Hooking into Component Life Cycle

- A component instance has a lifecycle.
- The lifecycle starts when Angular instantiates the component.
- The lifecycle continues with change detection.
- The lifecycle ends when Angular destroys the component instance and removes from DOM.
- Component **Creates, updates and destroys** instances.
- All phases of component are maintained by a sequence of events.
- These events are controlled with a set of methods known as Hook Method.

Hook Method	Purpose
ngOnChanges()	- Angular sets the value. - Binds the values to any property. - It gets notified with the changes in values by using “SimpleChanges” object. - It gets the previous value and current value. - It includes loading and any action performed in application.
ngOnInit()	- It will be called after the first ngOnChanges(). - Initialize the directive or component after Angular first displays the data-bound properties and sets.
ngDoCheck()	- Called immediately after ngOnChanges() or every change detection and also immediately after “ngOnInit()” on its first run. - It can Detect and act upon the changes that Angular can’t or won’t detect implicitly. - This is very regular while using custom events.
ngAfterContentInit()	- Called after ngOnInit() - Content into the components View (“.html”). - It can into any another directive view. “ng-template, ng-container”
ngAfterContentChecked()	- Called after ngAfterContentInit() and after every ngDoCheck() - This is responsible for **“Content Projection”** - It is a way to import HTML content from outside the component and insert that content into the components template at specific location. - After binding data to view on current component. It brings the

	content from external component or child component and renders into current component.
ngAfterViewInit()	- Called after "ngAfterContentChecked() - Respond after Angular Initializes the components views and its child views. - It responds to view changes. - It is view hierarchy during changes. - It identifies the changes in parent or child views.
ngAfterViewChecked()	- Rendering the final content to parent and child views.
ngOnDestroy()	- Cleanup the memory before destroying the component. - Unsubscribe to methods. - Detach the event handlers. - It is important to handle memory leaks.

FAQ: What is the purpose of ngOnInit()? Why not constructor()?
A: ngOnInt() performs complex initializations outside constructor. The initialized components are safe in ngOnInit().
Ex: ngOnInit() is a good place for a component to fetch the initial data from any remote location and load into memory, which can be kept safe and with restricted access.

FAQ: What is Change Detection?
FAQ: What is Content Projection?