Servisi

1. Uvod

Do sada smo podatke čuvali u promenljivama u sklopu Angular komponenti. Međutim, svrha komponenti je prikaz podataka. U ovoj lekciji ćemo uvesti pojam Angular servisa. Servisi treba da brinu o podacima u aplikaciji, tako da ćemo u njih izmestiti čuvanje podataka.

Za povezivanje komponente sa servisom ćemo koristiti tzv. injektiranje zavisnosti (dependency injection).

2. Servisi

U uvodu smo već napomenuli kako je svrha komponenti prikaz podataka. Operacije nad podacima ćemo zbog toga izdvojiti i implementirati putem servisa.

Servis se implementira u zasebnom fajlu i objedinjuje sve funkcionalnosti vezane za operacije nad podacima (učitavanje, pretraga, izmena, brisanje,...).

Servis se povezuje sa komponentom tzv. injektiranjem zavisnosti. Kada povežemo komponentu sa servisom, šablon će i dalje moći da se poveže (binding) sa podacima i svi podaci kojima raspolaže servis će biti ažurno prikazani na web stranici.

2.1 Implementacija Angular servisa

Servis je predstavljen dekoratorom i klasom (slično kao i komponenta). Dekorator @Injectable omogućava komponentama da zatraže injektiranje zavisnosti ka ovom servisu. U servisu omogućavamo pristup podacima, za šta ćemo i ovog puta iskoristiti listu objekata.

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

@Injectable()
export class CarService {
  cars: Car[] = [
    new Car('Red', 'Ford', 'Mustang');
    new Car('Black, 'Ford', 'Fiesta);
  ]
}
```

Nakon dodavanja servisa u Angular projekat, potrebno ga je povezati sa komponentom. Za ovo se koristi koncept dependency injection.

3. Dependency Injection

Dependency injection (injektiranje zavisnosti) je koncept kojim se olakšava modularizacija aplikacije. Injektiranjem zavisnosti su češće korišćeni elementi stavljeni na raspolaganje komponentama aplikacije.

Da bi komponenta koristila te druge elemente (npr. servise), potrebno je u njenom konstruktoru upotrebiti tipove tih elemenata za definisanje referenci. Na taj način će, prilikom instanciranja, komponenta zatražiti ove zavisnosti i potom sačuvati njihove reference za kasnije korišćenje.

Da bi se injektirala zavisnost, potrebno je u komponenti:

- importovati klasu iz odgovarajućeg fajla u kome je servis implementiran
- putem konstruktora zatražiti injektiranje zavisnosti ka servisu i referencu sačuvati u promenljivoj service

Potom se podaci kojima raspolaže servis mogu koristiti i u šablonu preko reference na servis i naziva promenljive sa podacima.

cars.servise.ts

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

@Injectable()
export class CarService {
   cars: Car[] = [
      new Car('Red', 'Ford', 'Mustang');
      new Car('Black, 'Ford', 'Fiesta);
   ]
}
```

carshop.component.ts

```
import { CarService } from './cars.service';

@Component({
   selector: 'carshop',
   template: './carshop.component.html',
   styleUrls: ['./carshop.component.css']
})
export class CarShopComponent {
   constructor(private service: CarService) {}
}
```

carshop.component.html

```
<ul>
   <li *ngFor="let car of service.cars">
      {{ car.model }}
   </li>
</ul>
```

Zadaci

Alat koji će se koristiti za pravljenje Angular aplikacija na vežbama je https://stackblitz.com/

1. Zaposleni je opisan imenom, prezimenom i radnim mestom. Kreirati servis i u njemu formirati listu sa podacima za troje zaposlenih. U komponentu injektovati zavisnost ka servisu i omogućiti ispis podataka o zaposlenima.

Domaći

1. Bioskopska projekcija je opisana nazivom filma, oznakom sale, početkom i trajanjem. Kreirati servis i u njemu formirati listu sa podacima za 5 projekcija. U komponentu injektovati zavisnost ka servisu i omogućiti ispis podataka o projekcijama.

Literatura

1. https://angular.io/
2. https://material.angular.io/
3. Materijali sa predavanja