Property Binding é uma funcionalidade do Angular que permite vincular propriedades de elementos ou componentes HTML a expressões ou variáveis definidas no componente TypeScript. Essa técnica é usada para atualizar dinamicamente as propriedades do DOM com base nos dados do componente, garantindo uma sincronização entre a interface do usuário (UI) e a lógica de negócios.

Funcionamento

O **Property Binding** é representado pelo uso de colchetes (`[ ]`) em torno do nome da propriedade do elemento HTML, como mostrado abaixo:

<input [value]="nome">

Neste exemplo:

- O valor da propriedade `value` do campo `<input>` será definido pelo valor da variável `nome` no componente TypeScript.
- Qualquer alteração no valor de `nome` será refletida automaticamente na propriedade `value`.

Principais Aplicações

Atualizar Atributos HTML Exemplo: Alterar o texto exibido em um botão.

<button [disabled]="isDisabled">Clique aqui</button>

1.
    - `disabled` é uma propriedade do botão.
    - `isDisabled` é uma variável no componente TypeScript que controla se o botão está habilitado ou não.
2. **Vincular Propriedades Personalizadas**

    - Propriedades customizadas de componentes também podem ser vinculadas usando `@Input`.

Exemplo Completo de Property Binding

HTML

```
<h1 [innerText]="titulo"></h1>
<img [src]="imagemUrl" [alt]="descricaoImagem">
<button [disabled]="isBotaoDesabilitado">Enviar</button>
<input [value]="nomeUsuario" (input)="atualizarNome($event)">
<p>Seu nome é: {{ nomeUsuario }}</p>
```

TypeScript

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

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  titulo = 'Exemplo de Property Binding'; // Controla o texto do <h1>
  imagemUrl = 'https://via.placeholder.com/150'; // URL da imagem no <img>
  descricaoImagem = 'Imagem de exemplo'; // Texto alternativo (alt) da
imagem
  isBotaoDesabilitado = false; // Controla se o botão está habilitado ou não
  nomeUsuario = ''; // Armazena o nome do usuário

  // Atualiza o nome do usuário com base na entrada do campo
  atualizarNome(event: Event): void {
    const input = event.target as HTMLInputElement;
    this.nomeUsuario = input.value;
  }
}
```

Propriedades Comuns que Podem Ser Vinculadas

Campos de Entrada (`<input>`, `<textarea>`)

```
 <input [value]="valor" [disabled]="isDisabled"
[placeholder]="placeholderText">
```

Imagens

```
 <img [src]="imagemUrl" [alt]="descricaoImagem">
```

Botões

```
 <button [disabled]="isDisabled" [innerText]="botaoTexto"></button>
```

Estilos Inline

```
 <div [style.backgroundColor]="corFundo"
[style.fontSize.px]="tamanhoFonte">
    Texto estilizado dinamicamente
</div>
```

Classes CSS

```
 <div [class.alert]="isAlerta" [class.success]="isSucesso">
    Texto com classes dinâmicas
</div>
```

Diferença entre Atributos e Propriedades

- **Atributos** são valores declarados no HTML, estáticos.
- **Propriedades** são valores representados pelo DOM e podem ser manipulados dinamicamente pelo JavaScript.

Exemplo:

```
<!-- Este é um atributo HTML -->
<input value="texto">

<!-- Este é um binding para a propriedade 'value' -->
<input [value]="variavelTexto">
```

Recursos Avançados

Two-Way Binding com ngModel

```
 <input [(ngModel)]="nome">
<p>O nome é: {{ nome }}</p>
```

1. Aqui, as alterações no campo de entrada são automaticamente sincronizadas com a variável `nome`.

Usando Funções no Binding

```
 <button [disabled]="calcularEstadoDoBotao()">Enviar</button>
```

2. A propriedade `disabled` será definida pelo valor retornado da função `calcularEstadoDoBotao`.

Conclusão: O **Property Binding** permite criar interfaces dinâmicas e interativas no Angular, conectando os dados da aplicação com as propriedades dos elementos do DOM de forma eficiente e reativa. É uma prática fundamental em aplicações Angular modernas.