**React面试题**

目录

[1.angularJs 和 React 区别 1](#_Toc20141)

[2.redux 中间件 2](#_Toc27304)

[3.redux 有什么缺点 2](#_Toc26207)

[4.React 组件的划分业务组件技术组件？ 2](#_Toc16629)

[5. React 生命周期函数 2](#_Toc20980)

[6.React 性能优化是哪个周期函数？ 3](#_Toc24289)

[8. diff 算法? 3](#_Toc20947)

[9.React 性能优化方案 4](#_Toc24448)

[10.简述 flux 思想 4](#_Toc10204)

[React 项目用过什么脚手架？Mern? Yeoman? 4](#_Toc19409)

[12.你了解 React 吗？ 5](#_Toc13639)

[13.React 解决了什么问题？ 5](#_Toc21393)

[14.React 的协议？ 5](#_Toc6306)

[15.了解 shouldComponentUpdate 吗？ 5](#_Toc6611)

[16.React 的工作原理？ 5](#_Toc21632)

[17.使用 React 有何优点？ 6](#_Toc7016)

[18. 展示组件(Presentational component)和容器组件(Container component)之间有 何不同？ 6](#_Toc3285)

[19.类组件(Class component)和函数式组件(Functional component)之间有何不 6](#_Toc9548)

[同？ 6](#_Toc6174)

[20.(组件的)状态(state)和属性(props)之间有何不同？ 7](#_Toc17333)

[21.应该在 React 组件的何处发起 Ajax 请求？ 7](#_Toc19129)

[22.在 React 中，refs 的作用是什么？ 7](#_Toc5761)

[23.何为高阶组件(higher order component)？ 8](#_Toc22049)

[24.使用箭头函数(arrow functions)的优点是什么？ 8](#_Toc7689)

[25. 为什么建议传递给 setState 的参数是一个 callback 而不是一个对象？ 8](#_Toc8938)

[26.除了在构造函数中绑定 this，还有其它方式吗？ 8](#_Toc27564)

[27.怎么阻止组件的渲染？ 9](#_Toc939)

[28. 当渲染一个列表时，何为 key？设置 key 的目的是什么？ 9](#_Toc15243)

[29.何为 JSX ？ 9](#_Toc20545)

# 1.angularJs 和 React 区别

React 对比 Angular 是思想上的转变，它也并不是一个库，是一种开发理念，组件化，分

治的管理，数据与 view 的一体化。它只有一个中心,发出状态，渲染 view，对于虚拟 dom

它并没有提高渲染页面的性能，它提供更多的是利用 jsx 便捷生成 dom 元素，利用组件

概念进行分治管理页面每个部分(例如 header section footer slider)

