Attribute Binding no Angular é uma técnica usada para alterar ou definir os atributos de um elemento HTML dinamicamente, com base em dados do componente. Ele permite associar os valores das propriedades do componente diretamente aos atributos do DOM, mantendo a interface sincronizada com o estado da aplicação.

Diferença entre Atributos e Propriedades

- **Atributos**: São valores declarados diretamente no HTML (como `title`, `id`, `disabled`, etc.). Eles permanecem fixos no DOM e representam o estado inicial do elemento.
- **Propriedades**: São os valores internos dos objetos DOM que podem mudar durante a execução da aplicação.

Exemplo:

- `<input disabled>` é um atributo.
- O `disabled` de um elemento `input` no DOM é uma propriedade.

Com o **Attribute Binding**, você pode modificar diretamente os **atributos**, especialmente os que não têm uma correspondência direta com as propriedades do DOM.

Sintaxe do Attribute Binding

[attributeName]="expression"

Aqui:

- `attributeName` é o nome do atributo a ser alterado.
- `expression` é uma expressão Angular ou uma variável definida no componente.

Exemplo Simples

Componente HTML:

<button [attr.title]="buttonTitle">Clique Aqui</button>

Componente TypeScript:

```
export class AppComponent {
  buttonTitle = 'Este é o botão!';
}
```

Explicação: O atributo `title` será definido dinamicamente como `"Este é o botão!"`. Ao passar o mouse sobre o botão, o texto aparecerá como uma dica (tooltip).

Casos de Uso Comuns

1. **Alterar atributos HTML personalizados ou não comuns**:

Exemplo:
<button [attr.aria-label]="ariaLabel">Botão Acessível</button>
ariaLabel = 'Botão para navegação';

    ○

2. **Gerenciar atributos não diretamente suportados como propriedades**:

Exemplo:
<input [attr.maxlength]="maxInputLength">
maxInputLength = 10;

    ○

3. **Controle de acessibilidade (A11y)**:

Exemplo: Adicionar ou alterar atributos ARIA:
<div role="dialog" [attr.aria-hidden]="isDialogOpen ? 'false' : 'true'">
  Conteúdo do diálogo

isDialogOpen = false;

    -
4. **Adicionar atributos personalizados para bibliotecas ou frameworks de terceiros**:

Exemplo:
<div [attr.data-custom]="customData">Div Personalizada
customData = 'valor-dinamico';

    -

Exemplo Completo: Alternar Habilitação Dinâmica de Botões

Componente HTML:

```
<button [attr.disabled]="isDisabled ? true : null">Clique Aqui</button>
<br>
<button (click)="toggleButton()">Alterar Estado</button>
```

Componente TypeScript:

```
export class AppComponent {
  isDisabled = true;

  toggleButton() {
    this.isDisabled = !this.isDisabled;
  }
}
```

Explicação:

- O botão será habilitado ou desabilitado dinamicamente com base no valor da variável `isDisabled`.
- O método `toggleButton` alterna o estado entre habilitado e desabilitado.

Limitações do Attribute Binding

1. **Não substitui propriedades diretamente**: Para atributos que têm propriedades DOM correspondentes (como `value` ou `checked`), é recomendado usar **Property Binding**.
2. **Pode causar problemas em atributos booleanos**: Por exemplo, `disabled` funciona melhor como propriedade.

Dicas

- Use **Attribute Binding** para atributos sem uma propriedade correspondente no DOM.
- Para atributos com equivalência direta como propriedade, prefira **Property Binding** (`[disabled]`, `[value]`, etc.).
- Combine **Attribute Binding** com eventos para criar interfaces interativas e dinâmicas.

Conclusão

O **Attribute Binding** é poderoso para manipular atributos não padrão, melhorar a acessibilidade e integrar elementos personalizados em sua aplicação Angular. Ele ajuda a criar uma interface mais dinâmica e adaptável, mantendo o controle preciso dos atributos HTML.