O que é Event Binding?

Event Binding é uma técnica no Angular que permite que você capture eventos do DOM (Document Object Model) em elementos HTML e os conecte a métodos do componente. Esses eventos podem ser disparados por ações do usuário, como cliques, digitação, movimentos do mouse ou eventos personalizados.

O Angular usa a sintaxe `(evento)="métodoDoComponente()"` para vincular o evento ao método.

Como Funciona o Event Binding?

Quando o evento ocorre, o Angular:

1. Captura o evento do DOM.
2. Chama o método especificado no componente.
3. Permite que o método manipule os dados ou altere o estado da aplicação com base no evento.

Exemplos de Uso do Event Binding

1. Clique de Botão

```
<button (click)="onButtonClick()">Clique aqui</button>

onButtonClick() {
  console.log('Botão clicado!');
}
```

- Quando o botão é clicado, a função `onButtonClick` é executada, exibindo "Botão clicado!" no console.

---

2. Evento de Digitação (keyup)

```
<input type="text" (keyup)="onKeyup($event)">

onKeyup(event: KeyboardEvent) {
  const value = (event.target as HTMLInputElement).value;
  console.log(`Texto atual: ${value}`);
```

```
}
```

- Cada vez que o usuário soltar uma tecla, o texto atual do campo será exibido no console.

---

3. Evento de Movimento do Mouse

```
<div (mousemove)="onMouseMove($event)" style="height: 100px; background: lightgray;">
  Mova o mouse aqui
</div>

onMouseMove(event: MouseEvent) {
  console.log(`Posição do mouse: X=${event.clientX}, Y=${event.clientY}`);
}
```

- Sempre que o mouse for movido sobre o `div`, a posição do ponteiro será exibida no console.

---

4. Alterar um Valor com Evento de Alteração (change)

```
<select (change)="onSelectChange($event)">
  <option value="opcao1">Opção 1</option>
  <option value="opcao2">Opção 2</option>
</select>

onSelectChange(event: Event) {
  const selectedValue = (event.target as HTMLSelectElement).value;
  console.log(`Opção selecionada: ${selectedValue}`);
}
```

- Exibe a opção selecionada pelo usuário no console.

---

5. Impedindo Comportamentos Padrão

```
<a href="https://example.com" (click)="preventDefault($event)">Clique aqui</a>

preventDefault(event: Event) {
  event.preventDefault();
```

```
  console.log('Comportamento padrão prevenido!');
}
```

- Quando o link é clicado, o navegador não navega para `https://example.com`.

---

6. Evento Personalizado com `@Output`

Em Angular, você pode criar seus próprios eventos personalizados em componentes usando `@Output`.

Componente Filho

```
import { Component, EventEmitter, Output } from '@angular/core';

@Component({
  selector: 'app-child',
  template: `<button (click)="notifyParent()">Notificar Pai</button>`,
})
export class ChildComponent {
  @Output() notify = new EventEmitter<string>();

  notifyParent() {
    this.notify.emit('Evento disparado pelo componente filho!');
  }
}
```

Componente Pai

```
<app-child (notify)="onChildNotify($event)"></app-child>

onChildNotify(message: string) {
  console.log(message); // Exibe 'Evento disparado pelo componente filho!'
}
```

Eventos Suportados

O Angular suporta uma ampla gama de eventos do DOM, incluindo:

- **Teclado**: `(keyup)`, `(keydown)`, `(keypress)`

- **Mouse**: `(click)`, `(dblclick)`, `(mousedown)`, `(mousemove)`, `(mouseup)`
- **Formulários**: `(input)`, `(change)`, `(submit)`
- **Eventos de Foco**: `(focus)`, `(blur)`
- **Eventos de Tela**: `(resize)`, `(scroll)`
- **Outros**: `(drag)`, `(drop)`, `(contextmenu)`

Combinação com Property Binding

Event Binding é frequentemente usado em conjunto com Property Binding para criar interatividade dinâmica. Por exemplo:

```
<input [value]="inputValue" (input)="onInputChange($event)">
<p>Texto atual: {{ inputValue }}</p>

inputValue = '';

onInputChange(event: Event) {
  this.inputValue = (event.target as HTMLInputElement).value;
}
```

- O campo de entrada e o parágrafo exibem dinamicamente o texto inserido.

Resumo do que Pode Ser Feito

1. Reagir a ações do usuário, como cliques, digitação e movimentos do mouse.
2. Capturar dados de eventos e usá-los no componente.
3. Prevenir comportamentos padrão ou propagações de eventos.
4. Criar eventos personalizados para comunicação entre componentes.
5. Criar interatividade em formulários, navegadores ou elementos da interface.

O **Event Binding** é uma das ferramentas mais importantes do Angular para criar aplicações interativas e responsivas.