by Kunal Sir

Component Life-Cycle

❖ Component Life Cycle:-

All Components and Directive have their life cycle as Angular creates , updates and destroy them.

Angular life cycle hooks begin with the instantiation of the component class by Angular.

Once the component instance is destroyed, and its displayed template is removed from the DOM object, the lifespan terminates

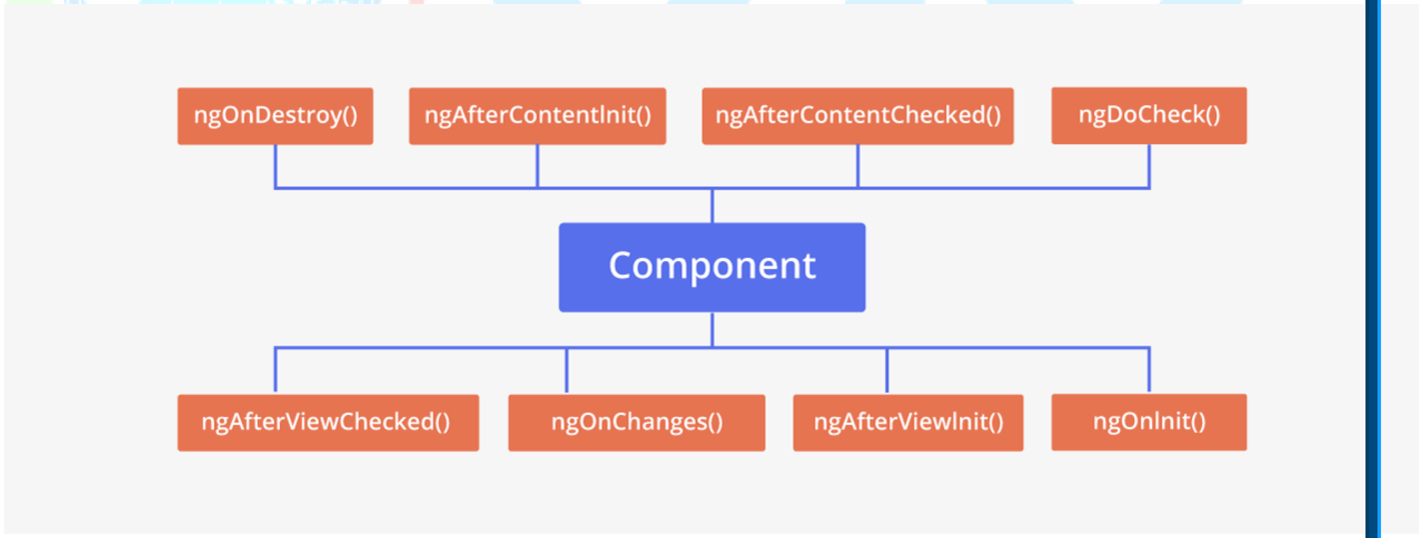
Angular provides some key Interfaces which provides hooks to handle the sensitive events.

Like form initialization, getting data from services, and as soon as component life cycle ends

We need to close the resources and perform cleanup activity.

Note :- constructor is not a life cycle hook , it is a typescript feature.

❖ Life Cycle Hooks:-



Component Life-Cycle

1. ngOnInit() :-

ngOnInit hook get invoke just before view is render on browser window.and just after class constructor and ngOnChanges hook get executed.

This hook is invoked once in component life span.

ngOnInit hook is mostly use for initialization purpose.

2. ngOnDestroy() :-

ngOnDestroy hook get invoke just before component complete its life span (removed from DOM object)

This hook is invoked once in component life span.

Using this hook we can perform cleanup activities.

3. ngOnChanges() :-

ngOnChanges hook invoke just after class constructor and just before ngOnInit hook get invoked.

Also if there is change detected in input property then this hook get invoked again.

This hook accept argument type of SimpleChanges.

4. ngDoCheck() :-

ngDoCheck hook is invoked just after ngOnInit hook get executed.

As similarly ngOnChanges , ngDoCheck also get invoked whenever changes are detected.

5. ngAfterContentInit() :-

ngAfterContentInit hook is invoked just before first ngDoCheck.

It can be use at the time of content projection.

Component Life-Cycle

6. ngAfterContentCheck() :-

ngAfterContentCheck hook will be invoke after every ngDoCheck.

