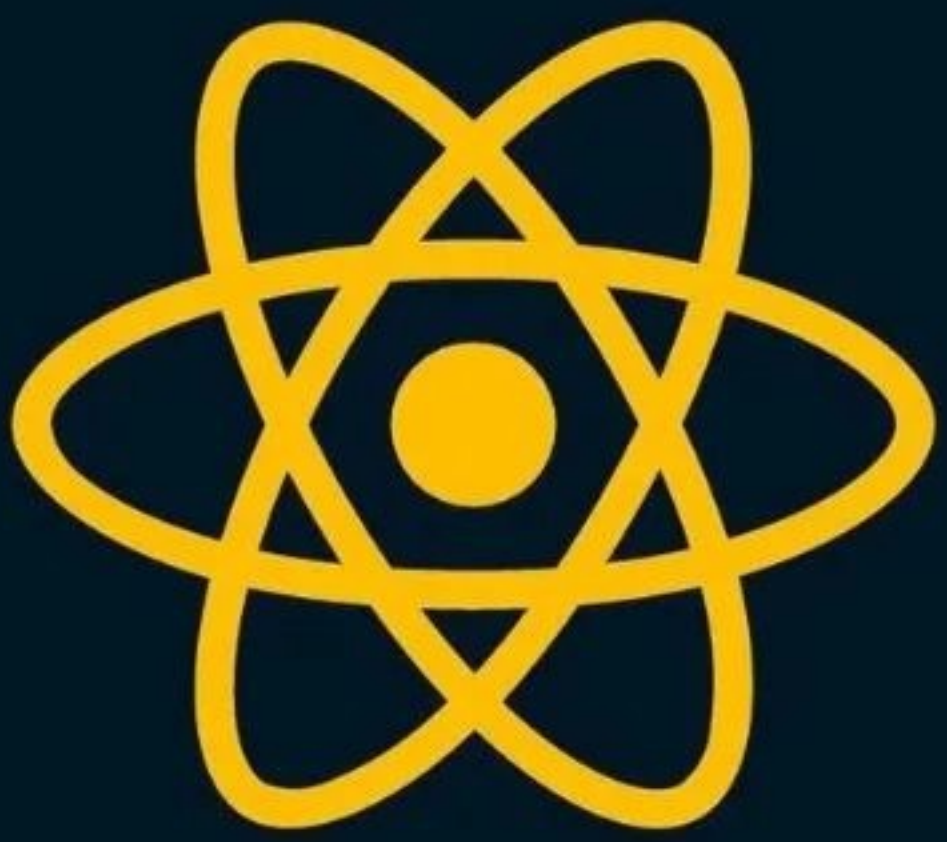
k Rakesh

@ Webdevloper

# useRef Hook



**k Rakesh**

**@ Webdevloper**

01

# Introduction

useRef is a React hook  that can be used in two ways:

- it can enable the access to the DOM element directly so we can manipulate it

- It can be used so save a value between re-renders in its current property that doesn't cause a re-render while being updated

Let's take a look some simple code to see the how we can use this hook in practice

**k Rakesh**

**@ Webdevloper**



# useState and onChange alternative

Below a "traditional" approach of changing the input value using useState and onChange handler

```
import './App.css';
import { useState } from 'react';

function App() {
  const [title, setTitle] = useState("")

  return (
    <div className='App'>
      <input
        type="text"
        value={title}
        onChange={(e) => setTitle(e.target.value)}
        placeholder="title"
      />
    </div>
  );
}

export default App;
```

k Rakesh
@ Webdevloper



# using useRef

Now let's take a look at the same example using useRef hook

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

function App() {
  const titleRef = useRef()

  return (
    <div className='App'>
      <input
        ref={titleRef}
        type="text"
        value={titleRef.current?.value}
        placeholder="title"
      />
    </div>
  );
}

