2023面试题

根据 遗忘曲线：如果没有记录和回顾，6天后便会忘记75%的内容

笔记正是帮助你记录和回顾的工具，不必拘泥于形式，其核心是：记录、翻看、思考

Vue

1. 钩子函数 生命周期

```JavaScript
beforeCreate(创建前)
created(创建后)
beforeMount(载入前)
mounted(载入后)
beforeUpdate(更新前)
updated(更新后)
beforeDestroy(销毁前)
destroyed(销毁后)
```

2. 数据双向绑定原理

```JavaScript
 vue数据双向绑定是通过数据劫持结合发布者-订阅者模式的方式来实现的.
  数据劫持、vue是通过Object.defineProperty()来实现数据劫持,
  其中会有getter()和setter方法; 当读取属性值时, 就会触发getter()方法,
  在view中如果数据发生了变化, 就会通过Object.defineProperty( )对属性设置一个setter函数,
  当数据改变了就会来触发这个函数;

```

3. 路由守卫

```JavaScript
1.全局路由守卫
beforeEach(to, from, next) 全局前置守卫，路由跳转前触发
beforeResolve(to, from, next) 全局解析守卫 在所有组件内守卫和异步路由组件被解析之后触发
afterEach(to, from) 全局后置守卫，路由跳转完成后触发

2.路由独享守卫
beforeEnter(to,from,next) 路由对象单个路由配置 ，单个路由进入前触发

3.组件路由守卫
beforeRouteEnter(to,from,next) 在组件生命周期beforeCreate阶段触发
beforeRouteUpdadte(to,from,next) 当前路由改变时触发
beforeRouteLeave(to,from,next) 导航离开该组件的对应路由时触发

4.参数
to:  即将要进入的目标路由对象
from:  即将要离开的路由对象
next(Function): 是否可以进入某个具体路由，或者是某个具体路由的路径
```

4. v-show 与 v-if 区别

```JavaScript
v-show和v-if的区别:
v-show是css切换，v-if是完整的销毁和重新创建。

使用
频繁切换时用v-show，运行时较少改变时用v-if

```

5. Vuex

```
Vuex有五个核心概念：state,getters,mutations,actions,modules

1. state：vuex的基本数据，用来存储变量

2. geeter：从基本数据(state)派生的数据，相当于state的计算属性

3. mutation：提交更新数据的方法，必须是同步的(如果需要异步使用action)。每个 mutation 都有一个字符串的 事件类型 (type) 和 一个 回调函数 (handler)。

回调函数就是我们实际进行状态更改的地方，并且它会接受 state 作为第一个参数，提交载荷作为第二个参数。

4. action：和mutation的功能大致相同，不同之处在于 ==》1. Action 提交的是 mutation，而不是直接变更状态。 2. Action 可以包含任意异步操作。

5. modules：模块化vuex，可以让每一个模块拥有自己的state、mutation、action、getters,使得结构非常清晰，方便管理。
```

6. 组件传值（父、子、兄弟间）

```
父组件向子组件传值:父组件通过属性的方式向子组件传值，子组件通过 props 来接收
子组件向父组件传值:子组件绑定一个事件，通过 this.$emit() 来触发
兄弟之间传值:使用的是$bus的传值方式
其他方:：缓存、Vuex
```

7. 怎么定义vue-router的动态路由？怎么获取传过来的动态参数?

```
在router目录下的index.js文件中，对path属性加上/:id。  使用router对象的params.id
```

8. 2.0和3.0的区别

```JavaScript
双向绑定:
  V2: 使用Object.defineProperty
  V3: 使用ES6的新特性proxy来劫持数据，当数据改变时发出通知

  根元素:
  V2: 必须要有一个根元素
  V3: 无要求

  diff算法:
  V2: 虚拟Dom全量比较
  V3: 增加了静态标记PatchFlag
```

生命周期

2.0 周期名称	3.0 周期名称	说明
beforeCreate	setup	组件创建之前
created	setup	组件创建完成
beforeMount	onBeforeMount	组件挂载之前
mounted	onMounted	组件挂载完成
beforeUpdate	onBeforeUpdate	数据更新，虚拟 DOM 打补丁之前
updated	onUpdated	数据更新，虚拟 DOM 渲染完成
beforeDestroy	onBeforeUnmount	组件销毁之前
destroyed	onUnmounted	组件销毁

