ANGULAR FRONTEND FEJLESZTÉS

5. forduló

A kategória támogatója: TCS - TATA Consultancy Services

RENDELKEZÉSRE ÁLLÓ IDŐ:

15:00

Ismertető a feladathoz

Felhasznált idő: 00:57/15:00 **Elért pontszám:** 0/20

1. feladat 0/9 pont

Az NGRX alapú webappunkban Zergeket szeretnénk megjeleníteni. Egyszerre 10 objectet (egy oldal) kérünk el a szervertől, majd minden gombnyomásra a következő oldalnyit. A gombra való kattintáskor egy getZergsAction-t dispatch-elünk a következő oldal indexével, majd a getZergsSuccess action a kapott Zergeket hozzáfűzi a store-ban lévő listához. A lekérést az alábbi effect-el oldottuk meg, de azt tapasztaltuk, hogy a türelmetlen felhasználók rövid idő alatt többször kattintanak a gombra, és ilyenkor egyes oldalak nem jelennek meg:

```
public getZergs$ = createEffect(() =>
  this.actions$.pipe(
    ofType(getZergsAction),
    switchMap(action => this.dataSvc.getZergs({ page: action.page }).pipe(
      map(zergs => getZergsSuccessAction({ zergs })),
      catchError(error => of(getZergsFailAction({ error })))
    )),
  )
);
```

Hogyan oldhatnánk meg a problémát?

Válaszok

- [x] A gombot disabledre állítjuk a **getZergsAction** reducerében, majd enabled-re a **getZergsSuccessAction** és **getZergsFailAction** reducerében.
- [x] Az effectben a **switchMap**-et **concatMap**-re cseréljük
- [ ] Az effectben a **switchMap** után teszünk egy **concatAll()** operátort
- [ ] Az effectben az **ofType** után teszünk egy **delay(1000)** operátort

Magyarázat

A probléma az, hogy a switchMap leiratkozik a belső Observable-ről (this.dataService.getCars()) amikor egy új getCarsAction érkezik be (a nevéből adódóan switch-el, azaz átvált egy másik observable-re).

Mivel a belső observable egy szerver lekérdezés, nem garantálja semmi, hogy hamarabb fog emittálni, mint ahogy a felhasználó mégegyszer kattintana a gombra.

"A gombot disabledre állítjuk a **getCarsAction** reducerében, majd enabled-re a **getCarsSuccessAction** reducerében." válasz jó: Ha a gomb disabled amíg tart a szerverről a lekérdezés garantáltan nem érkezhet be új action amíg az előző nem "végzett"

"Az effectben a **switchMap**-et **concatMap**-re cseréljük" válasz jó: A concatMap-el elérjük, hogy a belső observable-ről ne iratkozzon le az effect, hanem hozzákonkatenálja a következő oldalhoz tartozó kérést. (bővebben https://blog.angular-university.io/rxjs-higher-order-mapping/)

Frissítés (2021.11.22.): "Az effectben a **switchMap**-et **exhaustMap-**re cseréljük" válaszlehetőséget versenyzői visszajelzések alapján töröltük, mivel a kérdés egyértelműbb lett volna, ha ott az action implementáció is. Ennek hiányában a kérdés többféleképp volt értelmezhező.

2. feladat 0/4 pont

Készítettünk egy service-t, amit az Angular DI segítségével injektáltunk a **TerranComponent** komponensünkbe, ami a **TerranModule** modulban található. A service-t singletonként kellene használnunk, de azt tapasztaltuk, hogy minden komponens instance egy saját service instance-t kap. Mit csináltunk?

Válasz

- ( ) A service class-ról teljesen kihagytuk az **@Injectable()** dekorátort
- ( ) Az alábbi módon dekoráltuk a service classt: **@Injectable({ providedIn: 'root' })**
- ( ) Az alábbi módon dekoráltuk a service classt: **@Injectable({ providedIn: TerranModule })**
- ( ) Az alábbi módon derokoráltuk a service classt: **@Injectable()**, és a **TerranModule** providers tömbjébe tettük
- ( ) Az alábbi módon derokoráltuk a service classt: **@Injectable()**, és az **AppModule** (a root modul) providers tömbjébe tettük
- ( ) Az alábbi módon derokoráltuk a service classt: **@Injectable()**, és a **TerranComponent** providers tömbjébe tettük

Magyarázat

Az @Injectable() dekorátorral ellátott class-ok (nevezzük service-nek) csak akkor viselkednek NEM singletonként, ha egy komponens providers tömbjébe tesszük őket.

Ekkor csak az adott komponens számára érhető el a service (és a gyerekeinek), és minden komponens instance-nek egy új service instance-e lesz. (ElementInjector - bővebben https://angular.io/guide/hierarchical-dependency-injection)

3. feladat 0/7 pont

Meg szeretnénk jelentíteni egy listát cicákról, amennyiben az **isLoading** változó értéke **false**. A lista elemein a cica sorszámát (tömb index) és a cica nevét szeretnénk mutatni. Mik a hibák az alábbi kódban?

```
<div [class.odd]="isOdd" *ngIf="!isLoading" *ngFor="let zerg of zergs$ | async; count as count; odd as
isOdd; trackBy: trackByFn">
  Zerg {{index + 1}} of {{count}} | Name: {{ zerg.name }}
</div>
```

Válaszok

- [ ] Nincs **count** property-je az ngFor direktívának
- [x] Nem lehet egy elemen két strukturális direktíva egyszerre, az ***ngIf**-et egy wrapper elemre kell tenni
- [x] Az első interpolációban az **index** csak akkor érhető el, ha az *ngFor-t kiegészítjük "**index as index**"-szel
- [ ] Az **isOdd** változó csak a div-en belül érhető el, a class binding-ban nem
- [ ] A direktíva neve helytelen, ***ngFor** helyett ***ngRepeat** a helyes

Magyarázat

Nem lehet egy elemen két strukturális direktíva egyszerre, az ***ngIf**-et egy wrapper elemre kell tenni: Nem lehet egy elemen két struktúrális direktíva. Ilyenkor a legcélszerűbb az egyiket egy wrapper <ng-container> elemre tenni, ami nem fog új DOM elemet létrehozni.

Az első interpolációban az **index** csak akkor érhető el, ha az *ngFor-t kiegészítjük "**index as index**"-szel: A count változó csak akkor érhető el, ha deklaráljuk "count as count"-al az *ngFor-ban. Bővebben az *ngFor exportált értékeiről: https://angular.io/api/common/NgForOf

KÉSZÍTETTE

Megjelenés

Világos