# ANGULAR 7

Peter Gurský, peter.gursky@upjs.sk

# Editácia používateľov

- Chceme vytvoriť komponent na editáciu a pridávanie používateľov
  - Na nových URL
    - /users/edit/132
    - /users/new
- Stačí jeden, budú sa líšiť iba tým, či na začiatku je model používateľa prázdny alebo vyplnený používateľ
- Ako si prečítať číslo z URL adresy?

# Parametre URL:

- ak chceme poslať povinný parameter napr. pre URL adresu http://localhost:4200/users/edit/25
  - <a [**routerLink**]="['/users', 'edit', 25]">
  - app.routes.ts doplníme
    - { path: '/users/edit/:id', component: UserEdit }
  - v user-edit.ts :

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

export class UserEdit implements OnInit {

   constructor(private route: ActivatedRoute){}

   ngOnInit() {
      this.userId.set(this.route.snapshot.params['id']);
   }
}
```

táto syntax je (zatiaľ) ok, ale iba ak sa tento komponent už nebude používať inde

# Parametre URL:

- ak sa komponent znovupoužíva (nerenderujeme ho celý), napr. ak z jedného používateľa (rovnaké routovacie pravidlo) sa vieme prekliknúť na iného, počúvame na zmeny parametra:

```
import { ActivatedRoute, ParamMap } from '@angular/router';

ngOnInit() {
  this.route.paramMap.pipe(
    map((params: ParamMap) => params.get('id'))
  ).subscribe(userId => this.userId.set(userId));
}
```

# Parametre URL:

- ak sa komponent znovupoužíva (nerenderujeme ho celý), napr. ak z jedného používateľa (rovnaké routovacie pravidlo) sa vieme prekliknúť na iného, počúvame na zmeny parametra:

```
import { ActivatedRoute, ParamMap } from '@angular/router';
import { switchMap } from 'rxjs/operators';

ngOnInit() {
  this.route.paramMap.pipe(
    switchMap((params: ParamMap) =>
      this.usersService.getUser(+params.get('id')))
  ).subscribe(user => this.user.set(user));
}
```

# Parametre URL:

- ak sa komponent znovupoužíva (nerenderujeme ho celý), napr. ak z jedného používateľa (rovnaké routovacie pravidlo) sa vieme prekliknúť na iného, počúvame na zmeny parametra:

```
import { ActivatedRoute, ParamMap } from '@angular/router';
import { switchMap } from 'rxjs/operators';

ngOnInit() {
  this.route.paramMap.pipe(
    switchMap((params: ParamMap) =>
      this.usersService.getUser(+params.get('id')))
  ).subscribe(user => this.user.set(user));
}
```

Pre každú pritečenú hodnotu vráti prúd, z ktorého hodnoty majú odchádzať cez výstup.
Ak sa prúd vytvorený predchádzajúcou hodnotu ešte neukončil, násilne ho ukončí a posiela už iba hodnoty z nového prúdu

# Parametre URL:

- ak sa komponent znovupoužíva (nerenderujeme ho celý), napr. ak z jedného používateľa (rovnaké routovacie pravidlo) sa vieme prekliknúť na iného, počúvame na zmeny parametra:

```
import { ActivatedRoute, ParamMap } from '@angular/router';
import { switchMap } from 'rxjs/operators';

ngOnInit() {
  this.route.paramMap.pipe(
    switchMap((params: ParamMap) =>
      this.usersService.getUser(+params.get('id')))
  ).subscribe(user => this.user.set(user));
}
```

okrem **get(parameter)**, môžeme použiť aj metódy:
-**has(parameter)** – true, ak taký parameter máme
-**getAll(parameter)** – ak máme pre paremeter viac hodnôt
-**keys()** – vráti názvy parametrov

# nepovinné parametre

- Niekedy chceme v URL poslať nepovinné parametre napr. pre paginácu
  - http://localhost:4200/users;page=2;count=10
  - v kóde:
    - this.router.navigate(['/users', { page: 2, count: 10 }]);

matrix notácia cez ; – nie cez ? a &

```
import { ActivatedRoute, ParamMap } from '@angular/router';
import { switchMap } from 'rxjs/operators';

ngOnInit() {
  this.route.paramMap.pipe(
    switchMap((params: ParamMap) =>
      this.page.set(params.has('page') ? +param.get('page') : 1));
      this.count.set(params.has('count') ? +param.get('count') : 5));
      return this.usersService.getUsers();
    ).subscribe(users => this.users.set(users));
}
```

# Niektoré ďalšie parametre ActivatedRoute

- url
  - observable s poľom reťazcov častí cesty z URL v tejto trase
- data
  - observable s dátami pre danú trasu, vrátane dát získaných neskôr resolverom
  - do každej trasy vieme dať parameter data, ktorý môže obsahovať čokoľvek
- queryParamMap
  - observable s parametrami pre všetky trasy
  - parametre v URL uvedené za otáznikom v tvare ?atr1=val1&atr2=val2
- fragment
  - observable s hodnotou fragmentu/časti stránky – v URL je to hodnota za #
  - referencuje element s daným atribútom id