SERVICES AND DEPENDENCY INJECTION

A service is a class with specific purpose. Service helps achieve the following principles.

1. DRY- Do not Repeat Yourself. We should not make copies of same. For components needing the same data, it can be given in services and accessed by calling the service in the components.

2. Single Responsibility Principle- Components responsibility is to control views logic so we should not create data in it.

## Why Services?

* To share data among multiple components.
* To give application logic and this logic should be independent of individual components.
* For external interaction like connecting to a database.

# Dependency Injection

### DI as a framework

Injector is a container of all dependencies. We register all the dependencies with the injector.

### Steps in DI.

1. Define the service class.
2. Register with injector
3. Declare as dependency in the required classes.

### Hierarchial dependency in Angular

App Module

App Component

Component 1

Component 2

Using provider we can create an instance of the service.

App Module: By giving it in the App Module, same instance of the service is available Application-wide.

App Component: Same instance of service is available for all Components (but not for other services).

Components: Same instance of service is available for the particular Component and its child Components

# Injectable Decorator

Metadata present inside service class which tells Angular this service in itself have injected dependencies. If a service is injected into another service injectable decorator must be used.

@Injectable() is not there in components having dependencies because it has @Component() which tells Angular the might have dependencies.