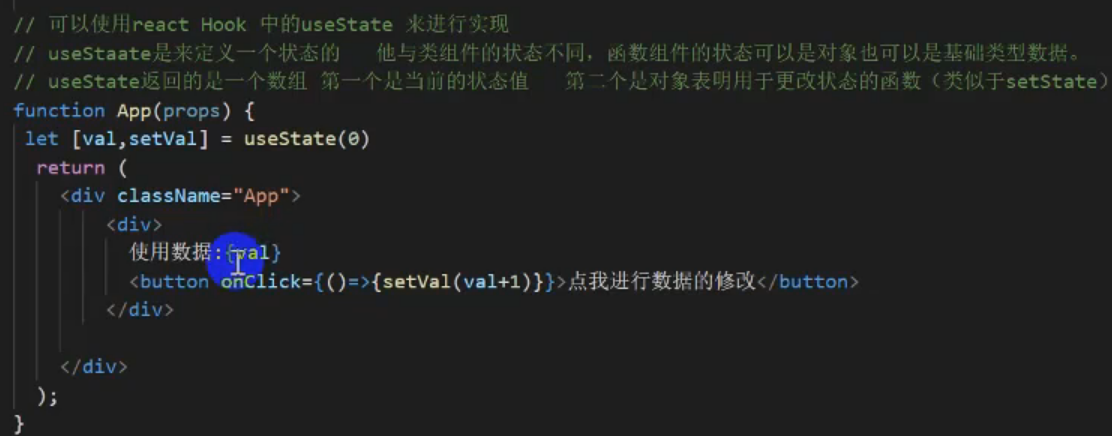
**Hook**

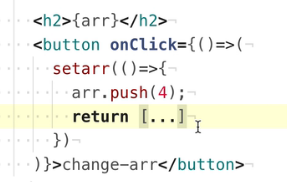
React16.8新增属性





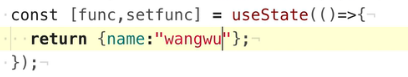
***\*useState(0)****的“0”是默认值*







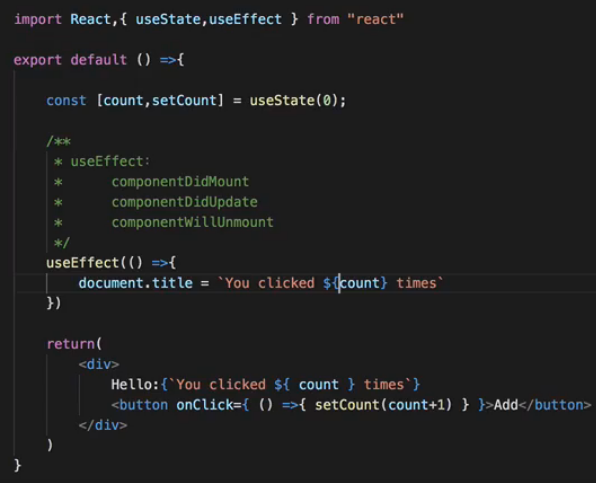
**useState里是函数时，func接收的是返回值**

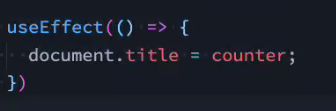




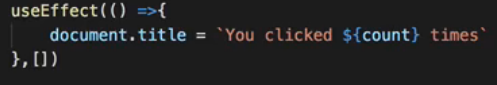
**Effect Hook（代表了三个生命周期函数，可以多个useeffect）**





