Learning and Knowledge Management

ANGULAR

Services and Dependency Injection

Ground Rules

For a successful class, please:

- Arrive on time.
- Turn off cell phones.
- Assist your colleagues; show respect to all individuals regardless of their skill and knowledge level.
- Do not use class time to surf the net, check e-mail, or use instant messaging.
- Adhere to attendance policy as directed by your local training coordinator.
- Make the most of face-to-face training; ask questions and share your own insights and expertise.
- Leave the room only during break times or in the event of an emergency.
- Manage expectations for your other responsibilities.

Module Objectives

At the end of this module, you should be able to:

- Understand what are services
- What is the need for the service
- Understand what are the uses for services
- What is Dependency Injection
- Understand on how to create services
- Injecting services
- Understand on the Injector hierarchy

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Services Overview

Services Overview

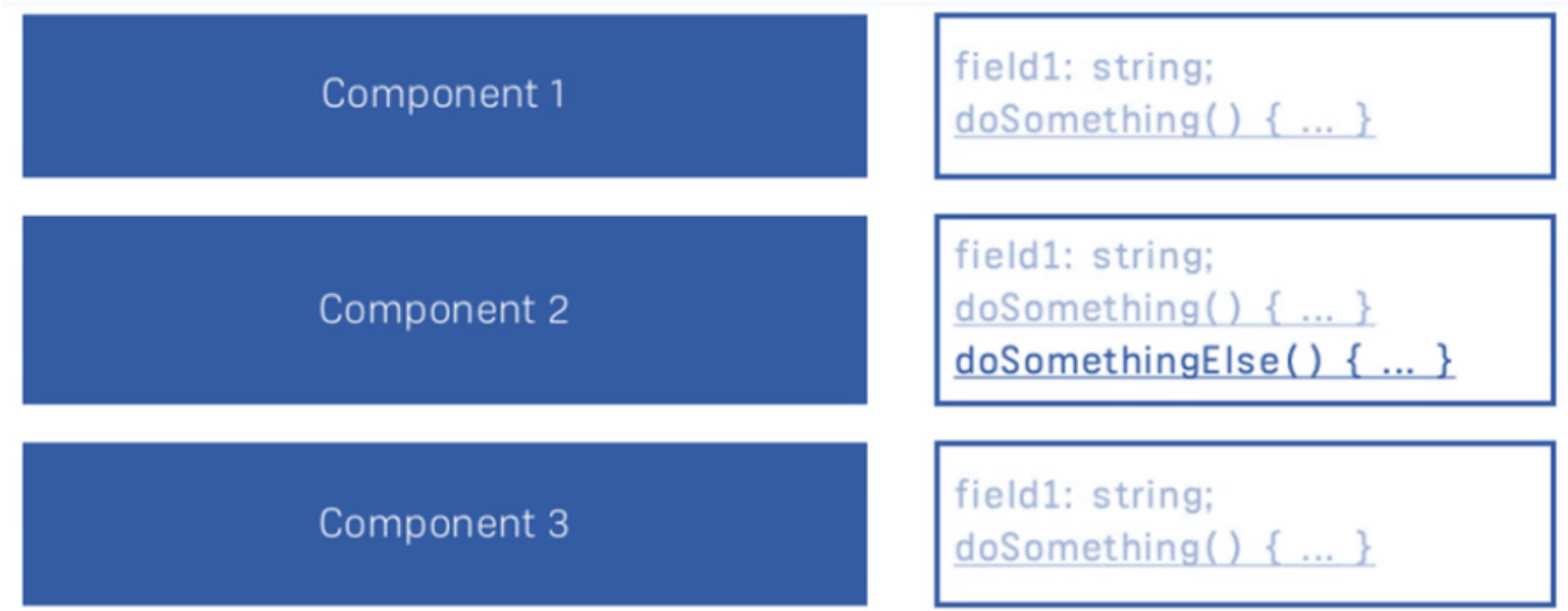
What is service?

