

## Services-Angular Notes

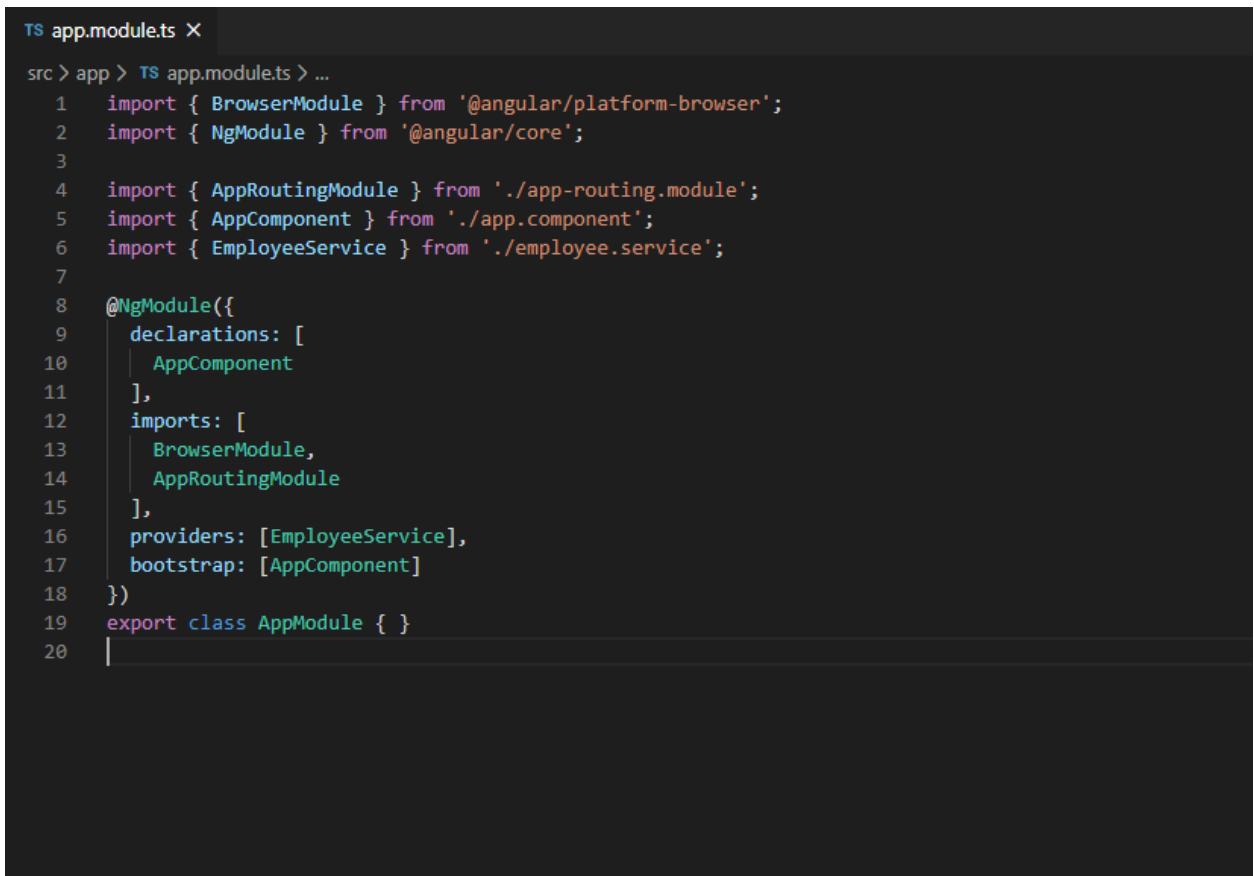
- A service is a class with a specific purpose.
- It is injected into other components and services that need it.
- Services are a great way to share information among classes that *don't know each other*.
- With services, we can access methods and properties across other components in the entire project.
- It must be declared as injectable using @Injectable decorator.
- When two different components need different instances of a service, it would be better to provide the service at the component level that needs the new and separate instance.
- A service is to be registered before it is used.
- A service provider is to be registered with the injector, or it won't know how to create the service.
- The providers property of @ Component or @ NgModule decorators is used to list services to be registered with the component or module.
- When a service is registered using module it is available to all components of that module.