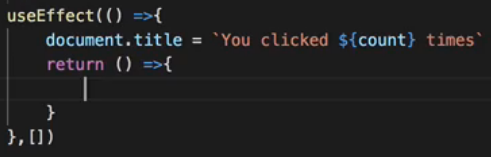


相当于componentDidMount+ componentDidUpdate

相当于componentDidMount（空数组，不依赖任何变量）



只监听count发生改变时，才会触发componentDidUpdate（依赖count）

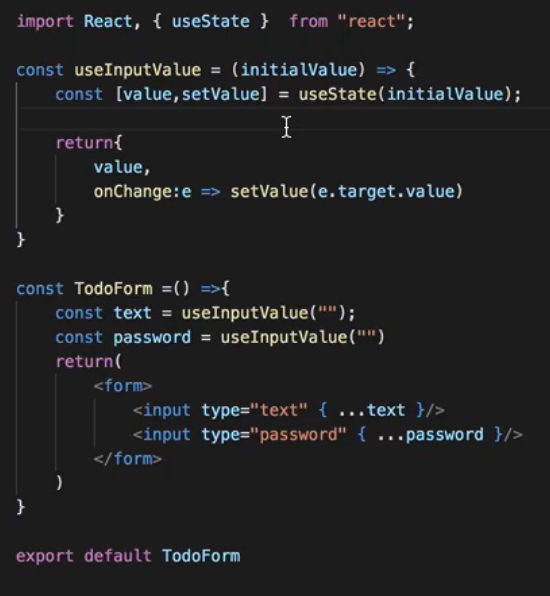
相当于 componentWillUnmount

组件销毁前清除定时器

多个useEffect时候



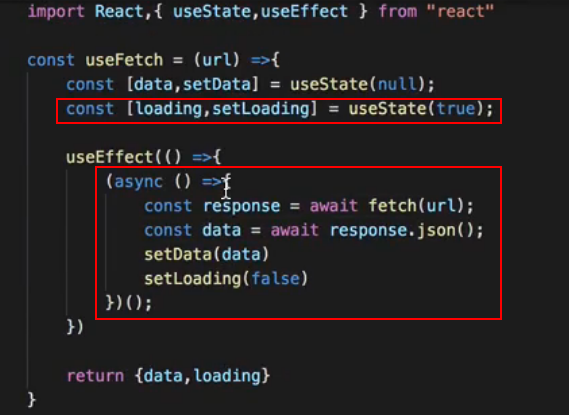
**例子：**



**网络请求**

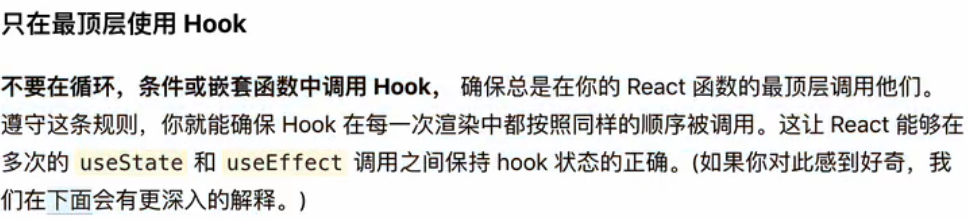
*\*使用一个函数包裹async防止报警告*

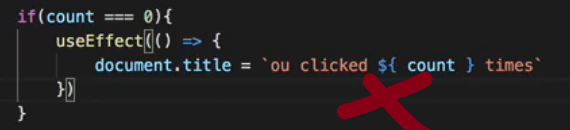
*\*定义一个loading来解决异步请求一开始找不到值的错误*