# 2.redux 中间件

中间件提供第三方插件的模式，自定义拦截 action -> reducer 的过程。变为 action ->

middlewares -> reducer 。这种机制可以让我们改变数据流，实现如异步 action ，

action 过

滤，日志输出，异常报告等功能。

常见的中间件：redux-logger：提供日志输出；

redux-thunk：处理异步操作；

redux-promise：

处理异步操作；actionCreator 的返回值是 promise

# 3.redux 有什么缺点

1.一个组件所需要的数据，必须由父组件传过来，而不能像 flux 中直接从 store 取。

2.当一个组件相关数据更新时，即使父组件不需要用到这个组件，父组件还是会重新

render，可能会有效率影响，或者需要写复杂的 shouldComponentUpdate 进行判断。

# 4.React 组件的划分业务组件技术组件？

根据组件的职责通常把组件分为 UI 组件和容器组件。UI 组件负责 UI 的呈现，容器组

件负责管理数据和逻辑。两者通过 React-Redux 提供 connect 方法联系起来。

# React 生命周期函数

一、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

componentWillMount：组件即将被装载、渲染到页面上

render:组件在这里生成虚拟的 DOM 节点

componentDidMount:组件真正在被装载之后

二、运行中状态：

componentWillReceiveProps:组件将要接收到属性的时候调用

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）

componentWillUpdate:组件即将更新不能修改属性和状态

render:组件重新描绘

componentDidUpdate:组件已经更新

三、销毁阶段：

componentWillUnmount:组件即将销毁

6.React 性能优化是哪个周期函数？

shouldComponentUpdate 这个方法用来判断是否需要调用 render 方法重新描绘 dom。因

为 dom 的描绘非常消耗性能，如果我们能在 shouldComponentUpdate 方法中能够写出更

优化的 dom diff 算法，可以极大的提高性能。

**7**.为什么虚拟 dom 会提高性能?

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。

具体实现步骤如下：

1.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；

2.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。

# diff 算法?

1.把树形结构按照层级分解，只比较同级元素。

2.给列表结构的每个单元添加唯一的 key 属性，方便比较。

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

4.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.

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

# 9.React 性能优化方案

1）重写 shouldComponentUpdate 来避免不必要的 dom 操作。

2）使用 production 版本的 React.js。

3）使用 key 来帮助 React 识别列表中所有子组件的最小变化



# 10.简述 flux 思想

Flux 的最大特点，就是数据的"单向流动"。

1.用户访问 View

2.View 发出用户的 Action

3.Dispatcher 收到 Action，要求 Store 进行相应的更新

4.Store 更新后，发出一个"change"事件

5.View 收到"change"事件后，更新页面

# React 项目用过什么脚手架？Mern? Yeoman?

Mern：MERN 是脚手架的工具，它可以很容易地使用 Mongo, Express, React and NodeJS

生成同构 JS 应用。它最大限度地减少安装时间，并得到您使用的成熟技术来加速开发。

# 12.你了解 React 吗？

了解，React 是 facebook 搞出来的一个轻量级的组件库，用于解决前端视图层的一些问

题，就是 MVC 中 V 层的问题，它内部的 Instagram 网站就是用 React 搭建的。

# 13.React 解决了什么问题？

解决了三个问题： 1.组件复用问题， 2.性能问题，3.兼容性问题：

# 14.React 的协议？

React 遵循的协议是“

BSD 许可证 + 专利开源协议”，这个协议比较奇葩，如果你的产

品跟 facebook 没有竞争关系，你可以自由的使用 React，但是如果有竞争关系，你的 React

的使用许可将会被取消

# 15.了解 shouldComponentUpdate 吗？

React 虚拟 dom 技术要求不断的将 dom 和虚拟 dom 进行 diff 比较，如果 dom 树比价大，

这种比较操作会比较耗时，因此 React 提供了 shouldComponentUpdate 这种补丁函数，如

果对于一些变化，如果我们不希望某个组件刷新，或者刷新后跟原来其实一样，就可以

使用这个函数直接告诉 React，省去 diff 操作，进一步的提高了效率。

# 16.React 的工作原理？

React 会创建一个虚拟 DOM(virtual DOM)。当一个组件中的状态改变时，React 首先会

通过 "diffing" 算法来标记虚拟 DOM 中的改变，第二步是调节(reconciliation)，会用 diff

的结果来更新 DOM。

# 17.使用 React 有何优点？

1.只需查看 render 函数就会很容易知道一个组件是如何被渲染的 JSX 的引入，使得组件的代码更加可读，也更容易看懂组件的布局，或者组件之间是

如何互相引用的

3.支持服务端渲染，这可以改进 SEO 和性能

4.易于测试

5.React 只关注 View 层，所以可以和其它任何框架(如 Backbone.js, Angular.js)一起使用

1. 展示组件(Presentational component)和容器组件(Container component)之间有 何不同？

1.展示组件关心组件看起来是什么。展示专门通过 props 接受数据和回调，并且几乎不

会有自身的状态，但当展示组件拥有自身的状态时，通常也只关心 UI 状态而不是数据

的状态。

2.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

# 19.类组件(Class component)和函数式组件(Functional component)之间有何不

# 同？

1.类组件不仅允许你使用更多额外的功能，如组件自身的状态和生命周期钩子，也能使

组件直接访问 store 并维持状态

2.当组件仅是接收 props，并将组件自身渲染到页面时，该组件就是一个 '无状态组件

(stateless component)'，可以使用一个纯函数来创建这样的组件。这种组件也被称为哑组

件(dumb components)或展示组件

20.(组件的)状态(state)和属性(props)之间有何不同？

1. State 是一种数据结构，用于组件挂载时所需数据的默认值。State 可能会随着时间的

推移而发生突变，但多数时候是作为用户事件行为的结果。

2. Props(properties 的简写)则是组件的配置。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(immutable)。组件不能改变自身的 props，但是可以把其子组件的 props 放在一起(统一管理)。Props 也不仅仅是数据--回调函数也可以通过 props