7. ngAfterViewInit() :-

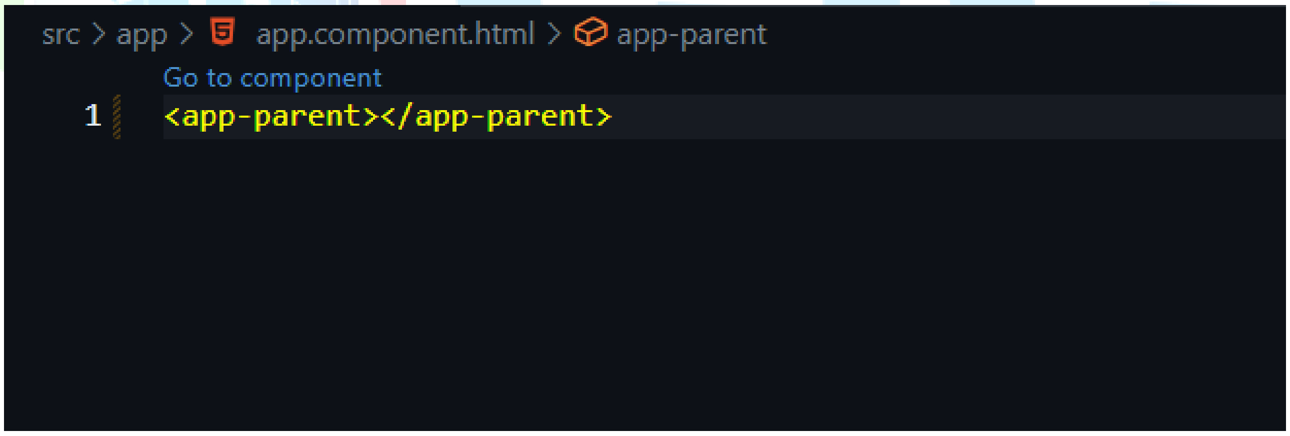
ngAfterViewInit will be invoked after first ngAfterContentCheck.

8. ngAfterViewCheck():-

ngAfterViewCheck will be invoked after ngAfterViewInit and after every subsequent of ngAfterContentCheck.

Example : -

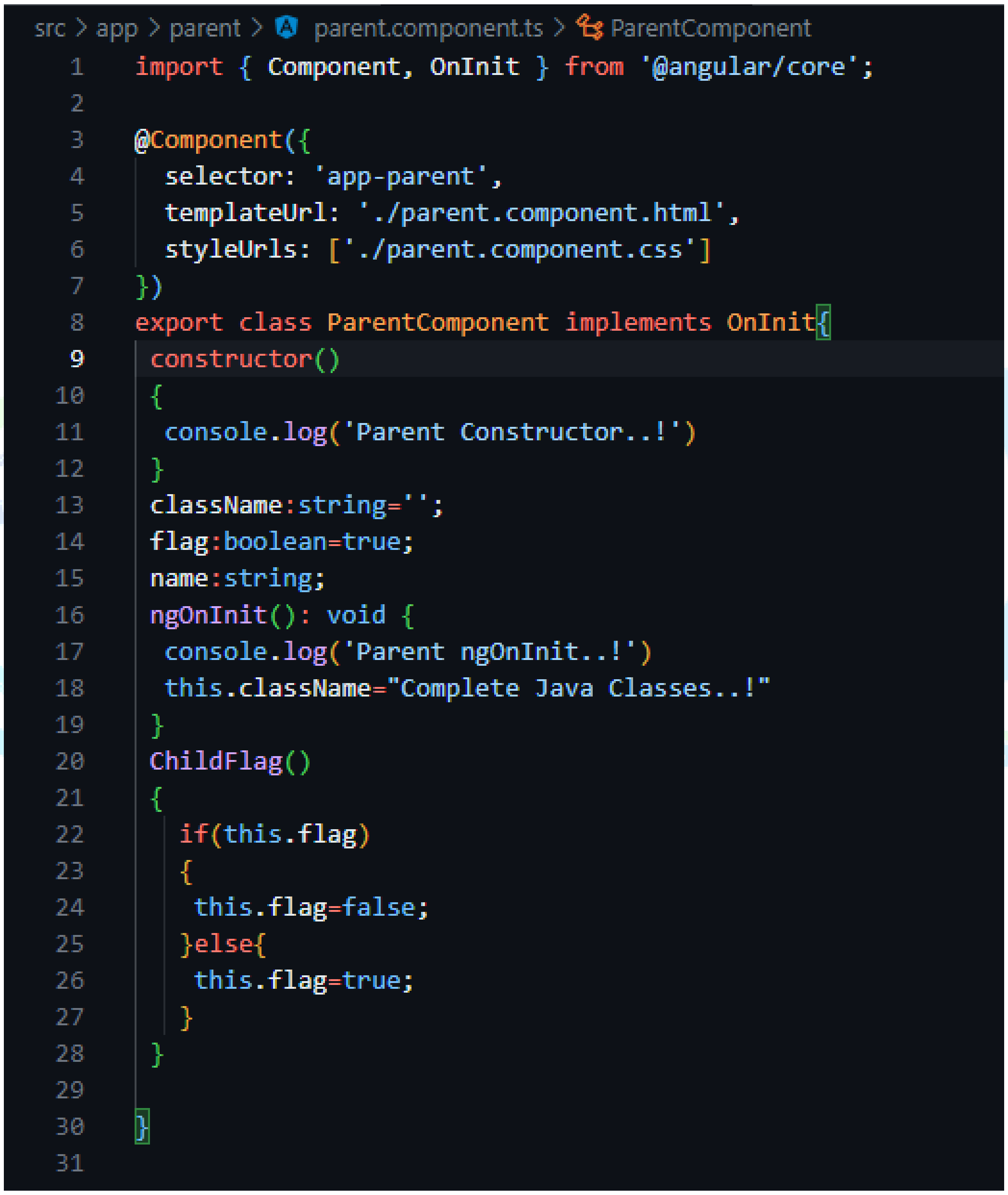
1. Snippet of app.component.html.



```html
<app-parent></app-parent>
```

