

- Em pessoa-form.html, (click)="salvar()" chama o método salvar de seu componente.
- pessoa-form.ts faz:
 - **@Output('criado')** pessoaSalva = **new EventEmitter()**; Onde é criado um objeto “pessoaSalva” do tipo EVENTO!
 - No método “salvar”, um objeto “pessoa” é criado normalmente com id e nome. Logo depois, a chamada **this.pessoaSalva.emit(pessoa)**; faz com o objeto “pessoaSalva” se ligue com app.component.html, passando como parâmetro uma pessoa.
- Em app.component.html, **<app-pessoa-form (criado) = 'aoSalvar(\$event)'></app-pessoa-form>** fica esperando ser chamado pelo emit, explicado anteriormente. Note que o objeto “pessoa” passado pelo emit é capturado pelo \$event.
- Finalmente, o objeto “pessoa” recebido no event é passado para app.component.ts, onde é adicionado ao array de **pessoas[]**, sendo assim visto por outros componentes filhos, como o pessoa-card, por exemplo.

CSS

- Como adicionar estilos CSS nos nossos componentes.
- Meta dados styleUrls
- Basta adicionar algo ao template indicado no metadados.
- Ou, você pode usar template literals dentro do próprio componente.

CSS

- Mexendo dentro do metadados:

```
@Component({  
  selector: 'app-pessoa-card',  
  templateUrl: './pessoa-card.component.html',  
  //styleUrls: ['./pessoa-card.component.css']  
  styles: [`  
    .card-body{  
      text-transform: uppercase;  
      color:blue;  
    }  
  `]  
})
```


CSS

- No arquivo css:

```
peessoa-form.component.html  TS pessoa-card.component.ts  # pessoa-form.component.css x
1  label{
2    color: red;
3  }
```

CSS com ngStyle

- Como adicionar estilos de forma dinâmica, através do ngStyle.
- Vamos trabalhar no pessoa-card.

CSS com ngStyle

- pessoa-card.html

```
<div class="card" [ngStyle]="getEstilosCard()">
  
  <div class="card-body">
    {{pessoa.nome}}
  </div>
</div>
```

CSS com ngStyle

- pessoa-card.ts

```
export class PessoaCardComponent {  
  
  @Input('obj-pessoa') pessoa:any;  
  
  getEstilosCard(){  
    return {  
      'border-width.px':this.pessoa.id,  
      backgroundColor:this.pessoa.id%2==0?'lightblue':'lightgreen'  
    };  
  }  
}
```

CSS com ngClass

- Classes dinâmicas.
- Vamos usar o componente “badge”, do bootstrap.

CSS com ngClass

- pessoa.card.html

```
<div class="card" [ngStyle]="getEstilosCard()">
  
  <div class="card-body">
    <span class="badge" [ngClass]="{'badge-danger':isAdmin(),'badge-
primary':!isAdmin()}">
      {{pessoa.nome}}
    </span>
  </div>
</div>
```

CSS com ngClass

- pessoa-card.ts

```
export class PessoaCardComponent {  
  
  @Input('obj-pessoa') pessoa:any;  
  
  isAdmin(){  
    return this.pessoa.nome.startsWith('J');  
  }  
  
  getEstilosCard(){  
    return {  
      'border-width.px':this.pessoa.id,  
      backgroundColor:this.pessoa.id%2==0?'lightblue':'lightgreen'  
    };  
  }  
}
```

E...

- Por hoje é só pessoal...