OnInit is an interface which returns Void. Called immediately after the change detector is triggered and the very first time.

All the components does not render at the same time.

Whenever the angular needs to use the component, it creates a new instance of it and starts using it and then destroys it.

So the question is, how do we get to know when it is getting rendered?

So whenever there is use of the selector in the page that is getting rendered at that time a new instance of the component is generated.

So the selector which is placed inside the page of any particular component, then the selector of the placed component acts as a child component.

This can be done through routing.. so the next question is .. Can there be two components rendering at the same time. Answer is Yes.

Through child component. Once its use is over , it gets destroyed.

So a lot of stages happens depending upon the requirement of the page use. Below are the stages

ngOnChanges

ngOninit

ngDocheck

ngAfterContettInit

ngAfterContentChecked

ngAfterViewInit

ngAfterviewChecked

ngOnDestroy.

Each hook has its own interface and to use it , we need to implement its interface

Eg: ngOnInit

Interface: OnInit

NgOnChanges()

Created a component – Home component . Placed its selector in the app component.html

Created another component as Author, placed its selector in the home component .

Created an @Input property data: number

Bind it in component.

In the parent component – Home component, html. Passé the value using binding

Rendered it

Example of parent and child component described above.

Now working with onchganges()

NgOnInit()

This will ne called once in the lifetime and before anything is displayed in the view.\

Oninit-api call, logic to peform service

Ctor—inject sevice,methods, no logic

NgDoCheck()— hits everytime a child component receives value from the parent component using the input decortator

While ngonchanges is able to detect only when there is a change in the value reference… meaning a new object is created .

Try to avoid use both of them together at once.