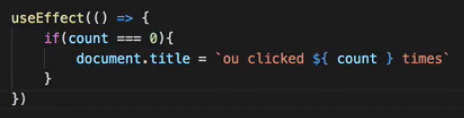




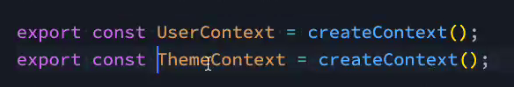
**Hook规则**

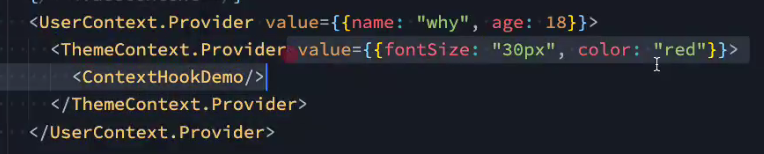


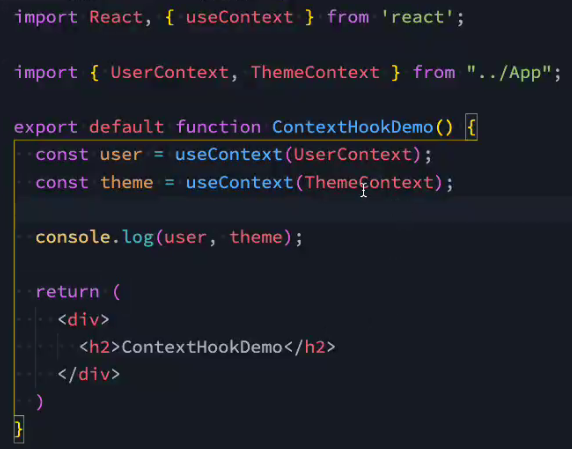
**错误**

**正确**

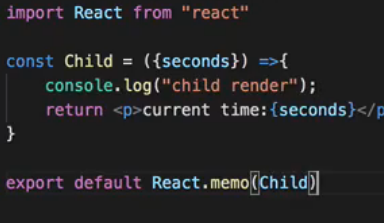
**useContext**



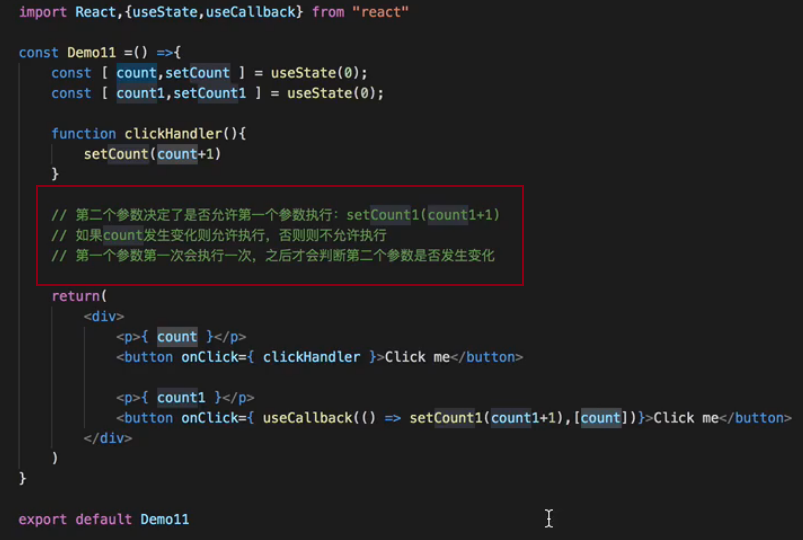


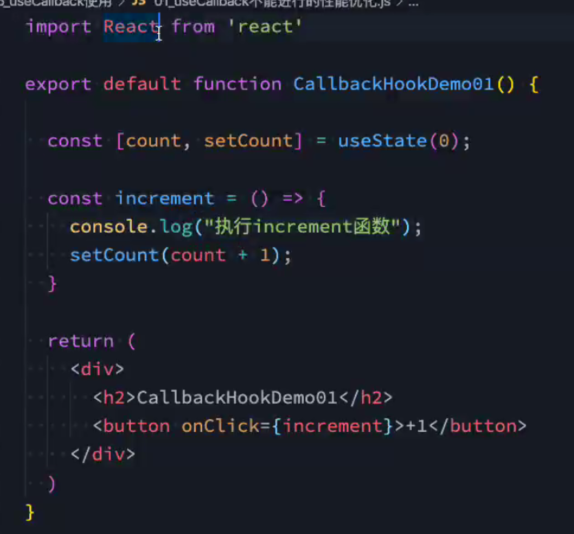


**React.Memo (与React.PureComponent类似，只能用于函数组件，不适合用于类组件)**

**防止Child组件过度重新渲染**

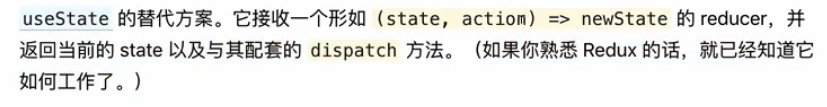
**useCallback**

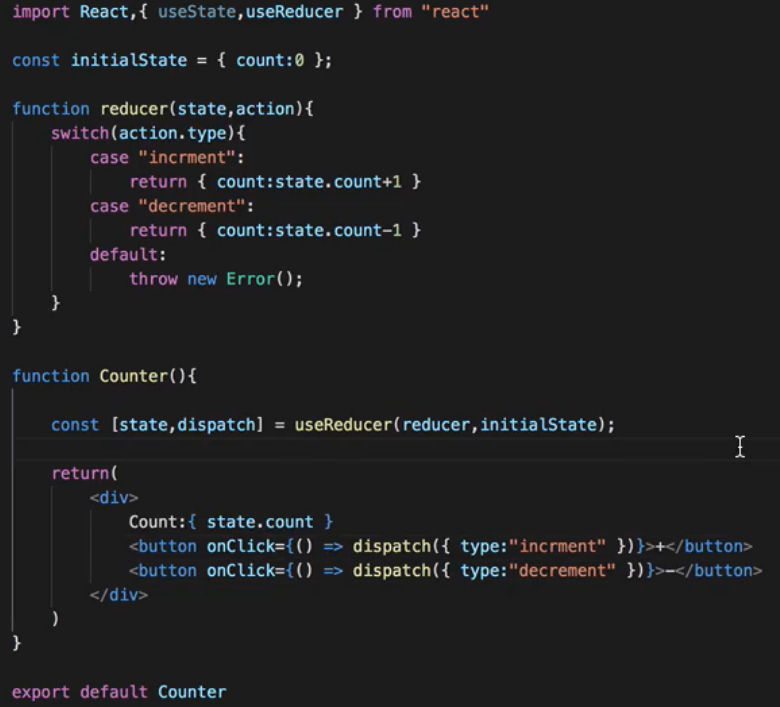
