APLICAÇÕES INTERATIVAS

RESOLUÇÃO DE EXERCÍCIOS DA AULA ANTERIOR

Exercícios de Event Binding

6. Crie 3 botões. Quando clicados, cada um deles deve atribuir um texto específico para uma propriedade. Use a propriedade num texto simples para exibir seu valor conforme os botões são clicados

Exercícios de Event Binding

7. Crie três botões e 3 divs. Ao serem clicados, os botões devem controlar quais classes cada div possui, alternando entre classes em que uma div está visível e as outras duas estão invisíveis, para cada uma das divs (alternando entre divs 1, 2 e 3)

ANGULAR
BINDINGS DE VIA DUPLA

Binding de Duas Vias

- Permite que alterações no template (edição do usuário) ou no código sejam refletidas de um para outro automaticamente
- Muito útil para utilização em caixas de texto e elementos onde o usuário entra com dados

Binding de Duas Vias

- Quando o usuário interagir na página com o componente (por exemplo, digitar um valor na caixa de texto), o valor será automaticamente atualizado em uma propriedade

Binding de Duas Vias

- Se a propriedade for alterada no código do componente, o valor também será atualizado automaticamente no HTML

Binding de Duas Vias

- A ligação de duas vias é declarada através da notação **[(atributo)]** (conhecida como "***banana in a box***" ou "banana na caixa")
- Como uma mistura dos dois *one-way bindings* (bindings de via única), formando o ***two-way binding*** **(ligação de duas vias)**

Binding de Duas Vias

- Para utilização do binding de duas vias, precisamos especificar qual atributo de um elemento HTML será ligado ao atributo do código-fonte da classe do componente

Binding de Duas Vias

- Para que o **valor** de elementos HTML (como caixas de texto) seja ligado a um atributo, normalmente utilizamos o elemento **ngModel**
- **O ngModel trata automaticamente o binding de duas vias usando o valor do componente HTML e os eventos de inserção de dados**

Binding de Duas Vias

```
<input type="text" [(ngModel)]="nome" >
```

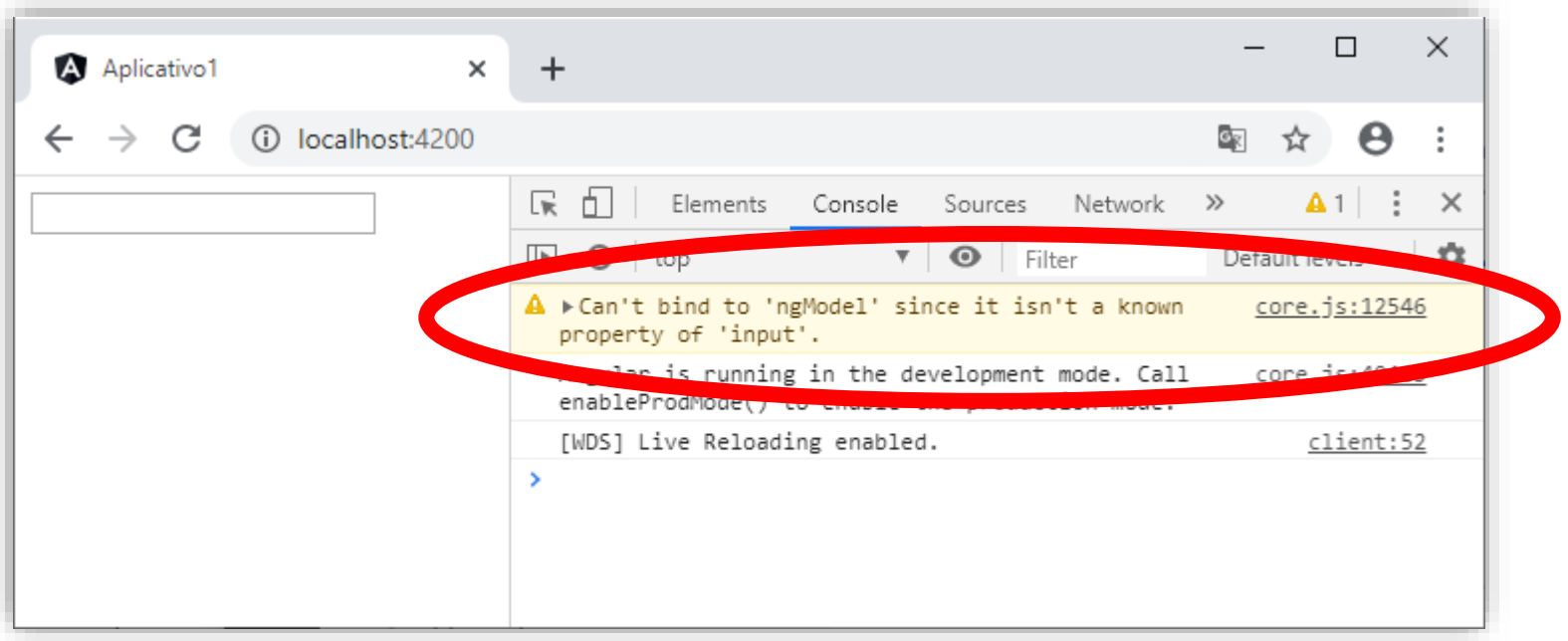
Binding de Duas Vias

```
<input type="text" [(ngModel)]="nome" >

export class TestComponent implements OnInit {

  nome = '';

  ...

}
```

