字节前端面试真题

字节

一面

> 说一下浏览器缓存

浏览器缓存分为**强缓存**和**协商缓存**，强缓存会直接从浏览器里面拿数据，协商缓存会先访问服务器看缓存是否过期，再决定是否从浏览器里面拿数据。

控制强缓存的字段有：Expires和Cache-Control，Expires 和 Cache-Control。

控制协商缓存的字段是：Last-Modified / If-Modified-Since 和 Etag / If-None-Match，其中 Etag / If-None-Match的优先级比Last-Modified / If-Modified-Since高。

cookie 与 session 的区别

Session 是在服务端保存的一个数据结构，用来跟踪用户的状态，这个数据可以保存在集群、数据库、文件中； Cookie 是客户端保存用户信息的一种机制，用来记录用户的一些信息，也是实现 Session 的一种方式。

详见：COOKIE和SESSION有什么区别?

> 浏览器如何做到 session 的功能的。

其实就是考察 http 怎么处理无状态是怎么处理的，具体可见 COOKIE和SESSION有什么区别? 里面的答案。

解释一下：csrf 和 xss

XSS：恶意攻击者往 Web 页面里插入恶意 Script 代码，当用户浏览该页之时，嵌入其中 Web 里面的 Script 代码会被执行，从而达到恶意攻击用户的目的。

CSRF：CSRF 攻击是攻击者借助受害者的 Cookie 骗取服务器的信任，可以在受害者毫不知情的情况下以受害者名义伪造请求发送给受攻击服务器，从而在并未授权的情况下执行在权限保护之下的操作。

详见：*前端安全面试题*

> 怎么防止 csrf 和 xss

详见：*前端安全面试题*

> 跨域的处理方案有哪些

常用的：jsonp、CORS、nginx 代理，完整的大概是九种，可见：九种跨域方式实现原理（完整版）

> CORS 是如何做的?
>
> 服务端设置 Access-Control-Allow-Origin 就可以开启 CORS。

> 对于 CORS，Get 和 POST 有区别吗?

其实想考察的就是什么时候会有**预检请求(option 请求)**。

> 了解 HTTPS 的过程吗?

推荐浪浪的 深入理解HTTPS工作原理

webpack 如何做性能优化

webpack 做性能优化主要是考虑打包体积和打包速度。

体积分析用 `webpack-bundle-analyzer` 插件，速度分析用：`speed-measure-webpack-plugin` 插件。

打包速度优化瓶子君的：玩转 webpack，使你的打包速度提升 90%。

es module 和 commonjs 的区别

高频题，考察 ES6 模块和 CommonJS 模块 的区别。关键点：1. 前者是值的引用，后者是值的拷贝。2.前者编译时输出接口，后者运行时加载。

推荐文章：前端模块化：CommonJS,AMD,CMD,ES6

> react 里如何做动态加载

`React.lazy` ，另外通过 webpack 的动态加载：`import()` 和 `ensure.require`

> 动态加载的原理是啥，就是 webpack 编译出来的代码

讲道理 webpack 动态加载就两种方式：`import()` 和 `require.ensure`，不过他们实现原理是相同的。

我觉得这道题的重点在于动态的创建 script 标签，以及通过 `jsonp` 去请求 **chunk**，推荐的文章是：webpack是如何实现动态导入的

笔试题：页面结构包括页头（永远在顶部）、主体内容、页脚，页脚永远在页面底部（不是窗口底部），即内容高度不够时，页脚也要保证在页面底部

常规题，考察基本的布局

笔试题：写 new 的执行过程

new 的执行过程大致如下：

1. 创建一个对象
2. 将对象的 _ *proto*_ 指向 构造函数的 prototype
3. 将这个对象作为构造函数的 this
4. 返回该对象。

```
function myNew(Con, ...args) {
  let obj = Object.create(Con.prototype)
  let result = Con.apply(obj, args)
```

```
  return typeof result === 'object' ? result : obj
}
```

笔试题：写一个处理加法可能产生精度的函数，比如 0.1 + 0.2 = 0.3

思路：对于浮点数在底层处理是有问题的，所以目的就是想办法将所以的浮点数转化为整数进行处理，同时乘以一个倍数(A)，然后加起来后再除以这个倍数(A)，这个倍数应该是两个数中最小的那个数的倍数，比如 0.1 + 0.02 ,那么应该同时乘以 100，变为 10 + 2，然后再将值除以 100。

1000000000 + 1000000000 允许返回字符串 处理大数

大数问题就是通过字符串来处理，从后往前加，然后处理进位的问题。

二面

聊项目

项目基本是问：

1. 项目难点以及怎么解决的
2. 项目有哪些亮点？

写一个 es6 的继承过程

这个题我觉得出得很好，很考察基本功。

```
// 这个是要实现的方法
createClass = fun(sons, super) {
        // TODO
        return fn;
}

// 这是个 es6 的一个例子，要实现 extends 的功能。
class Man extends Human {
        cons (args) {
                super(args)
          // xxxxx
        }

        speak() {
                console.log('')
        }
}

```

> 写一个大数相乘的解决方案。传两个字符串进来，返回一个字符串

```
function multi(str1, str2) {

}
```

这道题跟一面的时候思路差不多，只是进位的时候不一定是 1。

三面

> 聊项目
>
> 写一个防抖函数
>
> 算法题:leetcode-cn.com/problems/bu…

小节

字节果然是出了名的考算法题比较多的，基本每面都会算法题和编程题，对编程能力比较看重吧。

讲道理一面还是比较常规的，二三面因为都是团队 leader 和更高级别的，问的技术细节也比较少了，重点考察一些技术方案和项目的问题。