- A service in Angular is
  - A Typescript class
  - Encapsulates various methods to complete a task
  - Used to reuse data or logic across multiple components.
  - Used to either send or receive the data from and to a server
  - Act as a central repository to share the common code across the application
  - Can depend on other services which in turn might depend on Http Client service to fetch the data asynchronously

Services Overview

Without Services

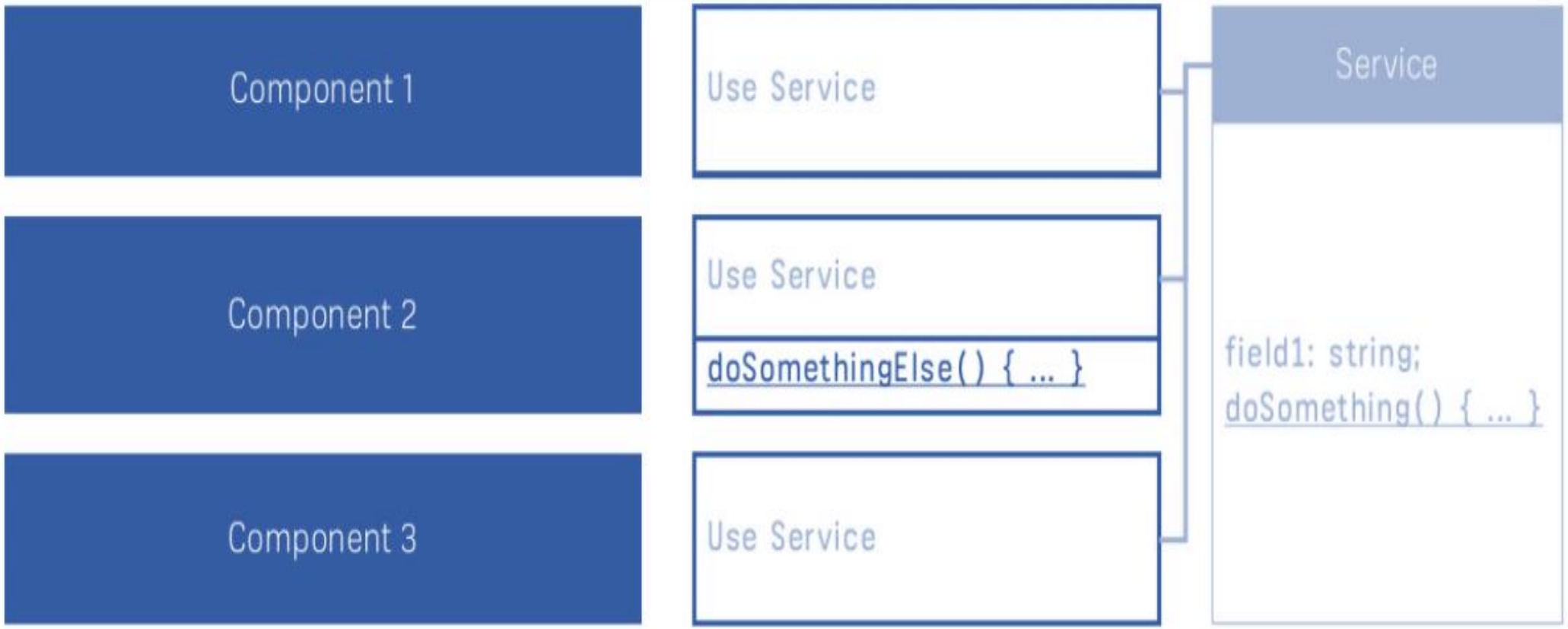
- Without services, logic or data-access needs to be duplicated across multiple components



Services Overview

With Services

- With Services, The Data and logic to access that data is implemented only once in a service, and that service can be used and reused by any/all the components in the application



Services Overview

What is the need for a service?

- Without services, logic or data-access may be duplicated across multiple components.

- With Services, The Data and logic to access that data is implemented only once in a service, and that service can be used and reused by any/all the components in the application.

Services Overview

Uses of Services

- Angular services are majorly used as
  - Promotes code reusability
  - Helps in sharing common data across multiple components
  - The business logic of the application can be separated from the rendering logic in the components
  - Provides ease in testing and debugging

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Dependency Injection

Dependency Injection

What is Dependency Injection?

