2021
整理题

PERSONAL RESUME

方向：前端

时间：2021/2/1

PERSONAL RESUME

笔试题整理-1

1、前端开发的优化问题

2、==与===区别

3、webpack：

4、PC 端/移动端常见的兼容性问题总结

5、Get 与 Post 的主要区别

6、浏览器输入 URL 到页面在用户面前，这上过程发生了什么

7、关于异步，解决方案

8、JavaScript 中，this 关键字的作用是什么？

9、填空

10、设置文字不换行，超出部分用... 代替样式为

11、block,inline,inline-block 区别?

12、如何使得英文单词不发生词内断行？

13、类型检测方法 typeof 和 instanceof 有什么区别

14、ＥＳ５的继承和ＥＳ６的继承有什么区别？

15、写出 ul ol dl 三种列表的 html 结构

16、css 代码优化

17、针对不同浏览器写出 box-shadow 的格式

18、HTML 有哪些新的页面元素？列给 5 个以上；

19、你认为良好的 CSS 架构目标是什么？

20、AJAX 是什么，有什么优点和缺点？

21、模块加载器有何优点，以及常模块加载器有哪些

22、介绍你所知道的 CSS hask 技巧

23、cookies、sessionStorage 和 localStorage 的区别

24、多个页面之间如果进行通信，组件之间有哪些传值的方法？

25、写出以下执行结果，谈谈上下文和作用域的区别？

26、写下结果：说说 Event Loop Task Queue?

27、Vue 中给 data 中的对象属性添加一个新的属性时会发生什么，如何解决？

28、判断真假:

29、foo 对象有 att 属性，那么获取 att 怪属性的值，以下哪些做法对

30、数组方法

31、String 对象

32、解释一下为什么需要清除浮动？清除浮动的方式

33、webpack 用来解决什么问题

34、ｎｕｌｌ和ｕｎｄｅｆｉｎｅｄ的区别

35、跟后端进行异步请求时，如何避免代码嵌套太深

36、http 协议中３０２状态是什么含义

37、new 操作符具体是干什么的

38、如何准确的判断这个变量是否为数组

39、同步和异步的区别

40、css 选择器优先级

41. documen.write 和 innerHTML 的区别

42. 如何规避 javascript 多人开发函数重名问题

43、js 的基础对象有那些，window 和 document 的常用的方法和属性列出来

44、 js 中如何定义 class,如何扩展 prototype?

45、写一段 js，判断是否是这样组成的：第一个必须是字母，后面可以是字母、数字、下划线，总长度为 5-20

46、哪些属性可以继承？

47、display:none 和 visibility:hidden 区别

48、谈谈以前端角度出发做好 SEO 需要考虑什么。

1、前端开发的优化问题

减少 http 请求次数：

Css Sprites：将多张图片合并为一幅单独的图片，

内联图片：Data url ，url 里包含了数据不需要在发送请求下载，二进制数据以 base64 的形式进行了一个字符串的体现。所以对比传统文件的体积来说大 1/3,如图有 10M，那么 DATA URL 形式就 13M 左右，所以文件很大不适合用 Data Url，兼容性较差，只支持 ie8 以上浏览器，浏览器不会缓存内联图片资源

适合：一般存小的数据，所以说一个浏览器图片很多且图片很小且资源多要发送很多次请求，这时用 data URL

缓存 AJAX：前端模板 JS+数据，减少由于 HTML 标签导致的带宽浪费，前端用变量保存 AJAX 请求结果，每次操作本地变量，不用请求，减少请求次数

缓存 DOM 节点查找的结果，减少对 DOM 操作，主要是减少 DOM 的重绘与回流（重排）

在事件绑定中，尽可能使用事件委托，减少循环给 DOM 元素绑定事件处理函数。if(e.target==nodeName){}

减少 DNS 查找：internet 是通过 IP 来查找服务器的，当在浏览器输入 IP 时连接到浏览器的 DNS 解析器会返回服务器的 IP 地址，在 DNS 查找完成之前，浏览器不能从主机名上下载到任何东西，响应时间依赖于 DNS 解析器反应有关这与距离和宽带速度也有关系。DNS 查找可被缓存起来提高性能。

使用外部 js 与 css 或每个页面类型创建单独脚本和样式表：会被浏览器缓存起来，公共 js 与 css 进行提取，避免冗余，

使用内容发布网络：分布多个不同地理位置的 WEB 服务器用于更有效向用户发布网容

将 Css 样式表放 head 中，避免白屏，将脚本放在底部(不会阻止页面内容的呈现)：脚本放顶部会阻塞其后面内容呈现

避免 CSS 表达示：动态设置属性且频繁求值影响加载性能拖慢网页的加载速度

避免重定向：将用户用一个 URL 重新路由到另一个 URL 上，301、302,重定向会使网页变慢。重定向(Redirect)就是通过各种方法将各种网络请求重新定个方向转到其它位置（如：网页重定向、域名的重定向、路由选择的变化也是对数据报文经由路径的一种重定向）。使用 www.cnblogs.com 而不是 cnblogs.com301 是永久重定向，而 302 是临时重定向

