Cheat Sheet – Services

Idea behind Services

Services can be used to centralize certain tasks, remove redundant code duplication or outsource "heavy" tasks.
Services are "normal" classes and don't receive any special metadata to "become services".

Services & Dependency Injection

In order to use services, they need to be injected into the components/ classes which should use them.
This is done by specifying them in the class constructor like this:

```
constructor(myService: MyService)
```

Angular 2's dependency injection framework (kind of a framework in a framework) takes care about the creation of the required instance(s).

In order to be able to create these instances, Angular 2 needs to know how to create such an instance. This is "explained" by specifying **providers**.
There are two easy ways to set up providers:

1. In the @Component/ @Directive metadata:

```
@Component({
    // …
    providers: [MyService]
})
```

2. In the providers array of the AppModule (to provide them Application-wide)

```
providers: [MyService]
```

Hierarchical Injector

Angular 2's Dependency Injector is hierarchical. This basically means, that providers may be set up on different "levels" of the application – leading to different outcomes.
All child components of a component will receive the same service instance, if the provider is set up on the parent component.
Vice versa, if providers are specified on each individual child component, different instances will be created.
It depends on your needs which behavior you want.
The highest possible provider level, is the providers array in the AppModule.