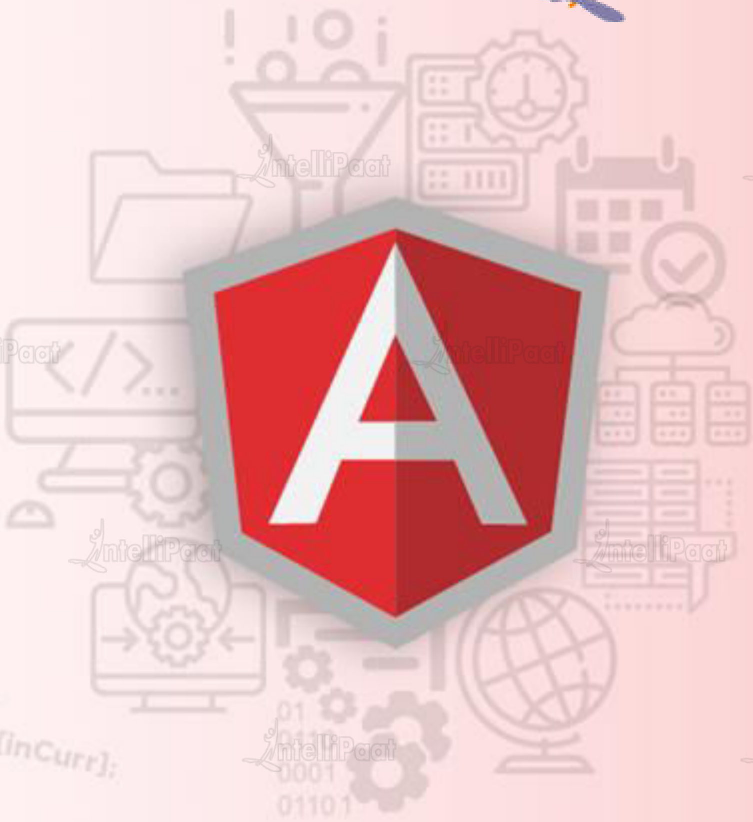
Directives



Agenda

01 What are Directives

02 Types of Directives

03 Component Directives

04 Structural Directives

05 Attribute Directives

06 Custom Directives

Directives

There are three kinds of directives in Angular:

•Components

- Component directives form the main class. It possesses the details about how the component should be instantiated, processed and utilized at runtime.

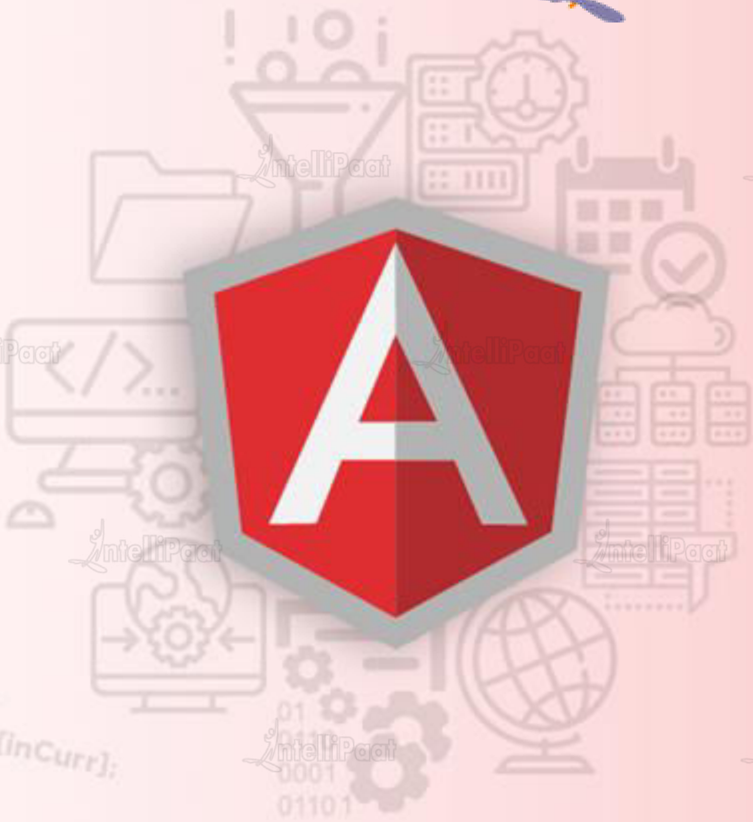
Structural

- change the DOM layout by adding and removing DOM elements.
- Ex: ngForOf ,ngIf, ngSwitch

