Como Criar Componentes Filhos no Angular 19

No Angular, um **componente filho** é um componente que faz parte de um componente principal (ou pai). Ele é útil para organizar o código e dividir a interface em partes reutilizáveis. Componentes filhos podem receber dados do componente pai e também enviar eventos de volta para ele.

Passos para Criar Componentes Filhos

1. Gerar o Componente Filho

Você pode criar um componente filho manualmente ou usando o CLI do Angular.

Com o Angular CLI:

Execute o comando:

- ng generate component nome-do-componente-filho

Exemplo:

- ng generate component filho

Isso criará:

- Um arquivo `.ts` para a lógica do componente (`filho.component.ts`).
- Um arquivo `.html` para o template do componente (`filho.component.html`).
- Um arquivo `.scss` (ou `.css`) para estilos (`filho.component.scss`).
- Um arquivo de teste (`filho.component.spec.ts`).

2. Configurar o Componente Filho

No arquivo `filho.component.ts`, configure a lógica do componente filho:

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

@Component({
  selector: 'app-filho', // Nome do componente usado no template do pai
  templateUrl: './filho.component.html',
  styleUrls: ['./filho.component.scss']
})
export class FilhoComponent {
  // Recebe dados do componente pai
  @Input() mensagemDoPai!: string;
```

```
  // Emite eventos para o componente pai
  @Output() respostaParaOPai = new EventEmitter<string>();

  // Método para enviar resposta ao pai
  enviarResposta() {
    this.respostaParaOPai.emit('Olá, Pai! Esta é a resposta do filho.');
  }
}
```

3. Usar o Componente Filho no Componente Pai

No **componente pai**, importe o filho e use seu `selector` no arquivo `.html` do pai.

1. Configurar o componente pai para enviar dados ao filho:

No arquivo `app.component.ts` (pai):

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.scss']
})
export class AppComponent {
  mensagemParaFilho = 'Olá, Filho!';

  // Método para receber dados do filho
  receberResposta(resposta: string) {
    console.log('Resposta do Filho:', resposta);
  }
}
```

2. Usar o filho no template do pai:

No arquivo `app.component.html`:

```
<!-- Passa dados para o filho usando property binding -->
<app-filho [mensagemDoPai]="mensagemParaFilho"
           (respostaParaOPai)="receberResposta($event)">
</app-filho>
```

4. Template do Componente Filho

No arquivo `filho.component.html`, exiba os dados recebidos e envie a resposta ao pai:

```
<p>{{ mensagemDoPai }}</p>
<button (click)="enviarResposta()">Responder ao Pai</button>
```

Explicação dos Decorators

1. **`@Input`**:
   Permite que o componente filho receba dados do componente pai.
   Exemplo: `[mensagemDoPai]="mensagemParaFilho"`
2. **`@Output`**:
   Permite que o componente filho envie eventos/dados para o componente pai.
   Exemplo: `(respostaParaOPai)="receberResposta($event)"`
3. **`EventEmitter`**:
   É usado com `@Output` para emitir eventos do filho para o pai.

Resultado

1. O componente pai envia a mensagem **"Olá, Filho!"** para o filho usando o decorator `@Input`.
2. O componente filho exibe a mensagem recebida e, ao clicar no botão, envia a resposta **"Olá, Pai! Esta é a resposta do filho."** para o pai usando o decorator `@Output`.

Quando Usar Componentes Filhos

- Para modularizar e reaproveitar partes da interface.
- Quando há necessidade de comunicação entre componentes, seja enviando dados para o filho ou recebendo eventos do filho.
- Em layouts complexos onde a separação de responsabilidades facilita a manutenção.