1. **精灵图(CSS Sprites)的优点和缺点**

**精灵图是一种网页图片应用处理方式。就是把网页中很多小背景图片整合到一张图片文件中，再利用CSS的“background-image”，“background-repeat”，“background-position”的组合进行背景图显示及定位，达到显示某一部分背景图的效果。**

**精灵图的优点：  
  1、减少图片的体积，因为每个图片都有一个头部信息，把多个图片放到一个图片里，就会共用同一个头部信息，从而减少了字节数。  
  2、减少了网页的http请求次数，从而加快了网页加载速度，提高用户体验。  
  3、解决了网页设计师在图片命名上的困扰，只需对一张集合的图片上命名就可以了，不需要对每一个小元素进行命名，从而提高了网页的制作效率。  
  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。**

**精灵图的缺点：  
  1.在图片合并的时候，你要把多张图片有序的合理的合并成一张图片，还要留好足够的空间，防止板块内出现不必要的背景；这些还好，最痛苦的是在宽屏，高分辨率的屏幕下的自适应页面，你的图片如果不够宽，很容易出现背景断裂；**

**2.在开发的时候比较麻烦，你要通过photoshop或其他工具测量计算每一个背景单元的精确位置，这是针线活，没什么难度，但是很繁琐；**

**3.在维护的时候比较麻烦，如果**[**页面背景**](http://baike.baidu.com/view/7105320.htm)**有少许改动，一般就要改这张合并的图片，无需改的地方最好不要动，这样避免改动更多的css，如果在原来的地方放不下，又只能（最好）往下加图片，这样图片的字节就增加了，还要改动css。**

**4.精灵图不能随意改变大小和颜色。精灵图改变大小会失真模糊，降低用户体验，css3新属性可以改变精灵图颜色，但是比较麻烦，并且新属性有兼容问题。现在一般都是用web字体(图标字体)来代替精灵图。**

1. **什么是vue全家桶**

**Vue + vue-router + vuex + axios + es6 + sass**

**3.doctype是什么，网页常见doctype及特点**

**DOCTYPE是document type(文档类型)的简写，在web设计中用来说明你用的XHTML或者HTML是什么版本。**

**常见类型：**

1. **过渡型(Transitional)：要求非常宽松，它允许你继续使用**[**HTML4**](https://www.baidu.com/s?wd=HTML4&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLnjR1mvf4PWTzuhuhPvfz0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3ErHDYn10vn1D)**.01的标识(但是要符合xhtml的写法)，完整代码如下：  
     
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "**[**http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd**](http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd)**">**
2. **严格型(Strict)：要求非常严格，你不能使用任何表现层的标识和属性，例如<br>，完整代码如下：  
     
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "**[**http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd**](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)**">**
3. **框架型(Frameset)：专门针对框架页面设计使用，如果你的页面中包含有框架（frameset），完整代码如下：  
     
   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "**[**http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd**](http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd)**">**

**3.什么是web语义化，有什么好处**

**Web语义化简单来说就是为了让页面对人和机器更友好，让人和机器更容易理解页面内容。**

1. **对机器来说，语义化能让机器更精确的知道页面中的重点和关键点。让机器更容易为人筛选出想要的部分。**
2. **对开发人员来说，更容开发和易维护页面。根据页面中的标签名和类名就能知道哪个部分放置了哪些内容，从而提高了开发和维护的效率。**

