JUNIORDev DIGITALNA DALMACIJA

React hooks > useRef

Reference

Glavna značajka prethodno spomenutog *useState* je to što svaka promjena stanja uzrokuje ponovno renderiranje komponente. Postoje slučajevi kada želimo izbjeći takvo ponašanje tj. želimo imati varijablu čija vrijednost ostaje sačuvana između renderiranja ali koja sama po sebi ne može uzrokovati renderiranje.

To možemo postići korištenjem ***useRef*** *hook*-a koji stvara internu referencu unutar komponente. Sintaksa je prilično jednostavna:

```
const mojaRef = useRef(0)
```

Ova naredba unutar komponente stvara referencu `mojaRef` u obliku objekta sa svojstvom `current` koje sadrži trenutnu vrijednost reference. Kao i kod *useState* početna vrijednost reference se navodi u pozivu samog *useRef hook*-a i ta početna vrijednost će se pridružiti referenci samo kod prvog renderiranja komponente, pri osvježavanju prikaza komponente referenca sadržava vrijednost koju je imala u prethodnom ciklusu.

Pogledajmo primjer sa dva brojača - jedan implementiran pomoću *useRef*, a drugi pomoću *useState*. Naravno, prije korištenja moramo uključiti oba *hook*-a:

App.jsx

```
import { useState, useRef } from 'react'

function App() {
  const [stateBrojac, postaviBrojac] = useState(0)
  const refBrojac = useRef(0)

  function uvecajRef(){
    refBrojac.current = refBrojac.current + 1;
```

```
      console.log("Uvećali smo brojač!")
    }

    function uvecajState(){
      postaviBrojac(b => b + 1)
    }

    return (
      <div>
        <p>Ref brojač: {refBrojac.current}</p>
        <button onClick={uvecajRef}>+</button>
        <p>State brojač: {stateBrojac}</p>
        <button onClick={uvecajState}>+</button>
      </div>
    )
  }

  export default App
```

Iz ovog primjera možemo uočiti nekoliko razlika između ova dva *hook*-a. Povećanjem *ref* brojača ne dolazi do ponovnog renderiranja, iako komponenta pamti svaku promjenu vrijednosti reference. To možemo vidjeti tako da uvećamo *state* brojač koji će uzrokovati ponovno renderiranje komponente i samim time će se prikazati aktualna vrijednost reference.

Ref brojač: 0
+
State brojač: 0
+

Također možete uočiti da suprotno od varijable stanja, referenci direktno mijenjamo vrijednost svojstva *current*, ne trebaju nam pomoćne funkcije za osvježavanje stanja.

DOM elementi

Još jedan primjer korištenja *useRef*-a je kada želimo stvoriti referencu na DOM element kojeg smo definirali u komponenti. Iako ovo ne bi trebali često raditi, ponekad nam je potrebna referenca na neki DOM element kako bi se npr. fokusirali na njega ili pomakli stranicu tako da bude vidljiv.

