REACT

6. forduló

A kategória támogatója: TCS - Tata Consultancy Services

Ismertető a feladathoz

Felhasznált idő: 00:00/21:00 **Elért pontszám:** 0/8

1. feladat 0/3 pont

Recoil-t használunk az alábbi Atom-okkal:

```
export const reposState = atom({
    key: 'repos',
    default: []
})

export const userIdAtom = atom({
    key: 'userId',
    default: 1
})
```

Az alábbi módon próbáljuk használni egy komponensben:

```
const [view, setView] = useRecoilState(userIdAtom);
```

Azonban az eslint Warning-ot jelez mert a `***setView***` függvényt nem használjuk. '***setView***' is assigned a value but never used.

Hogyan tudjuk ezt orvosolni, hogy ne kapjunk warning-ot, és hogy a kódunk átmenjen a code-review-n?

Válasz

- ○ Kikapcsoljuk az ***eslint*** ide vonatkozó '***rule***'-ját:

```
Turn off `eslintno-unused-atoms`
```

- ◉ Másik hook-al használjuk, aminek nincs "írás" oldala, azaz csak Read-Only:

```
const view = useRecoilValue(userIdAtom);
```

Ez a válasz helyes, de nem jelölted meg.

- Csinálunk egy dummy "valamit", amivel mégis hivatozunk a `***setView***`-ra, majd kiküldjük a kódot review-ra
- Simán kitöröljük a "setter" részét, és mehet a kód review-ra:

```
const [view] = useRecoilState(userIdAtom);
```

- Másik hook-al használjuk, aminek nincs "írás" oldala, azaz csak Read-Only:

```
const view = useRecoilStateRO(userIdAtom);
```

- Másik hook-al használjuk aminek nincs "írás" oldala, azaz csak Read-Only

```
const [view] = useRecoilReadOnlyAtom(userIdAtom);
```

Magyarázat

Ahhoz, hogy a warning eltünjön és a kód is átmenjen a review nem elég a problémát megkerülni, ezért azok a válaszok, amik a warning-ot csak "eltüntenik" **helytelenek**:

- Kikapcsoljuk az ***eslint*** ide vonatkozó '***rule***'-ját
- Simán kitöröljük a "setter" részét, és mehet a kód review-ra
- Csinálunk egy dummy "valamit", amivel mégis hivatozunk a `***setView***`-ra, majd kiküldjük a kódot review-ra

Az alábbi válaszokban szereplő hook-ok nem léteznek Recoil-ban ezért azok **helytelenek**:

- useRecoilStateRO
- useRecoilReadOnlyAtom

A **helyes** javítás tehát a **useRecoilValue** hook használata.

További információt, segítséget a feladat helyes megoldásához az alábbi dokumentációs linken találhattok:
https://recoiljs.org/docs/introduction/core-concepts

2. feladat 0/2 pont

Mi jelenik meg az alábbi `***selector***`-t használva...

```
const filteredTodoListState = selector({
    key: 'FilteredTodoList',
    get: ({get}) => {
        const filter = get(todoListFilterState);
        const list = get(todoListState);

        switch (filter) {
            case 'Show Uncompleted':
                return list.filter((item) => item.isComplete);
            case 'Show Completed':
                return list.filter((item) => !item.isComplete);
            default:
```

```
                return list;
            }
        },
    });
```

...ha a filter='Show Completed' és a listában 2 elem ***isComplete=true*** és 3 elem ***isComplete=false*** elem van.

Válasz

- ○ Az összes elem megjelenik
- ○ Csak 2 elem jelenik meg
- ● Csak 3 elem jelenik meg
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

A példakódban **"Show Completed"** filter esetében az alábbi selector fog lefutni:

```
return list.filter((item) => !item.isComplete);
```

Ez csak azokat az elemeket adja vissza, ahol az ***isComplete=false.***

Ezért a "Csak 3 elem jelenik meg" válasz **helyes**.

További információt, segítséget a feladat helyes megoldásához az alábbi dokumentációs linken találhattok:
https://recoiljs.org/docs/basic-tutorial/selectors

3. feladat 0/3 pont

Mire való az alábbi komponens?

```
function Observer(): React.Node {
    const snapshot = useRecoilSnapshot();
    useEffect(() => {
        console.debug('The following atoms were modified:');
        for (const node of snapshot.getNodes_UNSTABLE({isModified: true})) {
            console.debug(node.key, snapshot.getLoadable(node));
        }
    }, [snapshot]);

    return null;
}
```

Válasz

- ○ A kód nem csinál semmi hasznosat -> kuka
- ○ A komponens Time-Travel-t tud megvalósítani a Snapshot-al
- ○ A kód csak hibás
- ● Az összes State Change-t loggolja
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

A **useRecoilSnapshot** horog segítségével feliratkozhatunk a Recoil állapotunk összes változására. Ez lehetővé teszi pl. az összes változás debug-olását.

Ezért a "Az összes State Change-t loggolja" **helyes**.

A "A kód nem csinál semmi hasznosat -> kuka" válasz **helytelen** mivel, a kódnak valós funkciója van.

A "A komponens Time-Travel-t tud megvalósítani a Snapshot-al" válasz **helytelen** mivel, a kód nem módosítja és állítja vissza a state-et, csak loggol-ja azt.

A "A kód csak hibás" válasz helytelen, mivel a példakód helyes.

További információt, segítséget a feladat helyes megoldásához az alábbi dokumentációs linken találhattok:
https://recoiljs.org/docs/api-reference/core/useRecoilSnapshot