## Example 1

- Step 1: To create a service, we need to make use of the command line.

**ng g service employee**

- Step 2: Before creating a new service, we need to include the service created in the main parent **app.module.ts**.

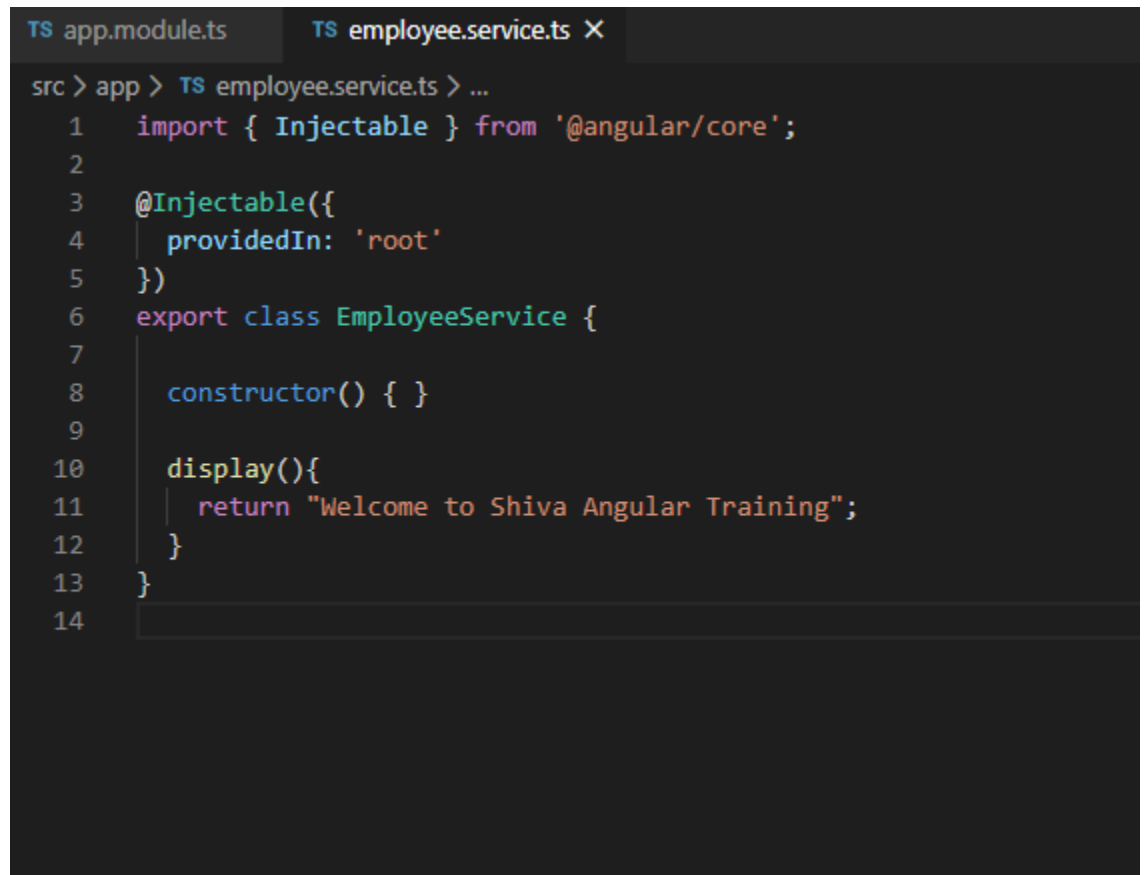


```
TS app.module.ts X
src > app > TS app.module.ts > ...
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { EmployeeService } from './employee.service';
7
8  @NgModule({
9    declarations: [
10     AppComponent
11   ],
12   imports: [
13     BrowserModule,
14     AppRoutingModule
15   ],
16   providers: [EmployeeService],
17   bootstrap: [AppComponent]
18 })
19 export class AppModule { }
20 |
```

We have imported the Service with the class name and the same class is used in the providers.

