

Refs in React – Accessing DOM Elements & Values



Refs (short for References) in React are used to directly access a DOM element or a React component instance — without using state or re-rendering the UI.

When to Use ref ?

- Focus an input field
- Trigger animations
- Scroll to an element
- Read values without triggering re-renders
- Use third-party DOM libraries

Basic Syntax

jsx

 Copy  Edit

```
import { useRef } from 'react'; function MyComponent() { const inputRef = useRef(null); // Create the ref
const handleClick = () => { inputRef.current.focus(); // Access the DOM node directly };
return ( <div> <input ref={inputRef} type="text" placeholder="Type something..." /> <button
onClick={handleClick}>Focus Input</button> </div> ); }
```

Example: Reading Input Value without State

jsx

 Copy  Edit

```
function FormRefExample() { const nameRef = useRef(); const handleSubmit = () => { alert(`Name:
${nameRef.current.value}`); }; return ( <div> <input ref={nameRef} type="text" placeholder="Enter
name" /> <button onClick={handleSubmit}>Submit</button> </div> ); }
```

useRef vs State

Feature	<code>useRef</code>	<code>useState</code>
Causes re-render	✗ No	✓ Yes
Stores DOM ref	✓ Yes	✗ No
Stores value	✓ (but hidden)	✓ (and visible in UI)

Summary

- Create with `useRef()`
- Attach to JSX element via `ref={refName}`
- Access it with `refName.current`
- Great for input focus, animations, scroll, or storing mutable values