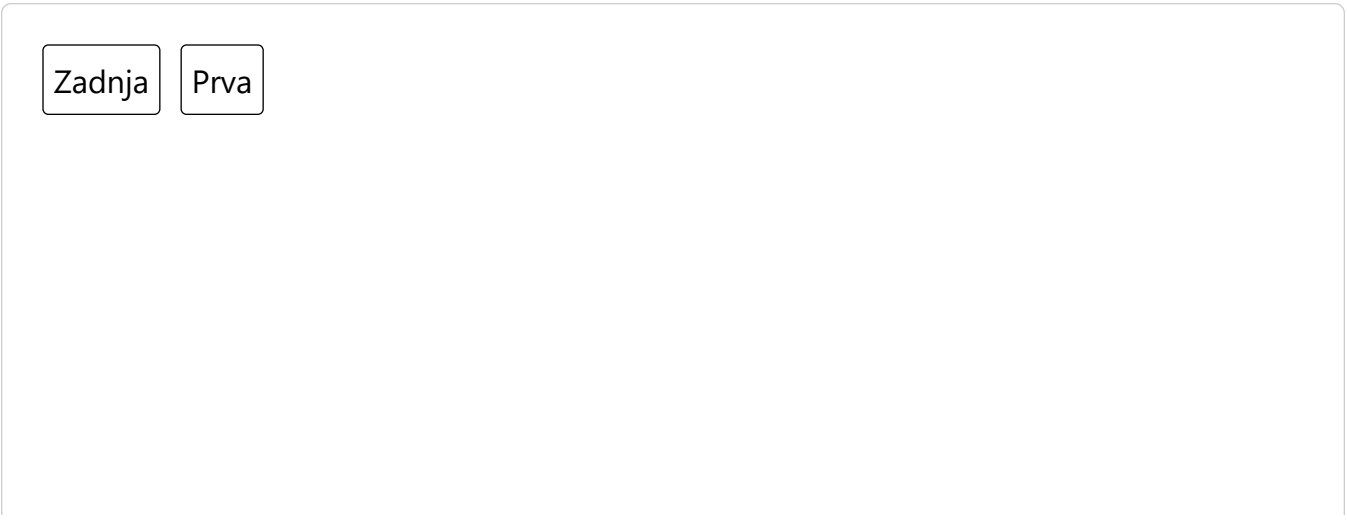
App.jsx

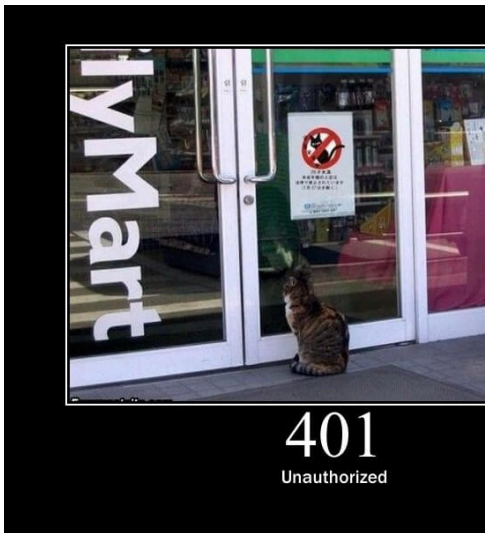
```jsx
import { useRef } from "react";
import "./App.css";

function App() {
  const prva = useRef();
  const zadnja = useRef();

  function naZadnju() {
    zadnja.current.scrollIntoView({behavior: "smooth"});
  }
  function naPrvu() {
    prva.current.scrollIntoView({behavior: "smooth"});
  }

  return (
    <div className="main">
      <div className="kontrole">
        <button onClick={naZadnju}>Zadnja</button>
        <button onClick={naPrvu}>Prva</button>
      </div>
      <div className="okvir">
        <img src='https://http.cat/204' alt='No Content' ref={prva} />
        <img src='https://http.cat/401' alt='Unauthorized' />
        <img src='https://http.cat/404' alt='Not Found' />
        <img src='https://http.cat/409' alt='Conflict' />
        <img src='https://http.cat/413' alt='Payload Too Large' ref={zadnja} />
      </div>
    </div>
  );
}
```





Korištenjem atributa ***ref*** unutar *div* elementa napravili smo vezu između ***useRef*** referenci definiranih na početku komponente i DOM elemenata. Nakon toga možemo tim DOM elementima pristupiti preko svojstva ***current*** njihove pripadajuće reference.

Vodite računa da je manipulacija DOM elemenata preko referenci samo pomoćno rješenje kada ne postoji druga opcija za postići željeni učinak. Većina takvih slučajeva se odnosi na situacije kada radimo ne-destruktivne akcije nad DOM elementima (fokus, scroll ili mjerenje dimenzija). Bilo kakve promjene DOM elemenata kojima upravlja React bi trebale biti izvedene na drugi način - npr. promjenom stanja.

⚠ Gore navedeni primjer sa *useRef* radi samo sa "običnim" DOM elementima. Ukoliko želite referencu na komponentu, potrebno je koristiti ***forwardRef*** *hook* ali ćemo ga za sada preskočiti.

Last updated on March 16, 2023

‹ useState useEffect ›