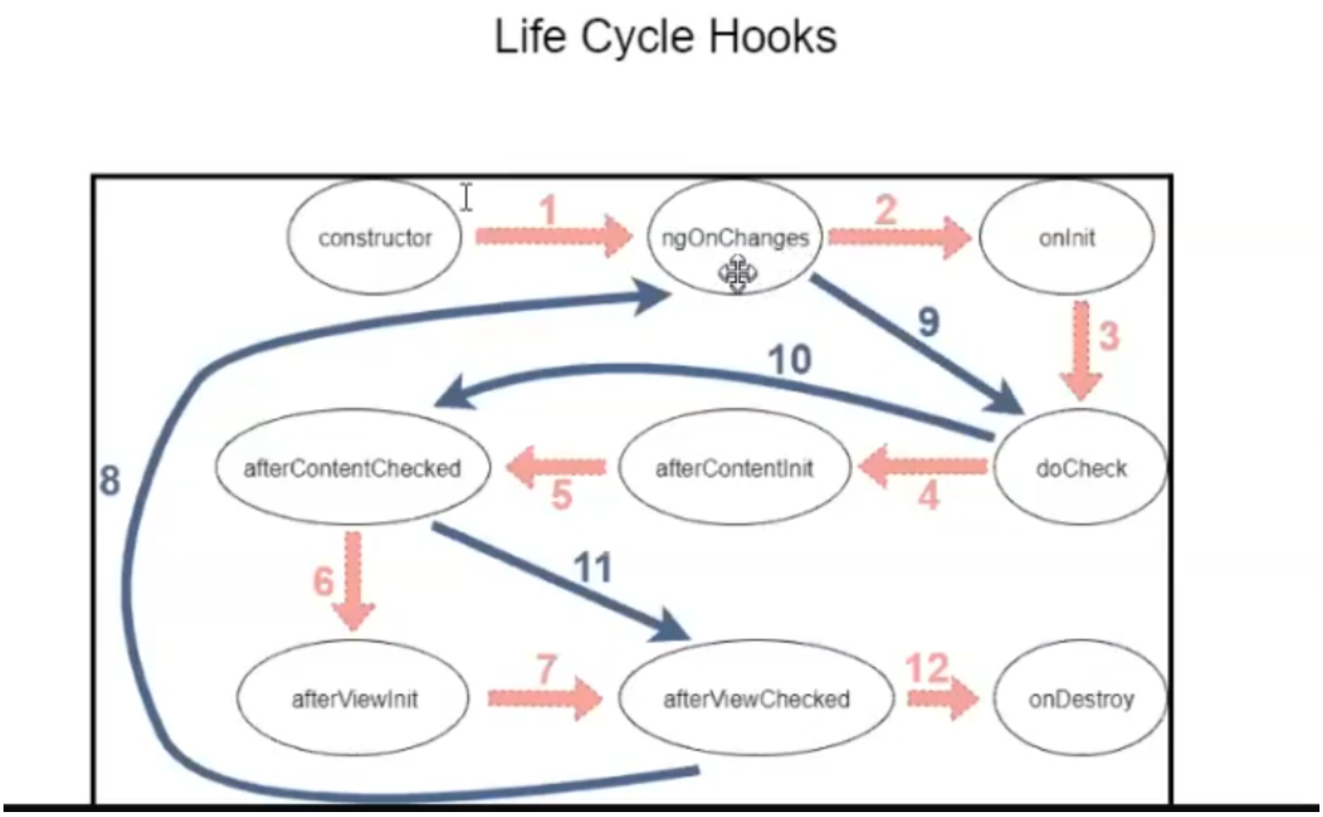
Lifecycle Methods



Angular provides a set of lifecycle hooks that allow you to tap into different stages of a component or directive's lifecycle. These hooks enable you to perform operations at specific moments, such as initialization, change detection, destruction, and more. Here's an overview of the lifecycle hooks available in an Angular component:

Initialization:

1. **ngOnChanges:**
   - Invoked when one or more bound input properties change. Receives a `SimpleChanges` object containing previous and current values.
2. **ngOnInit:**
   - Called once, after the first `ngOnChanges`. Useful for initialization logic such as retrieving data from services.
3. **ngDoCheck:**
   - Triggered during every change detection cycle after `ngOnChanges` and `ngOnInit`. Allows you to implement custom change detection logic.

Content Projection (For components with `<ng-content>`):

4. **ngAfterContentInit:**

    - Called after content (like projected content via `<ng-content>`) is initialized within a component.

5. **ngAfterContentChecked:**
    - Triggered after Angular checks the content projected into the component.

View (Template) Manipulation:

6. **ngAfterViewInit:**
    - Invoked after a component's view and child views are initialized. Useful for DOM manipulations or interacting with child components after view initialization.

7. **ngAfterViewChecked:**
    - Called after Angular checks the component's view and child views for changes.

Destruction:

8. **ngOnDestroy:**
    - Executed just before the component is destroyed. Useful for cleanup tasks like unsubscribing from observables or clearing timers.

Custom Pipes:

9. **ngOnDestroy:**
    - Executed just before the component is destroyed. Useful for cleanup tasks like unsubscribing from observables or clearing timers.

Directives:

For directives, the lifecycle hooks are similar to components but lack specific view-related hooks like `ngOnInit` and `ngOnDestroy`.

Usage:

You implement these hooks in your component or directive classes by providing method definitions:

```
import { Component, OnInit, OnDestroy, OnChanges, SimpleChanges } from
'@angular/core';

@Component({
  selector: 'app-example',
  template: '...'
})
export class ExampleComponent implements OnInit, OnDestroy, OnChanges {
  // Implement lifecycle hook methods here
  ngOnInit() {
    // Initialization logic
  }
  ngOnDestroy() {
    // Cleanup tasks
  }

  ngOnChanges(changes: SimpleChanges) {
    // React to input changes
  }
}
```

Each hook provides a specific entry point to perform actions or handle events during different stages of the component or directive's lifecycle. This allows you to manage component initialization, data changes, rendering, destruction, and more, ensuring your application behaves as intended at each stage.