避免方法：定义链接地址的 href 属性的时候，尽量使用最完整的、直接的地址

删除重复脚本：两次包含同一 JS 文件会影响性能。

少用全局变量

图片预载

避免在页面的主体布局中使用 table，table 要等其中的内容完全下载之后才会显示出来，显示比 div+css 布局慢

当需要设置的样式很多时设置 className 而不是直接操作 style

用 setTimeout 来避免页面失去响应

用 hash-table 来优化查找

能用 css 做的效果，不要用 js 做，能用原生 js 做的，不用第三方插件。只是用里面的一个小功能

避免使用 iframe：不仅不好管控样式，而且相当于在本页面又嵌套其他页面，消耗性能会更大。因为还回去加载这个嵌套页面的资源

在 js 中尽量减少闭包的使用，原因：使用闭包后，闭包所在的上下文不会被释放

减少对 cookie 的使用（最主要的就是减少本地 cookie 存储内容的大小），因为客户端操作 cookie 的时候，这些信息总是在客户端和服务端传递。如果上设置不当，每次发送一个请求将会携带 cookie

前端与后端进行数据交互时，对于多项数据尽可能基于 json 格式来进行传送。相对于使用 xml \来说传输有这个优势　目的：是数据处理方便，资源偏小

在基于 ajax 的 get 请求进行数据交互的时候，根据需求可以让其产生缓存（不是 304），这样在下一次从相同地址获取是数据时，取得就是上一次缓存的数据。（注意：很少使用，一般都会清空。根据需求来做）
在 js 封装过程中，尽量做到低耦合高内聚。减少页面的冗余代码
基于 script 标签下载 js 文件时，可以使用 defer 或者 async 来异步加载

附加：
解释下 css sprites ，如何使用
CSS Sprites 是一种网页图片处理的一种方式，就是把网页中一些背景图片整合到一张图片文件中，用 CSS" `background-image、background-repeat、background-position` 的组合进行背景定位。
优点：（1）减少网页的 http 请求，提高性能（2）减少图片的字节：多张图片合并成 1 张；
(3)减少了命名困扰：集合图起名就行，不用对每一个小元素进行命名提高制作效率；
(4)更换风格方便：只需要在一张或少张图片上修改图片的颜色或样式，整个网页的风格就可以改变，维护起来更加方便。
缺点：(1)图片合成比较麻烦；(2)背景设置时，需要得到每一个背景单元的精确位置，；
(3) 维护合成图片时，最好只是往下加图片，而不要更改已有图片。

Js 放<body>js+html</body>与<body>html+js</body>区别?
js+<body>html</body>:js 是阻塞的，也就是说，<script>对应的 js 必须加载完+执行完，才会继续加载下面的 html 的部分,如果你的 js 很大、很慢，那么放在上面，会导致页面会长时间空白
<body>html</body>js:唯一的区别就是放在</body>之后不标准，现在可以，但不定哪天官方突然抽风就不好使了,因为按照标准来说</body>之后不应该出现任何东西

2、==与===区别
== 表示相等 （值相等）
===表示恒等（类型和值都要相等）

3、webpack：
前端本身不支持像后端那样文件引用，使用 webpack 就可以实现这种功能。另外打包还会对代码做检查和压缩，达到优化的目的。
使用 webpack 过程：
a 生成 Packjson.json 执行命令 npm init –y；
文件作用：是 node 的项目描述文件，如项目依赖谁有哪些 scripts 常用
b 手动创建 webpack.config.js

```
css 用的 loader  css-loader style-loader
img 用的 loader(file-loader  或 url-loader)
less 用的 loader less-loader
vue 用的 loader vue-loader webpack webpack-cli
vue-style-loader vue-html-loader vue-template-compiler
vue-loader：编译 vue 文件必备的
webpack webpack-cli ：编译过程中需要用到，
vue-style-loader 编译 vue 里的样式，是 style-loader 的子级，有 style-loader 功能也有自己功能
vue-html-loader ：编译 vue 里 html 代码，编译 xxx.vue 文件里 template 中 div 开始的东西。
vue-template-compiler：编译 vue 模板，
```

基本配置：

```
module.exports = {
            mode: 'development',//开发模式
            entry: [],  //入口文件
            output: {},     //出口文件
            module: {//模块规则
                rules: [
              {test: /\.css$/i, use: ['style-loader', 'css-loader'] },
                ]
            }
}
```

文件作用:是 node 的一个模块要对外输出 json,所有的 webpack 配置都在这

1、webpack 与 vue-cli 区别:webpack 是自己配置。vue-cli 脚手架 构建项目不需要自己配置

webpack 如何打包更快，

用 happypack 提升项目的构建速度；happypack 只作用在 loader 上，使用多个进程同时对文件进行编译处理；比如：用 babel-loader 处理 js 类型的文件时，

可忽略 node_modules 文件夹；并且对该 loader 使用 happypack 处理；】

a. 安装 happypack 插件： npm install happypack --save-dev

// 创建 happypack 共享进程池，其中包含 6 个子进程,即多个 HappyPack 实例都使 用同一个共享进程池中的子进程去处理任务，以防止资源占用过多

```
const happyThreadPool = HappyPack.ThreadPool({ size: 6 });
```

b. 在 webpack.base.config.js 配置文件中引入：

```
const HappyPack = require('happypack');
```

c. 在 module 模块中配置时，对 js 类型的文件使用

```
module: {
   rules: [
   {
     test: /\.js$/,//把对.js 的文件处理交给 id 为 babel 的 HappyPack 的实例执行
     use: ['happypack/loader?id=babel'],//排除 node_modules 目录下的文件
     exclude: /node_modules/
    }]
 }
```

d. 在 plugins 插件中使用 happypack 实例

```
plugins: [
      new HappyPack({
//id 标识符，要和 rules 中指定的 id 对应起来，需要使用的 loader，用法和 rules 中 Loader 配置一样 // 可以直接是字符串，也可以是对象形式
       id: 'babel',
        loaders: ['babel-loader?cacheDirectory'],//共享进程池
        threadPool: happyThreadPool
      })
    ]
```

