There is ***no separate decorator*** to indicate that a class is a service class. We just define a class and when the data is needed, that class will be instantiated and the object will be used to invoke the functions. We can do this by explicitly creating an instance using the *new*keyword but it is better to use the ***dependency injection*** feature of Angular.

For Example:

***book.service.ts***

1. import {Book} from './book'; *// Importing the Book class type.*
2. export class BookService {
3. constructor() { }
4. getAllBooks():Book[] { *// Returns an Array of types*
5. return [
6. {"id": 1, "name": "HTML 5" },
7. {"id": 2, "name": "CSS 3" },
8. {"id": 3, "name": "Java Script" },
9. {"id": 4, "name": "Ajax Programming" }
10. ];
11. }
12. }

Let us see how are we going to use this in our *BookComponent*present in ***book.component.ts***

1. import { Component, OnInit } from '@angular/core';
2. import { BookService } from './book.service';
3. import { Book } from './book';
4. @Component({
5. selector: 'app-book',
6. template: `<div \*ngIf="bookListFromService">
7. <table>
8. <tr \*ngFor="let book of bookListFromService">
9. <th>{{book.id}}: </th>
10. <td>{{book.name}}</td>
11. </tr>
12. </table>
13. </div>`
14. })
15. export class BookComponent implements OnInit {
16. bookListFromService:Book[]; *// Will store the bookList to be displayed*
17. bookService:BookService; *// A variable of BookService type, will be used to populate bookList*
18. constructor() { }
19. ngOnInit() {
20. this.bookService = new BookService(); *// Creation of BookService instance using new keyword*
21. this.bookListFromService = this.bookService.getAllBooks(); *// Invoking the getAllBooks() of bookService to get data*
22. }
23. }

The **advantage**of using a service to get all details is that *the component doesn’t have to worry about how to fetch the data*. It will always invoke the *getAllBooks()* method to get the data.

When the source of data changes, it will be the responsibility of the *getAllBooks()*method to know how to fetch the data from the new source.

**The Problem**

**Observe** that, in line 24, how this service is being created. We are using the ***new***keyword to create the object of BookService i.e, *this.bookService = new BookService();*

This means as a **developer** we will be *creating and instantiating* services. This method poses a lot of work on the developer's part while maintaining Angular code because:

* The BookComponent is *tightly coupled* with BookService. This means that if tomorrow *BookService*needs to be changed somehow, for example some change in the constructor() signature, etc, the component code also needs to be modified to suit the change in service.
* Unit testing a component will become difficult since we will have to change the service class with a mock-service object, by manually changing the code every time we want to unit test the app.