export default App;
```

k Rakesh
@ Webdevloper



# What is the difference?

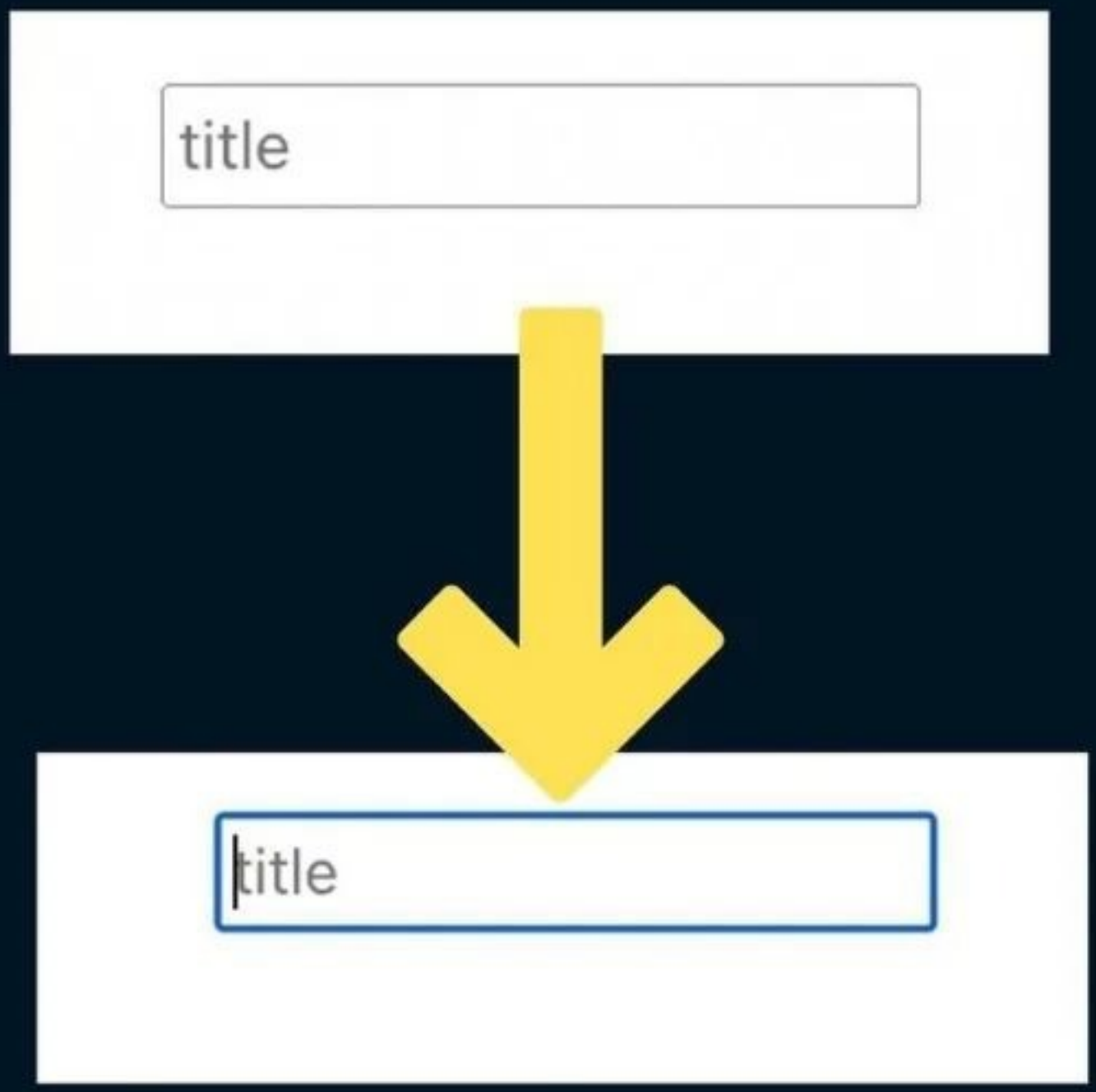
Both solutions are similar, however useState hook causes re-renders with every state change while useRef returns an object with a current property holding the actual value

k Rakesh
@ Webdevloper



# Add focus

To the following example let's add focus to the input. To visualize what we are trying to achieve let's take a look at the pictures below:



k Rakesh
@ Webdevloper



# Add useEffect hook

To add focus to our input we'll need to place some code in the useEffect hook

```
import './App.css';
import { useRef, useEffect } from 'react';

function App() {
  const titleRef = useRef()

  useEffect(()=> {
    titleRef.current.focus()
  }, [])

  return (
    <div className='App'>
      <input
        ref={titleRef}
        type="text"
        value={titleRef.current?.value}
        placeholder="title"
      />
    </div>
  );
}

export default App;
```

k Rakesh
@ Webdevloper



# Another example

Let's change the initial text after 3 seconds to a different one using useRef

```
import './App.css';
import { useRef, useEffect } from 'react';

function App () {
  const titleRef = useRef("Initial text");

  useEffect(function () {
    setTimeout(() => {
      titleRef.current.textContent = "Text changed"
    }, 3000);
  }, []);

  return (
    <div className="App">
      <h3 ref={titleRef}>Original title</h3>
    </div>
  )
}

export default App;
```

# Fallow For More Tips

# Save this Post For Later