传递。

# 21.应该在 React 组件的何处发起 Ajax 请求？

在 React 组件中，应该在 componentDidMount 中发起网络请求。这个方法会在组件第

一次“挂载”(被添加到 DOM)时执行，在组件的生命周期中仅会执行一次。更重要的是，

你不能保证在组件挂载之前 Ajax 请求已经完成，如果是这样，也就意味着你将尝试在

一个未挂载的组件上调用 setState，这将不起作用。在 componentDidMount 中发起网络

请求将保证这有一个组件可以更新了。

22.在 React 中，refs 的作用是什么？

Refs 可以用于获取一个 DOM 节点或者 React 组件的引用。何时使用 refs 的好的示例

有管理焦点/文本选择，触发命令动画，或者和第三方 DOM 库集成。你应该避免使用

String 类型的 Refs 和内联的 ref 回调。Refs 回调是 React 所推荐的。

# 23.何为高阶组件(higher order component)？

高阶组件是一个以组件为参数并返回一个新组件的函数。HOC 运行你重用代码、逻辑

和引导抽象。最常见的可能是 Redux 的 connect 函数。除了简单分享工具库和简单的

组合，HOC 最好的方式是共享 React 组件之间的行为。如果你发现你在不同的地方写

了大量代码来做同一件事时，就应该考虑将代码重构为可重用的 HOC。



# 24.使用箭头函数(arrow functions)的优点是什么？

1. 作用域安全：在箭头函数之前，每一个新创建的函数都有定义自身的 this 值(在构造

函数中是新对象；在严格模式下，函数调用中的 this 是未定义的；如果函数被称为“对

象方法”，则为基础对象等)，但箭头函数不会，它会使用封闭执行上下文的 this 值。

2. 简单：箭头函数易于阅读和书写

3. 清晰：当一切都是一个箭头函数，任何常规函数都可以立即用于定义作用域。开发者

总是可以查找 next-higher 函数语句，以查看 this 的值

1. 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。

# 26.除了在构造函数中绑定 this，还有其它方式吗？

可以使用属性初始值设定项(property initializers)来正确绑定回调，create-React-app 也是

默认支持的。在回调中你可以使用箭头函数，但问题是每次组件渲染时都会创建一个新

的回调。

# 27.怎么阻止组件的渲染？

在组件的 render 方法中返回 null 并不会影响触发组件的生命周期方法

# 当渲染一个列表时，何为 key？设置 key 的目的是什么？

Keys 会有助于 React 识别哪些 items 改变了，被添加了或者被移除了。Keys 应该被赋

予数组内的元素以赋予(DOM)元素一个稳定的标识，选择一个 key 的最佳方法是使用一

个字符串，该字符串能惟一地标识一个列表项。很多时候你会使用数据中的 IDs 作为

keys，当你没有稳定的 IDs 用于被渲染的 items 时，可以使用项目索引作为渲染项的

key，但这种方式并不推荐，如果 items 可以重新排序，就会导致 re-render 变慢



(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，在 ES2015 中，子类必须在 constructor

中调用 super()。传递 props 给 super() 的原因则是便于(在子类中)能在 constructor 访问

this.props。

# 29.何为 JSX ？

JSX 是 JavaScript 语法的一种语法扩展，并拥有 JavaScript 的全部功能。JSX 生产

React "元素"，你可以将任何的 JavaScript 表达式封装在花括号里，然后将其嵌入到 JSX

中。在编译完成之后，JSX 表达式就变成了常规的 JavaScript 对象，这意味着你可以在

if 语句和 for 循环内部使用 JSX，将它赋值给变量，接受它作为参数，并从函数中返回

它。