4、PC 端/移动端常见的兼容性问题总结

① 安卓浏览器看背景图片，有些设备会模糊，原因是手机的分辨率太小

解决方案： 用 2X 图片来代替 img 标签，然后 background-size: contain

② 防止手机中页面放大或缩小： 在 meta 中设置 viewport user-scalable = no

③ 上下拉滚动条卡顿： overflow-scrolling: touch;

④ 禁止复制选中文本： user-select: none;

⑤ 长时间按住页面出现闪退：-webkit-touch-callout: none;

⑥ 动画定义 3D 硬件加速： transform: translate 3d(0,0,0);

⑦ formate-detection 启动或禁止自动识别页面中的电话号码，content = "yes/no"

⑧ a 标签添加 tel 是拨号功能

⑨ 安卓手机的圆角失效： background-clip: padding-box;

⑩ 手机端 300ms 延迟： fastclick

① 横平时字体加粗不一致： text-size-adjust: 100%;

PC 端：

① rgba 不支持 IE8 用 opacity 属性代替 rgba 设置透明度

② 图片加 a 标签在 IE9 中出现边框 解决方案： img{border: none;}

③ IE6 不支持 display: inline-block 设置为： display: inline

④ position : fixed 不支持 IE5/IE6

⑤ IE6，Firfox 中，width = width + padding + border

⑥ min-height ie 6.0 不支持；ie 7.0 后的支持，但也可能会存在兼容性问题；

5、get 与 post 区别，何时用 post

1）get 是从服务器上获取数据，post 是向服务器传送数据。

2）get 是将参数数据加到 URL 中，用户可以看到。post 是将内容放置在 http 请求信息体内传送，用户看不到这个过程。

3）对于 get 方法，服务器端是用 Request.QueryString 获取变量的值，对于 post 方法，服务器端用 Request.Form 获取提交的数据。

4）get 传送的数据量较小，不能大于 2kb。post 传送的数据量较大，一般被默认为不受限制。但理论上，IIS4 中最大量为 80kb，IIS5 中为 100kb。“一个汉字字符是 2 个字节。1kb=1024 字节， 2k 就 1024 个字

5）get 安全性非常低，post 安全性较高。但是执行效率却比 post 方法好。

仅用于 POST 请求（1.无法使用缓存文件；2.向服务器发送大量数据；3.发送包含未知字符的用户输入时，post 比 get 更稳定也更可靠）

建议使用 get 方法的情况：1、get 方式的安全性较 Post 方式要差些，包含机密信息的话，建议用 Post 数据提交方式 2、在做数据查询时，建议用 Get 方式；而在做数据添加、修改或删除时，建议用 Post 方式；

6、浏览器输入 URL 到页面在用户面前，这上过程发生了什么

1. 在浏览器中输入网址；
2. 发送至 DNS 服务器并获得域名对应的 WEB 服务器的 IP 地址；
3. 与 WEB 服务器建立 TCP 连接；
4. 浏览器向 WEB 服务器的 IP 地址发送相应的 HTTP 请求；
5. WEB 服务器响应请求并返回指定 URL 的数据，或错误信息，如果设定重定向，则重定向到新的 URL 地址。
6. 浏览器下载数据后解析 HTML 源文件，解析的过程中实现对页面的排版，解析完成后在浏览器中显示基础页面。
7. 分析页面中的超链接并显示在当前页面，重复以上过程直至无超链接需要发送，完成全部显示。

7、关于异步，解决方案

a). 回调函数 b). 事件监听 c). 发布/订阅 d). Promise 对象 aynce/await

8、JavaScript 中，this 关键字的作用是什么？

关键字 this 指向当前对象。比如，顶级代码中的 this 指向全局对象；在指定元素事件的时候，this 指定当前发生事件的元素对象。对于嵌套函数，如果嵌套函数作为方法被调用，其 this 指向调用它的对象；如果作为函数调用，this 是全局对象或者为 undefined（严格模式下）。

```
var o = {a: 1,
    m: function () {console.log(this);
        f();//Window  作这函数调用
        function f() {
          console.log(this);
        }
    }
};
o.m();//{a: 1, m: f}作为方法调用
```

9、填空

为 div 设置类 a 与 b，应编写 HTML 代码? _ __

设置 CSS 属性 clear 的值为(none)可清除左右两边浮动

css 属性(margin)可为元素设置外补丁

写出定义标题的方法，br 标签在 XHTML 中语义是？ 换行

不换行必须设置哪些？ white-space: nowrap;

在使用 table 表现数据时，有时候表现出来的会比自己实际设置的宽度要宽，为些需要设置哪些属性值？

cellpadding=”0″，cellspacing=”0″

10、设置文字不换行，超出部分用... 代替样式为：

overflow: hidden; text-overflow: ellipsis; white-space: nowrap; width: 100px;

