Angular Rutiranje

Autori:
Milan Segedinac
Goran Savić

1. Rutiranje u veb aplikacijama

U veb alikacijama, *rutiranjem* se postiže da se aplikacija podeli na sekcije koje će biti prikazivane u zavisnosti od trenutnog URL-a u pregledaču. Ukoliko URL identifikuje stranicu na serveru, rutiranje uključuje slanje get zahteva za zadati URL i učitavanje i prikaz dobavljene stranice. Međutim, u jednostraničnim aplikacijama, HTML stranica se učitava samo jednom, a rutiranje podrazumeva logičku izmenu prikaza u aplikaciji.

Rutiranje je veoma važan aspekt razvoja aplikacija jer razdvaja sekcije aplikacije, pojednostavljuje održavanje stanje aplikacije i omogućuje da se sekcije apliakcije zaštite od neautorizovanog pristupa odgovarajućim pravilima rutiranja.

Rutiranje u Angular aplikacijama

U Angular aplikacijama rutiranje se realizuje zadavanjem *putanja* (eng. *paths*) za Angular komponente. Pošto će se rutiranje dešavati na klijentskoj strani, potrebno je konfigurisati Spring backend aplikaciju da, ukoliko se zahteva resurs koji ne postoji, bude izvršena redirekcija na index.html strnicu. Kod kojim se to postiže dat je listingom ispod.

```
@Bean

ErrorViewResolver  supportPathBasedLocationStrategyWithoutHashes()
{
  return new ErrorViewResolver() {

    @Override
          public  ModelAndView  resolveErrorView(HttpServletRequest
request,
      HttpStatus status, Map<String, Object> model) {
      return status == HttpStatus.NOT_FOUND
          ? new ModelAndView("index.html", Collections.<String,
            Object>emptyMap(), HttpStatus.OK) : null;
    }
  };
}
```

Deklarisanje ruta

U Angular aplikaciji korenska putanja ruta zadaje se postavljanjem base elementa index.html strance. U našoj aplikaciji postavljeno je `<base href="/">` što znači da će korenska ruta biti `"/"`.

U Angularu rutiranje je opcioni servis i nije deo Angular Core. Stoga je neophodno importovati odgovarajuće objekte (`Routes` i `RouterModule`) iz `'@angular/router'` u app.module.ts.

`Routes` objekat služi za predstavljanje pravila rutiranja: govori pozivom koje putanje će biti prikazana koja komponenta. Stoga je `Routes` objekat lista objekata koji sadrže atribut route i component. Objekat tipa `Routes` je prikazan listingom ispod. Vidimo da je

putanja 'main' ima postavljenu komponentu MainComponent, odnosno da će na URL http://localhost:8080/main biti prikazana MainComponent komponenta.

Prilikom zadavanja rute moguće je zadati i parametre putanja. Ruta za putanju 'record/:id' omogućuje prosleđivanje parametra id komponenti RecordDetailsComponent. Ukoliko bi korisnik uneo url http://localhost:8080/record/5 otvorila bi se komponenta RecordDetailsComponent sa prosleđenim parametrom 5.

Prilikom zadavanja ruta moguće je realizovati i redirekciju. U tom slučaju se za zadatu putanju postavlja atribut redirectTo. U primeru u listingu ispod vidimo da će se, ukoliko korisnik unese samo http://localhost:8080/ izvšiti redirekcija na putanju http://localhost:8080/main , odnosno da će se otvoriti MainComponent.

Obratite pažnju da poslednja ruta ima putanju '**'. To je vrednost koja će odgovarati bilo kojoj putanji. Postavljanjem ove putanje dobili smo da će svaki url koji počinje sa http://localhost:8080/ koji nije poklopljen ni sa jednom unetom rutom u aplikaciji otvarati komponentu PageNotFoundComponent. Ovde vidimo i da se poklapanje ruta vrši tako što se prvo pokuša poklapanje prve rute, pa druge,... da bi se na kraju pokušalo poklapanje poslednje rute.

https://angular-2-training-book.rangle.io/handout/routing/routeparams.html

```
import { RouterModule, Routes } from '@angular/router';

...
const appRoutes: Routes = [
  { path: 'record/:id', component: RecordDetailsComponent },
  { path: 'main', component: MainComponent },
  { path: '', redirectTo: 'main', pathMatch: 'full' },
  { path: '**', component: PageNotFoundComponent }
];
...
@NgModule({
  ...
  imports: [
...
    RouterModule.forRoot(
      appRoutes,
      { enableTracing: true } // <-- debugging purposes only
    )
  ],
...
})
```

Link na rutu

Postavljanje linka u Angular aplikaciji moguće je realizovati postavljanjem običnog <a> elementa sa href atributom. Međutim, Angular uvodi direktivu [routerLink] koja omogućuje interno rutiranje u aplikaciji i koja pojednostavljaju rad sa rutama (npr. postavljanje parametra putanje). Primer korišćenja routerLink direktive dat je listingom ispod. Vidimo da routerLink

direktiva prima niz čiji je prvi element string `'/record'`, a drugi element `record.id`. Ukoliko je vrednost `record.id` na primer 5, ovaj link bi upućivao na `http://localhost:8080/record/5`.

```
<a [routerLink]="['/record',record.id]">{{record.title}}</a>
```

Pristup parametrima rute

Da bi rutiranje bilo kompletno realizovano, potrebno je omogućiti i pristup parametrima rute. U Angularu se za to koristi poseban servis, `ActivatedRoute`, koji se importuje iz modula `'@angular/router'`. Primer pristupa parametrima rute prikazan je listingom ispod.

U konstruktoru komponente injektujemo `ActivatedRoute` u privatni atribut `route`. Preko ovog atributa možemo da pristupimo pristupimo parametrima putanje, odnosno atributu `params`. Ovaj atribut je tipa `Observable`, što znači da možemo da možemo da napišemo kod koji će se pozvati svaki put kada se parametri rute promene, što je urađeno kodom **`this.route.params.subscribe`**. Reakcija na promenu parametara putanje je poziv GET metode za preuzimanje vrednosti, odnosno `this.http.get(`/api/records/${this.id}`)`.

```
Import { ActivatedRoute } from '@angular/router';

...

@Component({
  selector: 'app-record-details',
  templateUrl: './record-details.component.html',
  styleUrls: ['./record-details.component.css']
})
export class RecordDetailsComponent implements OnInit {

  id: number;

  private sub: any;

  record: Record;

  isDataAvailable: boolean;

   constructor(private route: ActivatedRoute, private http: Http) {
}

  ngOnInit() {
        this.sub = this.route.params.subscribe(params => {
        this.isDataAvailable = false;
        //unarni + konvertuje string u number
        this.id = +params['id'];
        this.http.get(`/api/records/${this.id}`).subscribe(
         (res: Response) => {
           this.record=res.json();
           this.isDataAvailable = true;
         }
       );
     });
  }
}
```