**其他值改变并不会重新在渲染，只有count改变才会重新渲染**

 **函数这里会重复调用**

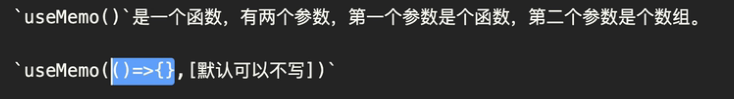


**useReducer**





**useMemo**



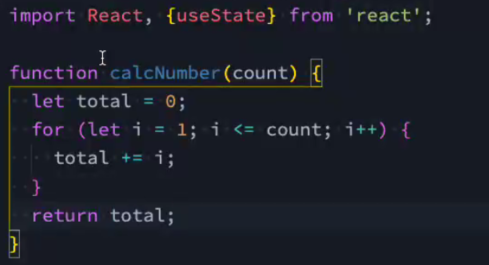
**第二个参数不写 ：会执行第一个参数的函数**

**第二个参数为空数组：不会执行第一个参数的函数**

**第二个参数（数组）内的值改变，会触发函数的执行**



**复杂计算时**



**只有count发生改变时才会重新计算循环，否则不会反复执行计算循环了。**

**传入子组件应用类型时**



**这里info会随着show变化不断创建新的对象，因此子组件也会重新渲染。**

**优化后**

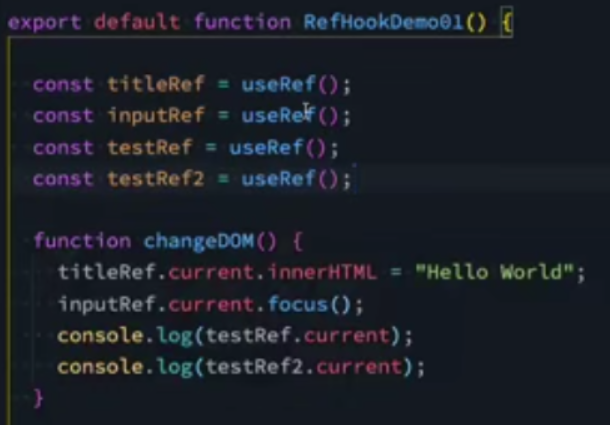


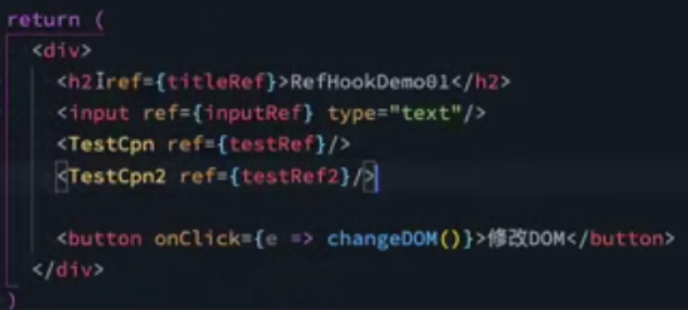