- Dependency Injection (DI) is an application design pattern
- The DI framework is used in the Angular application design to increase the modularity and efficiency
- Angular services are majorly used as
  - Promotes code reusability
  - Helps in sharing common data across multiple components
  - The business logic of the application can be separated from the rendering logic in the components
  - Provides ease in testing and debugging

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Create and Inject Services

Create and Inject Services

Step 1: Create a wrapper interface to maintain Product structure

- Create a new folder Products under src/app
- Create a class Product.ts under Products folder.
- This is used as a Product type to capture the data

- **Product.ts**

```
export interface Product {
   ProductId: number,
   ProductName : string,
   ProductPrice : number
}
```

Create and Inject Services

Step 2: Create a wrapper interface to maintain Product details

- Create ProductsData.ts class under Products folder to maintain the data for the products
- This way, now the Products data can be consumed by any component or a service in the application

•ProductsData.ts file

```
import { Product } from './product';

export const Products : Product[]  =
[
   {"ProductId":1231,"ProductName":"IPad","ProductPrice":12345.11},
   {"ProductId":1232,"ProductName":"IPhone","ProductPrice":34512.22},
   {"ProductId":1233,"ProductName":"IPod","ProductPrice":12343.33},
   {"ProductId":1234,"ProductName":"Moto G","ProductPrice":12344.44},
   {"ProductId":1235,"ProductName":"Moto X Play","ProductPrice":12345.55},
   {"ProductId":1236,"ProductName":"Moto GS5","ProductPrice":12346.66},
   {"ProductId":1237,"ProductName":"Moto GS5 Plus","ProductPrice":12347.77},
   {"ProductId":1238,"ProductName":"Samsung S8","ProductPrice":12348.88},
   {"ProductId":1239,"ProductName":"OnePlusOne","ProductPrice":12349.99},
   {"ProductId":1240,"ProductName":"Vivo Nova","ProductPrice":12340.45}
]
```

Create and Inject Services

Step 3: Create a service in Angular

- To create a new service in Angular, use the command

  - **ng generate service <service_name>**

- @Injectable decorator is used to define a class as service
- This decorator is used to provide the metadata to inject into a component as a dependency
- @Injectable decorator contains a property called **'providedIn'**
- The Angular CLI registers a provider with the root injector
- **-@Injectable{**

  providedIn:'root',

  })

- Create ProductService using the command

  - ng generate service product

- Add a service method to return the Products

Create and Inject Services

Step 3: ProductService.ts file

```
import { Injectable } from '@angular/core';
import { Products } from './products/productsData';

@Injectable({
  providedIn: 'root'
})
export class ProductService {

  constructor() { }

  getProducts(){
    return Products;
  }
}
```

Corresponding import statement

Service method getProducts()

Returns Products data from the import statement

Create and Inject Services

Step 4: Update the app.module.ts file with service changes

- Make the service available to the entire application
- Import the service in app.module.ts file with an import statement
- Inject the **ProductService** in the **'Providers'** array
-

Import the ProductService

Inject the ProductService in the providers array

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { ProductComponent } from './product/product.component';
Import { ProductService } from './product.service';

@NgModule({
  declarations: [
    AppComponent,
    ProductComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule
  ],
  providers: [ProductService],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Create and Inject Services

Step 5: Consume the service in the component

- Let’s consume the service in the ProductComponent
- Inject the ProductService at the component constructor by creating an instance of ProductService
- Create a products variable of Product array type to store the result
- Invoke the service method in the
- ngOnInit() hook
-

```
import { Component, OnInit } from '@angular/core';
import { ProductService } from '../product.service';
import { Product } from '../products/product';

@Component({
 selector: 'app-product',
 templateUrl: './product.component.html',
 styleUrls: ['./product.component.css']
})
export class ProductComponent implements OnInit {

 products: Product[];
 constructor(private productService: ProductService) { }

 ngOnInit(): void {
  this.products = this.productService.getProducts();
 }
}
```