**4.你知道的HTTP 请求方式有几种**

**HTTPRequestMethod共计17种**

**1.GET 请求指定的页面信息，并返回实体主体。**

**2.HEAD 类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头**

**3.POST 向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。**

**4.PUT 从客户端向服务器传送的数据取代指定的文档的内容。**

**5.DELETE 请求服务器删除指定的页面。**

**6.CONNECT HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。**

**7.OPTIONS 允许客户端查看服务器的性能。**

**8.TRACE 回显服务器收到的请求，主要用于测试或诊断。**

**9.PATCH 实体中包含一个表，表中说明与该URI所表示的原内容的区别。**

**10.MOVE 请求服务器将指定的页面移至另一个网络地址。**

**11.COPY 请求服务器将指定的页面拷贝至另一个网络地址。**

**12.LINK 请求服务器建立链接关系。**

**13.UNLINK 断开链接关系。**

**14.WRAPPED 允许客户端发送经过封装的请求。**

**15.LOCK 允许用户锁定资源，比如可以再编辑某个资源时将其锁定，以防别人同时对其进行编辑。**

**16.MKCOL 允许用户创建资源**

**17.Extension-mothed 在不改动协议的前提下，可增加另外的方法。**

**5.css选择器有哪些**

1. **简单选择器**

**通配符选择器 \***

**id选择器 #id**

**class选择器 .class**

**标签选择器 element**

1. **复合选择器**

**后代选择器 element element**

**子代选择器 element>element**

**兄弟选择器 element+element**

**并列选择器 element,element**

**伪类选择器 :link**

**属性选择器 [attribute]**

**6.css hack原理及常见hack**

**原理：CSS hack是一种类似作弊的手段，以欺骗浏览器的方式达到兼容的目的，是用浏览器的兼容性差异来解决浏览器的兼容性问题。**

**常见hack:**

1. **利用浏览器对相同代码的解析和支持的不同实现的hack**

**比如div{ \_width:80px }，在ie6中能识别并解析带下划线的属性，但是ie7及以上版本中识别不了。**

1. **以Firefox或Webkit特有的扩展样式实现的hack**

**如Firefox支持以-moz-开头的属性**

**Webkit内核支持以-webkit-开头的属性**

1. **以IE特有的条件注释为基础的hack**

**<!--[ifIE8]> <style type="text/css"> #test{  color:red;  }  </style><![endif]-->**

**比如这个样式，可以在ie8中生效，但是其他浏览器则不会生效**

**7.css中有哪些常见的继承属性**

* **文本**

1. **color(颜色，a元素除外)**
2. **direction(方向)**
3. **font（字体）**
4. **font-family（字体系列）**
5. **font-size（字体大小）**
6. **font-style（用于设置斜体）**
7. **font-variant（用于设置小型大写字母）**
8. **font-weight（用于设置粗体）**
9. **letter-spacing（字母间距）**
10. **line-height（行高）**
11. **text-align（用于设置对齐方式）**
12. **text-indent（用于设置首航缩进）**
13. **text-transform（用于修改大小写）**
14. **visibility（可见性）**
15. **white-space（用于指定如何处理空格）**
16. **word-spacing（字间距）**

* **列表**

1. **list-style（列表样式）**
2. **list-style-image（用于为列表指定定制的标记）**
3. **list-style-position（用于确定列表标记的位置）**
4. **list-style-type（用于设置列表的标记）**

* **表格**

1. **border-collapse（用于控制表格相邻单元格的边框是否合并为单一边框）**
2. **border-spacing（用于指定表格边框之间的空隙大小）**
3. **caption-side（用于设置表格标题的位置）**
4. **empty-cells（用于设置是否显示表格中的空单元格）**

* **页面设置（对于印刷物）**

1. **orphans（用于设置当元素内部发生分页时在页面底部需要保留的最少行数）**
2. **page-break-inside（用于设置元素内部的分页方式）**
3. **widows（用于设置当元素内部发生分也是在页面顶部需要保留的最少行数）**

* **其他**

1. **cursor（鼠标指针）**
2. **quotes（用于指定引号样式）**

**8.sessionStorage,localStorage,cookie区别**

* **共同点：都是保存在浏览器端，且同源的。**
* **区别：cookie数据始终在同源的http请求中携带（即使不需要），即cookie在浏览器和服务器间来回传递；cookie数据还有路径（path）的概念，可以限制cookie只属于某个路径下。存储大小限制也不同，cookie数据不能超过4k，同时因为每次http请求都会携带cookie，所以cookie只适合保存很小的数据，如会话标识。**
* **而sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存。sessionStorage和localStorage 虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。**
* **数据有效期不同，sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数据；cookie只在设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭。**
* **作用域不同，sessionStorage不在不同的浏览器窗口中共享，即使是同一个页面；localStorage 在所有同源窗口中都是共享的；cookie也是在所有同源窗口中都是共享的。Web Storage 支持事件通知机制，可以将数据更新的通知发送给监听者。Web Storage 的 api 接口使用更方便。**