Component Life-Cycle

2. **Snippet of parent.component.ts.**

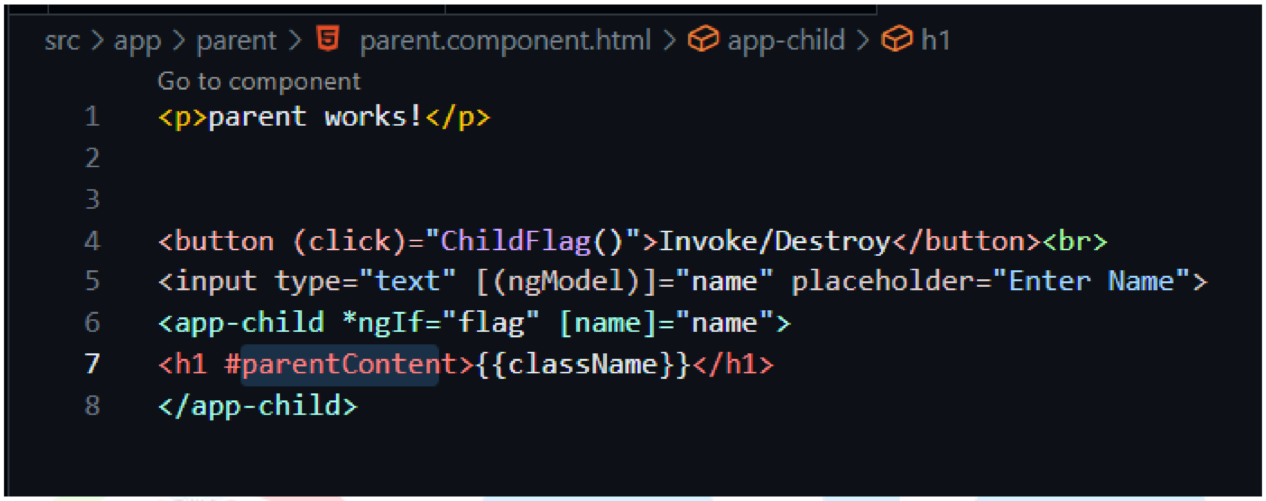


src > app > parent > parent.component.ts > ParentComponent

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

@Component({
  selector: 'app-parent',
  templateUrl: './parent.component.html',
  styleUrls: ['./parent.component.css']
})
export class ParentComponent implements OnInit{
 constructor()
 {
  console.log('Parent Constructor..!')
 }
 className:string='';
 flag:boolean=true;
 name:string;
 ngOnInit(): void {
  console.log('Parent ngOnInit..!')
  this.className="Complete Java Classes..!"
 }
 ChildFlag()
 {
   if(this.flag)
   {
    this.flag=false;
   }else{
    this.flag=true;
   }
 }

