🡪

React整个的渲染机制就是React会调用render()函数构建一棵Dom树. 在state/props发生改变的时候，默认情况下,由于shouldComponentUpdate总是返回true,当shouldComponentUpdate返回true的时候，render()函数会被再次调用渲染出另外一棵树，重新渲染所有的节点，构造出新的虚拟Dom tree跟原来的Dom tree用Diff算法进行比较，找到需要更新的地方批量改动【改动可能是改变DOM节点内容，或者替换DOM节点，一般是如果只是html元素内容文字不同，所含html元素，html元素属性值一样则React不会替换该DOM节点。如果React发现此次render出来的虚拟DOM内容，多或者少了html元素，多或者少了html元素属性值则会直接替换该DOM】再渲染到真实的DOM上，由于这样做就减少了对Dom的频繁操作，从而提升的性能。

🡪

React提供forceUpdate()方法，用于强制组件进行刷新,它将让组件跳过shouldComponentUpdate方法，直接进行render（）方法进行重新渲染。提供的目的是，有些时候无法通过调用this.setState({})方法来更新组件，但是某些外部数据发生了变化需要显示在组件的时候，调用forceUpdate()方法。

🡪

如果父组件的执行render方法，那么其所有的子组件默认情况下都会执行自身的render方法，不管子组件是否使用了父组件传来的props,或者接受的父组件props是否发生改变。

我们可以重定义子组件的 shouldComponentUpdate方法来防止子组件随着父组件重新渲染，从而减少调用子组件无必要的render方法，提高性能。

🡪

React的生命周期：就是React组件通过组件生命周期函数所经过的生命周期。组件生命周期会调用对应的函数。

我们主要将React的组件分为主要四个阶段，分别为Mounting阶段，Unmounting阶段，Props Change阶段，State Change阶段。

**Mounting Stage:**

render

ComponentWillMount

Class constructor

在产生组件前调用，只调用一次，在此阶段设置state不会引起重新渲染

ComponentDidMount

在组件渲染之后马上调用，此时DOM已经形成可以进行访问。通常在这个方法中使用Resful或者其他手段在远端获取数据提供给组件。

**Unmounting Stage:**

在组件卸载前调用，用于清理操作，例如移除事件计时器, 关闭网络连等等.当组件被unmount以后，React没有方法再恢复组件。唯有重新创建。

ComponentWillUnmount

**Props Changes Stage:**

当组件收到新的属性时候调用，在此方法中使用setState不会触发渲染

ShouldComponentUpdate

ComponentWillReceiveProps

这个方法在render方法前触发，用于决定重新渲染是否必要。主要用于性能优化。

ComponentWillUpdate

当组件收到新的属性或者状态的时候触发，setState方法不允许在此方法中调用。此方法主要用于准备更新而不是再次触发更新。

componentDidUpdate

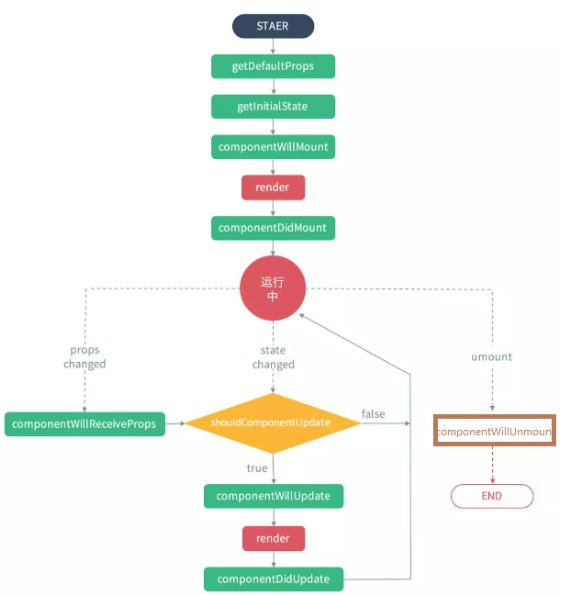
Render

此方法触发，当中更新flush到DOM当中。

**State Changes Stage:**

State Change的生命周期和Props的生命周期相似，除了没有ComponentWillReceiveProps。

将各个stage组合起来的整体流程如下图所示：



🡪

重写shouldComponentUpdate方法可以改变该组件的更新行为。规则为shouldComponentUpdate的第一个参数是新的props, 第二个参数是新的state. 如果该方法返回true则进行调用render方法，重新渲染新的DOM Tree，然后React进行DOM Diff算法。并且进入componentDidUpdate方法。如果返回false则放弃更新, 不调用render方法，并且由于不调用render方法渲染新的DOM Tree, React也不会进行DOM Diff比较。

🡪

getDerivedStateFromProps方法在mount阶段的render前调用，或者在更新阶段的shouldComponentUpdate方法前调用。

该方法用于在Props基础上构建一些state.该方法为静态方法，所以不能使用this关键字。该方法接受的第一个参数为props，以及第二个参数代表现在状态的state对象，要求返回一个新的state对象。

如果在组件中定义了该静态方法，则每当组件中的状态发生变化，它都将被执行根据props以及现在的state状态，计算出新的state对象。

🡪