- Step 3: Create a function in service class display() as below:-

```
display(){  
    return "Welcome to Shiva Angular Training";  
}
```



The screenshot shows a code editor with two tabs: 'TS app.module.ts' and 'TS employee.service.ts X'. The active tab is 'TS employee.service.ts X', which contains the following TypeScript code:

```
src > app > TS employee.service.ts > ...  
1  import { Injectable } from '@angular/core';  
2  
3  @Injectable({  
4    providedIn: 'root'  
5  })  
6  export class EmployeeService {  
7  
8    constructor() { }  
9  
10   display(){  
11     return "Welcome to Shiva Angular Training";  
12   }  
13 }  
14
```

- Step 4 : Now create two components with name 'first' and 'second' in the project.

**ng g component first**

**ng g component second**

- Step 5: To use the service in the new component created. In component, inject service into a property by using a private

parameter in constructor which is of service type. The **ngOnInit** function gets called by default in any component created. 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.

```
export class FirstComponent implements OnInit {  
  
  constructor(private _newMyService :EmployeeService) {  
  }  
  text:string;  
  
  ngOnInit(): void {  
    this.text=this._newMyService.display();  
  }  
  
  }
```

```

TS app.module.ts  TS employee.service.ts  TS first.component.ts X
src > app > first > TS first.component.ts > ...
1  import { Component, OnInit } from '@angular/core';
2  import { EmployeeService } from '../employee.service';
3
4  @Component({
5      selector: 'app-first',
6      templateUrl: './first.component.html',
7      styleUrls: ['./first.component.css']
8  })
9  export class FirstComponent implements OnInit {
10
11      constructor(private _newMyService :EmployeeService) { }
12      text:string;
13
14      ngOnInit(): void {
15          this.text=this._newMyService.display();
16      }
17
18  }
19

```

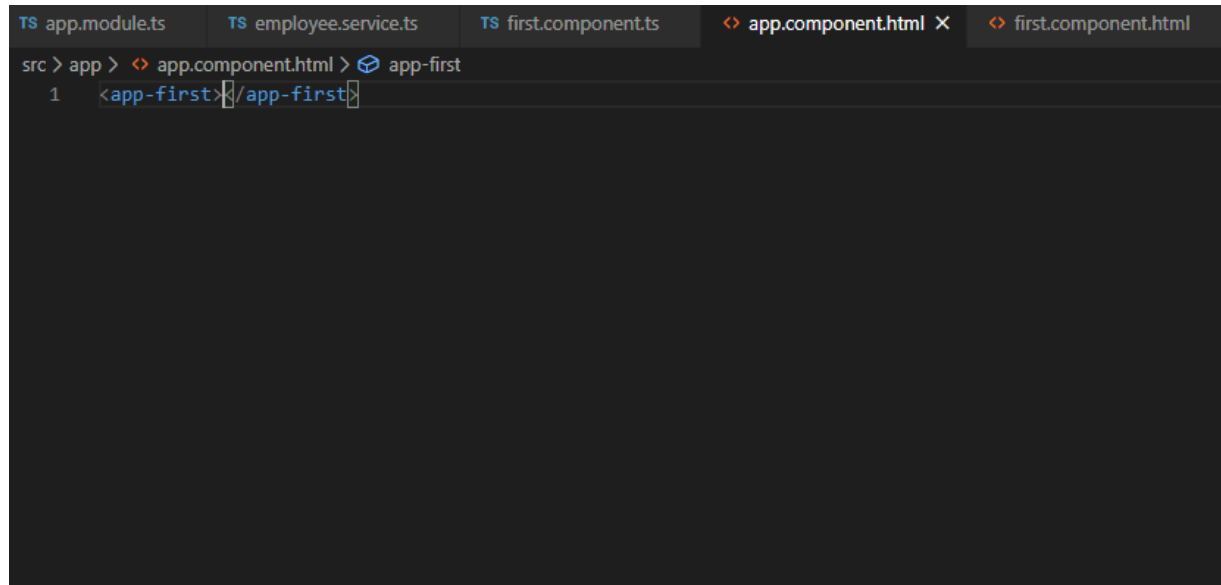
➤ Step 6: The text is displayed in the component html as follows –

```

TS app.module.ts  TS employee.service.ts  TS first.component.ts  <> first.component.html X
src > app > first > <> first.component.html > h1
1  <h1 style="color: green;">{{text}}</h1>

