O `@HostListener` é um decorador do Angular que permite capturar eventos do DOM dentro de uma diretiva. Ele pode receber diferentes **parâmetros** para fornecer mais controle sobre os eventos.

🔥 Exemplos e Exploração dos Parâmetros do @HostListener

🟢1️⃣ Capturando Eventos do Mouse (`click`, `mouseover`, `mouseout`)

Podemos capturar eventos do mouse para mudar o estilo de um botão quando o usuário interagir com ele.

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

@Directive({
  selector: '[appMouseEvents]'
})
export class MouseEventsDirective {

  @HostBinding('style.backgroundColor') bgColor: string = 'transparent';

  // Evento de clique
  @HostListener('click') onClick() {
    console.log('Botão clicado!');
    this.bgColor = 'blue';
  }

  // Mouse sobre o botão
  @HostListener('mouseover') onMouseOver() {
    console.log('Mouse entrou');
    this.bgColor = 'green';
  }

  // Mouse saiu do botão
  @HostListener('mouseout') onMouseOut() {
    console.log('Mouse saiu');
```

```
    this.bgColor = 'transparent';
  }
}
```

🟢 **Explicação:**

- `onClick()`: Muda a cor para **azul** quando o botão é clicado.
- `onMouseOver()`: Muda para **verde** quando o mouse passa sobre o botão.
- `onMouseOut()`: Retorna à cor **original** quando o mouse sai.

✅ Como Usar no HTML

```
<button appMouseEvents>Interaja Comigo</button>
```

🟢2️⃣ Capturando Eventos de Teclado (`keydown`, `keyup`)

Podemos capturar eventos do teclado para exibir o que o usuário está digitando.

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

@Directive({
  selector: '[appKeyEvents]'
})
export class KeyEventsDirective {

  // Captura quando uma tecla é pressionada
  @HostListener('keydown', ['$event']) onKeyDown(event: KeyboardEvent) {
    console.log(`Tecla pressionada: ${event.key}`);
  }

  // Captura quando uma tecla é solta
  @HostListener('keyup', ['$event']) onKeyUp(event: KeyboardEvent) {
    console.log(`Tecla liberada: ${event.key}`);
  }
}
```

🟢 **Explicação:**

- `onKeyDown(event: KeyboardEvent)`: Mostra no console qual tecla foi pressionada.
- `onKeyUp(event: KeyboardEvent)`: Mostra no console qual tecla foi solta.

✅ Como Usar no HTML

<input appKeyEvents type="text" placeholder="Digite algo...">

🟢3️⃣ Capturando Coordenadas do Mouse (`mousemove`)

Podemos capturar a posição exata do mouse na tela.

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

@Directive({
  selector: '[appMousePosition]'
})
export class MousePositionDirective {

  @HostListener('mousemove', ['$event']) onMouseMove(event: MouseEvent)
{
    console.log(`Mouse em X: ${event.clientX}, Y: ${event.clientY}`);
  }
}
```

🟢 **Explicação:**

- `event.clientX`: Posição horizontal do mouse na tela.
- `event.clientY`: Posição vertical do mouse na tela.

✅ Como Usar no HTML

```
<div appMousePosition style="height: 300px; background-color: lightgray;">
  Passe o mouse aqui!
</div>
```

🟢4️⃣ Capturando Eventos de Scroll (`window:scroll`)

Podemos capturar quando o usuário rola a página.

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

@Directive({
  selector: '[appScrollListener]'
})
export class ScrollListenerDirective {

  @HostListener('window:scroll', ['$event']) onScroll(event: Event) {
    console.log('Rolando a página!', window.scrollY);
  }
}
```

🟢 Explicação:

- `window:scroll`: Captura o evento de rolagem da página.
- `window.scrollY`: Retorna a posição vertical da rolagem.

✅ Como Usar no HTML

```
<div appScrollListener style="height: 2000px;">
  Role para testar!
</div>
```

🟢5️⃣ Desativando um Botão Temporariamente (`dblclick`)

Podemos desativar um botão temporariamente quando o usuário clicar duas vezes nele.

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

@Directive({
  selector: '[appDisableButton]'
})
export class DisableButtonDirective {
```

```
  @HostBinding('disabled') isDisabled = false;

  @HostListener('dblclick') onDoubleClick() {
    console.log('Botão desativado por 3 segundos!');
    this.isDisabled = true;

    setTimeout(() => {
      this.isDisabled = false;
      console.log('Botão reativado!');
    }, 3000);
  }
}
```

🟢 **Explicação:**

- `isDisabled`: Define se o botão está **desativado**.
- `onDoubleClick()`: Desativa o botão por **3 segundos** quando o usuário **clica duas vezes**.

✅ Como Usar no HTML

<button appDisableButton>Clique Duas Vezes</button>

🚀 Conclusão

O `@HostListener` é uma ferramenta poderosa no Angular que permite capturar eventos diretamente dentro de diretivas personalizadas. Ele pode ser usado para:
✅ Capturar eventos do **mouse** (click, mouseover, mouseout, mousemove).
✅ Capturar eventos do **teclado** (keydown, keyup).
✅ Monitorar eventos do **scroll** na página.
✅ Controlar **interações do usuário**, como **desativar botões** temporariamente.

Agora você pode explorar ao máximo os eventos no Angular! 🚀🔥