Ativando Binding de Duas Vias

- O binding de duas vias não vem ativado por padrão no Angular
- Ao abrir a página, o seguinte erro pode ser apresentado:

Ativando Binding de Duas Vias



Ativando Binding de Duas Vias

- Para ativa-lo, é necessário acrescentar um módulo extra a aplicação, o **FormsModule** (é preciso realizar este procedimento apenas uma vez por app)

Ativando Binding de Duas Vias

- Para ativar um módulo no Angular, deve-se editar o arquivo principal de módulos, o **app.module.ts**

Ativando Binding de Duas Vias

- Edite o arquivo **app.module.ts** e acrescente os seguintes elementos:
  - **Na parte superior, acrescente o import ao módulo**

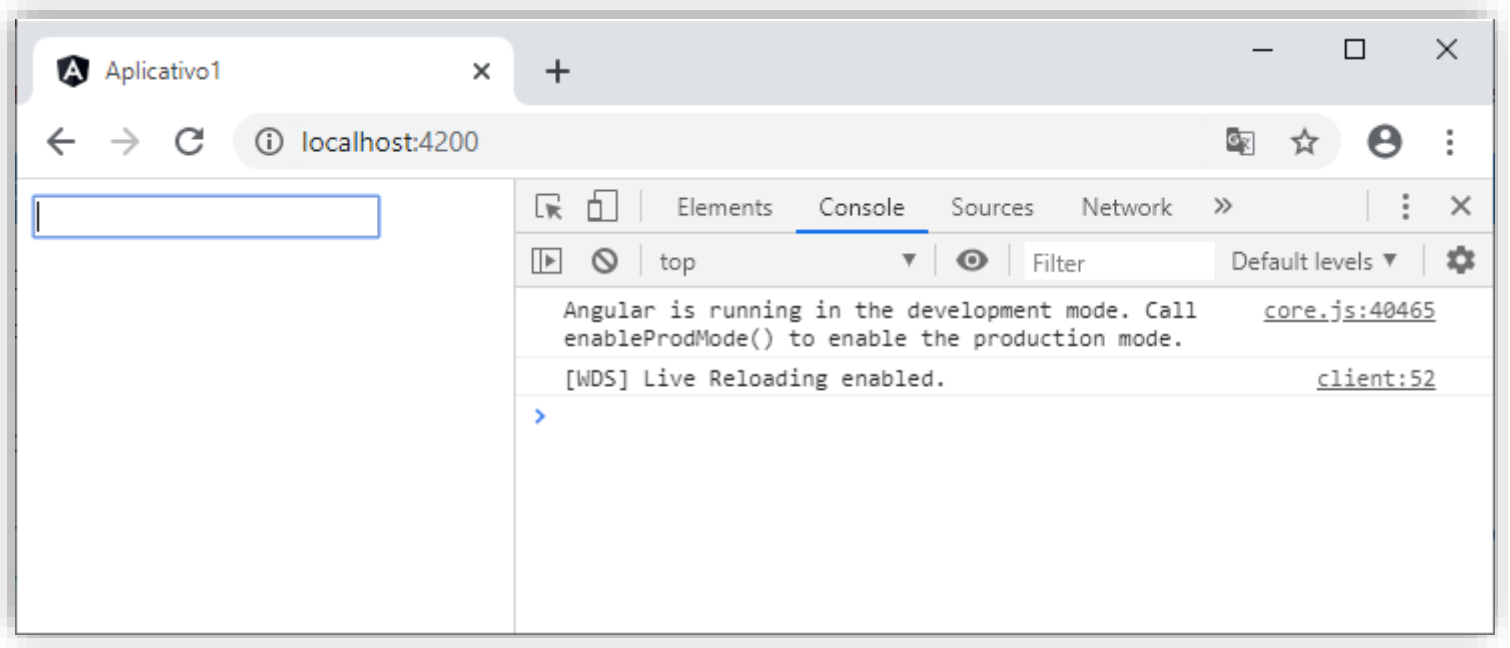
```
import { FormsModule } from '@angular/forms';
```