}

```

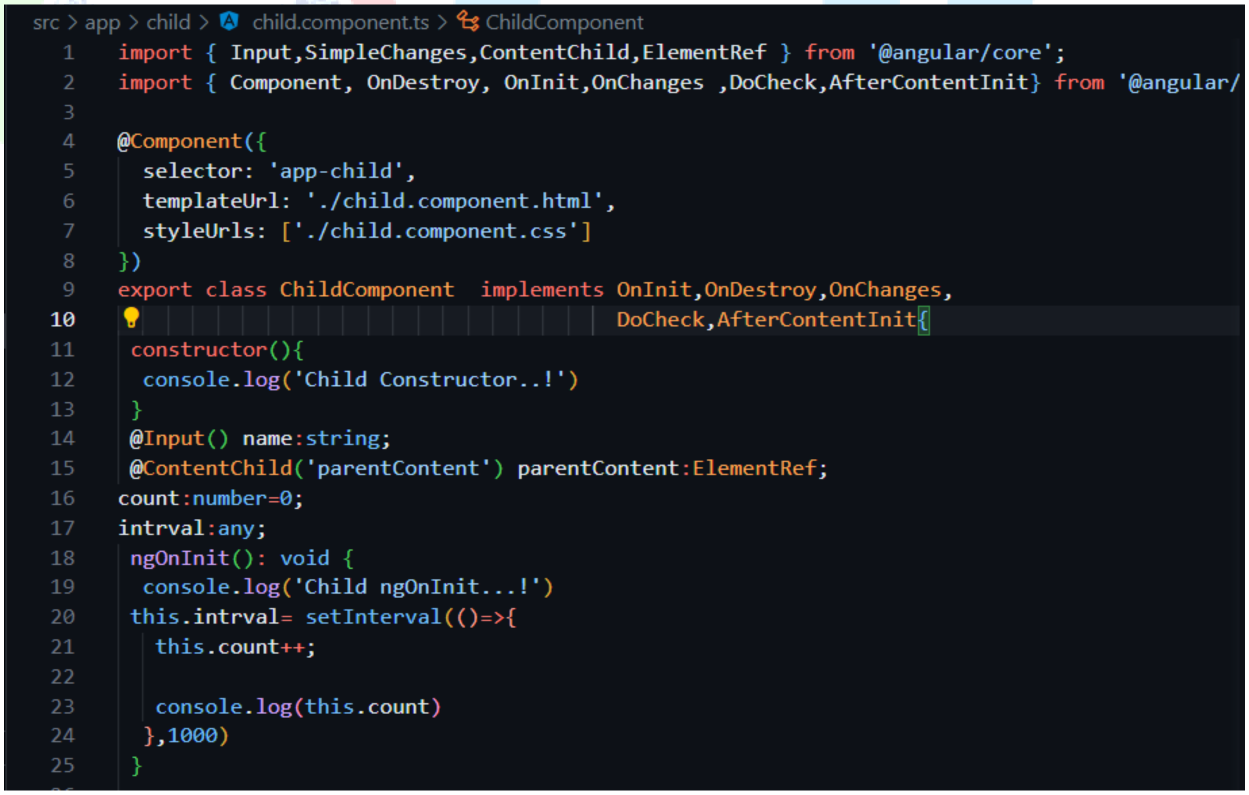
Component Life-Cycle

3. Snippet of parent.component.html.



