ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS
ΚΕΝΤΡΟ ΕΠΙΜΟΡΦΩΣΗΣ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗΣ

React JS
useRef Hook

Δαμιανός Μαρινάτος

Refs

- Όταν θέλουμε ένα component να "θυμάται" κάποια πληροφορία, αλλά **δεν θέλουμε αυτή η πληροφορία να προκαλεί νέα renders**, τότε μπορούμε να χρησιμοποιήσουμε ένα ref.
- Με απλά λόγια, ένα **ref** (συντομογραφία του *reference*) είναι ένας τρόπος για να έχουμε **άμεση πρόσβαση και αλληλεπίδραση με στοιχεία του DOM ή React components**.
- Όταν ένα ref συνδέεται με ένα στοιχείο React, μας δίνει τη δυνατότητα να το αναφέρουμε απευθείας στο πραγματικό DOM.

useRef

- Το useRef είναι ένα React Hook που χρησιμοποιείται για αποθήκευση τιμών που δεν επηρεάζουν το rendering.
- Μας επιτρέπει να κρατάμε αναφορές σε DOM στοιχεία ή μεταβλητές που δεν θέλουμε να κάνουν trigger re-render.
- Λειτουργεί σαν μια σταθερή αποθήκη δεδομένων που παραμένει ίδια η τιμή μεταξύ των renders.
- Πότε χρησιμοποιούμε useRef;
  - Όταν χρειαζόμαστε πρόσβαση σε DOM element (π.χ. input.focus())
  - Όταν θέλουμε να αποθηκεύσουμε μια τιμή ανάμεσα σε renders (π.χ. previous value)
  - Όταν θέλουμε να αποφύγουμε το re-render, δηλαδή να μην το αποθηκεύσουμε στο state.

useState vs useRef

- Το useState προκαλεί *re-render* του component όταν αλλάζει η τιμή του ενώ το useRef όχι.
- Το useState χρησιμοποιείται για τιμές που επηρεάζουν το UI, ενώ το useRef για τιμές που θέλουμε να θυμάται μεταξύ των renders χωρίς να επηρεάζουν το UI.
- Το useRef μπορεί να χρησιμοποιηθεί για άμεση πρόσβαση σε DOM στοιχεία (π.χ. input focus), ενώ το useState όχι.

useRef - Parameters


```
const ref = useRef(initialValue)
```

- **initialValue**: Η αρχική τιμή που θέλουμε να έχει η ιδιότητα current του ref object. Μπορεί να είναι οποιοσδήποτε τύπος. Αυτή η τιμή αγνοείται μετά το αρχικό render.
- Η useRef επιστρέφει ένα αντικείμενο με μία μόνο ιδιότητα, το **current**.
- Το current αρχικά έχει την τιμή που δώσαμε ως initialValue. Στη συνέχεια μπορούμε να την αλλάξουμε. Αν δώσουμε αυτό το ref object ως ref attribute σε κάποιο JSX στοιχείο, η React θα ορίσει το current ώστε να δείχνει σε αυτό το DOM στοιχείο.
- Σε επόμενα renders, η useRef θα επιστρέφει το ίδιο object

Παράδειγμα FocusInput (1)


```jsx
import { useRef } from "react";

const FocusInput : () => Element = () : Element => { Show usages new *
  const inputRef : RefObject<HTMLInputElement | null> = useRef<HTMLInputElement>( initialValue: null);

  const handleClick : () => void = () : void => { Show usages new *
    inputRef.current?.focus();
  };

  return (
    <>
      <div className="text-center space-x-4 mt-4">
        <input ref={inputRef} className="border px-4 py-2" />
        <button
          className="  bg-cf-dark-gray  text-white py-2 px-4 "
          onClick={handleClick}
        >
          Focus Input
        </button>
      </div>
    </>
  );
};

export default FocusInput; Show usages new *
```

- **useRef<HTMLInputElement>(null)**
  Δημιουργεί ένα ref που αρχικά δείχνει στο null, και δηλώνει ότι θα αναφέρεται σε ένα στοιχείο τύπου HTMLInputElement.
- **Ανάθεση του ref στο input**
  Το ref={inputRef} "δένει" το input με το ref. Έτσι, όταν το input εμφανιστεί στο DOM, η inputRef.current θα δείχνει σε αυτό το DOM στοιχείο.
- **Συνάρτηση handleClick**
  Όταν πατηθεί το κουμπί, καλείται η handleClick, η οποία κάνει focus στο input, αν υπάρχει (?).
- Το **?** είναι optional chaining: αν inputRef.current είναι null (π.χ. πριν το input φορτωθεί), δεν θα βγάλει error.
- Όταν πατήσουμε το κουμπί **"Focus Input"**, το input αποκτά focus (σαν να πατήσαμε πάνω στο input με το ποντίκι).