9. computed 与 watch 的区别

```JavaScript
computed支持缓存，相依赖的数据发生改变才会重新计算；watch不支持缓存，只要监听的数据变化就会触发相应操作

computed不支持异步，当computed内有异步操作时是无法监听数据变化的；watch支持异步操作

computed属性的属性值是一函数，函数返回值为属性的属性值，computed中每个属性都可以设置set与get方法。watch监听的数据必须是data中声明过或父组件传递过
```

10.Route和router的区别

```JavaScript
route:是路由信息对象，包括“path,parms,hash,name“等路由信息参数。
Router:是路由实例对象，包括了路由跳转方法，钩子函数等。
```

11.vue中数据变了但是视图不跟新怎么解决?

```JavaScript
原因:
1.数组数据变动：使用某些方法操作数组，变动数据时，有些方法无法被vue监测。
2.Vue 不能检测到对象属性的添加或删除。
3.异步更新队列：数据第一次的获取到了，也渲染了，但是第二次之后数据只有在再一次渲染页面的时候更新，并不能实时更新。

解决方案:
1.静默刷新(使用v-if的特性)
2.Vue.$set(官方推荐)
3.Vue.$forceUpdate(手动强制更新视图)
4.Object.assign(使用修改栈能触发视图更新的特性)
5.对于数组还可以使用splice方法

Vue对于数组的操作能识别变化的api包括push()、pop()、shift()、unshift()、splice()、sort()、reverse()这些都可被vue监测到
```

12. keep-alive

```JavaScript
1、什么是keep-alive?
keep-alive 是 Vue 的内置组件，当它包裹动态组件时，会缓存不活动的组件实例，而不是销毁它们。keep-alive 是一个抽象组件：它自身不会渲染成一个 DOM 元素，也不会出现在父组件链中

2、keep-alive的优点?
在组件切换过程中 把切换出去的组件保留在内存中，防止重复渲染DOM，减少加载时间及性能消耗，提高用户体验性。

3、keep-alive有三个属性

include ： 只有匹配的组件会被缓存
exclude ： 任何匹配的组件都不会被缓存
max ： 最多可以缓存多少组件实例

4、keep-alive的使用会触发两个生命周期函数?

这两个函数分别是
activated 当组件被激活（使用）的时候触发 可以简单理解为进入这个页面的时候触发
deactivated 当组件不被使用的时候触发 可以简单理解为离开这个页面的时候触发

```

13.vue-router 路由模式有几种?

```JavaScript
vue-router 有 3 种路由模式：hash、history、abstract：

hash: 使用 URL hash 值来作路由。支持所有浏览器，包括不支持 HTML5 History Api 的浏览器；
history : 依赖 HTML5 History API 和服务器配置。具体可以查看 HTML5 History 模式；
abstract : 支持所有 JavaScript 运行环境，如 Node.js 服务器端。如果发现没有浏览器的 API，路由会自动强制进入这个模式.

```

14.Vuex 页面刷新数据丢失

```JavaScript
本地存储
第三方插件解决
```

React

1. 区分Real DOM和Virtual（虚拟） DOM

真实DOM	虚拟DOM
更新慢	更新快
可以直接更新HTML	无法直接更新HTML
消耗内存更多	较少的内存消耗
元素更新，创建新的DOM	元素更新，更新JSX
DOM操作代价高	DOM操作简单

2. 类组件和函数组件之间有什么区别?

```JavaScript
类组件:

无论是使用函数或是类来声明一个组件，它决不能修改它自己的 props。
所有 React 组件都必须是纯函数，并禁止修改其自身 props。
React是单项数据流，父组件改变了属性，那么子组件视图会更新。
属性 props是外界传递过来的，状态 state是组件本身的，状态可以在组件中任意修改
组件的属性和状态改变都会更新视图。

函数组件:
函数组件接收一个单一的 props 对象并返回了一个React元素
函数组件的性能比类组件的性能要高，因为类组件使用的时候要实例化，而函数组件直接执行函数取返回结果即可。为了提高性能，尽量使用函数组件。
```

