State manager یکی از مهمترین جنبه های هر برنامه است. Stae 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 objectهستند.

اگر یک کاربر در تعامل با برنامه شما، stare خود را تغییر دهد، ممکن است UIپس از آن کاملاً متفاوت به نظر برسد، زیرا به جای state قبلی، با این stare جدید نشان داده می شود.
زمانی که برنامه بزرگتر می‌شود، پیچیدگیState manager به اشتراک گذاشته شده در بین مؤلفه ها دشوار می‌شود.
اگر توسعه‌دهندگان مقیاس‌پذیری را در ذهن نداشته باشند، پیدا کردن اینکه کجا مشکلی پیش می‌آید، 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. می‌توانید به stored stateدسترسی داشته باشید و state ها را update کنی

Hooks:
Hooks پس از انتشار نسخه‌ی 16.8 به ری‌اکت اضافه شده‌اند. آن‌ها به شما اجازه می‌دهند که از state و دیگر قابلیت های ری‌اکت بدون نوشتن کلاس استفاده کنید.
نمی توانیم 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. **Hook های** دیگری هم وجود دارند

Hooks rules:

Hook-1ها 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، مطمین می‌شوید که Hookها هنگامی که در یک کامپوننت رندر می‌شوند به یک ترتیب فراخوانی می‌شوند. این چیزیست که به ری‌اکت اجازه می‌دهد state hookها را بین چندین فراخوانی از `useState` و `useEffect` حفظ کند

Hook 2-ها 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.
- Hookها را از Hookها فراخوانی کنید

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

ری‌اکت به ترتیبی که Hookها فراخوانی می‌شود تکیه می‌کند.

Function component:

```
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 می‌کند