  - **Em "imports", insira o módulo FormsModule**

```
imports: [
  BrowserModule,
  FormsModule
],
```

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { TestComponent } from './test/test.component';
import { FormsModule } from '@angular/forms';

@NgModule({
  declarations: [
    AppComponent,
    TestComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Ativando Binding de Duas Vias



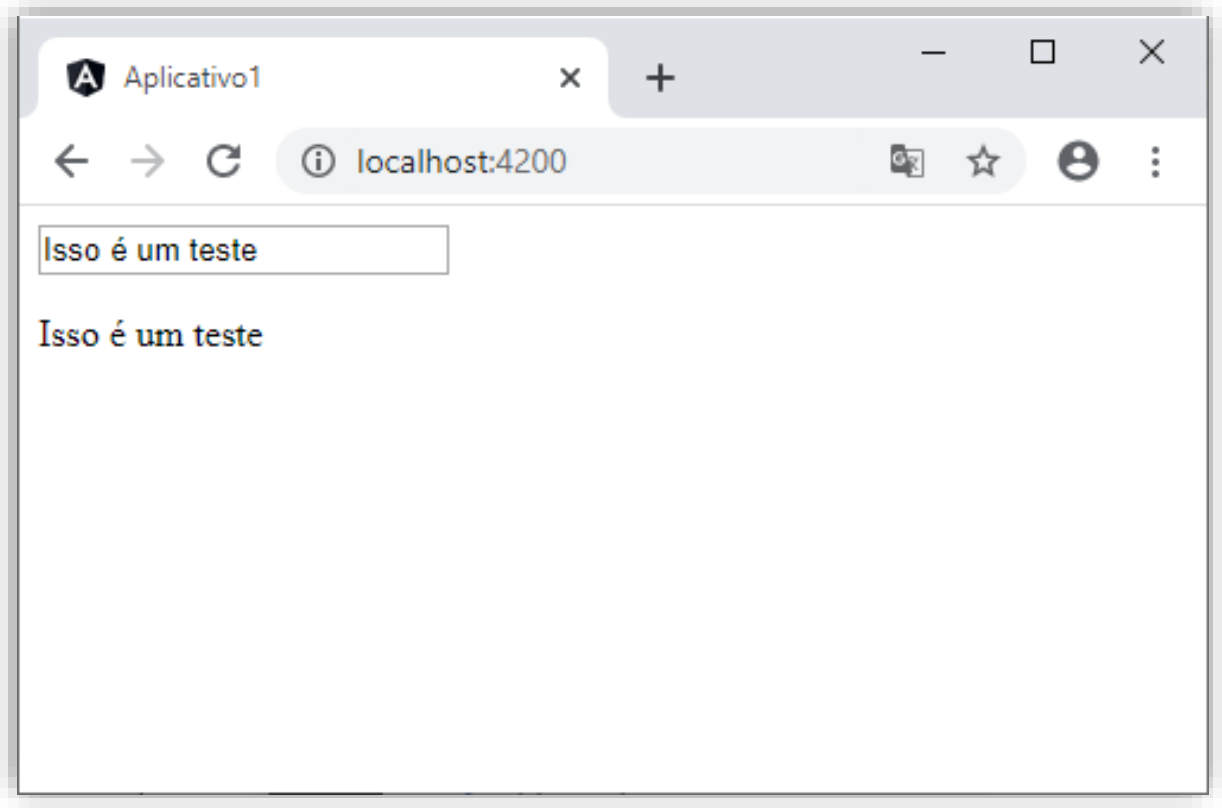
Duas Vias + Property Binding

- Se a propriedade usada no binding de duas vias (nome no nosso caso) tiver também um binding de via única (como um property binding ou interpolação) em outro elemento HTML, os valores digitados pelo usuário serão refletidos nesse elemento automaticamente

Duas Vias + Property Binding

```
<input type="text" [(ngModel)]="nome" >
<p>{{nome}}</p>

export class TestComponent implements OnInit {

  nome = '';

  ...

}
```

Duas Vias + Property Binding



Binding de Duas Vias

- O binding de duas vias nada mais é que uma união entre ambos os one-way bindings (da página para o código e do código para a página) numa forma mais compacta
- **É possível utilizar ambos os bindings separados, de forma mais extensa**

Outra Representação

```
<input type="text" [ngModel]="nome"
      (ngModelChange)="nome=$event" >
```

equivalente

```
<input type="text" [(ngModel)]="nome" >
```

Tipo de Bindings - Resumo

Tipo	Sintaxe	Categoria
Interpolação Property Binding	{{expressao}} [alvo]="expressao"	One-way (via única) do código para a tela
