🔍 Explorando o Parâmetro do `@HostBinding` - Adicionando Classes

O `@HostBinding` pode ser usado não apenas para manipular atributos e estilos, mas também para **adicionar ou remover classes CSS dinamicamente** em um componente ou diretiva do Angular.

📌1️⃣ Adicionando Classes com `@HostBinding('class')`

Podemos vincular uma classe diretamente ao elemento usando `@HostBinding('class')`.

Exemplo Básico

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

@Directive({
  selector: '[appClassBinding]'
})
export class ClassBindingDirective {
  @HostBinding('class') elementClass = 'default-class';
}
```

🔹 Essa diretiva adicionará a classe `'default-class'` a qualquer elemento que a utilize.
🔹 Exemplo de uso no HTML:

```
<p appClassBinding>Este parágrafo terá a classe 'default-class'</p>
```

📌2️⃣ Adicionando e Removendo Classes Condicionalmente

Podemos usar uma propriedade booleana para ativar ou desativar uma classe.

Exemplo: Alternar uma classe com base em uma condição

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

@Directive({
  selector: '[appWarning]'
})
export class WarningDirective {
  private isWarning = true; // Alterne para true/false para testar

  @HostBinding('class.warning') get warningClass() {
    return this.isWarning;
  }
}
```

- Se `isWarning` for `true`, a classe `"warning"` será adicionada ao elemento.
- Se for `false`, a classe será removida.
- Exemplo de uso no HTML:

```
<p appWarning>Este parágrafo terá a classe 'warning' se a condição for verdadeira.</p>
```

- CSS correspondente:

```
.warning {
  color: red;
  font-weight: bold;
}
```

📌3️⃣Vinculando Múltiplas Classes ao Mesmo Elemento

Podemos usar `@HostBinding('class')` para definir várias classes ao mesmo tempo.

Exemplo: Adicionando várias classes dinamicamente

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

@Directive({
  selector: '[appMultipleClasses]'
})
export class MultipleClassesDirective {
  @HostBinding('class') elementClasses = 'class1 class2 class3';
}
```

- Esse código adicionará **três classes** ao elemento ao mesmo tempo.
- Exemplo de uso no HTML:

```
<div appMultipleClasses>Este div terá as classes 'class1 class2 class3'</div>
```

- CSS correspondente:

```
.class1 { background-color: yellow; }
.class2 { border: 2px solid black; }
.class3 { font-size: 20px; }
```

📌4️⃣ Alternando Classes ao Passar o Mouse (`mouseover` e `mouseout`)

Podemos usar `@HostListener` para alternar classes dinamicamente com eventos.

Exemplo: Mudar a classe ao passar o mouse

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

@Directive({
  selector: '[appHoverEffect]'
```

```
})
export class HoverEffectDirective {
  @HostBinding('class.hovered') isHovered = false;

  @HostListener('mouseover') onMouseOver() {
    this.isHovered = true;
  }

  @HostListener('mouseout') onMouseOut() {
    this.isHovered = false;
  }
}
```

🔹 Isso adicionará a classe `"hovered"` ao elemento quando o mouse estiver sobre ele.
🔹 E removerá a classe quando o mouse sair.

🔹 Exemplo de uso no HTML:

```
<p appHoverEffect>Passe o mouse sobre mim!</p>
```

🔹 CSS correspondente:

```
.hovered {
  background-color: lightblue;
  transition: 0.3s;
}
```

📌5️⃣ Adicionando Classes com Base em uma Lista Dinâmica

Podemos definir um **array de classes** e alternar dinamicamente.

Exemplo: Classes dinâmicas com array

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

@Directive({
```

```
  selector: '[appDynamicClasses]'
})
export class DynamicClassesDirective {
  @HostBinding('class') elementClasses = '';

  constructor() {
    this.elementClasses = ['class1', 'class2'].join(' '); // Adiciona class1 e class2
  }
}
```

🔹 Aqui, transformamos um **array de classes** em uma string usando `.join(' ')`.

🔹 Exemplo de uso no HTML:

```
<div appDynamicClasses>Esse div tem classes dinâmicas</div>
```

🔹 CSS correspondente:

```
.class1 { color: green; }
.class2 { font-weight: bold; }
```

🚀 📌 Conclusão

🔹 `@HostBinding('class')` → Define uma ou várias classes no elemento.
🔹 `@HostBinding('class.nomeClasse')` → Ativa/desativa uma classe condicionalmente.
🔹 `@HostListener('mouseover')` → Pode ser combinado para adicionar/remover classes com eventos.
🔹 Podemos usar **arrays** para definir classes dinamicamente.

Com esses exemplos, agora você pode manipular **qualquer classe CSS dinamicamente** em seus componentes e diretivas no Angular! 🎯🔥