3. 生命周期（新版）

```JavaScript
React16废弃的生命周期有3个will:
componentWillMount
componentWillReceiveProps
componentWillUpdate

挂载
当组件实例被创建并插入 DOM 中时，其生命周期调用顺序如下:

constructor(): 在 React 组件挂载之前，会调用它的构造函数。
getDerivedStateFromProps(): 在调用 render 方法之前调用，并且在初始挂载及后续更新时都会被调用。
render(): render() 方法是 class 组件中唯一必须实现的方法。
componentDidMount(): 在组件挂载后（插入 DOM 树中）立即调用。
render() 方法是 class 组件中唯一必须实现的方法，其他方法可以根据自己的需要来实现。

更新
每当组件的 state 或 props 发生变化时，组件就会更新。
当组件的 props 或 state 发生变化时会触发更新。组件更新的生命周期调用顺序如下:

getDerivedStateFromProps(): 在调用 render 方法之前调用，并且在初始挂载及后续更新时都会被调用。根据 shouldComponentUpdate() 的返回值，判断 React 组件的输出是否受当前 state 或 props 更改的影响。
shouldComponentUpdate():当 props 或 state 发生变化时，shouldComponentUpdate() 会在渲染执行之前被调用。
render(): render() 方法是 class 组件中唯一必须实现的方法。
getSnapshotBeforeUpdate(): 在最近一次渲染输出（提交到 DOM 节点）之前调用。
componentDidUpdate(): 在更新后会被立即调用。
render() 方法是 class 组件中唯一必须实现的方法，其他方法可以根据自己的需要来实现。

卸载
当组件从 DOM 中移除时会调用如下方法:

componentWillUnmount(): 在组件卸载及销毁之前直接调用。
```

4. diff算法

```JavaScript
diff算法的本质：就是找出两个对象之间的差异，目的是尽可能做到节点复用。
上述中的对象:指的其实就是vue中的virtual dom（虚拟dom树），即使用js对象来表示页面中的dom结构。

三个层级策略:
1、tree层级：dom节点跨层级的移动操作特别少，可以将其忽略不计。
2、component层级：拥有相同类的两个组件会生成类似的树形结构，拥有不同类的两个组件将会生成不同的树形结构。
3、element层级：对于同一层级的一组子节点，可以通过唯一key进行区分。

```

5. 组件通信

```JavaScript
父传子：props；
子传父：子调用父组件中的函数并传参；
兄弟：利用redux实现和利用父组件
```

6. 什么是高阶组件?

高阶组件就是一个函数，且该函数接受一个组件作为参数，并返回一个新的组件。基本上，这是从React的组成性质派生的一种模式，我们称它们为“纯”组件， 因为它们可以接受任何动态提供的子组件，但它们不会修改或复制其输入组件的任何行为。

```JavaScript
高阶组件（HOC）是 React 中用于复用组件逻辑的一种高级技巧
高阶组件的参数为一个组件返回一个新的组件
组件是将 props 转换为 UI，而高阶组件是将组件转换为另一个组件
```

7. React Hooks

```JavaScript
(1) Hook是React 16.8.0版本增加的新特性/新语法
(2) 可以让你在函数组件中使用 state 以及其他的 React 特性

优势:
1.函数组件无this问题
2.自定义Hooks方便状态复用
3.副作用关注点分离

常用的Hook:
useState: 设置和改变state，代替原来的state和setState
useEffect: 代替原来的生命周期，componentDidMount，componentDidUpdate 和 componentWillUnmount 的合并版
useLayoutEffect: 与 useEffect 作用相同，但它会同步调用 effect
useMemo: 控制组件更新条件，可根据状态变化控制方法执行,优化传值
useCallback: useMemo优化传值，usecallback优化传的方法，是否更新
useRef: 跟以前的ref，一样，只是更简洁了
```

8. Redux

Redux 是 JavaScript 应用的状态容器，提供可预测的状态管理。
Redux并不只为react应用提供状态管理， 它还支持其它的框架。

```JavaScript
核心概念:
store、action、reducer

关键函数:
getState() ：用于获取当前最新的状态
subscribe() ：用于订阅监听当前状态的变化，然后促使页面重新渲染
dispatch() ：用于发布最新的状态

执行流程:
 (1) 用户通过事件触发ActionCreator制造action
 (2) 同时，用户触发的事件内调用dispatch来派发action
 (3) reducer接收action，并处理state返回newState
 (4) View层通过 getState() 来接收newState并重新渲染视图层

```

9. state 和 props

```JavaScript
props和 state 都是普通的 JavaScript 对象。它们都是用来保存信息的，这些信息可以控制组件的渲染输出，而它们的几个重要的不同点就是：
props: 是传递给组件的（类似于函数的形参），而 state 是在组件内被组件自己管理的（类似于在一个函数内声明的变量）。
props: 是不可修改的，所有 React 组件都必须像纯函数一样保护它们的 props 不被更改。 由于 props 是传入的，并且它们不能更改，因此我们可以将任何仅使用 props 的 React 组件视为pureComponent，也就是说，在相同的输入下，它将始终呈现相同的输出。
state: 是在组件中创建的，一般在 constructor中初始化 state
state: 是多变的、可以修改，每次setState都异步更新的。

