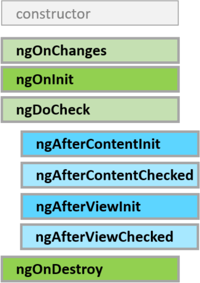
Lifecycle Hooks



A component has a lifecycle managed by Angular.

Angular creates it, renders it, creates and renders its children, checks it when its data-bound properties change, and destroys it before removing it from the DOM.

Angular offers lifecycle hooks that provide visibility into these key life moments and the ability to act when they occur.

A directive has the same set of lifecycle hooks, minus the hooks that are specific to component content and views.

Component lifecycle hooks overview

Directive and component instances have a lifecycle as Angular creates, updates, and destroys them. Developers can tap into key moments in that lifecycle by implementing one or more of the *lifecycle hook* interfaces in the Angular corelibrary.

Each interface has a single hook method whose name is the interface name prefixed with ng. For example, the [OnInit](https://angular.io/api/core/OnInit) interface has a hook method named ngOnInit() that Angular calls shortly after creating the component:

peek-a-boo.component.ts (excerpt)

content\_copyexport class PeekABoo implements [OnInit](https://angular.io/api/core/OnInit) {

constructor(private logger: LoggerService) { }

// implement [OnInit](https://angular.io/api/core/OnInit)'s `ngOnInit` method

ngOnInit() { this.logIt(`[OnInit](https://angular.io/api/core/OnInit)`); }

logIt(msg: string) {

this.logger.log(`#${nextId++} ${msg}`);

}

}

No directive or component will implement all of the lifecycle hooks and some of the hooks only make sense for components. Angular only calls a directive/component hook method *if it is defined*.

Lifecycle sequence

*After* creating a component/directive by calling its constructor, Angular calls the lifecycle hook methods in the following sequence at specific moments:

|  |  |
| --- | --- |
| **Hook** | **Purpose and Timing** |
| ngOnChanges() | Respond when Angular (re)sets data-bound input properties. The method receives a [SimpleChanges](https://angular.io/api/core/SimpleChanges) object of current and previous property values.  Called before ngOnInit() and whenever one or more data-bound input properties change. |
| ngOnInit() | Initialize the directive/component after Angular first displays the data-bound properties and sets the directive/component's input properties.  Called *once*, after the *first* ngOnChanges(). |
| ngDoCheck() | Detect and act upon changes that Angular can't or won't detect on its own.  Called during every change detection run, immediately after ngOnChanges() and ngOnInit(). |
| ngAfterContentInit() | Respond after Angular projects external content into the component's view.  Called *once* after the first ngDoCheck().  *A component-only hook*. |
| ngAfterContentChecked() | Respond after Angular checks the content projected into the component.  Called after the ngAfterContentInit()and every subsequent ngDoCheck().  *A component-only hook*. |
| ngAfterViewInit() | Respond after Angular initializes the component's views and child views.  Called *once* after the first ngAfterContentChecked().  *A component-only hook*. |
| ngAfterViewChecked() | Respond after Angular checks the component's views and child views.  Called after the ngAfterViewInit and every subsequent ngAfterContentChecked().  *A component-only hook*. |
| ngOnDestroy | Cleanup just before Angular destroys the directive/component. Unsubscribe Observables and detach event handlers to avoid memory leaks.  Called *just before* Angular destroys the directive/component. |