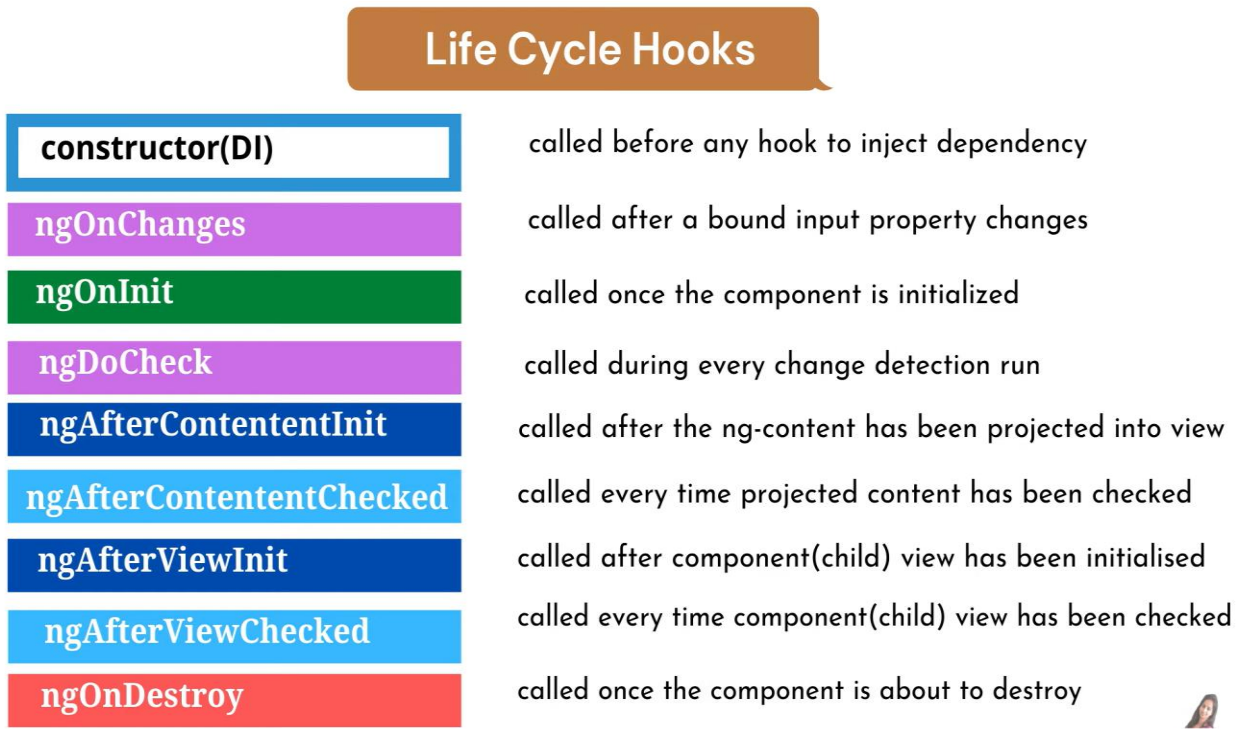
16

Component Lifecycle/Hooks

- ✓ Constructor
- ✓ ngOnchange
- ✓ ngOnInit
- ✓ ngDoCheck
    - o ngAfterContentInit
    - o ngAfterContentChecked
    - o ngAfterViewInit
    - o ngAfterViewChecked
- ✓ ngOnDestroy



Life Cycle Hooks

constructor(DI)	called before any hook to inject dependency
ngOnChanges	called after a bound input property changes
ngOnInit	called once the component is initialized
ngDoCheck	called during every change detection run
ngAfterContententInit	called after the ng-content has been projected into view
ngAfterContententChecked	called every time projected content has been checked
ngAfterViewInit	called after component(child) view has been initialised
ngAfterViewChecked	called every time component(child) view has been checked
ngOnDestroy	called once the component is about to destroy

