Registering Services with the Providers Array in Angular

## **Introduction**

In this guide, we will learn how to register service with the providers array. As you know, service is an essential concept in Angular if you want to communicate between two components that are not in a parent-child relationship. In such cases, you need to create a service and inject it into the component.

There are several methods for injecting a service into a component. Today, we're going to inject a service with the help of the providers array. If you have not heard this term before, don't worry; I'm going to cover everything from creating the service to injecting it into the providers array.

So let's get started by creating an Angular application.

## **Creating an Angular App**

To create an Angular app, you need to hit the below command in the Angular CLI.

1

ng new demo-app

console

Now we have a component called AppComponent. We'll create a service and register with AppComponent.

## **Creating a Service**

The below command will help you to create a service without making a unit test file because we're not going to perform any unit testing for now.

1

ng g s demo --skipTests=true

console

You can see our DemoService structure looks like the snippet below.

1

2

3

4

5

6

7

8

9

10

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

@Injectable({

providedIn: 'root'

})

export class DemoService {

constructor() { }

}

ts

## **@Injectable Decorator Helps to Inject the Service**

Here we have the @Injectable decorator. providedIn of the @Injectable decorator says that we have implemented the service on a global level, so we don't have to inject it in every component where we're going to use the service.

But in many cases, you don't want to inject it at the root level. There are many reasons, such as to improve the performance of an app. In thee cases, we need to manually inject the service in every component where we're going to use the service.

I'm going to remove the @Injectable decorator from the service and write a function to test the successful injection in the component without @Injectable.

Finally, our service will look like the snippet below.

1

2

3

4

5

6

7

8

export class DemoService {

constructor() { }

testFunction(){

console.log('Test function called:');

}

}

ts

## **Using Providers Array in the Component**

Now I'm going to inject the service in AppComponent by using the providers array and call the function of the service class to test the service injection in our component.

**app.component.ts**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

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

import { DemoService } from './demo.service';

@Component({

selector: 'app-root',

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

styleUrls: ['./app.component.scss'],

providers:[DemoService]

})

export class AppComponent implements OnInit {

constructor(private demoService: DemoService) { }

ngOnInit() {

this.demoService.testFunction();

}

}

ts

We've successfully injected our service in an AppComponent. Now we can run it to test.

Below is the output:

1

Test function called: demo.service.ts:6

console

We've successfully executed our application by injecting the service. It is possible to use this injected service in different components. You can create a different module in your application to add lazy loading, for example, and inject the service in the module. This will enable all components under this module to use the lazy loading service.

## **Using Providers Array in the Module**

Now we are going to use app.module.ts to show you a demo of how to inject a service on another module. Below is the code with the providers array.

**app.module.ts**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

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

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

import { FormsModule } from '@angular/forms'

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

@NgModule({

declarations: [

AppComponent,

],

imports: [

BrowserModule,

FormsModule

],

providers: [DemoService],

bootstrap: [AppComponent]

})

export class AppModule { }

ts

You can see the same output by injecting the service on the module rather than every single component.

## **Conclusion**

This guide is all about how to inject a service with the help of the provider's array. I hope you have enjoyed it a lot. You can read more about it [here](https://angular.io/guide/providers).