



# Angular Services

---

.NET

*Services are a great way to share information among classes that don't know each other and to make requests to an API. Components should never fetch or save data directly.*

[HTTPS://ANGULAR.IO/TUTORIAL/TOH-PT4#WHY-SERVICES](https://angular.io/tutorial/toh-pt4#why-services)

# Dependency Injection – Services and Injectables

<https://angular.io/guide/glossary#dependency-injection-di>  
<https://angular.io/guide/dependency-injection>

---

**Components** should always delegate data access to a **Service**. A **Service** can get data from an API web service, local storage, a mock data source, and more.

**Services** are integral to Angular. A **service** is an instance of a class that you can make available to any part of your application using **Angular's Dependency Injection** system.

A **Service** is your portal to persist data and have methods to access that data.

The **@Injectable()** decorator accepts a metadata object for the service, the same way the **@Component()** decorator does for component classes.

```
1  import { Injectable } from
2  import { Hero } from './hero
3  import { HEROES } from './mo
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class HeroService {
9
10     0 references
11     getHeroes(): Hero[] {
12       return HEROES;
13     }
14
15     0 references
```

# Dependency Injection – Services and Injectables

<https://angular.io/tutorial/toh-pt4#provide-the-heroservice>

<https://angular.io/guide/dependency-injection>

<https://angular.io/guide/architecture-services>

---

Services must be registered with Angular's Dependency Injection system before they can be injected into a **Component**.

By default, the **Angular CLI** command **ng generate service** registers a **provider** with the **root** injector for your **Service** by including **provider** metadata that's **providedIn: 'root'** in the **@Injectable()** **decorator** of the **Service Component**.

When a **Service** is provided at the root level, Angular creates a single, shared instance of the **Service** and injects it into any class that asks for it.

Angular will also remove any unused **Services**.

```
1  import { Injectable } from
2  import { Hero } from './hero
3  import { HEROES } from './me
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class HeroService {
9
10     0 references
11     getHeroes(): Hero[] {
12       return HEROES;
13     }
14
15     0 references
```

# Angular – Use DI to Get a Service

<https://angular.io/tutorial/toh-pt4>

---

## Create a service:

1. Create a **Service**:
  - `ng generate service <ServiceName>`.
2. Import the **Injectable** symbol into the **Service Component**  
To allow the **Service** to be injected into **Components**:
  - `import { Injectable } from '@angular/core';`
3. Import the **Service** into the **Component** where it will be used:
  - `import { ServiceName } from '../relative/location';`
4. Inject the **Service** into the constructor of the **Component** where it will be used:
  - `constructor(private ServiceVariableName: ServiceName) {}`.

Use `ngOnInit()` to access and retrieve data from a service on instantiation of the **Component** instead of using the constructor.

```
2 import { Hero } from '../hero';
3 import { HeroService } from '../hero.service';
```

```
0 references | 1 reference
constructor(private heroService: HeroService) {}

1 reference
getHeroes(): void {
  this.heroes = this.heroService.getHeroes();
}

6 references
ngOnInit(): void {
  this.getHeroes();
}
```