Binding de Eventos	(alvo)="declaracao"	One-way (via única) da tela para o código
Duas Vias	[(alvo)]="expressao"	Duas vias (do código para a tela e da tela para o código)

Exercícios de Binding de Duas Vlas

1. Crie 2 caixas de texto. Ao digitar um valor nas duas caixas de texto, a soma dos valores deve aparecer automaticamente num elemento p abaixo das mesmas

Exercícios de Binding de Duas Vlas

2. Crie uma caixa de texto, onde o usuário deverá digitar uma raça de cachorro. Quando uma raça válida for digitada, uma imagem da mesma deverá ser exibida. Faça isso para pelo menos 3 raças

BINDING EM OUTROS ELEMENTOS DE ENTRADA HTML

Binding em Elementos HTML

- Vimos como efetuar bindings de duas vias com elementos de entrada simples (input), mas é possível utilizar todos os elementos de formulário HTML para entrada de dados em Angular, como veremos a seguir

Entrada de Texto

- Elementos de entrada de texto (textarea, password, etc) são utilizados da mesma forma como o input type=“text” padrão
- Outros tipos de input (type=“*”, como number) também podem ser utilizados sem maiores problemas

Selects

```
<select [(ngModel)]="cidade">
    <option value="saopaulo">São Paulo</option>
    <option value="santos">Santos</option>
    <option value="campinas">Campinas</option>
    <option value="sorocaba">Sorocaba</option>
</select>
<p>{{cidade}}</p>
```

```
...
export class TestComponent implements OnInit {
  cidade = '';

  ...
}
```

