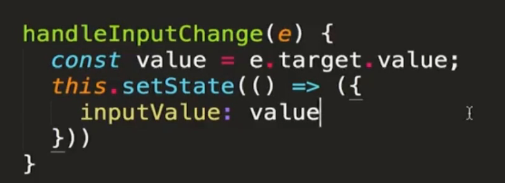
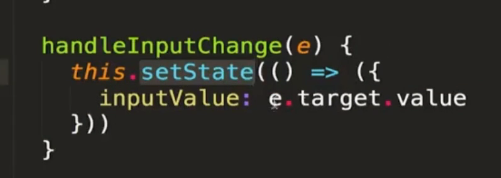


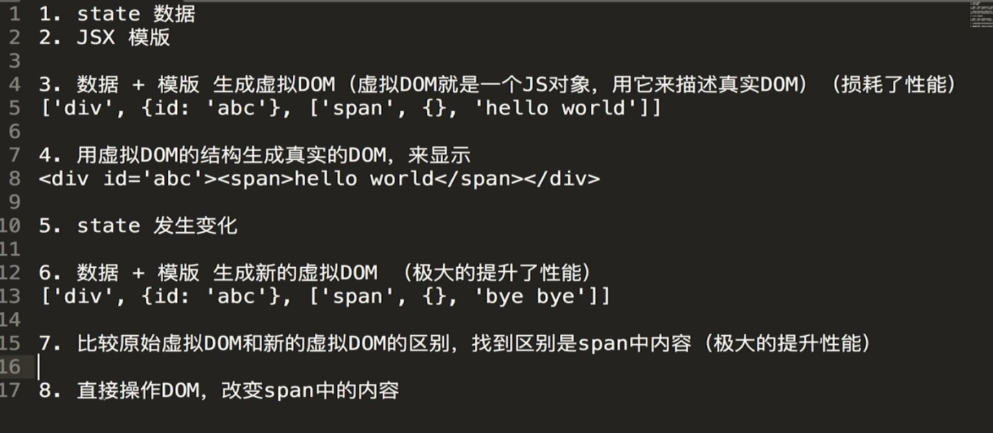
通过 React 元素处理事件跟在 DOM 元素上处理事件非常相似。但是有一些语法上的区别：

* React 事件使用驼峰命名，而不是全部小写。
* 通过 JSX , 你传递一个函数作为事件处理程序，而不是一个字符串。
* 在 React 中你不能通过返回 false, 阻止默认行为。必须明确调用 preventDefault
* 当使用 React 时，你一般不需要调用 addEventListener 在 DOM 元素被创建后添加事件监听器。相反，只要当元素被初始渲染的时候提供一个监听器就可以了。
* 在 JavaScript 中，类方法默认没有 [绑定](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_objects/Function/bind) 的。如果你忘记绑定 this.handleClick 并将其传递给onClick，那么在直接调用该函数时，this 会是 undefined 。(通过：属性初始化、bind、箭头函数3个方法都可以完成，推荐属性初始化 )