```

10. 受控组件和非受控组件

```JavaScript
受控组件: 是React控制的组件,input等表单输入框值不存在于 DOM 中，而是以我们的组件状态存在。每当我们想要更新值时，我们就像以前一样调用setState。
不受控制组件:是您的表单数据由 DOM 处理，而不是React 组件，Refs 用于获取其当前值;
```

微信小程序

1. 小程序组件传值

2. bindtap 和 catchtap 区别

```JavaScript
bind事件绑定不会阻止冒泡事件向上冒泡
catch事件绑定可以阻止冒泡事件向上冒泡
```

3. 简述下 wx.navigateTo(), wx.redirectTo(), wx.switchTab(), wx.navigateBack(), wx.reLaunch() 区别

```JavaScript
wx.navigateTo() : 保留当前页面，跳转到应用内的某个页面。但是不能跳到 tabbar 页面
wx.redirectTo() : 关闭当前页面，跳转到应用内的某个页面。但是不允许跳转到 tabbar 页面
wx.switchTab() : 跳转到 TabBar 页面，并关闭其他所有非 tabBar 页面
wx.navigateBack() : 关闭当前页面，返回上一页面或多级页面。可通过 getCurrentPages() 获取当前的页面栈，决定需要返回几层
wx.reLaunch() : 关闭所有页面，打开到应用的某个页面。

```

4. 生命周期函数

```JavaScript
onLaunch: 生命周期回调—监听小程序初始化
onReady: 生命周期回调—监听页面初次渲染完成
onLoad: 生命周期回调—监听页面加载
onShow:生命周期回调—监听小程序启动或切前台
onHide:生命周期回调—监听小程序切后台
onUnload:生命周期回调—监听页面卸载

```

5. 关于分包

```JavaScript
目前小程序分包大小有以下限制：
整个小程序所有分包大小不超过 20M
单个分包/主包大小不能超过 2M
```

如何分包

CSS

1. CSS三角形

```JavaScript
主要：宽高设置为0，由边框来控制大小，然后边框颜色改为透明，然后更改一边的边框颜色为自己想要的颜色，就能实现三角形效果
如上三角 △
triangle-up {
 width: 0;
 height: 0;
 border-left: 50px solid transparent;
 border-right: 50px solid transparent;
 border-bottom: 100px solid red
}
```

2. CSS优先级如何排序?

```JavaScript
优先级如下:

!important>style（内联）>Id（权重100）> class（权重10）>标签（权重1）。
  同类别的样式中，后面的会覆盖前面的。
```

3. position的值?

```JavaScript
静态定位 - static
  相对定位 - relative
  绝对定位 - absolute
  固定定位 - fixed
```

4. 弹性布局 设置 垂直 水平居中

```JavaScript
display: flex;
 justify-content: center;
 align-items: center;
```

ES6 & 浏览器 & 基础知识

1. 数据类型都有什么

```JavaScript
1.基本数据类型：数字(Number)、字符串（String）、布尔(Boolean)、空（Null）、未定义（Undefined）；还有ES6新增的：Symbol(表示独一无二的值)、BigInt（任意精度整数）。

2.引用数据类型：对象(Object)、数组(Array)、函数(Function)
```

2. 深浅拷贝

```JavaScript
浅拷贝：只复制引用，而未复制真正的值
深拷贝：对目标的完全拷贝，不像浅拷贝那样只是复制了一层引用，就连值也都复制了

目前实现深拷贝的主要是利用 JSON 对象中的 parse 和 stringify
const originArray = [1,2,3,4,5];
const cloneArray = JSON.parse(JSON.stringify(originArray));
console.log(cloneArray === originArray); // false
```

3. let、var、const的区别

关键字	变量提升	块级作用域	重复声明同名变量	重新赋值
var	√	x	√	√
let	x	√	x	√
const	x	√	x	x

4. 判断数据类型有几种方法

5. 闭包

```
闭包指有权访问另一个函数作用域中变量的函数。简单理解就是，一个作用域可以访问另外一个函数内部的局部变量
优点:
1）可以减少全局变量的定义，避免全局变量的污染
2）能够读取函数内部的变量
3）在内存中维护一个变量，可以用做缓存

缺点:
1）造成内存泄露
2）闭包可能在父函数外部，改变父函数内部变量的值。
3）造成性能损失
```

