Pemrograman Web Modern

Insan Taufik, S.Kom., M.Kom.

State Management Dasar (useState Hook)

Sub-Materi

Konsep State sebagai data internal yang dapat berubah. Pengenalan Hooks. Penggunaan hook useState untuk mengelola state di komponen fungsional.

Tujuan Pembelajaran (Capaian)

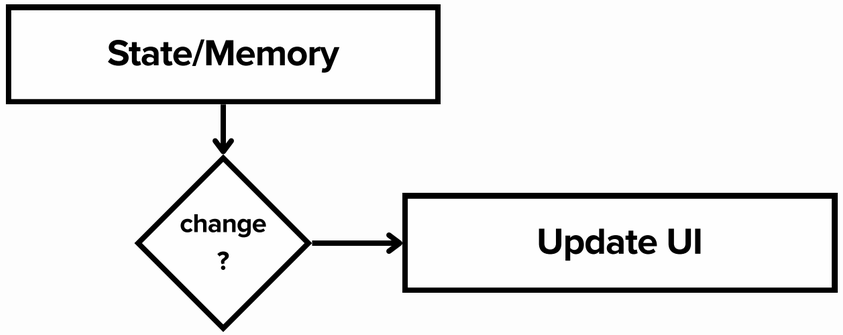
Mampu membuat aplikasi interaktif dengan mengubah data menggunakan state melalui hook useState.

Apa itu State?

Definisi State:

- Data internal yang dimiliki oleh komponen
- Dapat berubah selama lifecycle komponen
- Perubahan state akan trigger re-render komponen
- Local vs Global state

Analoginya:



Perbedaan Props vs State

Props	State
Data dari parent komponen	Data internal komponen
Immutable (tidak bisa diubah)	Mutable (bisa diubah)
Changes trigger re-render	Changes trigger re-render
Functional parameters	Component's memory

Pengenalan React Hooks

Apa itu Hooks?

- Fungsi khusus yang memungkinkan menggunakan state dan fitur React lainnya di functional components
- Diperkenalkan di React 16.8
- Tidak perlu class components untuk state management

Beberapa Hooks Populer:

- useState - untuk state management
- useEffect - untuk side effects
- useContext - untuk context API
- useReducer - untuk complex state logic

useState Hook - Sintaks Dasar

```jsx
import { useState } from 'react';

// Sintaks dasar
const [state, setState] = useState(initialValue);

// Contoh
const [count, setCount] = useState(0);
const [name, setName] = useState('');
const [isLoggedIn, setIsLoggedIn] = useState(false);
```

Penjelasan:

- useState returns array dengan 2 element
- Element pertama: current state value
- Element kedua: function untuk update state
- Never modify state directly!

Aturan Penggunaan useState

✅ **BENAR:**

```jsx
setCount(count + 1);
setName('John Doe');
setIsLoggedIn(true);
```

❌ **SALAH:**

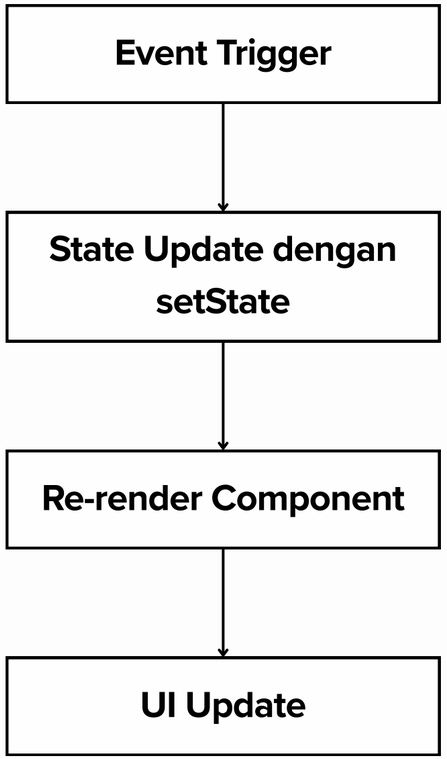
```jsx
count = count + 1; // Langsung modify state
```

Functional Update (disarankan):

```jsx
setCount(prevCount => prevCount + 1);
```

State Update dan Re-render

Flow State Update:



Karakteristik:

- State updates are asynchronous
- Multiple state updates mungkin digabungkan (batched)
- State bersifat immutable artinya setiap kali state diperbarui, sebuah salinan baru dari state tersebut dibuat, dan bukan mengubah state yang sudah ada

Best Practices useState

1. Gunakan multiple state variables untuk data yang tidak related
Pisahkan state yang tidak saling berhubungan agar kode lebih jelas dan update tidak memicu re-render yang tidak perlu.

```
const [count, setCount] = useState(0);
const [theme, setTheme] = useState("light");
```

2. Group related data dalam object state
Kalau datanya saling berkaitan, satukan dalam satu objek agar lebih mudah dikelola.

```
const [user, setUser] = useState({ name: "", age: 0 });
```

3. Functional updates untuk state yang bergantung pada previous state
Jika nilai baru tergantung pada nilai lama, gunakan callback form dari setState.

```
setCount(prevCount => prevCount + 1);
```

4. Initialize state dengan nilai yang sesuai
Selalu beri nilai awal yang tepat sesuai tipe datanya.

```
const [items, setItems] = useState([]);     // array
const [user, setUser] = useState(null);     // object belum ada
const [loading, setLoading] = useState(true); // boolean
```

Ringkasan

Yang telah dipelajari:

- ✓ Konsep state sebagai data internal
- ✓ Pengenalan React Hooks
- ✓ Penggunaan useState untuk state management
- ✓ Aturan dan best practices