Diretiva com Múltiplos Seletores no Angular

No Angular, podemos definir **múltiplos seletores** para que a mesma diretiva funcione em diferentes elementos e atributos. Isso torna a diretiva mais flexível e reutilizável.

Como Definir Múltiplos Seletores?

Podemos fazer isso passando uma lista de seletores no `@Directive({ selector: '...' })` separados por vírgula.

Exemplo 1: Aplicando a mesma diretiva a diferentes elementos e atributos

Aqui, criaremos uma diretiva que adiciona um efeito de **zoom** quando o usuário passa o mouse sobre um elemento. Ela pode ser aplicada tanto a `<img>`, `<div>` quanto a qualquer elemento que tenha o atributo `appHoverZoom`.

Código da Diretiva

```
import { Directive, ElementRef, HostListener, Renderer2 } from
'@angular/core';

@Directive({
  // A diretiva pode ser aplicada como:
  // 1. Um atributo em qualquer elemento: [appHoverZoom]
  // 2. Um elemento personalizado: app-Avatar-Zoom
  // 3. Aplicada diretamente a imagens: img
  selector: '[appHoverZoom], app-Avatar-Zoom, img'
})
export class HoverZoomDirective {
  constructor(private el: ElementRef, private renderer: Renderer2) {
    // Adiciona uma transição suave para animação
    this.renderer.setStyle(this.el.nativeElement, 'transition', 'transform 0.3s
ease-in-out');
  }
```

```
  @HostListener('mouseenter') onMouseEnter() {
    // Aumenta o tamanho do elemento ao passar o mouse
    this.renderer.setStyle(this.el.nativeElement, 'transform', 'scale(1.1)');
  }

  @HostListener('mouseleave') onMouseLeave() {
    // Retorna ao tamanho normal quando o mouse sai
    this.renderer.setStyle(this.el.nativeElement, 'transform', 'scale(1)');
  }
}
```

Como Usar a Diretiva?

Agora podemos aplicar a diretiva de **três maneiras diferentes** no HTML:

1️⃣ Como um atributo em qualquer elemento

```
<img src="assets/imagens/unicornio.jpg" appHoverZoom>
<div appHoverZoom> Passe o mouse aqui! </div>
```

2️⃣ Como um elemento personalizado

```
<app-Avatar-Zoom>
  <img src="assets/imagens/unicornio.jpg">
</app-Avatar-Zoom>
```

3️⃣ Aplicada automaticamente a todas as imagens (`img`)

```
<img src="assets/imagens/gato.jpg">
<img src="assets/imagens/cachorro.jpg">
```

✅ **Todas as imagens da página receberão o efeito de zoom automaticamente!**

Exemplo 2: Mudando a Cor de Fundo e o Borda em Diferentes Elementos

Agora, vamos criar uma diretiva que altera a **cor de fundo** e a **borda** do elemento.
Podemos aplicá-la a **botões**, **divs** e **parágrafos** (p).

Código da Diretiva

```
import { Directive, ElementRef, HostListener, Renderer2 } from '@angular/core';

@Directive({
  // Pode ser aplicada a botões, divs e parágrafos
  selector: '[appHighlight], button, div, p'
})
export class HighlightDirective {
  constructor(private el: ElementRef, private renderer: Renderer2) {
    this.renderer.setStyle(this.el.nativeElement, 'transition', 'all 0.3s ease-in-out');
  }

  @HostListener('mouseenter') onMouseEnter() {
    // Muda a cor de fundo e a borda ao passar o mouse
    this.renderer.setStyle(this.el.nativeElement, 'backgroundColor', '#FFD700'); // Amarelo ouro
    this.renderer.setStyle(this.el.nativeElement, 'border', '2px solid #FF8C00'); // Laranja escuro
  }

  @HostListener('mouseleave') onMouseLeave() {
    // Retorna ao normal quando o mouse sai
    this.renderer.setStyle(this.el.nativeElement, 'backgroundColor', 'transparent');
    this.renderer.setStyle(this.el.nativeElement, 'border', 'none');
  }
}
```

Como Usar a Diretiva?

Podemos aplicá-la de diferentes formas no HTML:

1️⃣ Como um atributo

<button appHighlight> Passe o mouse aqui </button>
<div appHighlight> Destaque essa caixa
<p appHighlight> Esse parágrafo terá um destaque ao passar o mouse

2️⃣ **Aplicada automaticamente a botões, divs e parágrafos**
<button> Clique Aqui </button>
 Destaque automático
 Este parágrafo também terá um efeito

✅ **Todos os botões (`<button>`), `<div>` e `<p>` terão efeito automaticamente!**

O que mais podemos fazer com múltiplos seletores?

Além dos exemplos acima, podemos:

✔️ **Criar diretivas que funcionam em vários tipos de elementos**
✔️ **Aplicar diferentes efeitos com base no tipo de elemento**
✔️ **Modificar estilos, adicionar classes ou manipular propriedades do DOM**

Exemplo 3: Diferentes Efeitos para Diferentes Elementos

Podemos verificar **qual tipo de elemento** está sendo modificado e aplicar diferentes efeitos.
Aqui, por exemplo, **botões** ficam maiores, **imagens** ganham sombra e **divs** mudam de cor.

```
import { Directive, ElementRef, HostListener, Renderer2 } from '@angular/core';

@Directive({
  selector: '[appMultiEffect], button, img, div'
})
export class MultiEffectDirective {
  constructor(private el: ElementRef, private renderer: Renderer2) {
```

```
    this.renderer.setStyle(this.el.nativeElement, 'transition', 'all 0.3s
ease-in-out');
  }

  @HostListener('mouseenter') onMouseEnter() {
    if (this.el.nativeElement.tagName === 'BUTTON') {
      // Botões aumentam de tamanho
      this.renderer.setStyle(this.el.nativeElement, 'transform', 'scale(1.2)');
    } else if (this.el.nativeElement.tagName === 'IMG') {
      // Imagens ganham uma sombra
      this.renderer.setStyle(this.el.nativeElement, 'boxShadow', '5px 5px 10px
rgba(0,0,0,0.5)');
    } else if (this.el.nativeElement.tagName === 'DIV') {
      // Divs mudam de cor
      this.renderer.setStyle(this.el.nativeElement, 'backgroundColor',
'#ADD8E6'); // Azul claro
    }
  }

  @HostListener('mouseleave') onMouseLeave() {
    this.renderer.setStyle(this.el.nativeElement, 'transform', 'none');
    this.renderer.setStyle(this.el.nativeElement, 'boxShadow', 'none');
    this.renderer.setStyle(this.el.nativeElement, 'backgroundColor',
'transparent');
  }
}
```

Usando no HTML:

```
<button appMultiEffect> Botão com Efeito </button>
<img src="assets/imagens/unicornio.jpg" appMultiEffect>
<div appMultiEffect> Caixa com Efeito </div>
```

✅ **Cada elemento receberá um efeito diferente ao passar o mouse!**

Conclusão

Com **múltiplos seletores**, podemos criar **diretivas poderosas e reutilizáveis**, aplicando efeitos a **diferentes elementos** sem precisar criar várias diretivas separadas.

O que aprendemos?

- Como definir múltiplos seletores na diretiva
- Como aplicar a diretiva a diferentes elementos
- Como criar efeitos diferentes dependendo do tipo de elemento