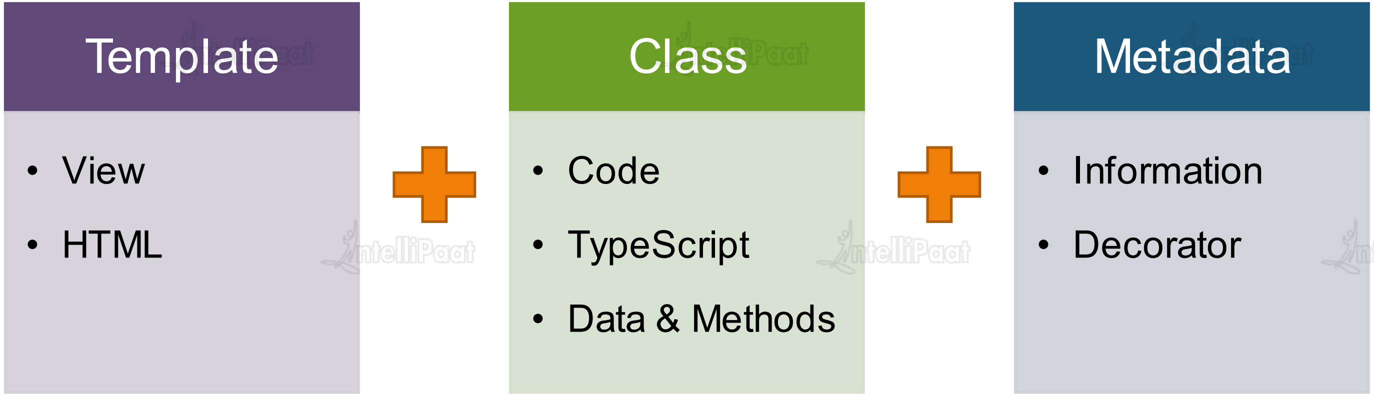
Attribute

- change the appearance or behavior of an element, component.
- Ex ngStyle, ngClass, ngModel.

Component Directives



Component



Template

- View
- HTML

Class

- Code
- TypeScript
- Data & Methods

Metadata

- Information
- Decorator

Structural Directives

Structural Directives

What are Structural Directives?

- They shape or reshape the DOM's *structure*, typically by adding, removing, or manipulating elements.
- Structural directives are easy to recognize. An asterisk (*) precedes the directive attribute name. like *ngIf, *ngForOf, *ngSwitch
- These directive belong to @angular/common/src/directives

Built-in Structural Directives

•ngIf

- conditionally add or remove an element from the DOM

ngSwitch

- a set of directives that switch among alternative views

ngForOf

- repeat a template for each item in a list

ngIf Syntax

- **Simple form:**

```
<div *ngIf="condition">...</div>
<ng-template [ngIf]="condition"><div>...</div></ng-template>
```

- **Form with else block:**

```
<div *ngIf="condition; else elseBlock">...</div>
<ng-template #elseBlock>...</ng-template>
```

- **Form with then and else block:**

```
<div *ngIf="condition; then thenBlock else elseBlock"></div>
<ng-template #thenBlock>...</ng-template>
 <ng-template #elseBlock>...</ng-template>
```

- **Form with storing the value locally:**

```
<div *ngIf="condition as value; else elseBlock">{{value}}</div>
<ng-template #elseBlock>...</ng-template>
```

ngSwitch Syntax

```
<div [ngSwitch]=“color">

<div *ngSwitchCase=“red”>You picked red</div>

<div *ngSwitchCase=“blue”>You picked blue</div>

<div *ngSwitchCase=“green”>You picked green</div>

<div * ngSwitchDefault>Try again</div>

</div>
```

ngFor Syntax

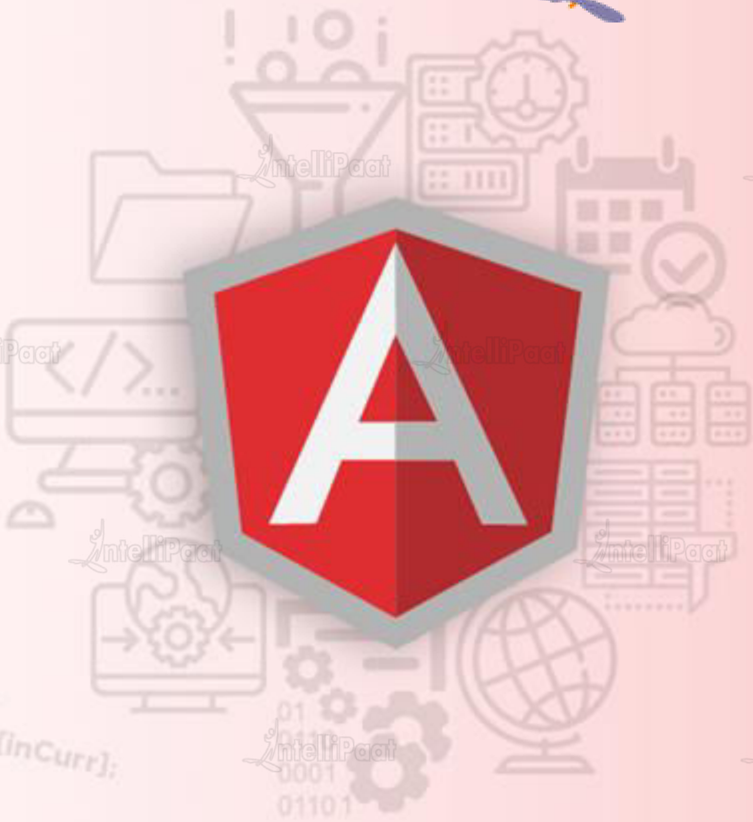
```
<li *ngFor="let user of users; index as i; first as isFirst">

{{i}}/{{users.length}} {{user}} {{isFirst}}

</li>
```

