

This cheat sheet is for the course [Learn C# Full Stack Development with Angular and ASP.NET](#) by Jannick Leismann.

ANGULAR SERVICE

The goal of **angular services** is to allow various application components to share functionality and data. They are singleton objects.

Encapsulation

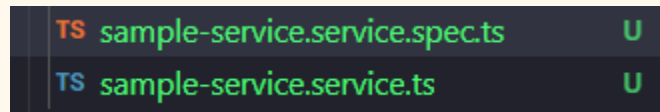
They aid in the encapsulation of data access, business logic, and other non-view-related tasks.

Reusability

Services facilitate code reuse by giving components a common location to share code.

Creating a Service

Use the Angular CLI to generate a service: **ng generate service service-name** or **ng g s service-name**.



```
ng generate service sample-service
TS sample-service.service.spec.ts  U
TS sample-service.service.ts       U
```

sample-service.service.ts

```
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})

export class SampleServiceService {

  constructor() { }

}
```

A class can be marked as a service that is ready for injection by using the **@Injectable** decorator.

providedIn: 'root' indicates that the service is accessible from anywhere in the application and doesn't require explicit provisioning in any module.

Dependency Injection

```
import { Component } from '@angular/core';

import { SampleService } from '../sample.service';

@Component({
  selector: 'app-example',
  templateUrl: './example.component.html',
  styleUrls: ['./example.component.css']
})

export class ExampleComponent {
  constructor(private sampleService: SampleService) { }
}
```

The dependency injection mechanism built into Angular enables using services within components or other services.

Usage

```
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})

export class SampleServiceService {
  getMessage(): string {
    return 'Hello from SampleService!';
  }
}
```

```

import { Component } from '@angular/core';
import { SampleService } from '../sample.service';

@Component({
  selector: 'app-example',
  template: '<p>{{ message }}</p>',
})
export class SampleComponent {
  message: string;

  constructor(private sampleService: SampleService) {
    this.message = this.sampleService.getMessage();
  }
}

```

Once a service is **injected** into a **component**, you can use its **methods** and **properties**.