# React笔记

## React生命周期

|  |
| --- |
| 实例化  getDefaultProps  getInitialState  componentWillMount  render  componentDidMount  存在期  componentWillReceiveProps  shouldComponentUpdate  componentWillUpdate  componentDidUpdate  销毁时  componentWillUnmount |

## React调用事件处理方法时，怎么调用到this

|  |
| --- |
| 1. this.foo.bind(this)  2. 可以使用箭头函数  3. 可以使用lodash-decorators里面的Bind装饰器 |

## React为什么要bind(this)

|  |
| --- |
| 1. 作用域的问题，foo() {} 直接使用this,作用域仅在该方法内部,foo() {}里面使用外部成员，需要bind(this)  2. 与 const foo = () => {}里的this作用域不一样 |

## React render工作

|  |
| --- |
| 主要分首次渲染与更新来说，说的比较多，不赘述，网上react源码分析一大堆... |

## react注意

|  |
| --- |
| 1. JSX做表达式判断时候，需要强转为boolean类型  // 如果不使用 !!b 进行强转数据类型，会在页面里面输出 0  render() {  const b = 0;  return <div>  {  !!b && <div>这是一段文本</div>  }  </div>  }  2、尽量不要在 componentWillReviceProps 里使用 setState，如果一定要使用，那么需要判断结束条件，不然会出现无限重渲染，导致页面崩溃。(实际不是componentWillReviceProps会无限重渲染，而是componentDidUpdate)  3、给组件添加ref时候，尽量不要使用匿名函数，因为当组件更新的时候，匿名函数会被当做新的prop处理，让ref属性接受到新函数的时候，react内部会先清空ref，也就是会以null为回调参数先执行一次ref这个props，然后在以该组件的实例执行一次ref，所以用匿名函数做ref的时候，有的时候去ref赋值后的属性会取到null。  详情见  4、遍历子节点的时候，不要用 index 作为组件的 key 进行传入。 |

## react的虚拟dom是怎么实现的

|  |
| --- |
| 首先说说为什么要使用Virturl DOM，因为操作真实DOM的耗费的性能代价太高，所以react内部使用js实现了一套dom结构，在每次操作在和真实dom之前，使用实现好的diff算法，对虚拟dom进行比较，递归找出有变化的dom节点，然后对其进行更新操作。为了实现虚拟DOM，我们需要把每一种节点类型抽象成对象，每一种节点类型有自己的属性，也就是prop，每次进行diff的时候，react会先比较该节点类型，假如节点类型不一样，那么react会直接删除该节点，然后直接创建新的节点插入到其中，假如节点类型一样，那么会比较prop是否有更新，假如有prop不一样，那么react会判定该节点有更新，那么重渲染该节点，然后在对其子节点进行比较，一层一层往下，直到没有子节点 |

## react 的渲染过程中，兄弟节点之间是怎么处理的？也就是key值不一样的时候

|  |
| --- |
| 通常我们输出节点的时候都是map一个数组然后返回一个ReactNode，为了方便react内部进行优化，我们必须给每一个reactNode添加key，这个key prop在设计值处不是给开发者用的，而是给react用的，大概的作用就是给每一个reactNode添加一个身份标识，方便react进行识别，在重渲染过程中，如果key一样，若组件属性有所变化，则react只更新组件对应的属性；没有变化则不更新，如果key不一样，则react先销毁该组件，然后重新创建该组件。 |

## react diff算法(三层diff)

|  |
| --- |
|  |

## React中的controlled component 和 uncontrolled component区别（受控组件和不受控组件）

|  |
| --- |
| 受控组件  <https://link.juejin.im/?target=https%3A%2F%2Freact.docschina.org%2Fdocs%2Fforms.html>  非受控组件  [https://link.juejin.im/?target=https%3A%2F%2Freact.docschina.org%2Fdocs%2Funcontrolled-components.html](https://link.juejin.im/?target=https%3A%2F%2Freact.docschina.org%2Fdocs%2Funcontrolled-components.html)) |

## react-router内部实现机制

|  |
| --- |
| react-router的实现原理  <https://link.juejin.im/?target=https%3A%2F%2Fblog.csdn.net%2Ftangzhl%2Farticle%2Fdetails%2F79696055> |

## setState是异步的还是同步的

|  |
| --- |
|  |

## 子组件和父组件componentDidMount哪一个先执行

|  |
| --- |
|  |

## redux的一般流程

## react组件的优化

## react组件的通信

## react 的virtual dom和diff算法的实现方式

## MVC、MVVM了解么，数据双向绑定和单向绑定实现方式

## ****mvc和mvvm****

|  |
| --- |
| <https://link.juejin.im/?target=http%3A%2F%2Fwww.ruanyifeng.com%2Fblog%2F2015%2F02%2Fmvcmvp_mvvm.html>  <https://juejin.im/post/593021272f301e0058273468>  <http://www.ruanyifeng.com/blog/2015/02/mvcmvp_mvvm.html> |

## react-router实现方式，单页面应用相关东西

## react的ssr了解么？大概怎么实现

## react的理念是什么

|  |
| --- |
| 拿函数式编程来做页面渲染 |

## react设计思路

## 为什么虚拟DOM比真实DOM性能好

## react常见的通信方式

## redux整体的工作流程

## redux和全局对象之间的区别

## redux数据回溯设计思路

## react性能优化

## 介绍pureComponet

## 介绍Function Component

## React数据流

## props和state的区别

## 介绍react context

## React15/16.x的区别

## 重新渲染render会做些什么

## 哪些方法会触发react重新渲染

## state和props触发更新的生命周期分别有什么区别

## setState是同步还是异步

## 对无状态组件的理解

## 介绍Redux工作流程

## 对react看法，它的优缺点