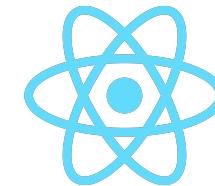




Pemrograman Sisi Cleint

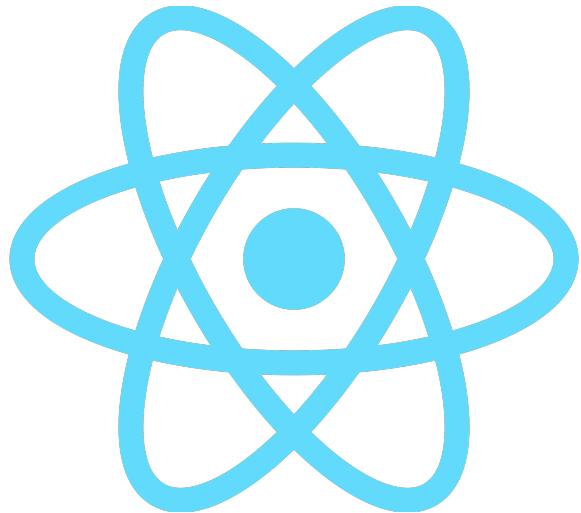
ReactJS State

Muhammad Syaifur Rohman, S.Kom, M.CS





Definisi



FYI

State adalah sebuah objek JavaScript yang digunakan untuk merepresentasikan data yang berubah sepanjang waktu dalam sebuah komponen React. State adalah salah satu konsep utama dalam React yang memungkinkan kita untuk membuat komponen yang dinamis dan interaktif.



Konsep State

“Dalam pengembangan aplikasi web, seringkali kita perlu mengelola data yang berubah, seperti input pengguna, status aplikasi, atau data yang diambil dari server.

Dengan menggunakan state, kita dapat memperbarui tampilan komponen berdasarkan perubahan data tersebut tanpa harus melakukan rendering ulang seluruh komponen.”



State

```
import { useState } from "react"

function App() {
  const [count, setCount] = useState(0);

  function handleClick(){
    setCount(count +1)
  }

  return (
    <div>
      Hitungan: {count}
      <button onClick={handleClick}>Klik!</button>
    </div>
  )
}

export default App
```

Untuk membuat state dalam sebuah komponen React, kita menggunakan fungsi `useState()` yang disediakan oleh React Hooks.

Untuk mengakses nilai state di dalam komponen, kita menggunakan variabel yang merepresentasikan state tersebut. Dalam contoh di atas, variabel `count` digunakan untuk merepresentasikan nilai state.

Untuk memperbarui nilai state, kita menggunakan fungsi setter yang diberikan oleh React Hooks. Dalam contoh di atas, kita menggunakan `setCount` untuk memperbarui nilai state `count`.

Studi Kasus State

```
import React, { useState } from 'react';

function App() {
  const [inputValue, setInputValue] = useState('');

  const handleChange = (event) => {
    setInputValue(event.target.value);
  };

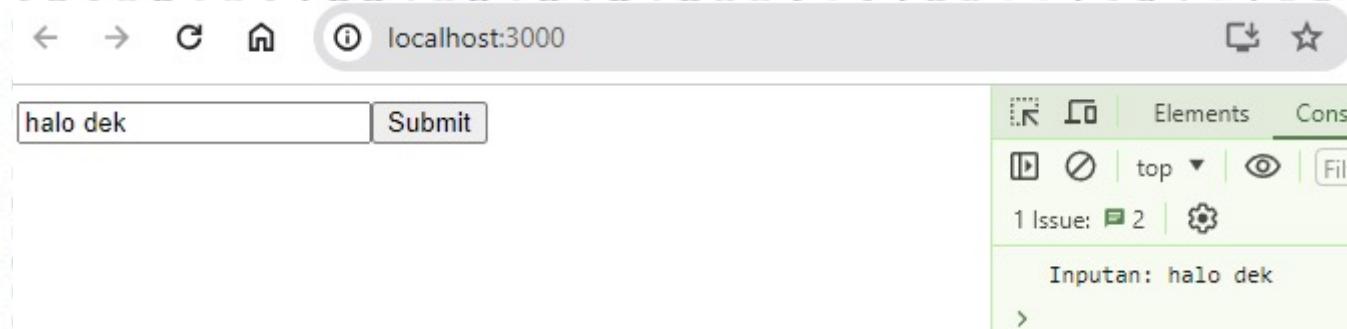
  const handleSubmit = (event) => {
    event.preventDefault();
    console.log('Submitted value:', inputValue);
    // Lakukan sesuatu dengan nilai input
  };

  return (
    <form onSubmit={handleSubmit}>
      <input type="text" value={inputValue} onChange={handleChange} />
      <button type="submit">Submit</button>
    </form>
  );
}

export default App
```