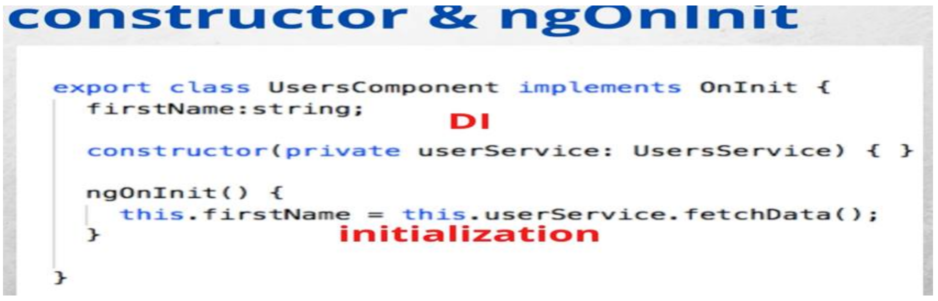
What are Component Hooks/Lifecycle?

- Every component in angular has a life-cycle , a series of stages that goes through from initialization to destruction
- Life-cycle hooks events occur at each stage. As a result, we can use these hook events various stages of our application to achieve fine control over the components.
- Hooks can implement by using interfaces

what is constructor?

- it is special method which will invoked automatically whenever object is created
- it is typescript feature not angular
- it is used to dependency injection (di)
- it executes in first of hooks after that start hooks

example;



constructor & ngOnInit

```
export class UsersComponent implements OnInit {
  firstName:string;

  constructor(private userService: UsersService) { }

  ngOnInit() {
    this.firstName = this.userService.fetchData();
  }
}
```

What is ngOnInit?

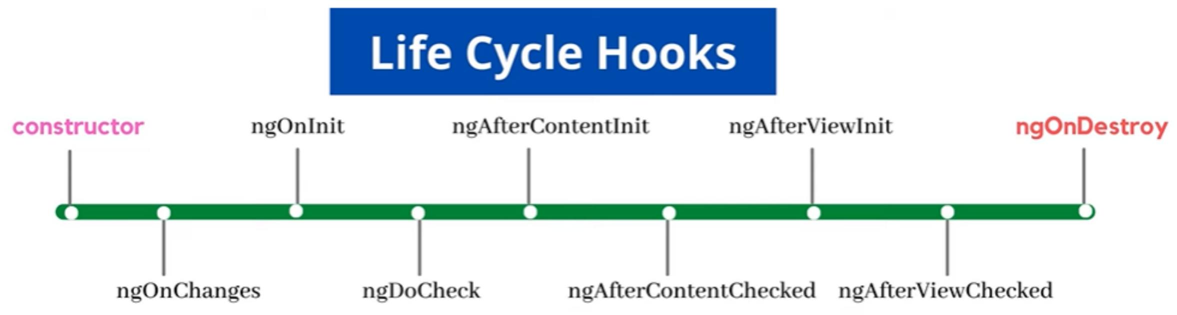
- Called on initialization
- `OnInit` is a lifecycle hook that is called after Angular has initialized all data-bound properties of a directive.
- `ngOnInit()` method to handle any additional initialization tasks.
- This hook is called when the first change detection is run on the component.

Example

```
export class BComponent implements OnInit {
 ngOnInit() {
 console.log("ngOnInit called")
 }
}
```

Difference between constructor and ngOnInit

constructor	ngOnInIt
➢ Should use only for dependency injection ➢ It is part of typescript ➢ Binding not happened till we can access only variables	➢ Should use for handle any initial logic that need to be executed ➢ It is part of angular framework ➢ Binding with the UI is done ➢ So can access all data bound properties



ngOnChanges

- this method is called once on components creation and then every time changes are detected in one of the component input properties
- It Receives a Simplechanges Object as a parameter which contains information regarding the which of the input properties has changed- in case we have more than one – current and previous value

ngDoCheck

- During the change detection , when angular checks components input properties for change, it uses === for dirty checking.
- For arrays/object this means the references only are the dirty checked, since product array isn't change. Hence ngOnchange will not execute for that the solution we can use ngDocheck
- Detect and act upon changes that angular cant or wont detects its own
- Called during the every change detection run, immediately after ngOnchanges and ngOninit