index: number: The index of the current item in the iterable.
first: boolean: True when the item is the first item in the iterable.
last: boolean: True when the item is the last item in the iterable.
even: boolean: True when the item has an even index in the iterable.
odd: boolean: True when the item has an odd index in the iterable.

Attribute Directives



Attribute Directives

- Attribute directives listen to and modify the behavior of other HTML elements, attributes, properties, and components.
- They are usually applied to elements as if they were HTML attributes, hence the name. like: ngClass, ngStyle and ngModel.

Built-in Attribute Directives

•ngClass

- add and remove a set of CSS classes dynamically for a element.

ngStyle

- add and remove a set of HTML styles dynamically for a element.

ngModel

- two-way data binding to an HTML form element

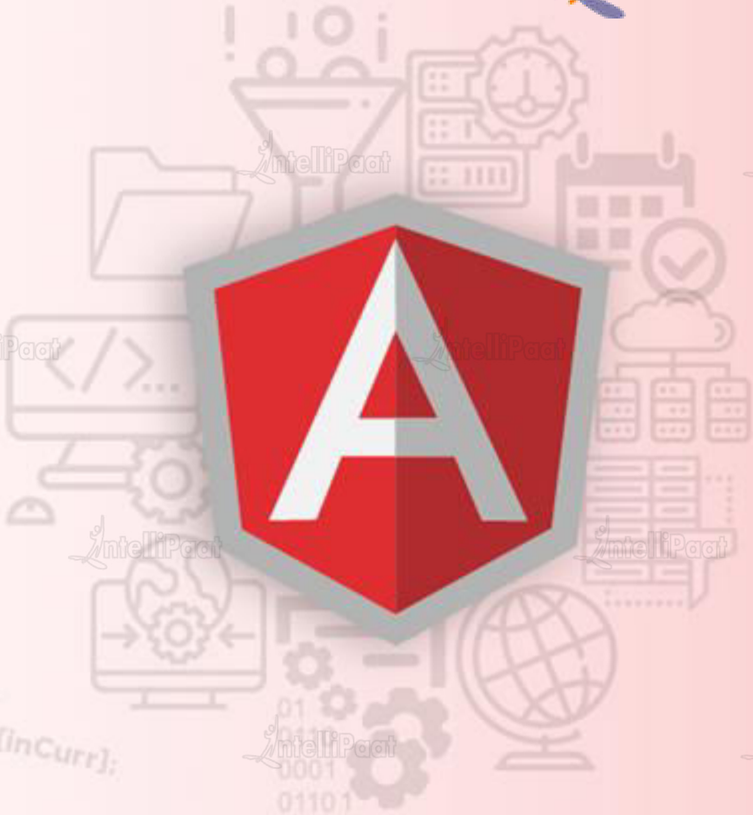
ngStyle Syntax

- <some-element [ngStyle]="{'font-style’: ‘italic’ , 'max-width.px': widthExp}">...</some-element>
- <some-element [ngStyle]="{‘font-size.px’: 24}">...</some-element>
- <some-element [ngStyle]="objExp">...</some-element>

ngClass Syntax

- <some-element [ngClass]="{‘text-success’: true}">...</some-element>
- <button [ngClass]="{‘btn btn-primary’}">...</some-element>
- <button [ngClass]="{[‘btn’, ‘btn-primary’]}">...</some-element>

Custom Directives



Custom Directives

- Create the directive class file in a terminal window with the CLI command

  ng generate directive appHighLight
- Apply in template :

  <p appHighLight></p>

```
import { Directive, ElementRef } from '@angular/core’;

 @Directive({ selector: '[appHighlight]'  })

export class HighlightDirective

{ constructor(el: ElementRef)

 { el.nativeElement.style.backgroundColor = 'yellow’; }

 }
```

Custom Directives using Listeners

```
import { Directive, ElementRef } from '@angular/core’;

 @Directive({ selector: '[appHighlight]'  })

export class HighlightDirective
{constructor(private e1:ElementRef) {
}
@HostListener('mouseenter') onMouseEnter() {
this.highlight('yellow');
}
@HostListener('mouseleave') onMouseLeave() {
this.highlight(null);
}
private highlight(color:  string) {
this.e1.nativeElement.style.backgroundColor = color;
} }
```

Demo

IntelliPaat

☎ India : +91-7847955955

US : 1-800-216-8930 (TOLL FREE)

✉ sales@intellipaat.com

24X7 Chat with our Course Advisor