11、block, inline, inline-block 区别?

block:块级元素独占一行主要有 、<ul>、<li>、<p>、<fieldset>、<form>、<h1>、<hr>、<iframe>、<ol>、<pre>、<table>、<tr>、<td>，nav、aside、header、footer、section、article、ul-li、address 能够识别宽高

inline-block：行内块状元素。行内块状元素综合了行内元素和块状元素的特性但是各有取舍，不自动换行，不独占一行，但是可以设置宽高

inline：行内元素。与其他行内元素共享一行，因此无法设置宽高。对 margin 仅设置左右方向有效，上下无效；padding 设置上下左右都有效，即会撑大空间；不会自动进行换行。、<a>、<b>、<img>、
、<button>、、<textarea>、<select>

12、如何使得英文单词不发生词内断行？

不断行:word-wrap: break-all;

断行：word-wrap: break-word;

13、类型检测方法 typeof 和 instanceof 有什么区别

typeof: 返回值是一个字符串，用来说明变量的数据类型。一般只能返回如下几个结果：number, boolean, string, function, object, undefined。

instanceof: 返回值为布尔值;用于判断一个变量是否属于某个对象的实例。

instanceof 运算符用于检测构造函数的 prototype 属性是否出现在某个实例对象的原型链上。

14、ＥＳ５的继承和ＥＳ６的继承有什么区别？

ES5 的继承时通过 prototype 或构造函数机制来实现。ES5 的继承实质上是先创建子类的实例对象，然后再将父类的方法添加到 this 上（Parent.apply(this)）。

ES6 的继承机制完全不同，实质上是先创建父类的实例对象 this（所以必须先调用父类的 super()方法），然后再用子类的构造函数修改 this。

具体的：ES6 通过 class 关键字定义类，里面有构造方法，类之间通过 extends 关键字实现继承。子类必须在 constructor 方法中调用 super 方法，否则新建实例报错。因为子类没有自己的 this 对象，而是继承了父类的 this 对象，然后对其进行加工。如果不调用 super 方法，子类得不到 this 对象。

15、写出 ul ol dl 三种列表的 html 结构

前两个都是 li

```
<dl><dt>标题</dt><dd>内容 1</dd></dl>
```

16、css 代码优化

margin-top:20px;margin-right:5px;margin-left:5px;margin-bottom:20px;font-style:italic;
font-weight:bold;font-size:1em;line-height:140%;font-family:'lucida Grande',sans-serif;
color:#333333;

参考：

margin: 20px5px;
font: italic bold 1em/140%'Lucida Grande', sans-serif;
color: #369;

17、针对不同浏览器写出 box-shadow 的格式

-webkit-box-shadow: 3px 3px 3px;
-moz-box-shadow: 3px 3px 3px;
box-shadow: 3px 3px 3px;

18、HTML 有哪些新的页面元素？列给 5 个以上；

Article、 header footer figure（一组媒体及标签文字） mark video section embed 外部插件或对象

19、你认为良好的 CSS 架构目标是什么？

好的 CSS 架构目标并不同于开发一个好的应用程序，它必须是可预测、可重用、可维护和可伸缩的。

可预测意味着可以像预期的那样规范自己的行为。当你添加或者修改某个规则时，它并不会影响到没有指定的部分。对于一个小网站来说，一些微乎其微的改变并不算什么。而对于拥有成千上万个页面的大网站来说，可预测却是必须的。

可重用：CSS 规则应具备抽象和解耦性，这样你就可以在现有的基础上快速构建新的组件，无需重新修改编码模式。

可维护：当把新组件放置到网站上，并且执行添加、修改或者重新设计操作时，无需重构现有 CSS，并且新添加的 X 并不会打破原有页面的 Y 组件。

可扩展：当网站发展到一定规模后，都需要进行维护和扩展。可扩展的 CSS 意味着网站的 CSS 架构可以由个人或者团队轻易地管理，无需花费太多的学习成本。

20、AJAX 是什么，有什么优点和缺点？

优点：

页面局部刷新，提高用户体验度；

使用异步方式与服务器通信，具有更加迅速的响应能力；

减轻服务器负担；

基于标准化的并被广泛支持的技术，不需要下载插件或者小程序。

缺点：不支持浏览器 back 按钮；安全问题；对搜索引擎的支持比较弱。

AJAX 干掉了 Back 和 History 功能，即对浏览器机制的破坏。无法回到前一个页面状态

Ajax 就如同对企业数据建立了一个直接通道。使开发者不经意间暴露比以前更多的数据和服务器逻辑

对搜索引擎的支持比较弱。如果使用不当，AJAX 会增大网络数据的流量，从而降低整个系统的性能。

21、模块加载器有何优点，以及常模块加载器有哪些

require 加载模块。

module.exports 对外暴露接口。

一个文件就是一个模块，具备独立的作用域，不会污染全局。

22、介绍你所知道的 CSS hask 技巧

```
.bb{
 background-color:#f1ee18;/*所有识别*/
.background-color:#00deff\9; /*IE6、7、8 识别*/
+background-color:#a200ff;/*IE6、7 识别*/
_background-color:#1e0bd1;/*IE6 识别*/
}
```

