web前端面试题附答案002-说一下react组件间的数据传递

答：

一：其实这个问题已经问了很多年了，但还是有很多人会挂在这道题上。网上的答案也是一搜一大把，如果你用过react做过一两个项目，不是特别小的项目的话，如果你平时认真总结，也能说的差不多。但很明显，前端发展到今天，很多回答是不能满足面试官的胃口的。

面试官有的他可能对react也不熟，所以你要回答出你的气场，回答有条理，第一种，第二种等等；面试官可能只是想听一下你的简单回答，那么到目前为止，网上的答案或者你平时总结一些也都可以答完这道题，但随着疫情严重，环境恶劣，仅仅回答条目怕是难以让面试官对你倾心；还有的面试官比较苛刻，或者基于现在大环境的考量，我今天写一下我通过和各大厂同学交流后，大家希望得到的答案。

很多面试官在面试的时候比较反感的就是指导性回答，比如让你说一下数组去重，你简单的说了一句indexof，就像是在指导别人工作一样，可能平时工作这样子一点问题也没有，但面试的时候，面试官或者一面的技术面，他最大程度的决定着你能不能通过面试。很多人如果技术面过了，后边的老板面，HR面反而好通过。后边我还会为大家写老板面的相关文章。

所以总结起来就是，react组件间的数据传递这道题，在线上面试的前提下，在你不写代码的前提下，在只是语音描述的前提下，如何让面试官觉得舒服。正所谓人无我有，人有我有，其实内卷早就存在，只不过这个词当时不存在而已。

二、开始按条理回答

1. 共享式传递

因为react组件最终打出来的业务代码js会

下方，所以我们在顶部添加window.name可以使全局react组件达到共享的目的，所以与之类似的还有localstorage，sessionStorage,cookie等，都是为了共享

2. 普通的数据修改传递

如果做的项目过于简单，是没有必要非得引入redux的，要知道引入rudex也是需要成本的。其实也就是dedux的初级版本。例如在当前组件发起ajax请求，当服务端修改数据后，

目标组件再请求数据，达到传递效果

3. 与上一条类似，在某种场景下不发ajax请求，为什么不发呢，只是前端自己闭环做事情和前后端联调所需成本毕竟不同。你可以通过node的fs去写文件，写内容，在目标组件再次读取这个通过fs写的文件内容

4. 可能你也不想在业务代码js顶部写window.name类似的全局变量，但因为redux代码来的冗余，那么建议在项目src根目录下添加store.js，当然只是取名为store.js，其实跟redux那个sotre不是一回事。这一步的store.js只是定义一些变量数据，当某个组件需要的时候再去import引入，当然也可以修改store.js里的变量。当目标组件需要用到的时候呢，再import这个store.js文件。这里已经不是单纯的window.name那样全局变量的概念了，而是通过模块化维护一套运行在react组件顶部的变量

5. 然后就开始说基于react技术栈的组件传递了。那么面试官的问题是react组件之间的数据传递，为什么前面还要啰嗦那么多呢。因为你的思路越多，可能就是面试官希望团队中寻求的那个人，证明你经验丰富，证明你热爱总结。要知道除了技术面，老板很多也是技术出身，所以他希望看到一个思路开阔的同学，这会让他眼前一亮。

6. 接下来就是父组件传递给子组件，使用过的都知道，父组件通过,而子组件通过解构props就可以拿到name值。我想现在几乎所有的答题者这一步也就是这么回答。但你有没有想过，在目前的大环境下，你这个职位有多少人在争取，而且他们可能已经好几个月没工作了，天天在家学习，还有很多大厂毕业生可能也是你的竞争对手，所以还是那个内卷原则，人无我有，人有我优。你就不能说到这里就结束。首先这个Child组件，我们传入了name=“jim”，子组件会有constructor函数，这个函数里同构super获取props这个对象，对象中包含着我们要传递的name值。那么这个constructor函数当然也可以不写，但是不手动写，代码还是会隐式添加上。而super函数中的props是一个参数，是父组件传递过来的对象，所以子组件没有this指向，必须通过super函数，将子组件做为this.props来指向父组件，这样也就拿到了传递过来的name值，而这也是react单项数据流的传输。

