🎨 Explorando o Parâmetro do `@HostBinding` - Adicionando Estilos

O `@HostBinding` permite manipular diretamente os estilos de um elemento dentro de uma diretiva. Isso é útil quando queremos adicionar ou modificar estilos dinamicamente sem precisar recorrer ao **ngStyle** ou classes CSS.

📌1️⃣ Alterando Estilos Diretamente

Podemos usar `@HostBinding('style')` para definir estilos diretamente em um elemento.

Exemplo: Adicionando um fundo vermelho

```
import { Directive, HostBinding } from '@angular/core';

@Directive({
  selector: '[appRedBackground]'
})
export class RedBackgroundDirective {
  @HostBinding('style.backgroundColor') bgColor = 'red';
}
```

🔹 Esse código faz com que **qualquer elemento** com essa diretiva tenha um fundo vermelho.

🔹 **Uso no HTML**:

```
<p appRedBackground>Este parágrafo tem fundo vermelho!</p>
```

📌2️⃣ Adicionando Múltiplos Estilos

Podemos definir **vários estilos ao mesmo tempo** usando um objeto.

Exemplo: Adicionando múltiplos estilos

```
import { Directive, HostBinding } from '@angular/core';

@Directive({
  selector: '[appStyleMultiple]'
})
export class StyleMultipleDirective {
  @HostBinding('style') elementStyles = {
    backgroundColor: 'yellow',
    color: 'blue',
    padding: '10px',
    borderRadius: '5px'
  };
}
```

- Esse código adicionará **vários estilos ao mesmo tempo** no elemento.
- **Uso no HTML**:

```
<p appStyleMultiple>Esse parágrafo tem múltiplos estilos aplicados!</p>
```

📌3️⃣ Alterando Estilos Condicionalmente

Podemos modificar estilos dinamicamente com base em uma condição.

Exemplo: Mudando a cor do texto com base em um booleano

```
import { Directive, HostBinding } from '@angular/core';

@Directive({
  selector: '[appConditionalStyle]'
})
export class ConditionalStyleDirective {
  private isHighlighted = true; // Altere para false para testar

  @HostBinding('style.color') get textColor() {
    return this.isHighlighted ? 'green' : 'black';
  }
```

}

- Se `isHighlighted` for `true`, o texto será **verde**.
- Se for `false`, o texto será **preto**.

- **Uso no HTML**:

<p appConditionalStyle>Esse texto pode ser verde ou preto, dependendo da condição.</p>

📌4️⃣ Alterando Estilos com Eventos

Podemos usar `@HostListener` para mudar estilos quando o usuário interage com o elemento.

Exemplo: Mudar a cor de fundo ao passar o mouse

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

@Directive({
  selector: '[appHoverStyle]'
})
export class HoverStyleDirective {
  @HostBinding('style.backgroundColor') bgColor = 'transparent';

  @HostListener('mouseover') onMouseOver() {
    this.bgColor = 'lightblue';
  }

  @HostListener('mouseout') onMouseOut() {
    this.bgColor = 'transparent';
  }
}
```

- Quando o mouse passar sobre o elemento, o fundo ficará **azul-claro**.
- Quando o mouse sair, o fundo voltará ao **transparente**.

- **Uso no HTML**:

<p appHoverStyle>Passe o mouse sobre mim!</p>

📌5️⃣ Alterando Estilos Com Base em Propriedades da Diretiva

Podemos permitir que o usuário defina um valor personalizado dentro do HTML.

Exemplo: Alterando a cor da borda dinamicamente

```
import { Directive, HostBinding, Input } from '@angular/core';

@Directive({
  selector: '[appBorderStyle]'
})
export class BorderStyleDirective {
  @Input() borderColor: string = 'black';

  @HostBinding('style.border') get borderStyle() {
    return `2px solid ${this.borderColor}`;
  }
}
```

- Aqui, o usuário pode passar um **valor de cor** no HTML.
- **Uso no HTML**:

```
<p appBorderStyle borderColor="purple">Esse texto tem borda roxa!</p>
<p appBorderStyle borderColor="red">Esse texto tem borda vermelha!</p>
```

📌6️⃣ Alterando Propriedades Específicas

Podemos definir estilos com **unidades específicas**, como `px`, `em` ou `%`.

Exemplo: Alterando o tamanho da fonte dinamicamente

```
import { Directive, HostBinding } from '@angular/core';
```

```
@Directive({
  selector: '[appFontSize]'
})
export class FontSizeDirective {
  @HostBinding('style.fontSize.px') fontSize = 20;
}
```

- 🔹 Aqui, definimos o **tamanho da fonte em pixels**.
- 🔹 **Uso no HTML**:

```
<p appFontSize>Esse texto tem fonte de 20px!</p>
```

🚀 📌 Conclusão

- 🔹 `@HostBinding('style')` → Define múltiplos estilos ao mesmo tempo.
- 🔹 `@HostBinding('style.nomeDoEstilo')` → Permite definir um estilo específico.
- 🔹 Podemos usar **valores dinâmicos** para alterar os estilos de acordo com eventos ou condições.
- 🔹 Podemos usar **parâmetros no HTML** para personalizar os estilos de cada elemento.

Agora você pode **controlar estilos dinamicamente** no Angular com `@HostBinding`! 🚀🔥