**生命周期顺序**

指令和组件的实例有一个生命周期：新建、更新和销毁。   
当Angular使用构造函数新建一个组件或指令后，就会按下面的顺序在特定时刻调用这些生命周期钩子方法：

**ngOnChanges()**

用处：当Angular设置数据绑定输入属性发生变化时响应。  
时机：当被绑定的输入属性的值发生变化时调用，不过首次调用是会发生在ngOnInit()之前的。  
ngOnChanges()方法获取了一个对象，它可以同时观测1个/多个绑定的属性值

**ngOnInit()**

用处：在Angular第一次显示数据绑定和设置指令/组件的输入属性之后，初始化指令/组件。   
时机：在第一轮ngOnChanges()完成之后调用，只调用一次。  
要明确一点就是，虽然接口中有一个constructor构造函数，但是在这里最好只对局部变量进行初始化之外什么都不做，通过ngOnInit()获取初始数据，而且是放在构造函数后面。

**ngDoCheck()**

用处：检测那些Angular自身无法捕获的变更  
时机：在每个Angular变更检测周期中调用，ngOnChanges()和ngOnInit()之后。  
谨慎使用，因为一些你意想不到的事情也会被视为变更了

**ngAfterContentInit()和ngAfterContentChecked()**

只适用于组件  
用处：在外来内容被投影到组件中之后/投影组件内容的变更检测之后调用  
时机：   
ngAfterContentInit()--第一次ngDoCheck()之后调用，且只调用一次；   
ngAfterContentChecked()--每次ngDoCheck()之后调用，如果需要调用ngAfterContentInit()，则在ngAfterContentInit()调用之后

**ngAfterViewInit()和ngAfterViewChecked()**

只适用于组件   
用处：初始化完/检测变更完组件视图及其子视图之后调用。

**ngOnDestroy**

用处：销毁指令/组件之前调用并清扫

释放那些不会被垃圾收集器自动回收的各类资源的地方。取消那些对可观察对象和DOM事件的订阅。停止定时器。注销该指令曾注册到全局服务或应用级服务中的各种回调函数。