Checkboxes

```
<input id="enviar-email" type="checkbox" [(ngModel)]="email" />
<label for="enviar-email">Enviar E-mail</label>

<input id="contato-telefonico" type="checkbox" [(ngModel)]="contTel" />
<label for="Contato Telfônico">Entrar em Contato por Telefone</label>


<p>Email: {{email}}</p>
<p>Telefone: {{contTel}}</p>
```

```
...
export class TestComponent implements OnInit {
  email = false;
  contTel = true;

  ...

}
```

radio

```
<p>
    <input type="radio" value="masc" id="gen-masc" [(ngModel)]="genero" >
    <label for="gen-masc">Masculino</label>

    <input type="radio" value="fem" id="gen-fem" [(ngModel)]="genero" >
    <label for="gen-fem">Feminino</label>

    <input type="radio" value="out" id="gen-out" [(ngModel)]="genero" >
    <label for="gen-out">Outro</label>
 </p>

<p>Gênero: {{genero}}</p>
```

```
...
export class TestComponent implements OnInit {
  genero = '';

  ...

}
```

Exercícios de Binding em Outros Elementos

3. Crie um select contendo alguns cursos do Senac e uma caixa de texto (nome). Quando for selecionado um curso e texto for digitado, informações na parte inferior devem ser atualizadas, exibindo o curso selecionado e o nome do candidato

Exercícios de Binding em Outros Elementos

- Crie 3 caixas de texto e rádios contendo as operações básicas. Quando valores forem digitados nas caixas de texto, a operação selecionada deve ser realizada e o resultado exibido na 3ª caixa

DIRETIVAS ESTRUTURAIS

Diretivas Estruturais

- Diretivas estruturais são responsáveis por modificar o layout HTML de acordo com condições específicas
- Elas são capazes de remodelar a estrutura do DOM, geralmente adicionando, removendo ou manipulando elementos

Diretivas Estruturais

- Para utilização de uma diretiva, é necessário aplica-la a um elemento hospedeiro
- A diretiva deve afetar esse elemento e também todos seus descendentes

Diretivas Estruturais

- É fácil reconhecer uma diretiva estrutural em um arquivo de template Angular, já que todas as diretivas são tratadas como atributos HTML e a grande maioria possui um asterisco (*) antes de seu nome de atributo

Diretivas Estruturais

- O Angular permite que suas próprias diretivas sejam criadas, ainda que as principais diretivas utilizadas sejam as *diretivas embutidas*
- *As diretivas embutidas existentes no Angular são *ngIf, *ngFor e ngSwitch*

**ngIf*

- O **ngIf* permite que um elemento HTML (bem como todos seus elementos filhos) seja processado apenas caso uma ou mais condições sejam satisfeitas

**ngIf*

- Esta diretiva pode ser utilizada em qualquer elemento HTML e fará com que seus elementos filhos não sejam exibidos (nem processados) caso a condição não seja satisfeita

**ngIf*

- Elementos não processados não são inseridos no DOM, nem exibidos ao usuário ou tem eventos monitorados
- Em termos práticos, é como se o trecho de HTML “não existisse”

**ngIf*

- Normalmente, utilizamos o **ngIf* numa “div” ou num container ao redor dos elementos cujo processamento deva ser condicional, mas é possível utilizá-lo diretamente no elemento desejado

*ngIf

```
<div *ngIf="condicao">

</div>
```

Exemplo 01

```
        <input type="text" [(ngModel)]="nome">

        <p *ngIf="nome">
            Nome: {{nome}}
        </p>
```

```
export class Exemplo01Component implements OnInit {
  nome = '';

  constructor() { }

  ngOnInit(): void {

  }

}
```

Exemplo 02

```
<button (click)="exibirComponente('exercicio01')">Exercício 01</button>
<button (click)="exibirComponente('exercicio02')">Exercício 02</button>
<button (click)="exibirComponente('exemplo01')">Exemplo 01</button>

<app-exercicio01 *ngIf="componenteAtual === 'exercicio01'"></app-exercicio01>
<app-exercicio02 *ngIf="componenteAtual === 'exercicio02'"></app-exercicio02>
<app-exemplo01 *ngIf="componenteAtual === 'exemplo01'"></app-exemplo01>
```

```
export class AppComponent {
  componenteAtual = 'exercicio01';

  exibirComponente(nome: string) {
    this.componenteAtual = nome;
  }
}
```

