AngularServices: Dependency Injection : Code reusability for utility methods:

ng g service myservice

C:\projectA7\angular7-app>ng g service myservice

CREATE src/app/myservice.service.spec.ts (348 bytes)

CREATE src/app/myservice.service.ts (138 bytes)

Following are the files created which are shown at the bottom ― myservice.service.specs.ts and myservice.service.ts.

**myservice.service.ts**

import { Injectable } from '@angular/core';

@Injectable({

providedIn: 'root'

})

export class MyserviceService {

constructor() { }

showTodayDate() {

let ndate = new Date();

return ndate;

}

}

Here, the injectable module is imported from the @angular/core. It contains the @Injectable method and a class called MyserviceService. We will create our service function in this class.

Before creating a new service, we need to include the service created in the main parent **app.module.ts**.

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppRoutingModule , RoutingComponent} from './app-routing.module';

import { AppComponent } from './app.component';

import { NewCmpComponent } from './new-cmp/new-cmp.component';

import { ChangeTextDirective } from './change-text.directive';

import { SqrtPipe } from './app.sqrt';

import { MyserviceService } from './myservice.service';

@NgModule({

declarations: [

AppComponent,

NewCmpComponent,

],

imports: [

BrowserModule,

AppRoutingModule

],

providers: [MyserviceService],

bootstrap: [AppComponent]

})

export class AppModule { }

We have imported the Service with the class name, and the same class is used in the providers. Let us now switch back to the service class and create a service function.

In the service class, we will create a function which will display today’s date. We can use the same function in the main parent component app.component.ts and also in the new component new-cmp.component.ts that we created in the previous chapter.

In the above service file, we have created a function **showTodayDate**. Now we will return the new Date () created. Let us see how we can access this function in the component class.

**app.component.ts**

import { Component } from '@angular/core';

import { MyserviceService } from './myservice.service';

@Component({ selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

title = 'Angular 7 Project!';

todaydate;

constructor(private myservice: MyserviceService) {}

ngOnInit() {

this.todaydate = this.myservice.showTodayDate();

}

}

The ngOnInit function gets called by default in any component created. The date is fetched from the service as shown above. To fetch more details of the service, we need to first include the service in the component ts file.

We will display the date in the .html file as shown below −

**app.component.html**

{{todaydate}}

<app-new-cmp></app-new-cmp>

Let us now see how to use the service in the new component created.

**New.cmp.component.ts**

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

import { MyserviceService } from './../myservice.service';

@Component({

selector: 'app-new-cmp',

templateUrl: './new-cmp.component.html',

styleUrls: ['./new-cmp.component.css']

})

export class NewCmpComponent implements OnInit {

newcomponent = "Entered in new component created";

todaydate;

constructor(private myservice: MyserviceService) { }

ngOnInit() {

this.todaydate = this.myservice.showTodayDate();

}

}

In the new component that we have created, we need to first import the service that we want and access the methods and properties of the same. Check the code highlighted. todaydate is displayed in the component html as follows −

**new-cmp.component.html**

<p>

{{newcomponent}}

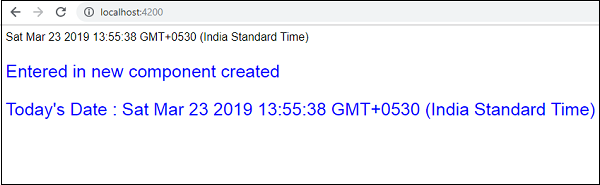
</p>

<p>

Today's Date : {{todaydate}}

</p>

The selector of the new component is used in the app.component.html file. The contents from the above html file will be displayed in the browser as shown below −



If you change the property of the service in any component, the same is changed in other components too. Let us now see how this works.