23、cookies、sessionStorage 和 localStorage 的区别

一、 背景介绍

Cookie：意思小甜饼，非常小，大小限制为 4KB 左右，是网景公司的一位员工在 1993 年 3 月的发明。它的主要用途是保存登录信息，比如登录某个网站可以看到“记住密码”，这通常就是通过在 Cookie 中存入一段识别用户身份的数据来实现的。

localStorage：是 HTML5 标准中新加入的技术，它并不是什么划时代的新东西。早在 IE 6 时代，就有一个叫 userData 的东西用于本地存储，而当时考虑到浏览器的兼容性，更通用的方案是使用 Flash。而如今，localStorage 已经被大多数浏览器所支持

sessionStorage：它与 localStorage 的接口类似，但保存数据的生命周期与 localStorage 不同。我们都应该知道 Session 是“会话”。而 sessionStorage 是一个前端的概念，它只是可以将一部分数据在当前会话中保存下来，刷新页面数据依旧存在。但当页面关闭后，sessionStorage 中的数据就会被清空。

二、 三者的区别

特性	Cookie	localStorage	sessionStorage
数据的生命期	可设置失效时间，默认是关闭浏览器后失效	除非被清除，否则永久保存	仅在当前会话下有效，关闭页面或浏览器后被清除
存放数据大小	4K左右	一般为5MB	一般为5MB
与服务器端通信	每次都会携带在HTTP头中，如果使用cookie保存过多数据会带来性能问题	仅在客户端（即浏览器）中保存，不参与和服务器的通信	仅在客户端（即浏览器）中保存，不参与和服务器的通信
易用性	需要程序员自己封装，源生的Cookie接口不友好	源生接口可以接受，亦可再次封装来对Object和Array有更好的支持	源生接口可以接受，亦可再次封装来对Object和Array有更好的支持

三、 应用场景

因为考虑到每个 HTTP 请求都会带着 Cookie 的信息，所以 Cookie 当然是能精简就精简！比较常用的一个应用场景就是判断用户是否登录。针对登录过的用户，服务器端会在他登录时往 Cookie 中加入一段加密过的唯一辨识单一用户的辨识码，下次只要读取这个值就可以判断当前用户是否登录啦。曾经还使用 Cookie 来保存用户在电商网站的购物车信息，如今有了 localStorage，似乎它在这个方面便可以给 Cookie 放个假了~

而另一方面 localStorage 接替了 Cookie 管理购物车的工作，同时也能胜任其他一些工作。比如 HTML5 游戏通常会产生一些本地数据，localStorage 则是非常适合做这个工作的。如果遇到一些内容特别多的表

单，为了优化用户体验，我们可能要把表单页面拆分成多个子页面，然后按步骤引导用户填写。这时候 sessionStorage 的作用就发挥出来了。

cookie 数据保存在客户端，session 数据保存在服务器端。

之前 Cookie 实现，localStorage：适用于长期存储数据，浏览器关闭后数据不丢失，数据只能由用户删除；sessionStorage：存储的数据在浏览器关闭后自动删除。

24、多个页面之间如果进行通信，组件之间有哪些传值的方法？

1、父组件往子组件传值 props

2、子组件往父组件传值，通过 emit 事件

3、vuex 进行传值

vuex 主要是是做数据交互，父子组件传值可以很容易办到，但是兄弟组件间传值（兄弟组件下又有父子组件），或者大型 spa 单页面框架项目，页面多并且一层嵌套一层的传值，异常麻烦，用 vuex 来维护共有的状态或数据会显得得心应手。

4、provide/inject：主要解决了跨级组件间的通信问题，不过它的使用场景，主要是子组件获取上级组件的状态，跨级组件间建立了一种主动提供与依赖注入的关系。

5、$attrs/$listeners

$attrs：包含了父作用域中不被 prop 所识别 (且获取) 的特性绑定 (class 和 style 除外)。当一个组件没有声明任何 prop 时，这里会包含所有父作用域的绑定 (class 和 style 除外)，并且可以通过 v-bind="$attrs" 传入内部组件。通常配合 interitAttrs 选项一起使用。

$listeners：包含了父作用域中的 (不含 .native 修饰器的) v-on 事件监听器。它可以通过 v-on="$listeners" 传入内部组件

25、为什么要对 URL 进行编码

是因为 Url 中有些字符会引起歧义，Url 的编码格式采用的是 ASCII 码

encodeURIComponent 编码的字符范围要比 encodeURI 的大，encodeURIComponent 对特殊字符编码解决用 ASCII 吗表示为:3D:= 26:&现在有这样一个问题，如果我的参数值中就包含=或&这种特殊字符的时候该怎么办。比如说“name1=value1”,其中 value1 的值是“va&lu=e1”二手字符串，那么实际在传输过程中就会变成这样“name1=va&lu=e1”。我们的本意是就只有一个键值对，但是服务端会解析成两个键值对，这样就产生了奇异。URL 编码只是简单的在特殊字符的各个字节前加上%，例如，我们对上述会产生奇异的字符进行 URL 编码后结果：“name1=va%26lu%3D”，这样服务端会把紧跟在“%”后的字节当成普通的字节，就是不会把它当成各个参数或键值对的分隔符。

```
console.log(encodeURIComponent("va&lu=e1"))// va%26lu%3De1
```

26、jsonp 原理是什么

其背后原理就是利用了 script 标签不受同源策略的限制，在页面中动态插入了 script，script 标签的 src 属性就是后端 api 接口的地址,并且以 get 的方式将前端回调处理函数名称告诉后端，后端在响应请求时会将回调返还，并且将数据以参数的形式传递回去。

