ANGULAR LIFECYCLE HOOKS

VIVEKANAND P V

INTRODUCTION

- Runtime handles the creation and change detection
- Data-binding and template techniques add additional complexity
- User wants notification about what's going on

IN THE BEGINNING

Component starts its life as an object of the class. Assuming there are no field dependencies, the constructor is the first lifecycle hook. At this point, it’s just an instance of the class and not yet a full-blown component.

TEMPLATE RENDERING STARTS

After the instantiation of the class, the runtime proceeds to start rendering the template associated with the component. At this time, ngOnInit() lifecycle hook is called.

TEMPLATE RENDERING ENDS

Since the start, all the data-bound properties between template and component are bound. When the rendering reaches the end, ngAfterViewInit() will be called.

COMPONENT DESTRUCTION

If the component goes out of the scope, ngOnDestroy() will be called. In case of routing implementation, components are destroyed while the app is still alive. In case of a one-component example, the component is destroyed when the app comes to halt. Thus ngOnDestroy() won't be visible.

SPECIAL NOTE ON LIFECYCLE HOOKS

- Lifecycle hooks are methods which are reserved for the runtime to call. User should avoid calling these methods.
- Usually, most lifecycle hooks are simple notifications. User takes appropriate actions at these junctures.
- Lifecycle hooks are called at runtime. Therefore it is enough to just provide the method. But the best practice would be to implement the interface explicitly.

INPUT PROPERTY-BINDING AND COMPONENT LIFECYCLE

- These can change often
- For any change in any of the input bound properties, ngOnChanges(…) will be called
- Provides a data-structure of SimpleChanges type
- Runs before ngOnInit() and runs subsequently on input-bound properties change

SO FAR: COMPONENT WITH INPUT-PROPERTY BINDING

- constructor
- ngOnChanges(…) // first run
- ngOnInit()
  - *ngOnChanges(…)* // subsequent changes
- ngAfterViewInit()
- ngOnDestroy()

CHANGE-DETECTION CYCLE

- Hook is ngDoCheck()
- Runs on any data-bound activity (property/value/event)
- First run after ngOnInit() but before ngAfterViewInit()
- Second run after ngAfterViewInit()
- Subsequent runs at every data-bound change
- Use with caution; using ngOnChanges(…) and ngOnDoCheck() flags a warning

SO FAR: CHANGE-DETECTION CYCLE

- constructor
- …
- ngOnChanges(…) // subsequent changes, depends
  - ngDoCheck()
- …

CONTENT PROJECTION

- Hook is ngAfterContentInit()
- Runs once when the projected content is rendered
- ngAfterContentInit() is called only once, after first ngDoCheck()

SO FAR: CONTENT-PROJECTION

- …
- ngOnInit()
- ngDoCheck()
  - ngAfterContentInit() // only once, depends
  - …
- …

CHANGE-DETECTION CYCLE: COMPLETION

- Hooks are ngAfterViewChecked() and ngAfterContentChecked()
- Run whenever change-detection cycle completes
- Change detection starts at ngDoCheck() and compltes for the component and its children at ngAfterViewChecked()
- Completes at ngAfterContentChecked() for projected content

CHANGE-DETECTION CYCLE: COMPLETION ORDER

- ngAfterViewChecked() is called at least once after ngAfterViewInit() and subsequently at the end of every change detection cycle
- ngAfterContentChecked() is called after ngAfterContentInit()

SO FAR: CHANGE-DETECTION CYCLE COMPLETION

- …
- ngOnInit()
- ngDoCheck()
  - ngAfterContentInit() // only for projected content
  - ngAfterContentChecked() // only for projected content
  - ngAfterViewInit()
  - ngAfterViewChecked() // at every cycle completion
- …

OVERVIEW

- constructor // only once
- ngOnChanges() // first call for input-bound properties
- ngOnInit() // only once
- ngDoCheck() // first change-detection start
  - ngAfterContentInit() // only once
  - ngAfterContentChecked() // end of change-detection for projected content
  - ngAfterViewInit() // only once
  - ngAfterViewChecked() // end of change-detection for component and its children
- ngOnDestroy() // only once

INPUT-BOUND PROPERTY CHANGE (SUBSEQUENT)

- ngOnChanges() // first call for input-bound properties
  - ngDoCheck() // first change-detection start
    - ngAfterContentChecked() // end of change-detection for projected content
    - ngAfterViewChecked() // end of change-detection for component and its children

CHANGE-DETECTION CYCLE (SUBSEQUENT)

- ngDoCheck() // first change-detection start
  - ngAfterContentChecked() // end of change-detection for projected content
  - ngAfterViewChecked() // end of change-detection for component and its children