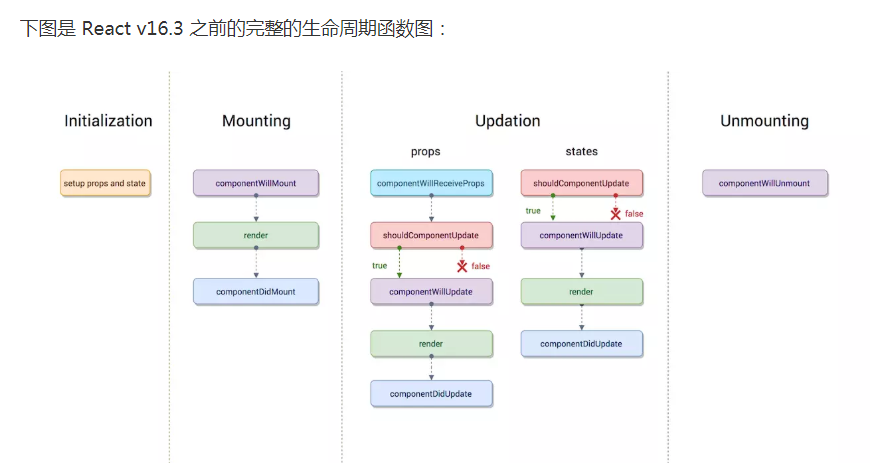
SetState(函数)异步更新

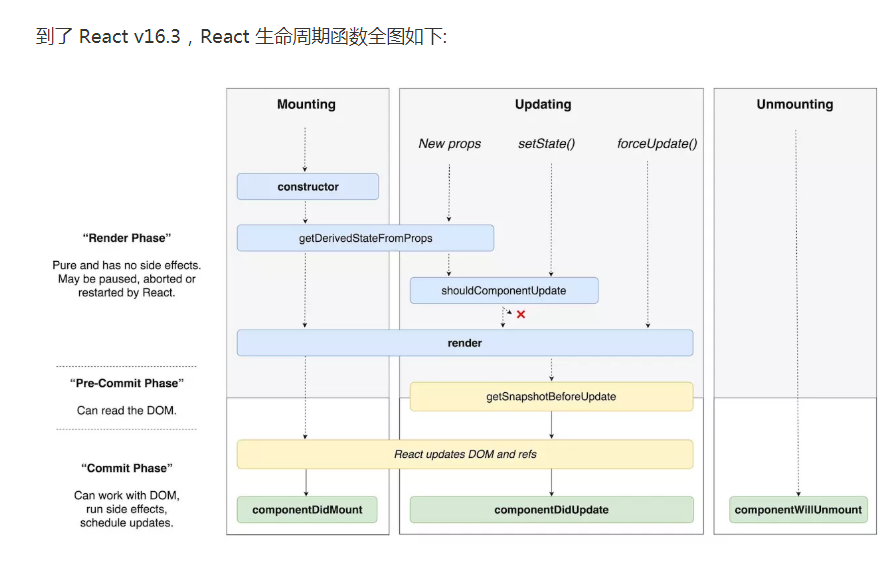


* componentDidUpdate(object prevProps, object prevState)
* shouldComponentUpdate(object nextProps, object nextState)：组件判断是否重新渲染时调用



shouldComponentUpdate(). shouldComponentUpdate should not be used when extending React.PureComponent. Please extend React.Component if shouldComponentUpdate is used.





static getDerivedStateFromProps(nextProps, prevState) {

//根据nextProps和prevState计算出预期的状态改变，返回结果会被送给setState

}

说getDerivedStateFromProps取代componentWillReceiveProps是不准确的，因为componentWillReceiveProps只在Updating过程中才被调用，而且只在因为父组件引发的Updating过程中才被调用（往上翻看第一个图）；而getDerivedStateFromProps在Updating和Mounting过程中都会被调用。

React v16.3还引入了一个新的声明周期函数getSnapshotBeforeUpdate(prevProps, prevState)

getSnapshotBeforeUpdate，这函数会在render之后执行，而执行之时DOM元素还没有被更新，给了一个机会去获取DOM信息，计算得到一个snapshot，这个snapshot会作为componentDidUpdate的第三个参数传入。

getSnapshotBeforeUpdate(prevProps, prevState) {

console.log('#enter getSnapshotBeforeUpdate');

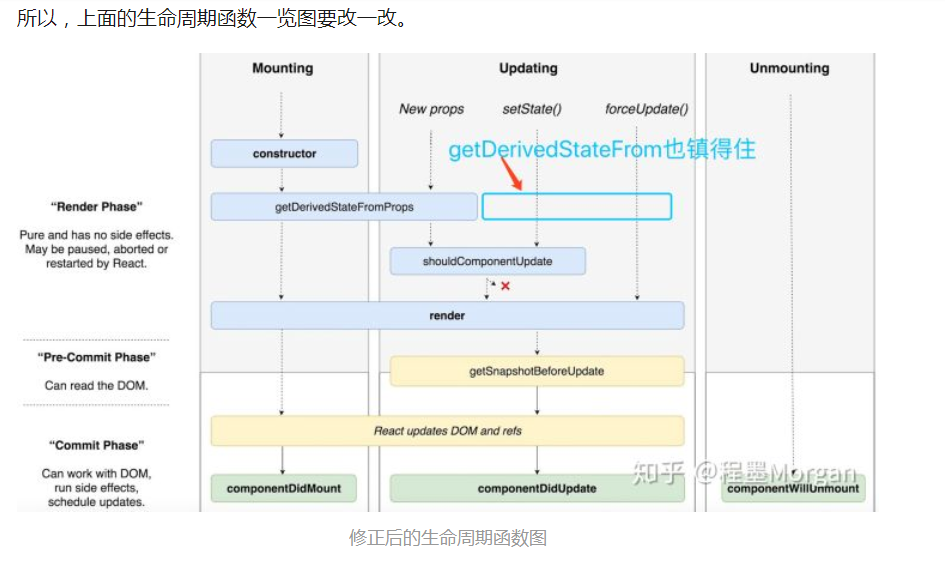
return 'foo';

