15

Content Projection

- What is Content Projection?
- Implementations of Content Projection
    - Single slot
    - Multi-slot
    - Conditional
- Ng-container Vs Ng-Template

What is Content Projection?

- Content projection is a method of importing HTML content from outside a component and inserting it into the component's template in certain content.
- Content projection allows a directive to use templates while still being able to clone the original content and add it to the DOM, which is a programming interface for HTML and XML documents and represents a page where programs can change the document structure, style, and content.

Single-slot Content Projection

- The term single-slot content projection refers to the creation of a component into which only one component can be projected.

Example – Child Component

```
@Component({
  selector: 'app-demo',
  template: `
    <h2>Single-slot content projection</h2>
    <ng-content></ng-content>
  `
})
...
```

Parent Component

Users of this component may now project their own message into the component with the <ng-content> element in place. As an example:

```
<app-demo>
  <p>Projected content!</p>
</app-demo>
```

Here, Projected content! will be displayed instead of <ng-content></ng-content>

Multi-slot Content Projection

- A component may have several slots.
- Each slot can have its own CSS selector that specifies what content gets into it.
- Multi-slot content projection is the name given to this pattern.
- You must indicate where you want the projected content to appear using this pattern.
- This is accomplished by utilizing the select property of <ng-content>.

Example-

Child Component

```
@Component({
  selector: 'app-demo',
  template: `
    <h2>Multi-slot content projection</h2>

    Default:
    <ng-content></ng-content>

    Question:
    <ng-content select="[heading]"></ng-content>
  `
})
...
```

Parent –

Here, one of the <ng-content> contains select="[heading]. i.e. it will only project the data which contains the heading attribute.

```
<app-demo>
  <p heading>
    Content Projection in Angular
  </p>
  <p>Projected content!</p>
</app-demo>
```

Conditional Content Projection

- If your component has to conditionally render content or render content several times, you should set it up to accept a <ng-template> element containing the conditionally rendered content.
- It is not suggested to utilize a <ng-content> element in these circumstances because when a component's consumer delivers the content, the content is always initialized, even if the component does not specify a <ng-content> element or if that <ng-content> element is inside of a ngIf statement.
- You may have your component directly render content based on any condition you want, as many times as you like, using a <ng-template> element. Angular will not render a <ng-template> element's content unless it is expressly rendered.
  Example –

```
@Component({
  selector: 'app-component',
  template: `
   <div *ngIf="isLoggedIn; else loggedOut">
      Welcome back.
   </div>

    <ng-template #loggedOut>
      Please login.
    </ng-template>
  `,
})
export class AppComponent {
  isLoggedIn = true;
}
```

Here **<ng-template>** will render only when the condition failed.

Ng-Container Vs Ng-Template

Ng-Content	Ng-Template
• ng-container will serve as a container for elements, It will accept structural directives like ngFor, ngIf etc. But it will never get rendered to the DOM.	• ng-template allows us to create template content, this also will not render to the DOM until we specifically/conditionally add it to the DOM.

ALL THE BEST