Hasil

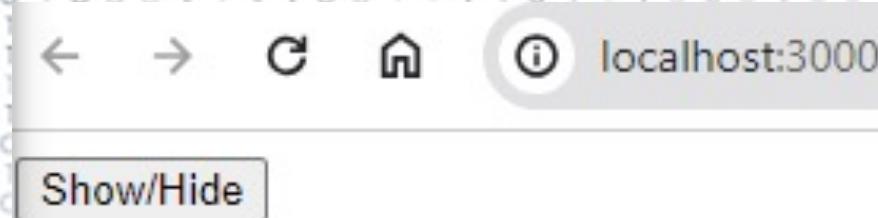
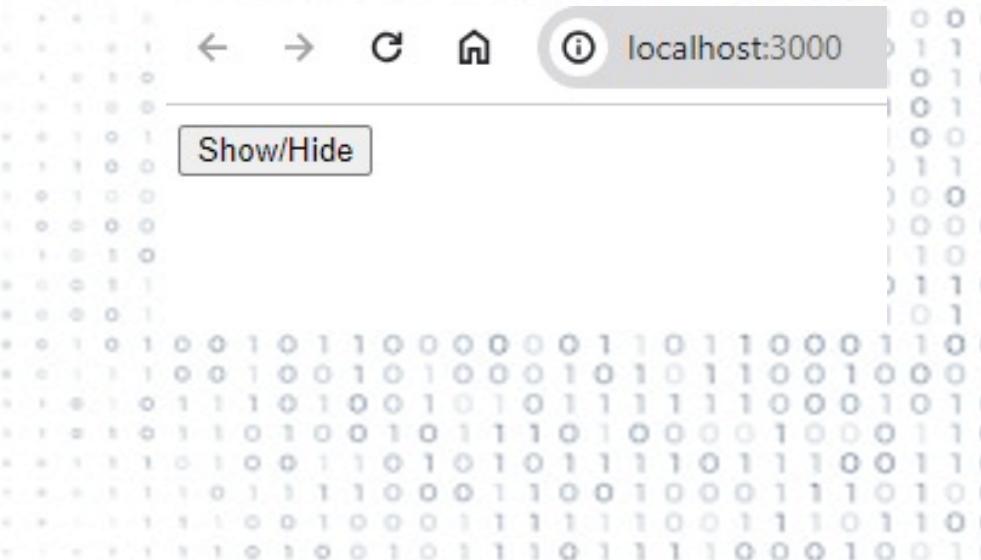


Studi Kasus State

```
● ● ●  
  
import React, { useState } from 'react';  
  
function App() {  
  const [isVisible, setIsVisible] = useState(true);  
  
  const toggleVisibility = () => {  
    setIsVisible(!isVisible);  
  };  
  
  return (  
    <div>  
      <button onClick={toggleVisibility}>Show/Hide</button>  
      {isVisible && <h1>Hello, World!</h1>}  
    </div>  
  );  
}  
  
export default App
```



Hasil



Punten!

