Angular Lifecycle Hooks - Summary & Use Cases

Angular components go through a series of lifecycle phases. These hooks allow developers to tap into key moments during a component's life.

Below is the list of lifecycle hooks in the order they are called, with their typical use cases and examples.

constructor

Use Case: Sets up dependency injection. Avoid data logic.

ngOnChanges

Use Case: Called when @Input() properties change. Useful for reacting to changes.

ngOnInit

Use Case: Used for component initialization. Ideal for API calls or setting defaults.

ngDoCheck

Use Case: Custom change detection logic. Use with caution for performance.

ngAfterContentInit

Use Case: Fires once after content (ng-content) is projected.

ngAfterContentChecked

Use Case: Called after every check of projected content.

ngAfterViewInit

Use Case: Fires once after component view and child views are initialized.

ngAfterViewChecked

Use Case: Called after every check of component's views.

ngOnDestroy

Use Case: Cleanup: Unsubscribe, clear intervals, etc.

Lifecycle Timeline

Angular Lifecycle Hooks Timeline

constructor Sets up dependency injection. Avoid data logic.

ngOnChanges Called when @Input() properties change. Useful for reacting to changes.

ngOnInit Used for component initialization. Ideal for API calls or setting defaults.

ngDoCheck Custom change detection logic. Use with caution for performance.

ngAfterContentInit Called once after content (ng-content) is projected.

ngAfterContentChecked Called after every check of projected content.

ngAfterViewInit Called once after component view and child views are initialized.

ngAfterViewChecked Called after every check of component's views.

ngOnDestroy Cleanup: Unsubscribe, clear intervals, etc.