27. JavaScript 有哪几种数据类型

参考答案：

简单：Number，Boolean，String，Null，Undefined

引用：Object，Array，Function

28、判断真假:

null instanceof Object、 null === undefined、 null==undefined、 NaN==NaN

结果 false false true false

因为：typeof null ->Object

typeof NaN ->number

29、foo 对象有 att 属性，那么获取 att 怪属性的值，以下哪些做法对

getAttribute() 和 attr() 都是获取元素属性的方法前者 js 写法后者 jquery 写法

这里不能用

正确写法：

foo.att、foo['att']

错误写法：

foo('att')、foo{'att'}、foo['a','t','t']、foo.getAttribute('att')、foo.attr('att')

30、数组方法。以下哪个方法用于 Array 中删除元素，并向数组添加新元素－>splice（位置，删除个数，添加项）

arr.splice(位置，删除几个，添加新项)方法向/从数组中添加/删除项目，然后返回被删除的项目。

arrayObject.slice(start,end)含头不含尾， 方法可从已有的数组中返回选定的元素。

push()末尾添加一项或多项

shift()开头删除

pop()末尾删除最后一项

unshift()开头添加一项或多项元

arrayObject.join(要使用的分隔符)

var arr = ["George","John","Thomas"]console.log(arr.join(".")//George.John.Thomas

concat() 方法用于连接两个或多个数组。不会改变现有的数组,而仅仅会返回被连接数组的一个副本。

arrayObject.concat(arrX,arrX,.....,arrX)

var a = [1,2,3];

document.write(a.concat(4,5)); //1,2,3,4,5

findIndex 返回匹配下标第一个,返回的是索引值，没有符合条件元素时 findIndex()返回的是-1

find 返回某个匹配东西 查找第一个符合条件的匹配元素，而 find()返回的是 undefined。

语法：arr.findIndex(function(item, index, arr), thisValue)

```
var ages = [3, 10, 18, 20]
var i = ages.find(value => value > 4)
console.log(i) // 10
var ages = [3, 10, 18, 20]
var i = ages.findIndex(value => value > 4)
console.log(i) // 1
var arr = [
{ id: 1, name: '衣服', price: 69.9, count: 5 },
{ id: 2, name: '裤子', price: 29.9, count: 15 },
{ id: 3, name: '鞋子', price: 19, count: 25 }
]
console.log(arr.find(item => item.id == 2)) //{id: 1, name: "衣服", price:
69.9, count: 5}
console.log(arr.findIndex(item => item.id == 2)) //1
```

toString() 方法可把数组转换为字符串，并返回结果。

arrayObject.toString()

var arr = ["George","John","Thomas"]

console.log(arr.toString())//George,John,Thomas

arr.reverse(): 颠倒数组中元素的顺序，原先第一个元素现在变成最后一个，同样原先的最后一个元素变成现在的第一个。

slice(start,end-1): 从已有数据中选中元素，返回新数组，含头不含尾，

sort: 按照字母顺序对数组排序，支持传入指定排序方法的函数作为参数。

arr.reduce() ：从左往右执行 arr.reduce(prev,item,index,arr)； prev 上次计算的结果

arr.valueOf: 和 toString 相似，将数组作为字符串返回

for....of....： arr.keys() 数组下标 ,arr.entries() 数组某一项

arr.find(): 查找，找出第一个符合条件的数组成员，如果没返回有找到，undefined,return 返回一个值

arr.findIndex(): 找的是位置， 没找到返回-1

arr.fill() 填充,arr.fill(填充的东西, 开始位置, 结束位置不包含);

arr.includes() 返回 true/false

Array.from()方法就是将一个类数组对象或者可遍历对象转换成一个真正的数组

for...in 语句来遍历数组内的元素。

```
let arrayLike = {0: 'tom',1: '65', 2: '男',3: ['jane', 'john', 'Mary'],
'length': 4}let arr = Array.from(arrayLike) console.log(arr) // ['tom','65','男
',['jane','john','Mary']]
```

所谓类数组对象，最基本的要求就是具有 length 属性的对象。

addEventListener :true - 事件句柄在捕获阶段执行

map 返回一个新数组主要是对数组每个元素的操作，有 return，

```
var array1 = [1, 4, 9, 16]; const map1 = array1.map(x => x * 2);
console.log(map1);// [2, 8, 18, 32
```

for/in 主要是对象键值的一些遍历，对象,

不要使用 for/in 语句来循环数组的索引，你可以使用 for 语句替代。

forEach 的应用只要是数组的简单遍历

```
var arr = [1, 2, 3, 4, 5];
arr.forEach(function (currentValue, index, arr) {//当前，索引，所有值
        console.log(currentValue)//1 2 3 4 5
})
```

for..of 遍历数组，for of 无法循环遍历对象

```
var arr = ['nick', 'freddy', 'mike', 'james'];
for (var key in arr) { console.log(key);//0123}
for (var item of arr) {console.log(item);//nick freddy mike james}
```

31、String 对象

indexOf() 方法可返回某个指定的字符串值在字符串中首次出现的位置。

str.indexOf(查找值，从哪开始可选) 找不到返-1，找到返位置

concat() 方法用于连接两个或多个字符串。str.concat(str1,str2)

var str1="Hello "

var str2="world!"

