x.subscribe(

(response)=>{},

(error)=>{}

);

If we get promise, instead of Observable,

.then(

(response)=>{},

(error)=>{}

);

Observable vs Promise

We can create an observable, whenever we want to execute a method asynchronously.

1. ngOnChanges –

This event is called every time whenever there is a change in the value of control of the input. The change in the value of a property that is bound fires this event. A data map containing the changes, i.e. the previous and the current value of the property within a SimpleChange.

Properties

The components having an input can use the method.

Whenever there is a change in the value of the input, the method gets invoked.

2. ngOnInit

After the initialization of the component or the display of the properties of the bound data, the event of ngOnInit is initialized. The event is called once only after the event of ngOnChanges. Therefore, the event is mainly responsible for initializing the components data.

Properties

The data in a component is initialized by this event.

The method is called when the values of the input are set.

Angular CLI has by default added this hook to all its components.

The method can be called only once.

3. ngDoCheck

After the check on the input properties of the angular components, the triggering of the ngDoCheck event occurs. It is mainly done to detect or act upon any changes that angular fails to detect. Based on the developer’s logic, the check can be implemented. Therefore, implementation of any custom change logic or algorithms for detection of change in any components is allowed by the event.

Properties

For any detection of changes angular runs this method.

The method is called for detecting changes.

4. ngAfterContentInit –

After the projection of the external content within the view of the component, the event ngAfterContentInit is called. For checking all the component bindings for the first time, this method gets executed for the first time. Its execution follows as soon as the execution of the method ngDoCheck(). The child component is usually linked with this method.

Properties

Initially, the method is called after ngDoCheck.

Work is done by the content initialization.

5. ngAfterContentChecked –

The projection of the external content into the component is followed by checking the projected content. The method is called every time the check in the content is made through the mechanism of change detection of angular. It is executed after the execution of the method ngAfterContentInit(). Also, the method is executed after the subsequent execution of ngDoCheck(). It is usually linked with the initialization of the child components.

Properties

To get started, the method waits for the completion of ngContentInit.

Execution is done after ngDocheck.

6. ngAfterViewInit

With the initialization of the angular components and the child components, the method of ngAfterViewInit is called. After the execution of the ngAfterContentChecked method, ngAfterViewInit method is called for the first time. The method is applicable only to the components of angular.

Properties

Only once the call for the method is generated after the view of the component is initialized.

7. ngAfterViewChecked –

Just after the ngAfterViewInit method, the ngAfterViewChecked method is called. Whenever the change detection method of angular does its checks over the components, the method of ngAfterViewChecked gets executed. The method is also executed after the execution of the ngAfterContentChecked(). Also, when the binding of the directives of the child component is changed, the method gets executed.

Properties

The call is generated after the initialization and checking.

Work of the method is started after the completion of every method of ngDocheck.

8. ngOnDestroy

The method is called just before the destruction of the components by angular. To avoid scenarios of memory leaks, the method is able to detach event handlers, and also useful in unsubscribing observables. Only for once, the method is called to remove the component from the DOM.

Properties

The call is generated just before the removal of components from DOM.

Interfaces in Angular Lifecycle

The class of the components can be used for defining the angular lifecycle hooks methods. However, with the help of the interfaces, the methods can be called upon. The names of the interfaces are similar to that of the method names, omitting the prefix “ng”. This is because of the presence of a typescript interface with each of the methods of the lifecycle hooks. For example, the interface of ngOnInit is called OnInit. Only one event of the lifecycle hook is defined through a single interface. Further, the compiler doesn’t throw any compilation error when the interfaces are not implemented.

https://www.upgrad.com/blog/life-cycle-of-angular-components/