Create products variable of Products() type

Inject ProductService at constructor

Invoke the service method for the initialization

Create and Inject Services

Step 5: Consume the service in the component

- Display the output in the Product.Component.html file

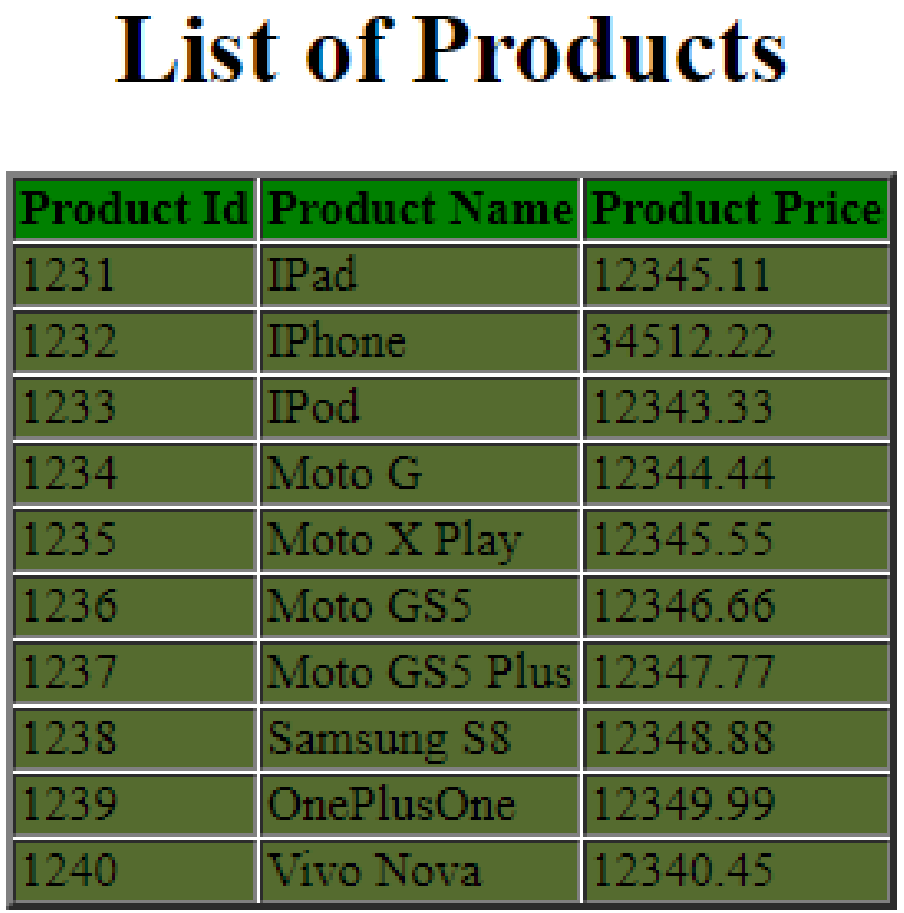
```
<h1 style="text-align: center;" >List of Products</h1>
<div align="center">
   <table border="2">
      <thead style="background-color: green;">
         <tr>
            <th>Product Id</th>
            <th>Product Name</th>
            <th>Product Price</th>
         </tr>
      </thead>
      <tbody style="background-color:darkolivegreen;">
         <tr *ngFor="let product of products">
            <td >{{product.ProductId}}</td>
            <td >{{product.ProductName}}</td>
            <td >{{product.ProductPrice}}</td>
         </tr>
      </tbody>
   </table>
</div>
```

Create and Inject Services

OUTPUT

-

-



List of Products

Product Id	Product Name	Product Price
1231	IPad	12345.11
1232	IPhone	34512.22
1233	IPod	12343.33
1234	Moto G	12344.44
1235	Moto X Play	12345.55
1236	Moto GS5	12346.66
1237	Moto GS5 Plus	12347.77
1238	Samsung S8	12348.88
1239	OnePlusOne	12349.99
1240	Vivo Nova	12340.45

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Injector Hierarchy and Services

Injector Hierarchy and Services

Step 3: Create a service in Angular

- Dependency Injection in Angular performs hierarchical injection.
- It has different scenarios .
    - Service at App Module : same instance of the service is available across the application
    - Service at App Component : same instance of the service is available to App Component along with its child components
    - Service at Other component : the injection is similar to the service injection at App Component
- This dependency will be resolved by the DI framework whenever a service class is instantiated
- By default, the DI searched for a provider starting from component’s local injector and then bubble up till it reaches the root injector

Topic List

#No	Module Topics
1	Services Overview
2	Dependency Injection
3	Create and Inject Service
4	Injector Hierarchy and Services
5	Activity on Services

Activity on Services

Activity

Complete an activity on Services

- **Refer the Hands On Workbook :**

  LKM_MEAN_Angular_Services_HONWorkbook.docx

Reference

Heading	Description
Services	Link

Module Summary

Now, you should be able to:

- Understand what are services
- What is the need for the service
- Understand what are the uses for services
- What is Dependency Injection
- Understand on how to create services
- Injecting services
- Understand on the Injector hierarchy

Thank You