Exercícios de **ngIf*

1. Crie uma caixa de texto para entrada de números. Quando um número positivo for digitado, exiba a frase “número positivo” abaixo da caixa de texto

Exercícios de **ngIf*

2. Modifique o exercício anterior para que sejam exibidas as frases “número positivo”, “número negativo” ou “zero” abaixo da caixa de texto, de acordo com as entradas do usuário

Exercícios de **ngIf*

3. Crie um select que permita selecionar o tipo de contato preferencial do usuário (Whatsapp ou email). Cada tipo de contato deverá exibir um campo de texto específico, que permita ao usuário digitar seu valor adequadamente

Exercícios de **ngIf*

4. Crie uma calculadora, utilizando dois campos de texto e 4 botões (+, -, * e /). Quando clicados, os botões deverão realizar as operações matemáticas adequadas e exibir o resultado em texto, na parte inferior. Apenas exiba os botões e o resultado se houver números válidos nos campos de texto

*ngFor

- *ngFor é uma diretiva de repetição
- Corresponde a uma maneira de apresentar visualmente uma lista ou sequência de itens através da repetição de elementos HTML com estes valores

**ngFor*

- Definindo um bloco HTML que especifica como um único item deve ser exibido, basta instruir o Angular a usar esse bloco como um modelo para renderizar cada item da lista
- O texto atribuído ao **ngFor* é a instrução que guia o processo do repetidor

*ngFor

- Normalmente, o valor do atributo *ngFor será “let item of array”
- Nesse caso, cada item HTML repetido corresponderá ao item de cada posição do array

*ngFor

```
<ul>
      <li *ngFor="let item of array">
            {{item}}
      </li>
</ul>
```

*ngFor (caso precise do índice)

```
<ul>
        <li *ngFor="let item of array; let i = index;">
                {{i}} {{item}}
        </li>
</ul>
```

Exemplo 03

```
<ul >
    <li*ngFor="let animal of arrayBichos; let i = index;">{{i}} {{animal}}</li>
</ul>
```

```
export class Exemplo03Component implements OnInit {

  arrayBichos = ['Cachorro', 'Gato', 'Tartaruga', 'Pássaro'];

  constructor() { }

  ngOnInit(): void {
  }

}
```

Exemplo 04

```
<table border="1" >
    <tr *ngFor="let filme of arrayFilmes; let i = index;">
        <td>{{filme}}</td>
        <td>{{arrayNotas[i]}}</td>
    </tr>
</table>
```

```
export class Exemplo04Component implements OnInit {

  arrayFilmes = ['Parasita', 'Ex_Machina', 'Divertida Mente', 'Forrest Gump'];
  arrayNotas = [10, 9, 9, 8];

  constructor() { }

  ngOnInit(): void {
  }

}
```

Exercícios de **ngFor*

5. Itere por um vetor contendo um array de nomes de imagens, exibindo cada imagem de forma sequencial. Quando clicada, uma imagem deve exibir seu nome num alerta

Exercícios de **ngFor*

6. Crie um array contendo 4 países do mundo. Mostre o array em formato de lista na página. Quando um país for clicado, mostre um alert contendo sua capital

Exercícios de **ngFor*

7. Crie o template de uma div contendo o título do post e seu conteúdo. Faça um array de títulos e um array de posts contendo 3 elementos cada. Faça com que os posts e títulos sejam exibidos na tela

Exercícios de **ngFor*

8. Crie um array de nomes vazio, junto a uma caixa de texto e um botão. Ao clicar no botão, o nome deve ser adicionado no array de nomes e aparecer numa lista de nomes ordenada (ol)


"That's all Folks!"