```html
<p>parent works!</p>


<button (click)="ChildFlag()">Invoke/Destroy</button><br>
<input type="text" [(ngModel)]="name" placeholder="Enter Name">
<app-child *ngIf="flag" [name]="name">
<h1 #parentContent>{{className}}</h1>
</app-child>
```

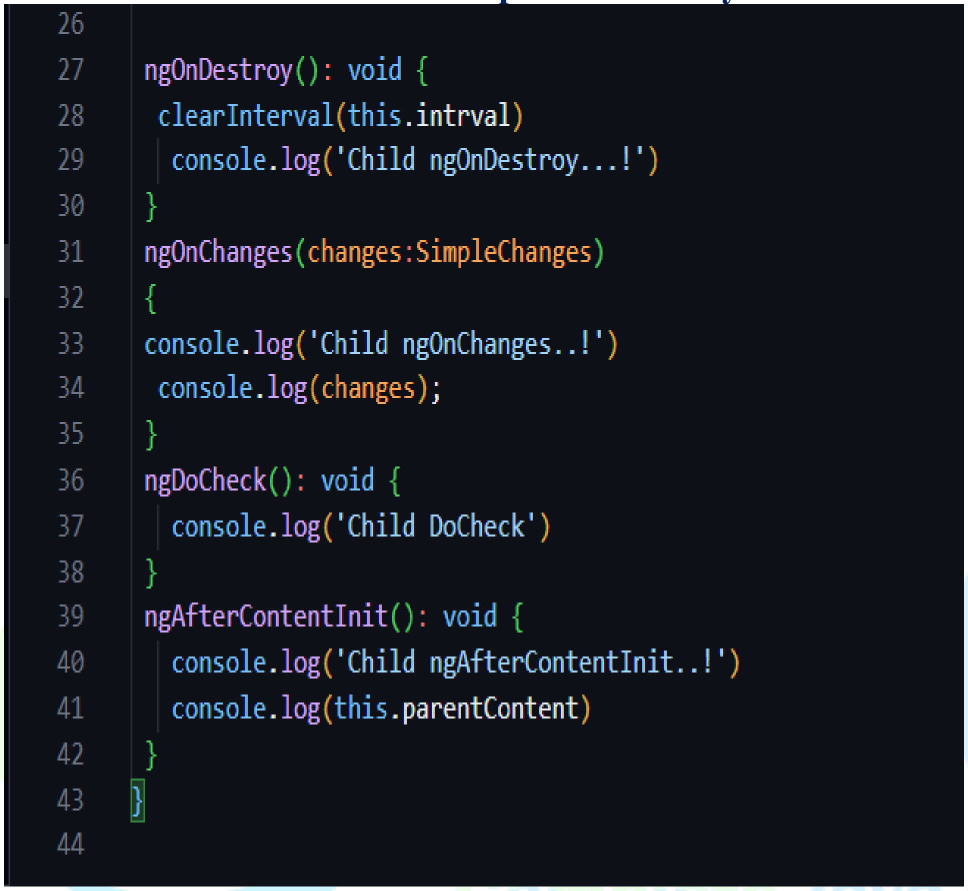
4. Snippet of child.component.ts.



```typescript
import { Input,SimpleChanges,ContentChild,ElementRef } from '@angular/core';
import { Component, OnDestroy, OnInit,OnChanges ,DoCheck,AfterContentInit} from '@angular/

@Component({
  selector: 'app-child',
  templateUrl: './child.component.html',
  styleUrls: ['./child.component.css']
})
export class ChildComponent  implements OnInit,OnDestroy,OnChanges,
                                        DoCheck,AfterContentInit{
 constructor(){
  console.log('Child Constructor..!')
 }
 @Input() name:string;
 @ContentChild('parentContent') parentContent:ElementRef;
count:number=0;
intrval:any;
 ngOnInit(): void {
  console.log('Child ngOnInit...!')
 this.intrval= setInterval(()=>{
   this.count++;

   console.log(this.count)
  },1000)
 }
```

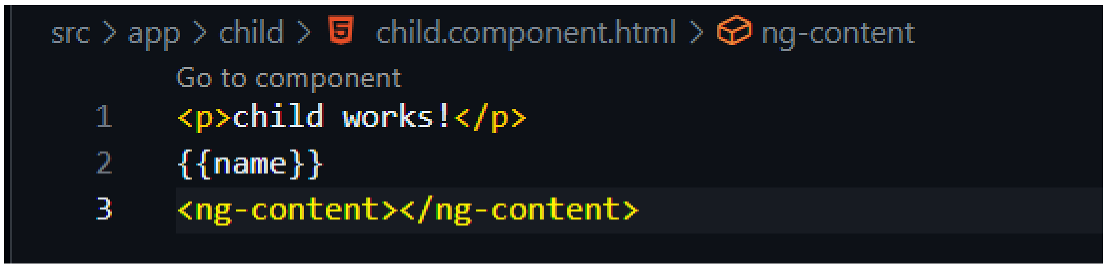
by Kunal Sir

Component Life-Cycle