Studi Kasus State



```
import React, { useState } from 'react';

function App() {
  const [items, setItems] = useState(['Apel', 'Gwedhang', 'Jeyuk']);
  const [newItem, setNewItem] = useState('');

  const addItem = () => {
    if (newItem.trim() !== '') {
      setItems([...items, newItem]);
      setNewItem('');
    }
  };

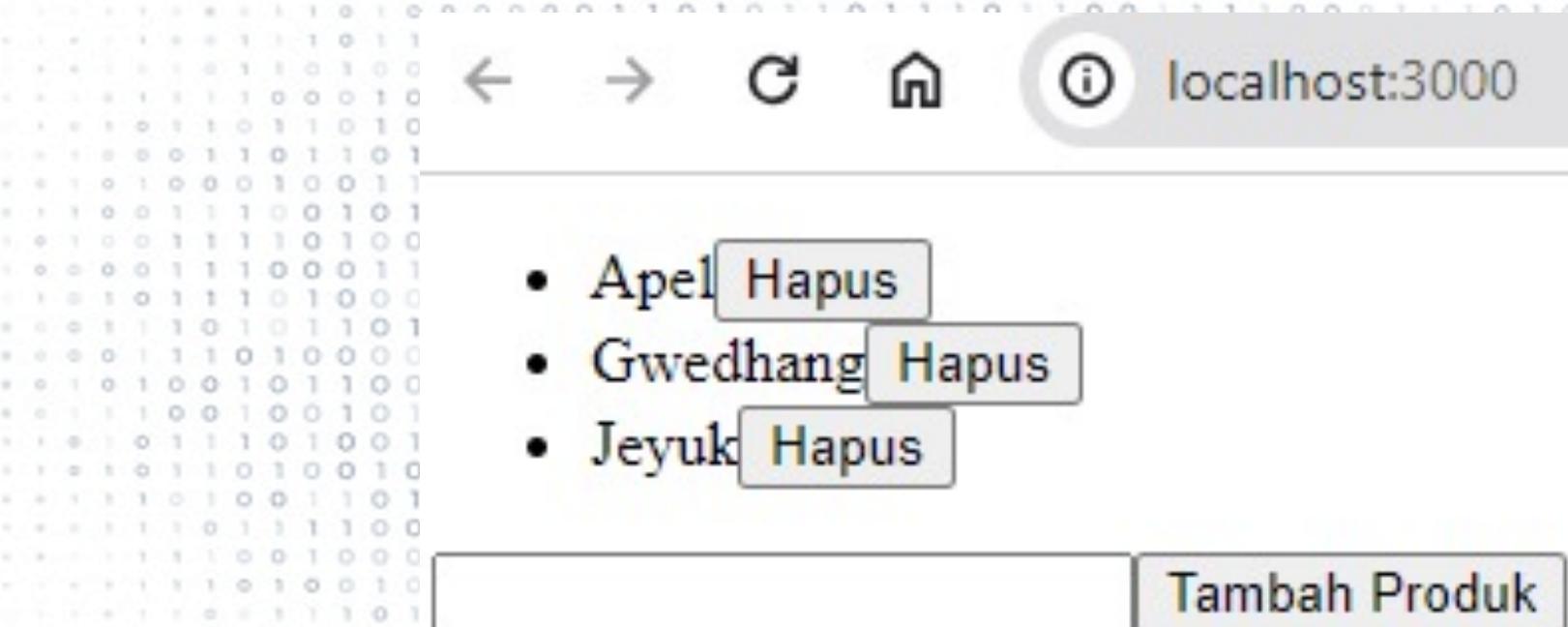
  const removeItem = (index) => {
    const updatedItems = items.filter((item, i) => i !== index);
    setItems(updatedItems);
  };
}
```



```
return (
  <div>
    <ul>
      {items.map((item, index) => (
        <li key={index}>
          {item}
          <button onClick={() => removeItem(index)}>Hapus</button>
        </li>
      ))}
    </ul>
    <input
      type="text"
      value={newItem}
      onChange={(e) => setNewItem(e.target.value)}
    />
    <button onClick={addItem}>Tambah Produk</button>
  </div>
);

export default App
```