6. webpack相关面试题

```
webpack 的作用就是处理依赖，模块化，打包压缩文件，管理插件。

1、webpack打包原理

  把所有依赖打包成一个 bundle.js 文件，通过代码分割成单元片段并按需加载。

2、webpack的优势

  （1） webpack 是以 commonJS 的形式来书写脚本滴，但对 AMD/CMD 的支持也很全面，方便旧项目进行代码迁移。
  （2）能被模块化的不仅仅是 JS 了。
  （3） 开发便捷，能替代部分 grunt/gulp 的工作，比如打包、压缩混淆、图片转base64等。
  （4）扩展性强，插件机制完善
```

7. typeof null和undefined结果

```
typeof null => 'object'
typeof undefined => 'undefined'

null === undefined => false
null == undefined  => true
```

8. cookie localstorage seesionstorage区别

```JavaScript
1.大小
cookie：  4K左右，很小很小;
sessionStorage 和 localStorage：5M;

2.有效期
cookie：使用expire设置过期时间
sessionStorage：浏览器关闭则清空，生命周期为仅在当前对话下
localStorage：不手动清空则不会清除，生命周期为永远

3.是否会将数据发给服务器
cookie：每次访问都会传送cookie给服务器，即使是不需要的时候，这样会浪费带宽
sessionStorage 和 localStorage：不传送
```

9.移动端点击事件300MS延迟问题

```JavaScript
1.下载fastclick的包
2.禁用浏览器缩放
3.通过touchstart和touchend模拟实现
```

10. 防抖和节流

```JavaScript
防抖(debounce)：触发高频事件后 n 秒内函数只会执行一次，如果 n 秒内高频事件再次被触发，则重新计算时间
节流(throttle)：高频事件触发，但在 n 秒内只会执行一次，所以节流会稀释函数的执行频率

区别：防抖动是将多次执行变为最后一次执行，节流是将多次执行变成每隔一段时间执行。
```

11. 常见code码

```JavaScript
200 - 请求成功
301 - 资源（网页等）被永久转移到其它URL
403 - Forbidden 服务器理解请求客户端的请求，但是拒绝执行此请求
404 - 请求的资源（网页等）不存在
500 - 内部服务器错误
502 - Bad Gateway 作为网关或者代理工作的服务器尝试执行请求时，从远程服务器接收到了一个无效的响应

```

12. http 和 https 的区别

```JavaScript
1）HTTP 明文传输，数据都是未加密的，安全性较差，HTTPS 数据传输过程是加密的，安全性较好。
2）使用 HTTPS 协议需要到 CA（Certificate Authority，数字证书认证机构） 申请证书，一般免费证书较少，因而需要一定费用
3）HTTP 页面响应速度比 HTTPS 快，主要是因为 HTTP 使用 TCP 三次握手建立连接，客户端和服务器需要交换 3 个包，而 HTTPS除了 TCP 的三个包，还要加上 ssl 握手需要的 9 个包，所以一共是 12 个包。
4）HTTP 和 HTTPS 使用的是完全不同的连接方式，用的端口也不一样，前者是 80，后者是 443。
5）HTTPS 其实就是建构在 SSL/TLS 之上的 HTTP 协议，所以，要比较 HTTPS 比 HTTP 要更耗费服务器资源。

```

13. 前端优化策略

```JavaScript
1、减少http请求数
2、将脚本往后挪，减少对并发下载的影响
3、避免频繁的DOM操作
4、压缩图片
5、gzip压缩优化，对传输资源进行体积压缩(html,js,css)
6、按需加载
7、组件化
8、减少不必要的Cookie（Cookie存储在客户端，伴随着HTTP请求在浏览器和服务器之间传递，由于cookie在访问对应域名下的资源时都会通过HTTP请求发送到服务器，从而会影响加载速度，所以尽量减少不必要的Cookie。）

```

14.什么是跨域?

```JavaScript
跨域解决方法:

1、jsonp方式
2、代理服务器的方式
3、服务端允许跨域访问(CORS)
4、取消浏览器的跨域限制
```

大厂真题 & 高级前端进阶

小米秋招面试题（卷1+卷2）

B站前端笔试题（卷1）

58同城前端笔试