console.log(str1.concat(str2)) //Hello world!

slice()提取字符串的片断，并在新的字符串中返回被提取的部分。

str.slice(start,end)含头不含尾

```
var str="Hello happy world!"
console.log(str.slice(6))//happy world!
console.log(str.slice(6,11))//happy
```

split() 把字符串分割为字符串数组。

str.split(分隔符，返回长度可选

```
document.write(str.split(" ") //How,are,you,doing,today?
document.write(str.split(" ",3))  //How,are,you
```

substr() 从起始索引号提取字符串中指定数目的字符。

str.substr(start,length)

```
var str="Hello world!"
document.write(str.substr(3))//lo world!
document.write(str.substr(3,7))//lo worl
```

substring() 提取字符串中两个指定的索引号之间的字符。

str.substring(start,stop)

```
var str="Hello world!"
document.write(str.substring(3))//lo world!
document.write(str.substring(3,7))//lo w
```

toLowerCase() 把字符串转换为小写。

toUpperCase() 把字符串转换为大写。

toString() 返回字符串。stringObject 的原始字符串值。一般不会调用该方法。

32、解释一下为什么需要清除浮动？清除浮动的方式

浮动导致元素已不在普通流中，所以在排列布局的时候文档中的普通流表现的和浮动元素不存在一样，当浮动元素的高度超出包含框的时候，会出现包含框不会自动撑高来包裹浮动元素，即所谓的“高度塌陷”。换种说法：父元素的高度是由子元素撑开的，且子元素设置了浮动，父元素没有设置浮动，子元素脱离了标准的文档流，那么父元素的高度会将其忽略，如果不清除浮动，父元素会出现高度不够，那样如果设置 border 或者 background 都得不到正确的解析

清除浮动:在父元素的最后加一个冗余元素并为其设置 clear:both

给父元素添加 overflow:hidden ||auto

1. 额外标签法（在最后一个浮动标签后，新加一个标签，给其设置 clear：both；）（不推荐）

2. 父级添加 overflow 属性（父元素添加 overflow:hidden）（不推荐）

3. 使用 after 伪元素清除浮动（推荐使用）

```
.clearfix:after{/*伪元素是行内元素 正常浏览器清除浮动方法*/
 content: "";
 display: block;
 height: 0;
 clear:both;
visibility: hidden;
}
.clearfix{
*zoom: 1;/*ie6 清除浮动的方式 *号只有 IE6-IE7 执行，其他浏览器不执行*/
}
```

4. 使用 before 和 after 双伪元素清除浮动

```
.clearfix:after,.clearfix:before{content: "";display: table;}
.clearfix:after{ clear: both;}
.clearfix{ *zoom: 1;}
```

33、webpack 用来解决什么问题

前端本身不支持像后端那样文件引用，使用 webpack 就可以实现这种功能。另外打包还会对代码做检查和压缩，达到优化的目的。

使用 webpack 过程：

1、 生成 Packjson.json 执行命令 npm init –y；

文件作用：是 node 的项目描述文件，如项目依赖谁有哪些 scripts 常用

2、 手动创建 webpack.config.js

34、null 和 undefined 的区别

null 是 javascript 关键字，空值，typeof 返回 object,可以表示数字，字符串和对象“无值”定义一个对象，未赋值前为 null

undefined 是预定义的全局变量，为“未定义”，变量一种取值，表示没有初始化。

当查询对象属性，数组元素值时，返回 undefined 时表示属性或元素不存在；

如果函数没返回值也返回 undefined

需要注意的是，虽然 null 和 undfined 是不同的，但是因为都表示“值的空缺”，两者可以互换。因此＝＝为 true,＝＝＝为 false

35、跟后端进行异步请求时，如何避免代码嵌套太深

现在出现了一种比较好的写法， 就是用 Promise.js 来简单来写

```
var asncy = function (text) {
    var promise = new Promise(function (resolve, reject) {
        setTimeout(function () {
            console.log(text);
            resolve();
        }, 1000)
    });
    return promise;
}
asncy ("1").then(function () {
    return asny("2");
}).then(function () {
    return asny("3");
}).then(function () {
    console.log("done");
});
```

36、http 协议中 3 0 2 状态是什么含义

该资源原本确实存在，但已经被**临时**改变了位置,换而言之，就是请求的资源暂时驻留在不同的 URI 下

37、new 操作符具体是干什么的

（1）创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。

（2）属性和方法被加入到 this 引用的对象中。

（3）新创建的对象由 this 所引用，并且最后隐式的返回 this 。

38、如何准确的判断这个变量是否为数组

只有 instanceof 才能判断一个对象是否是真正的数组

```
Arr instanceof Array;   //true
a.constructor === Array;    //true
Object.prototype.toString.call(a) === '[object Array]';//true
Array.isArray(a);    //true
```

39、同步和异步的区别

异步操作：同时进行多个操作，用户体验好（如用户名检查，输入完就检查）. 异步缺点： 好用但写起来麻烦。

同步操作：一次只能进行一次操作，用户体验不好，按顺序执行,优点：清晰，

40、css 选择器优先级

! important ID 选择器 class 选择器 标签选择器 伪类选择器

41. documen.write 和 innerHTML 的区别

document.write 只能重绘整个页面

innerHTML 可以重绘页面的一部分

42. 如何规避 javascript 多人开发函数重名问题

参考答案：

(1) 可以开发前规定命名规范，根据不同开发人员开发的功能在函数前加前缀

(2) 将每个开发人员的函数封装到类中，调用的时候就调用类的函数，即使函数重名只要类名不重复就 ok

43、js 的基础对象有那些，window 和 document 的常用的方法和属性列出来

参考答案：

String,Number,Boolean,Date,Math,Array,RegExp

Window:

