1)­ stateها یک راه ساده برای ارتباط و اشتراک گذاری داده ها بین کامپوننت ها هستند . و ساختار دادهمشخص برای برنامه ایجاد میکند که میتوانیم آن را بخوانیم و بنویسیم .هم فانکشن کامپوننتها و هم کلاس کامپوننت ها میتوانند از استیت استفاده کنند .

state در واقع یک شیء است که property های متخلفی را در خود نگه می دارد. ما می توانید هر نوع داده ای (آرایه، رشته و...) را در آن قرار دهید. خصوصیت state یک خصوصیت خاص است؛ از این جهت که اگر تغییری در آن ایجاد شود react تمام DOM را دوباره render می کند در واقع در نتیجه عملکرد کاربر تغییر میکند . در این حالت setState تفاوت های بین آرگومان خود و state قبلی را پیدا کرده و آن را اصلاح می کند اما آنقدر هوشمند است که به otherState دست نمی زند (چون آن را تغییر نداده ایم). این رفتار که با هررفتار کاربر استیت تغییر میکند اگر برنامه ما بزرگ شود پیگیری همه وابستگیها مشکل است . و برای رفع این مشکل کتابخانه هایی برای مدیریت stateها ارائه شده است .

2) فانکشن کامپوننت ها :

خوانایی فانکشنال کامپوننت ها بالاتر از کلاس کامپوننت هاست .

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

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

تا قبل از نسخه 16.8 و معرفی hooks نمی توانستیم از [state و lifecycle methodes](https://reactjs.org/docs/state-and-lifecycle.html)  استفاده کنید.ولی با معرفی hooks این مشکل نیز بر طرف شد و می توانیم تمامی کلاس کامپوننت ها را با فانکشن کامپوننت بنویسیم.

3) هوک ها به نسخه ۱۶.۸ اضافه شده‌اند. هوک‌ها توابعی هستند که با استفاده از آن‌ها می‌توانیم بدون نوشتن کلاس، از state و دیگر ویژگی‌های ری‌اکت استفاده کنیم. نام آن‌ها همیشه با use شروع می‌شود

برای مثال، هوک useState  این امکان را می‌دهد که از state در فانکشن کامپوننتها استفاده کنیم. هوک‌های دیگری هم وجود دارند .

یاuseEffect با استفاده از این Hook, به ری‌اکت اعلام می‌کنید که کامپوننت ما بعد از رندر نیاز به انجام کاری دارد ،که همان “effect” خودمان است) و بعد از به‌روز رسانی فراخوانی‌اش می‌کند. ) می‌توانیم با useEffect  به عنوان ترکیبی از componentDidMount ، componentDidUpdate  و  componentWillUnmount  نگاه کنیم.