腾讯篇——

1、Vuex 中的重要核心属性有哪些？

四大核心属性，以及一个子模块管理属性，state 存放数据，Mutations 同步修改 state 里面的数据，actions 异步修改数据，但需要调用 Mutations 里面的方法修改，getters 数据过滤器，对数据进行整理，还有一个 modules，当数据过多和复杂时，将数据用模块化分开，在住文件中用 modules 引用

2、CSS 中实现元素水平垂直居中的方式有哪些？

设置绝对定位，然后上下左右值设为 0，margin 为 auto。

使用 flex 布局，justify-content:center，align-items:center。

绝对定位，设置 left，top 为 50%，transform:translate(-50%，-50%)

子元素 display:inline-block，再在父元素上 text-aligen:center

设置文本的话，行高＋text-align

3、JS 放在 head 里和放在 body 里有什么区别？

JS 应该放到 body 内的最下方，浏览器遇到 script 标签会阻塞进行下载，因此如果文件较大，会导致白屏。因此放在 body 内的下方，保证页面渲染后，再加载 JS

4、JavaScript 的参数是按照什么方式传递的？

按值传递，如果传个对象的话看起来好像是按引用了，其实只不过是函数内参数变量也指向了那个地址而已，如果说在函数内部将参数变量设置为新值，会发现对原引用类型数据没有影响，所以只是看起来按引用了。最终还是按值。

5、**promise 和 async 有什么区别？**

① promise 是一个对象，async 是一个函数

② promise 属于 ES6，async 属于 es7

③ promise 处理异步是另一种地狱回调，async 是测底拉平，更加优雅好用

④ promise 是 async 的底层

6、JS 哪些操作会造成内存泄露？

参考答案：

① 意外的全局变量

② 闭包

③ 没有清理的 Dom 引用

④ 被遗忘的定时器或回调

⑤ 子元素存在引用

7、常见的 SPA 首屏优化方式有哪些？

参考答案：

首屏优化的核心是加载和解析的性能。

加载优化的核心是 资源体积 和 首屏资源数量。

解析优化的核心是 资源体积 和 代码的执行性能。

可见，资源体积会同时影响加载和解析的性能。

一、加载优化:

加载方面最有效的方式是缓存的合理使用，再快的网速和再大的并发数也不如直接跳过下载阶段。

其他优化有:

①合理分包

在还使用 HTTP1.X 的情况下，衡量包的体积和并发数量，利用 webpack 的 Code Split 技术探索出合适的包体积和资源数量。

②启用 CDN 加速

让不同地区的人都可以享受稳定可靠的下载速度。

③静态资源分域

将图片，音视频等静态资源放到不同域名加载，防止加载时受浏览器并发数限制; 同时还可以实现 cookie 隔离，在安全和加载速度上都有极大好处。

④DNS 预解析: 合理采用 DNS 预解析

域名解析(DNS 寻址)是一个相对比较耗时的过程。

理想状态下，浏览器通过查询系统 hosts 或者进一步询问本地 DNS 服务器就可以获得结果(服务器 IP)，但也很有可能要经历完整的寻址过程: 分别与根 DNS 服务器，域 DNS 服务器，以及解析 DNS 服务器进行多次交互才能获得结果。

所以启用 DNS 预解析可以很大程度上节省域名寻址的时间(但凡事有度，过犹不及)，如果搭配静态资源分域，会有更优秀的变现。

⑤尽量启用 HTTP2

HTTP2 提供了多路复用可以突破并发限制，头部字段压缩也可以减少传输的数据体积，二进制数据传输也可以让多种数据类型选择更合适的方式传输。

二、解析优化

解析优化分为资源体积优化以及运行时的性能优化。

资源体积优化:

基本的方式有，压缩字体，图片以及音视频等静态文件体积等等。

另外，现代化的工程基本上都用了 Webpack 或其他构建工具，以 Webpack 为例，其实 Webpack 以及社区的大佬们都为我们的工程提供了很多的优化。

8、Vue3.0 性能提升主要是通过哪几个方面体现的？

①响应式系统提升

vue2 在初始化的时候，对 data 中的每个属性使用 defineprope ry 调用 getter 和 setter 使之变为响应式对象。如果属性值为对象，还会递归调用 defineproperty 使之变为响应式对象。

vue3 使用 proxy 对象重写响应式。proxy 的性能本来比 defineproperty 好，proxy 可以拦截属性的访问、赋值、删除等操作，不需要初始化的时候遍历所有属性，另外有多层属性嵌套的话，只有访问某个属性的时候，才会递归处理下一级的属性。

优势：

可以监听动态新增的属性；

可以监听删除的属性 ；

可以监听数组的索引和 length 属性；

②编译优化

优化编译和重写虚拟 dom，让首次渲染和更新 dom 性能有更大的提升 vue2 通过标记静态根节点,优化 diff 算法 vue3 标记和提升所有静态根节点,diff 的时候只比较动态节点内容 Fragments, 模板里面不用创建唯一根节点,可以直接放同级标签和文本内容

静态提升
patch flag, 跳过静态节点,直接对比动态节点,缓存事件处理函数
③源码体积的优化
vue3 移除了一些不常用的 api，例如：inline-template、filter 等 使用 tree-shaking

9、Composition Api 与 Vue 2.x 使用的 Options Api 有什么区别？
Options Api
包含一个描述组件选项（data、methods、props 等）的对象 options；
API 开发复杂组件，同一个功能逻辑的代码被拆分到不同选项 ；
使用 mixin 重用公用代码，也有问题：命名冲突，数据来源不清晰；
composition Api
vue3 新增的一组 api，它是基于函数的 api，可以更灵活的组织组件的逻辑。
解决 options api 在大型项目中，options api 不好拆分和重用的问题。