```

- Step 7: The selector of the new component is used in the app.component.html file.



The screenshot shows a code editor with several tabs at the top: 'app.module.ts', 'employee.service.ts', 'first.component.ts', 'app.component.html', and 'first.component.html'. The 'app.component.html' tab is active. The editor content shows the file path 'src > app > app.component.html' and a search bar with 'app-first'. Below the search bar, line 1 contains the HTML element '<app-first> /app-first'.

- Step 8: Next, navigate to your project's folder and run the local development server using the following command:

**ng serve**

- A local development server will start listening on the <http://localhost:4200/> address. Open your web browser and navigate to the <http://localhost:4200/> address to see your app up and running. The contents from the above html file will be displayed in the browser as shown below –

## Welcome to Shiva Angular Training

- In the second component, the service can be accessed using the similar steps

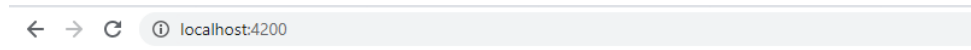
```
src > app > second > TS second.component.ts > SecondComponent
1  import { Component, OnInit } from '@angular/core';
2  import { EmployeeService } from '../employee.service';
3
4  @Component({
5    selector: 'app-second',
6    templateUrl: './second.component.html',
7    styleUrls: ['./second.component.css']
8  })
9  export class SecondComponent implements OnInit {
10
11    constructor(private _newMyService :EmployeeService) { }
12    text:string;
13    ngOnInit(): void {
14      this.text=this._newMyService.display();
15    }
16
17  }
18
```

```
src > app > second > <> second.component.html > ...  
1   <h1 style="color: red;">{{text}}</h1>  
2
```

```
src > app > <> app.component.html > app-second  
1   <app-first></app-first>  
2   <app-second></app-second>
```

- The contents from the above html file will be displayed in the browser as shown below –





## Example 2

Use the employee data specified as an array in the service class in the components.

- Step 1: To create a service, we need to make use of the command line.

**ng g service employee**

- Step 2: Before creating a new service, we need to include the service created in the main parent **app.module.ts**.

```

src > app > TS app.module.ts > AppModule
1  import { BrowserModule } from '@angular/platform-browser';
2  import { NgModule } from '@angular/core';
3
4  import { AppRoutingModule } from './app-routing.module';
5  import { AppComponent } from './app.component';
6  import { EmployeeService } from './employee.service';
7  import { FirstComponent } from './first/first.component';
8  import { SecondComponent } from './second/second.component';
9
10 @NgModule({
11   declarations: [
12     AppComponent,
13     FirstComponent,
14     SecondComponent
15   ],
16   imports: [
17     BrowserModule,
18     AppRoutingModule
19   ],
20   providers: [EmployeeService],
21   bootstrap: [AppComponent]
22 })
23 export class AppModule { }
24

```

- Step 3: Create an array of employee with employee data in service class as below:-

```

export class EmployeeService {
  employees :any[];

  constructor() {
    this.employees=[{
      empid:`100`,ename:`siva`
    },

```

```

{
  empid:`101`,ename:`sirisha`
},{
  empid:`102`,ename:`rani`
},
]
}
}

```

```

src > app > TS employee.service.ts > EmployeeService > constructor > ename
1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class EmployeeService {
7    employees :any[];
8
9    constructor() {
10     this.employees=[{
11       empid:`100`,ename:`siva`
12     },
13     {
14       empid:`101`,ename:`sirisha`
15     },{
16       empid:`102`,ename:`rani`
17     },
18
19
20   ]
21
22   }
23
24
25 }
26

```

- Step 4 : Now create two components with name 'first' and 'second' in the project.

