Keys 是 React 用于追踪哪些列表中元素被修改、被添加或者被移除的辅助标识。

在代码中调用 setState 函数之后，React 会将传入的参数对象与组件当前的状态合并，然后触发所谓的调和过程（Reconciliation），重新渲染。

初始化阶段：

getDefaultProps:获取实例的默认属性

getInitialState:获取每个实例的初始化状态

shouldComponentUpdate:组件接受到新属性或者新状态的时候（可以返回 false，接收数据后不更新，阻止 render 调用，后面的函数不会被继续执行了）

为什么虚拟 dom 会提高性能?(必考)

虚拟 dom 相当于在 js 和真实 dom 中间加了一个缓存，利用 dom diff 算法避免了没有必要的 dom 操作，从而提高性能。

用 JavaScript 对象结构表示 DOM 树的结构；然后用这个树构建一个真正的 DOM 树，插到文档当中当状态变更的时候，重新构造一棵新的对象树。然后用新的树和旧的树进行比较，记录两棵树差异把 2 所记录的差异应用到步骤 1 所构建的真正的 DOM 树上，视图就更新了。

### react diff 原理（常考，大厂必考）

把树形结构按照层级分解，只比较同级元素。

给列表结构的每个单元添加唯一的 key 属性，方便比较。

React 只会匹配相同 class 的 component（这里面的 class 指的是组件的名字）

合并操作，调用 component 的 setState 方法的时候, React 将其标记为 dirty.到每一个事件循环结束, React 检查所有标记 dirty 的 component 重新绘制.

选择性子树渲染。开发人员可以重写 shouldComponentUpdate 提高 diff 的性能。

Refs 是 React 提供给我们的安全访问 DOM 元素或者某个组件实例的句柄。

展示组件关心组件看起来是什么。只关心 UI 状态而不是数据的状态。

容器组件则更关心组件是如何运作的。容器组件会为展示组件或者其它容器组件提供数据和行为(behavior)，它们会调用 Flux actions，并将其作为回调提供给展示组件。容器组件经常是有状态的，因为它们是(其它组件的)数据源。

### 类组件：自身的状态和生命周期钩子

### 函数式组件： '无状态组件(stateless component)'，这种组件也被称为哑组件(dumb components)或展示组件

props 由父组件传递给子组件，并且就子组件而言，props 是不可变的

State 是一种数据结构，用于组件挂载时所需数据的默认值。

高阶组件是一个以组件为参数并返回一个新组件的函数。

### 为什么建议传递给 setState 的参数是一个 callback 而不是一个对象

因为 this.props 和 this.state 的更新可能是异步的，不能依赖它们的值去计算下一个 state。

### (在构造函数中)调用 super(props) 的目的是什么

在 super() 被调用之前，子类是不能使用 this 的，传递 props 给 super() 的原因则是便于(在子类中)能在 constructor 访问 this.props。

在 React 组件中，应该在 componentDidMount 中发起网络请求。组件上调用 setState才起作用。不能操作未挂载的dom



Flux 的最大特点，就是数据的"单向流动"