Hasil



Studi Kasus State

```
import React, { useState } from 'react';

function App() {
  const [isOnline, setIsOnline] = useState(true);

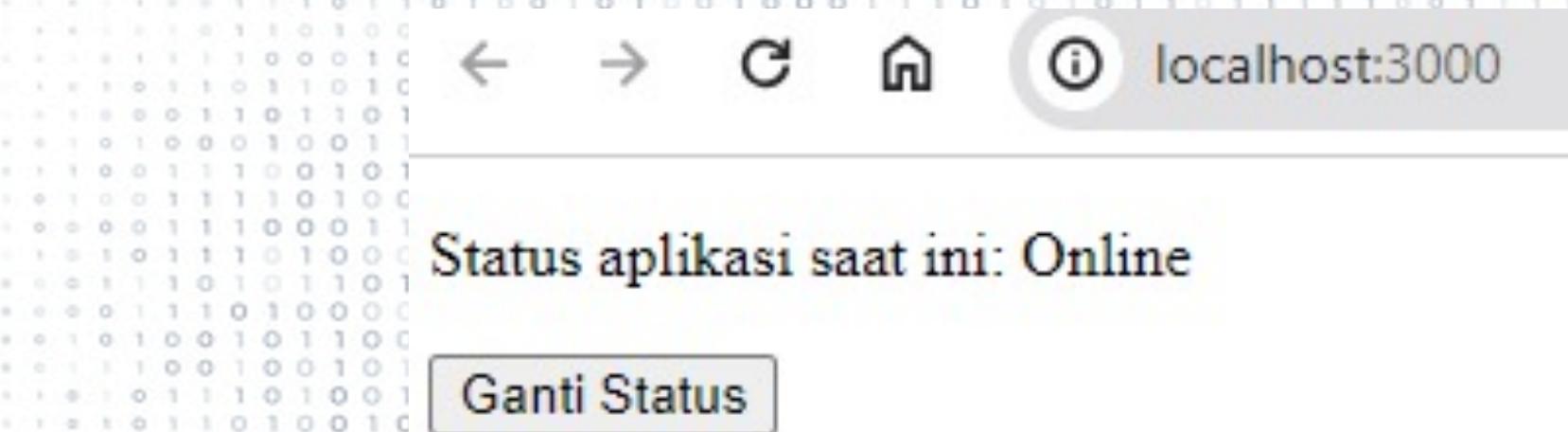
  const toggleStatus = () => {
    setIsOnline(!isOnline);
  };

  return (
    <div>
      <p>Status aplikasi saat ini: {isOnline ? 'Online' : 'Offline'}</p>
      <button onClick={toggleStatus}>Ganti Status</button>
    </div>
  );
}

export default App
```



Hasil





Penutup



State adalah salah satu fitur utama yang membuat ReactJS sangat kuat dan fleksibel untuk pengembangan aplikasi web interaktif. Dengan menggunakan state, pengembang dapat membuat komponen yang responsif dan dinamis berdasarkan interaksi pengguna dan perubahan data. Melalui berbagai kasus penggunaan seperti mengelola input form, menampilkan atau menyembunyikan komponen, mengelola list data, dan mengatur status aplikasi, kita dapat melihat betapa pentingnya memahami dan mengelola state dengan baik dalam aplikasi React.



Penutup

Gunakan State Secara Bijaksana

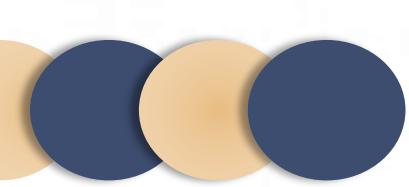
- Simpan state hanya di tempat yang diperlukan. Jika data tidak berubah, simpan di props atau variabel biasa.
- Hindari mengelola state yang sama di banyak tempat. Tempatkan state di komponen induk dan gunakan props untuk meneruskannya ke komponen anak.

Manajemen State Global

- Untuk aplikasi besar, pertimbangkan menggunakan manajemen state global seperti Context API, Redux, atau MobX untuk mengelola state yang dibutuhkan di banyak tempat dalam aplikasi.

Konsisten dengan Struktur State

- Pertahankan struktur state yang konsisten dan mudah dipahami. Gunakan objek untuk mengelompokkan state yang terkait.



Referensi

1. Eran Krinsbruner, **A Frontend Web Developer's Guide to Testing**, 2022
2. Adam Boduch , Roy Derks , Mikhail Sakhniuk, **React and React Native - Fourth Edition**, 2022
3. Maya Shavin , Raymond Camden, **Frontend Development Projects with Vue.js 3 - Second Edition**, 2023
4. Waweru Mwaura, **End-to-End Web Testing with Cypress**, 2021
5. <https://roadmap.sh/frontend>
6. <https://jsonplaceholder.typicode.com/posts>



Thanks

Muhammad Syaifur Rohman, S.Kom, M.CS