**ng g component first**

**ng g component second**

- 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.

```
export class FirstComponent implements OnInit {  
  
    constructor(private _newMyService :EmployeeService  
    ) { }  
    employees:any[];  
  
    ngOnInit(): void {  
        this.employees=this._newMyService.employees;  
    }  
  
}
```

```

src > app > first > TS first.component.ts > FirstComponent
1  import { Component, OnInit } from '@angular/core';
2  import { EmployeeService } from '../employee.service';
3
4  @Component({
5      selector: 'app-first',
6      templateUrl: './first.component.html',
7      styleUrls: ['./first.component.css']
8  })
9  export class FirstComponent implements OnInit {
10
11      constructor(private _newMyService :EmployeeService) { }
12      employees:any[];
13
14      ngOnInit(): void {
15          this.employees=this._newMyService.employees;
16      }
17
18  }
19

```

- Step 5: To display the data in firstcomponent.html

```

<table border="1">
  <tr><th>Employee Id</th><th>Employee Name</th></tr>
  <tr *ngFor="let em of employees">
    <td>
      {{em.empid}}
    </td>
    <td>
      {{em.ename}}
    </td>
  </tr>
</table>

```

</tr>  
</table>

```
src > app > first > <> first.component.html > table
1  <table border="1">
2    <tr><th>Employee Id</th><th>Employee Name</th></tr>
3    <tr *ngFor="let em of employees">
4      <td>
5        {{em.empid}}
6      </td>
7      <td>
8        {{em.ename}}
9      </td>
10   </tr>
11 </table>
```

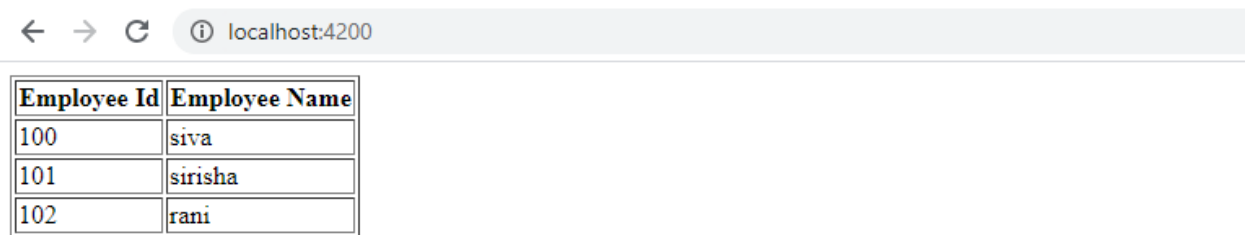
- Step 7: The selector of the new component is used in the app.component.html file.

```
src > app > <> app.component.html > ...
1  <app-first></app-first>
2
```

- Step 8: Next, navigate to your project's folder and run the local development server using the following command:

**ng serve**

- A local development server will start listening on the `http://localhost:4200/` address. Open your web browser and navigate to the `http://localhost:4200/` address to see your app up and running. The contents from the above html file will be displayed in the browser as shown below –



The screenshot shows a web browser window with the address bar displaying 'localhost:4200'. The main content area contains a table with two columns: 'Employee Id' and 'Employee Name'. The table has three rows of data.

| Employee Id | Employee Name |
|-------------|---------------|
| 100         | siva          |
| 101         | sirisha       |
| 102         | rani          |