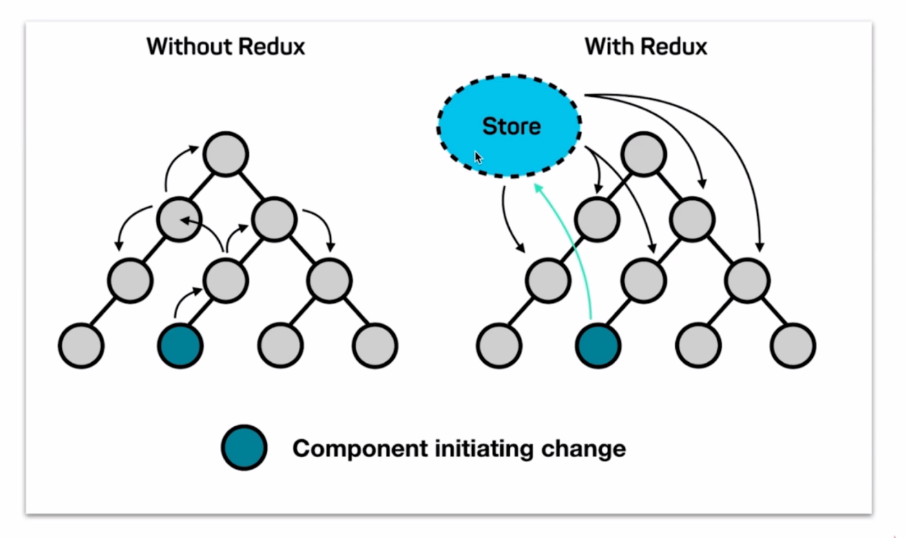
}

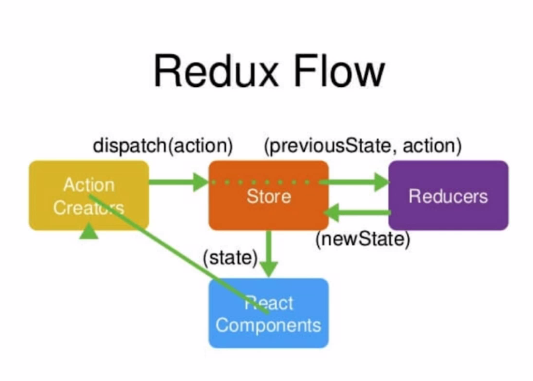
componentDidUpdate(prevProps, prevState, snapshot) {

console.log('#enter componentDidUpdate snapshot = ', snapshot);

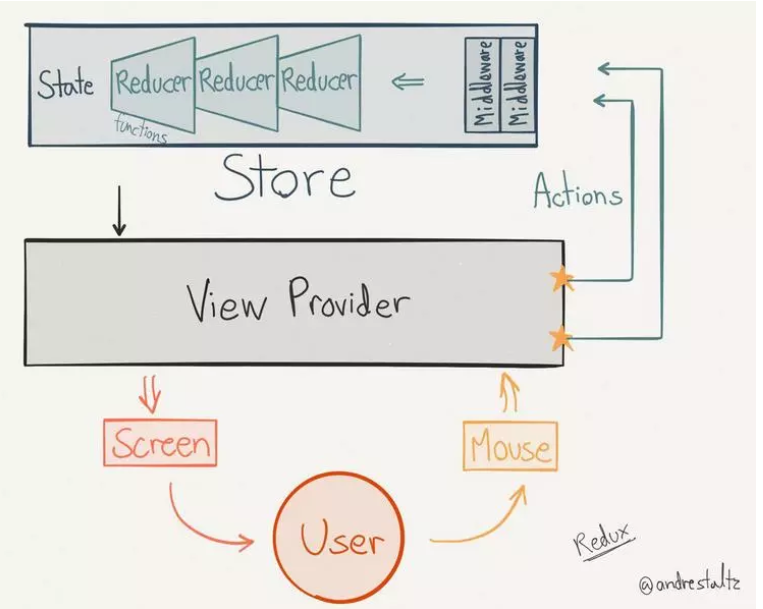
}

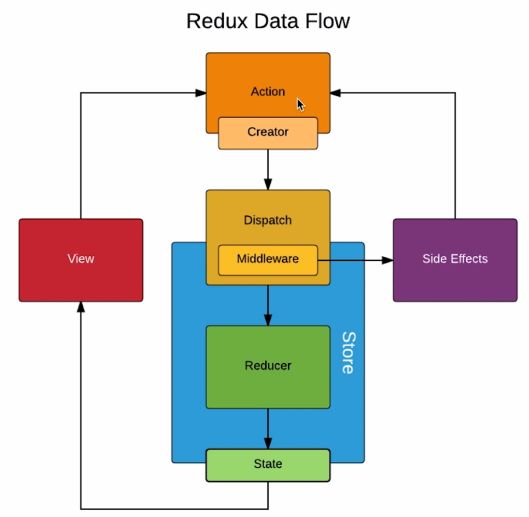






Redux中间件：





阿帕奇服务器：