方法：setInterval,setTimeout,clearInterval,clearTimeout,alert,confirm,open

属性：name,parent,screenLeft,screenTop,self,top,status

Document：

方法：

createElement,execCommand,getElementById,getElementsByName,getElementByTagName,write,writeln

属性：cookie,doctype,domain,documentElement,readyState,URL

44、 js 中如何定义 class,如何扩展 prototype?

参考答案：

Ele.className = “***”; //***在 css 中定义，形式如下：.*** {…}

A.prototype.B = C;

A 是某个构造函数的名字

B 是这个构造函数的属性

C 是想要定义的属性的值

如何添加 html 元素的事件,有几种方法.

（1） 为 HTML 元素的事件属性赋值

（2） 在 JS 中使用 ele.on*** = function() {…}

（3） 使用 DOM2 的添加事件的方法 addEventListener 或 attachEvent

45、写一段 js，判断是否是这样组成的：第一个必须是字母，后面可以是字母、数字、下划线，总长度为 5-20

```
var reg = /^[a-zA-Z][a-zA-Z_0-9]{4,19}$/;
reg.test("ala__ala__ala__ala__");//true
```

46、CSS 哪些属性可以继承？

1）文本相关属性：font-family、 font-size、 font-style、font-variant, font-weight、
font、 letter-spacing、line-height、text-align、 text-indent、text-transform、
word-spacing、color;

47、display:none 和 visibility:hidden 区别

1.display:none 是彻底消失，不在文档流中占位，浏览器也不会解析该元素；visibility:hidden 是视觉上消失了，可以理解为透明度为 0 的效果，在文档流中占位，浏览器会解析该元素；

2.使用 visibility:hidden 比 display:none 性能上要好，display:none 切换显示时 visibility，页面产生回流（当页面中的一部分元素需要改变规模尺寸、布局、显示隐藏等，页面重新构建，此时就是回流。所有页面第一次加载时需要产生一次回流），而 visibility 切换是否显示时则不会引起回流。

46、谈谈以前端角度出发做好 SEO 需要考虑什么。

参考答案：

1、了解搜索引擎如何抓取网页和如何索引网页

你需要知道一些搜索引擎的基本工作原理，各个搜索引擎之间的区别，搜索机器人（SE robot 或叫 web crawler）如何进行工作，搜索引擎如何

对搜索结果进行排序等等。

2、Meta 标签优化

主要包括主题（Title），网站描述(Description)，和关键词（Keywords）。还有一些其它的隐藏文字比如 Author（作者），Category（目录），

Language（编码语种）等。

3、如何选取关键词并在网页中放置关键词

搜索就得用关键词。关键词分析和选择是 SEO 最重要的工作之一。首先要给网站确定主关键词（一般在 5 个上下），然后针对这些关键词进行优

化，包括关键词密度（Density），相关度（Relavancy），突出性（Prominency）等等。

4、了解主要的搜索引擎

虽然搜索引擎有很多，但是对网站流量起决定作用的就那么几个。比如英文的主要有 Google，Yahoo，Bing 等；中文的有百度，搜狗，有道等。

不同的搜索引擎对页面的抓取和索引、排序的规则都不一样。还要了解各搜索门户和搜索引擎之间的关系，比如 AOL 网页搜索用的是 Google 的搜索技

术，MSN 用的是 Bing 的技术。

5、主要的互联网目录

Open Directory 自身不是搜索引擎，而是一个大型的网站目录，他和搜索引擎的主要区别是网站内容的收集方式不同。目录是人工编辑的，主要

收录网站主页；搜索引擎是自动收集的，除了主页外还抓取大量的内容页面。

6、按点击付费的搜索引擎

搜索引擎也需要生存，随着互联网商务的越来越成熟，收费的搜索引擎也开始大行其道。最典型的有 Overture 和百度，当然也包括 Google 的广告

项目 Google Adwords。越来越多的人通过搜索引擎的点击广告来定位商业网站，这里面也大有优化和排名的学问，你得学会用最少的广告投入获得最

多的点击。

7、搜索引擎登录

网站做完了以后，别躺在那里等着客人从天而降。要让别人找到你，最简单的办法就是将网站提交（submit）到搜索引擎。如果你的是商业网

站，主要的搜索引擎和目录都会要求你付费来获得收录（比如 Yahoo 要 299 美元），但是好消息是（至少到目前为止）最大的搜索引擎 Google 目前还
是免费，而且它主宰着 60%以上的搜索市场。

8、链接交换和链接广泛度（Link Popularity）

网页内容都是以超文本（Hypertext）的方式来互相链接的，网站之间也是如此。除了搜索引擎以外，人们也每天通过不同网站之间的链接来
Surfing（“冲浪”）。其它网站到你的网站的链接越多，你也就会获得更多的访问量。更重要的是，你的网站的外部链接数越多，会被搜索引擎认为
它的重要性越大，从而给你更高的排名。

9、标签的合理使用