```
    jumlah.value = 1;
  }
};
</script>
```

Perbandingan Keduanya:

Aspect	Options API	Composition API
Struktur	Lebih terpisah (berbasis properti)	Lebih terintegrasi (semua di `setup`)
Fleksibilitas	Mudah untuk proyek kecil	Lebih fleksibel untuk proyek besar
Pembelajaran	Cocok untuk pemula	Cocok untuk yang lebih mahir

Kalau dari kode kamu, gaya penulisan sudah **Composition API** karena menggunakan `script setup`. Ini adalah gaya modern Vue.js. 😊

Mau bahas lebih detail tentang satu gaya tertentu?

jelaskan kak kalau di react js dan di vue js itu kan secara konsep sama yaa? nah maksud saya adalah kalau seperti onMounted di react js seperti apa ? dst intinya adalah konsep penulisan di vue js saya di convert ke react

Benar sekali, **React.js** dan **Vue.js** memiliki konsep yang mirip untuk mengelola komponen, tetapi dengan sintaks dan pendekatan yang sedikit berbeda. Saya akan membandingkan penulisan di Vue.js dengan cara menulis yang setara di React.js. Berikut ini adalah perbandingannya:

---

1. Lifecycle Method

- **Vue.js**: Menggunakan hooks seperti `onMounted`, `onUpdated`, `onUnmounted`, dll.
- **React.js**: Menggunakan `useEffect`.

Contoh Vue.js:

```vue
<script setup>
import { onMounted } from "vue";

onMounted(() => {
  console.log("Component mounted");
});
</script>
```

Setara di React.js:

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

function Component() {
  useEffect(() => {
    console.log("Component mounted");
  }, []); // Empty dependency array means it runs once, like onMounted.

  return <div>React Component</div>;
}
```

---

2. State Management

- **Vue.js**: Menggunakan `ref` atau `reactive` untuk mengelola state.
- **React.js**: Menggunakan `useState` untuk mengelola state.

Contoh Vue.js:

```vue
<script setup>
import { ref } from "vue";

const count = ref(0);
```

```
const increment = () => {
  count.value++;
};
</script>

<template>
  <div>
    <p>Count: {{ count }}</p>
    <button @click="increment">Increment</button>
  </div>
</template>
```

Setara di React.js:

jsx

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

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

  const increment = () => {
    setCount((prevCount) => prevCount + 1);
  };

  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={increment}>Increment</button>
    </div>
  );
}
```

---

3. Computed Properties

- **Vue.js**: Menggunakan `computed` untuk menghitung nilai berdasarkan state.
- **React.js**: Menggunakan fungsi biasa atau `useMemo` untuk nilai yang dihitung.

Contoh Vue.js:

```vue
<script setup>
import { ref, computed } from "vue";

const count = ref(10);

const doubleCount = computed(() => count.value * 2);
</script>

<template>
  <p>Double Count: {{ doubleCount }}</p>
</template>
```

Setara di React.js:

```jsx
import { useMemo, useState } from "react";

function Component() {
  const [count, setCount] = useState(10);

  const doubleCount = useMemo(() => count * 2, [count]);

  return <p>Double Count: {doubleCount}</p>;
}
```

---

4. Event Handling

- **Vue.js**: Menggunakan atribut `@event` seperti `@click`.
- **React.js**: Menggunakan atribut `onEvent` seperti `onClick`.

Contoh Vue.js:

```vue
```

```
<template>
  <button @click="handleClick">Click me</button>
</template>

<script setup>
const handleClick = () => {
  console.log("Button clicked");
};
</script>
```

Setara di React.js:

jsx

```jsx
function Component() {
  const handleClick = () => {
    console.log("Button clicked");
  };

  return <button onClick={handleClick}>Click me</button>;
}
```

---

5. API Calls

- **Vue.js**: Bisa menggunakan `onMounted` untuk memulai API call.
- **React.js**: Menggunakan `useEffect`.

Contoh Vue.js:

vue

```
<script setup>
import { ref, onMounted } from "vue";
import axios from "axios";

const data = ref([]);

onMounted(async () => {
```

```
  const response = await axios.get("https://api.example.com/data");
  data.value = response.data;
});
</script>
```

Setara di React.js:

jsx

```jsx
import { useState, useEffect } from "react";
import axios from "axios";

function Component() {
  const [data, setData] = useState([]);

  useEffect(() => {
    async function fetchData() {
      const response = await axios.get("https://api.example.com/data");
      setData(response.data);
    }
    fetchData();
  }, []);

  return <div>{JSON.stringify(data)}</div>;
}
```

---

6. Routing

- **Vue.js**: Menggunakan `vue-router`.
- **React.js**: Menggunakan `react-router-dom`.

Contoh Vue.js:

vue

```vue
<template>
  <router-link to="/about">Go to About</router-link>
</template>
```

Setara di React.js:

```jsx
import { Link } from "react-router-dom";

function Component() {
  return <Link to="/about">Go to About</Link>;
}
```

Kesimpulan: Vue.js dan React.js memiliki konsep dasar yang mirip, tetapi pendekatan penulisannya berbeda:

- **Vue.js** lebih deklaratif dan terstruktur.
- **React.js** lebih berbasis fungsi dan eksplisit.

Keduanya sama-sama kuat, tinggal disesuaikan dengan preferensi dan kebutuhan proyek. 😊