7. 子组件给父组件传递。其实有点类似回调函数了，子组件通过<Child oneHandleClick={e => this.count()}。而子组件通过props拿到oneHandleClick，就可以通过子组件自身的onClick去执行这个回调函数，然后执行到父组件去。

8. 兄弟组件，兄弟组件其实是父传子，子传父，父再传兄弟的过程。我想如果你完整的说出了前2条，而这一条你这样简练的回答，面试官也会觉得你总结上没有问题。其实面试时间有限，一般1个小时左右吧。所以该长了长，该短了少描述是一个不错的选择。但是短并非一味地简短，而是让面试官明白，你的简短概括其实他已经知道你要说的答案了。

9. 父传更深层次的子组件，如果你只说一句context就想结束，那恐怕不行。一定要注意，面试不能指导性回答，要诚恳回答。说自己如何使用，使用的东西是什么逻辑，甚至要讲优缺点。接着说context,你可以说给祖宗组件设定childContextTypes 和 getChildContext()，react自动深层次向下传递，当组件某个层次的子组件定义了contextTypes的时候，说明这个子组件有所需要，即可达到深层次数据传递。当然，这只是使用，由于现在是线上面试技术居多，你这么说，如果对面是一个明白人，肯定知道你至少是用过的了。但你一定要给自己加戏，我就是要说context的逻辑。但是谨记，面试讲究一个气场，意思就是你可以源码很少看过，但这个逻辑你要很自信的说出来。首先通过Reacct.createContext创建context对象，当然这个对象有一定的数据结构，但最重要的是保存值用的currentValue，还有provide和consumer这3个要说出来。而刚刚说使用场景的时候，getChildContext函数就是在给currentValue赋值，而react组件在最初生成的时候，是有个fiber树概念的，fiber是一个存有很多属性的用来表示整个react组件树的对象，所以在创建content的同时，还会去走react本身的修改数据的方法，这样修改了fiber树以后，子组件fiber节点自然就可以拿到了。

10. redux，项目比较复杂的时候会用到。虽然之前说过redux会给项目带来负担，但当项目比较复杂的时候，使用它带来的效果会比优于所谓的负担值。引入reudx大概会增加不到10K的代码体积吧。首先还是说redux的流转过程，业务组件里有个按钮，触发一个dispatch操作，然后发送的是action，而这个action可同步，可异步，继续向store流转，store接收到action后，又触发reducer去改变state数据，然后又把数据流转回最初的业务组件，闭环。但这个过程可能面试官不太想问，因为大家都知道，他可能会顺带问问你有没有遇到过什么问题，但其实正常使用，没有问题啊。React本身是单向数据流转，通过redux呢变成了双向数据流转，这个思路才是重点。千万谨记，面试的时候脑子里要有逻辑，而不是部分实现代码，回答要顺畅。Redux的逻辑大概就是集中某个地方识别变量或者数据的变化，预先注册回调函数，等感知到数据变化后调用回调函数。再具体点就是创建store对象的时候，会有几个内置函数，getState用来获取最新state的值，dispatch用来触发监听函数，subscribe用来注册监听函数。而要使用redux呢，react最外层组件要包一层provide组件，然后再借助刚刚说的context回答方案达到数据传递。

结语：怎么样？会不会觉得回答的太笼统，但不管你看几遍源码，概括起来就是这样子的。因为面试官肯定会准备概念题和实例题，概念题嘛，就是说概念，时间限制，线上语音面试的限制，无法让你一行一行念代码，而其实很多面试官他也没有看过源码，但他知道大概流程或者某一部分细节。真正到了实例题，肯定就不是这种笼统逻辑了，我们后面说。