@Directive

Siddharth Ajmera **@SiddAjmera**

Content:

- Directives
- Angular's Built-In Directives
- Custom Structural Directives
- Custom Attribute Directives

Directives

- Directives are a way of attaching behaviour to DOM elements
- Directives are decorated with the @Directive decorator
- Can be used in other components and directives
- Directives are registered in the Declarations array of an NgModule
- Directives are configured through the metadata passed to the @Directive decorator
- Directives can implement Lifecycle hooks to control their runtime behaviour

Types of Directives

- Components
  - Directives a template
  - Most common directive used throughout the app
- Structural Directives
  - Directives that change the DOM layout by adding/removing elements
  - Prefixed with an asterisk. Eg. *ngFor, *ngIf, *ngSwitch
- Attribute Directives
  - Directives that change the appearance or behaviour of an element
  - Used as attributes of elements. Eg. ngStyle, ngClass

Built-in Directives

- Structural
  - NgFor
  - NgIf
  - NgSwitch

- Attribute
  - NgClass
  - NgStyle
  - NgNonBindable

Custom Directives

- Custom Directives are classes that are decorated with the @Directive decorator
- Contain the metadata ‘selector’ which is enclosed in [] to specify it as an attribute
- For attribute directives, we have access to the dom elements through elementRef, which can be updated. (Use renderer)
- We can access the properties of host element using @HostBinding, and register eventlisteners on the host element using @HostListerner
- We can pass data to the directive using @Input decorator
- In structural directives, we have access to the template as templateRef and to the view container as ViewContainerRef
- We can add the template to the container using vcRef.createEmbeddedView() and empty the container using vcRef.clear()