**Usecalback和usememo区别**



**useRef**

**引用DOM**



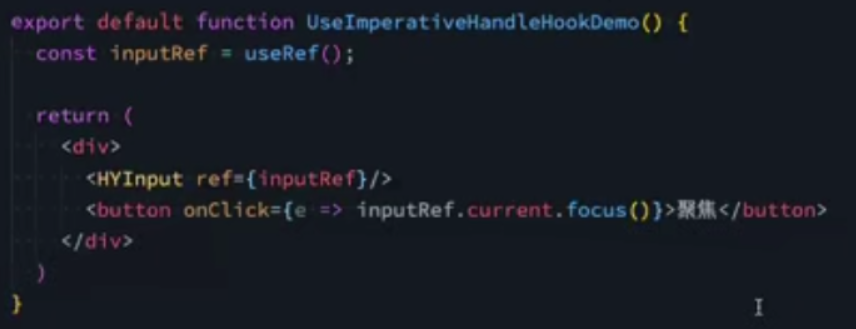


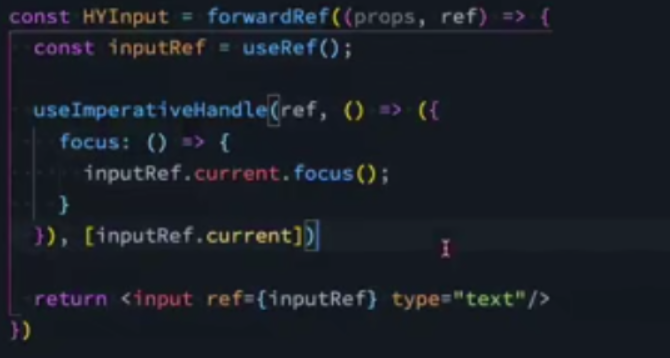
**引用其他数据**



一开始numRef.current一开始为0，显示0，count变为了10，这时候执行useEffect后numRef.current变为10但为渲染，当下次点击时候显示10，count变为20并显示

**useImperativeHandle**





这样子是为了不让父组件随意操控子组件

**useLayoutEffect**

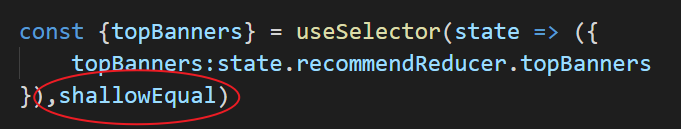


这里会先显示0，再立马变为随机数

这里直接变为随机数

**useDispatch,useSelector（将组件与store相关联）**





**通过shallowEqual进行浅层比较来提高性能**

