

what is react State manager-1

State manager یکی از مهمترین جنبه های هر برنامه است. State برنامه تعیین می کند که کاربران چه چیزی را ببینند، برنامه چگونه ظاهر شود، چه داده هایی ذخیره شود و غیره. کامپوننت های React دارای یک built-in state object هستند.

اگر یک کاربر در تعامل با برنامه شما، state خود را تغییر دهد، ممکن است الپس از آن کاملاً متفاوت به نظر برسد، زیرا به جای state قبلی، با این state جدید نشان داده می شود. زمانی که برنامه بزرگ تر می شود، پیچیدگی State manager به اشتراک گذاشته شده در بین مؤلفه ها دشوار می شود.

اگر توسعه دهندگان مقیاس پذیری را در ذهن نداشته باشند، پیدا کردن اینکه کجا مشکلی پیش می آید، واقعاً سخت است. به همین دلیل است که در برنامه خود به State manager نیاز دارید.

توسعه دهندگان React state را ایجاد و حفظ می کنند. React با معرفی hooks و redux ها به این سمت حرکت کرده است. هنگامی که state باید در میان بسیاری از مؤلفه ها به اشتراک گذاشته شود و ممکن است مشکلات به وجود بیایند.

Redux:

Redux برای حل این مشکل خاص ایجاد شده است. این یک store فراهم می کند که همه state های برنامه شما را نگه می دارد. هر جزء می تواند به stored states ذخیره شده بدون ارسال آن از یک component به component دیگر دسترسی پیدا کند. در اینجا یک نمای ساده از نحوه عملکرد Redux وجود دارد.

Store in Redux

Store، state برنامه را نگه می دارد. می توانید به stored state دسترسی داشته باشید و state ها را update کنید

Hooks:

Hooks پس از انتشار نسخه 16.8 به ری اکت اضافه شده اند. آن ها به شما اجازه می دهند که از state و دیگر قابلیت های ری اکت بدون نوشتن کلاس استفاده کنید.

نمی توانیم از Hooks در کلاس ها استفاده کنید. ولی با استفاده از آن ها دیگر احتیاجی به نوشتن کلاس نخواهید داشت.

در حقیقت Hook یک تابع ویژه است که به ما اجازه می دهد از امکانات ری اکت استفاده کنیم. برای مثال، `useState` به ما این امکان را می دهد که از state در `function component` استفاده کنیم. Hook های دیگری هم وجود دارند

منبع:

[/https://www.loginradius.com/blog/async/react-state-management](https://www.loginradius.com/blog/async/react-state-management)

Function component-2

```
const Example = (props) => {  
  return <div />;  
}
```

یا به این شکل:

```
function Example(props) {  
  return <div />;  
}
```

نوع دیگری از کامپوننت ها که

۱- فاقد `this` هستند

۲- با استفاده از `hooks` می تواند از `state` استفاده مثل `useState` که دقیقا امکانات مشابه `this.state` در `class`

`component` را فراهم می کند

۳- استفاده از آن راحت تر از `class component` هاست

۴- تست کردن فانکشنال کامپوننت ها ساده تر از کلاس کامپوننت هاست.

۵- کدهای نهایی ترجمه شده (و حتی سورس کدها) با فانکشنال خیلی کوتاه تر میشه.

۶- استفاده از فانکشنال کامپوننت ها به رعایت `Best Practice` ها کمک میکنه

در یک کامپوننت تابعی، کلید واژه `this` وجود ندارد، پس ما نمی توانیم مقدار `this.state` را بخوانیم و یا مقداری را به آن اختصاص دهیم. در عوض، ما می توانیم از هوک `useState` به طور مستقیم در کامپوننت خود استفاده کنیم:

```
const [count, setCount] = useState(0);
```

منبع :

[https://www.google.com/url?](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAkQAQ&url=https%3A%2F%2Fwww.robinwieruch.de%2Freact-function-component%2F&usg=AOvVaw3Ik92PqA0Aignqave6Z5mq)

[sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAkQAQ&url=https%3A%2F%2Fwww.robinwieruch.de%2Freact-function-component%2F&usg=AOvVaw3Ik92PqA0Aignqave6Z5mq](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAkQAQ&url=https%3A%2F%2Fwww.robinwieruch.de%2Freact-function-component%2F&usg=AOvVaw3Ik92PqA0Aignqave6Z5mq)

[https://www.google.com/url?](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAoQAQ&url=https%3A%2F%2Fwww.geeksforgeeks.org%2Freactjs-functional-components%2F&usg=AOvVaw3_OO4WeAT1OE5tlh_ty25m)

[sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAoQAQ&url=https%3A%2F%2Fwww.geeksforgeeks.org%2Freactjs-functional-components%2F&usg=AOvVaw3_OO4WeAT1OE5tlh_ty25m](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwjhlqv19rn2AhW9RvEDHeXWBfQQFnoECAoQAQ&url=https%3A%2F%2Fwww.geeksforgeeks.org%2Freactjs-functional-components%2F&usg=AOvVaw3_OO4WeAT1OE5tlh_ty25m)

[?https://fa.reactjs.org/docs/hooks-state.html](https://fa.reactjs.org/docs/hooks-state.html)

Hooks in react-3

Hooks پس از انتشار نسخه‌ی 16.8 به ری‌اکت اضافه شده‌اند. آن‌ها به شما اجازه می‌دهند که از state و دیگر قابلیت‌های ری‌اکت بدون نوشتن کلاس استفاده کنید. نمی‌توانیم از Hooks در کلاس‌ها استفاده کنید. ولی با استفاده از آن‌ها دیگر احتیاجی به نوشتن کلاس نخواهید داشت.

در حقیقت Hook یک تابع ویژه است که به ما اجازه می‌دهد از امکانات ری‌اکت استفاده کنیم. برای مثال، `useState` به ما این امکان را می‌دهد که از state در `function component` استفاده کنیم. Hook‌های دیگری هم وجود دارند

Hooks rules:

۱- Hook‌ها را فقط در بالاترین سطح فراخوانی کنید
Hook‌ها را درون لوپ، شرط، یا توابع در هم تنیده فراخوانی نکنید به جای آن، همیشه آن‌ها را در بالاترین سطح توابع ری‌اکت، پیش از هر `return` زودهنگام، فراخوانی کنید. با پیروی از این قوانین، مطمئن می‌شوید که Hook‌ها هنگامی که در یک کامپوننت رندر می‌شوند به یک ترتیب فراخوانی می‌شوند. این چیز نیست که به ری‌اکت اجازه می‌دهد state hook‌ها را بین چندین فراخوانی از `useState` و `useEffect` حفظ کند

۲- Hook‌ها را فقط در توابع ری‌اکتی فراخوانی کنید
Hook‌ها را در توابع عادی جاوااسکریپت فراخوانی نکنید. به جای آن، می‌توانید:
• Hook‌ها را در توابع ری‌اکت فراخوانی کنید.

• Hook ها را از Hook ها فراخوانی کنید

با پیروی از این قوانین، متوجه می شوید که تمام منطق stateful به وضوح در منبع کد یک کامپوننت قابل مشاهده است.

ری اکت به ترتیبی که Hook ها فراخوانی می شود تکیه می کند.

منبع:

[/https://www.loginradius.com/blog/async/react-state-management](https://www.loginradius.com/blog/async/react-state-management)