```
  ngOnDestroy(): void {
   clearInterval(this.intrval)
    console.log('Child ngOnDestroy...!')
  }
  ngOnChanges(changes:SimpleChanges)
  {
  console.log('Child ngOnChanges..!')
   console.log(changes);
  }
  ngDoCheck(): void {
    console.log('Child DoCheck')
  }
  ngAfterContentInit(): void {
    console.log('Child ngAfterContentInit..!')
    console.log(this.parentContent)
  }
}
```

5. Snippet of child.component.html.



src > app > child > child.component.html > ng-content

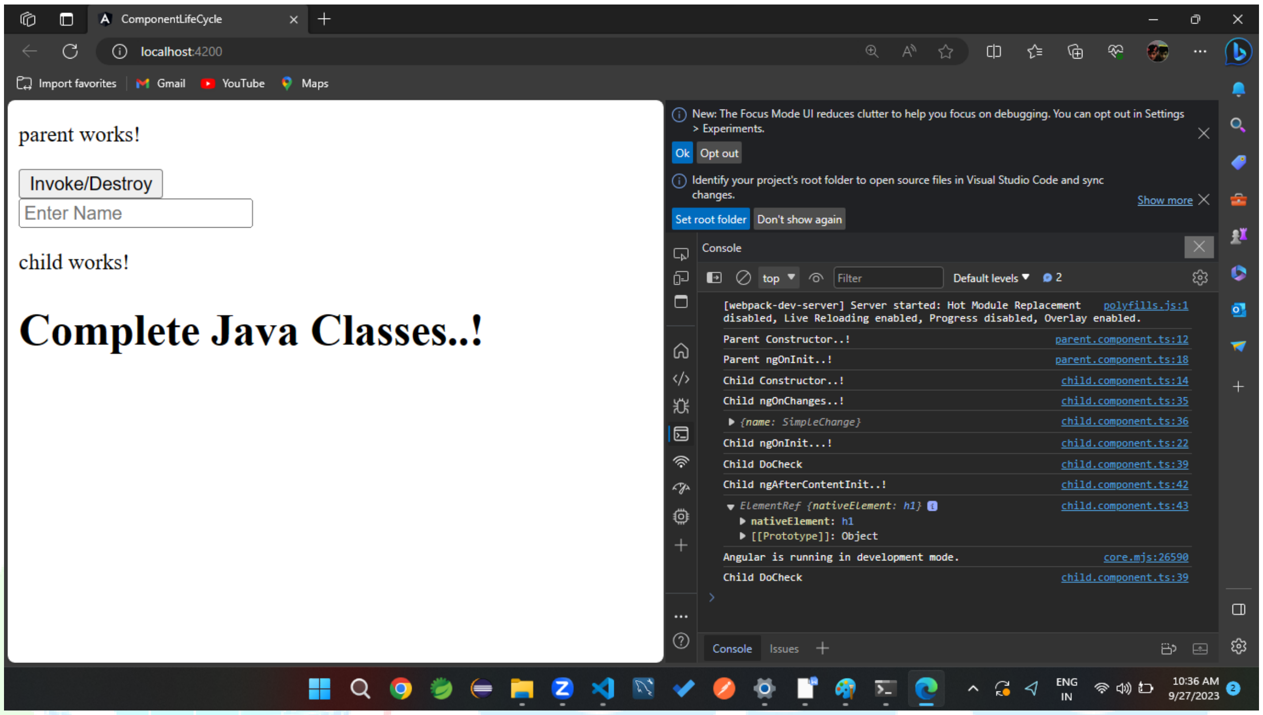
Go to component

```
<p>child works!</p>
{{name}}
<ng-content></ng-content>
```

by Kunal Sir

Component Life-Cycle

6. OUTPUT:-



parent works!

Invoke/Destroy

Enter Name

child works!

Complete Java Classes..!

```
[webpack-dev-server] Server started: Hot Module Replacement   polyfills.js:1
disabled, Live Reloading enabled, Progress disabled, Overlay enabled.
Parent Constructor..!                                 parent.component.ts:12
Parent ngOnInit..!                                    parent.component.ts:18
Child Constructor..!                                   child.component.ts:14
Child ngOnChanges..!                                   child.component.ts:35
 ▶ {name: SimpleChange}                                child.component.ts:36
Child ngOnInit...!                                     child.component.ts:22
Child DoCheck                                          child.component.ts:39
Child ngAfterContentInit..!                            child.component.ts:42
 ▼ ElementRef {nativeElement: h1}                      child.component.ts:43
   ▶ nativeElement: h1
   ▶ [[Prototype]]: Object
Angular is running in development mode.                      core.mjs:26590
Child DoCheck                                          child.component.ts:39
```

ComponentLifeCycle
localhost:4200
Import favorites
Gmail
YouTube
Maps
New: The Focus Mode UI reduces clutter to help you focus on debugging. You can opt out in Settings > Experiments.
Ok
Opt out
Identify your project's root folder to open source files in Visual Studio Code and sync changes.
Show more
Set root folder
Don't show again
Console
top
Filter
Default levels
2
Console
Issues
10:36 AM
9/27/2023