**9.JavaScript有哪些数据类型**

**原始类型有6种 object,number,string,boolean,null,undefined**

**10.输出今天是星期几**

**var str = "今天是星期" + "日一二三四五六".charAt(new Date().getDay());alert(str);**

**11.如何判断一个变量是否为数组**

**arr.constructor === Array 为true说明是数组类型**

**arr instanceof Array 为true则是数组类型**

**Object.prototype.toString.call(arr) === '[object Array]'; 为true为数组类型**

**Array.isArray(arr) 为true则为数组类型**

**12.实现一个js深度克隆函数**

**//深度克隆**

**function deepClone(obj){**

**var result={},oClass=isClass(obj);**

**for(key in obj){**

**var copy=obj[key];**

**if(isClass(copy)=="Object"){**

**result[key]=arguments.callee(copy);**

**}else if(isClass(copy)=="Array"){**

**result[key]=arguments.callee(copy);**

**}else{**

**result[key]=obj[key];**

**}**

**}**

**return result;**

**}**

**function isClass(o){**

**if(o===null) return "Null";**

**if(o===undefined) return "Undefined";**

**return Object.prototype.toString.call(o).slice(8,-1);**

**}**

**//克隆一个数组**

**var arr=["a","b","c"];**

**var oNew=deepClone(arr);**

**console.log(oNew);//Object {0: "a", 1: "b", 2: "c"}**

**13.给定一个日期，页面打印倒计时**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="utf-8">**

**<title>倒计时</title>**

**</head>**

**<body>**

**<input type="text" name="time" id="time" value="2017-09-22" />**

**<input type="button" name="okbtn" id="okbtn" value="确认" />**

**<br />**

**<p id="textp">这里显示倒计时</p>**

**</body>**

**<script type="text/javascript">**

**var okbtn=document.getElementById("okbtn");**

**textp=document.getElementById("textp");**

**okbtn.onclick=function(){**

**var time=document.getElementById("time");**

**var timevalue=time.value;**

**//通过正则表达式确认输入格式是否正确**

**var patt=/^(\d{1,4})(-)(\d{1,2})\2(\d{1,2})$/;**

**if(patt.test(timevalue)==false){**

**//如果不正确**

**textp.innerHTML="输入格式不满足YYYY-MM-DD";**

**return false;**

**}else{**

**textp.innerHTML="这里显示倒计时";**

**}**

**//获取输入的年月日**

**var timearray=timevalue.split("-");**

**//ShowLeftTime(timearray[0],timearray[1],timearray[2]);**

**setInterval(function(){ShowLeftTime(timearray[0],timearray[1],timearray[2]);},1000);**

**}**

**function ShowLeftTime(year,month,date){**

**//得出剩余时间**

**var now=new Date();**

**var endDate=new Date(year,month-1,date);**

**var leftTime=endDate.getTime()-now.getTime();**

**var leftsecond=parseInt(leftTime/1000);**

**var day=Math.floor(leftsecond/(60\*60\*24));**

**var hour=Math.floor((leftsecond-day\*24\*60\*60)/3600);**

**var minute=Math.floor((leftsecond-day\*24\*60\*60-hour\*3600)/60);**

**var second=Math.floor(leftsecond-day\*60\*60\*24-hour\*60\*60-minute\*60);**

**//显示剩余时间**

**textp.innerHTML="距离"+year+"年"+month+"月"+date+"日"**

**+"还有"+day+"天"+hour+"小时"+minute+"分"+second+"秒";**

**}**

**</script>**

**</html>**

**14.数组去重**

**//利用indexOf**

**var aa=[1,3,5,4,3,3,1,4]**

**function arr(arr) {**

**var result=[]**

**for(var i=0; i<arr.length; i++){**

**if(result.indexOf(arr[i])==-1){**

**result.push(arr[i])**

**}**

**}**

**console.log(result)**

**}**

**arr(aa)**

**//循环嵌套**

**function fn(arr) {**

**var result = [];**

**var flag;**

**for (var i = 0, len = arr.length; i < len; i++) {**

**flag = false;**

**for (var j = 0, len = result.length; j < len; j++) {**

**if (arr[i] == result[j]) {**

**flag = true;**

**break;**

**}**

**}**

**if (!flag) {**

**result.push(arr[i]);**

**}**

**}**

**return result;**

**}**

**15.求数组中最大值，最小值和重复次数最多的数值**

**var arr = [1,2,2,3,5,5,5,6];**

**var max = arr[0]; // 数组中的最大值**

**var min = arr[0]; // 数组中的最小值**

**var mostCount; // 数组中出现次数最多的元素**

**var temp = {};**

**var num = 0;**

**for(var i=arr.length-1; i>=0; i--){**

**if(max<arr[i]){ // 得到最大值**

**max = arr[i]**

**}**

**if(min>arr[i]){ // 得到最小值**

**min = arr[i]**

**}**

**if(temp[arr[i]]){ // 得到元素出现的次数，并组成obj**

**temp[arr[i]] = tempObj[arr[i]]+1;**

**}else{**

**temp[arr[i]] = 1;**

**}**

**}**

**for(var key in temp){ // 从对象中得到数组中出现最多的元素**

**if(!mostCount){**

**mostCount = key;**

**}else if(temp[mostCount]<temp[key]){**

**mostCount = key;**

**}**

**}**

**console.log("最大值为"+max+"; 最小值为"+minVal+"; 次数最多的为"+mostCount);**

**17.请概括什么情况下打印控制台会提示undefined**

**1.当打印的变量声明未赋值时**

**2.当打印的执行函数没有返回值时**

**18.原生实现数组快速排序**

**var times=0;**

**var quickSort=function(arr){**

**//如果数组长度小于等于1无需判断直接返回即可**

**if(arr.length<=1){**

**return arr;**

**}**

**var midIndex=Math.floor(arr.length/2);//取基准点**

**var midIndexVal=arr.splice(midIndex,1);//取基准点的值,splice(index,1)函数可以返回数组中被删除的那个数arr[index+1]**

**var left=[];//存放比基准点小的数组**

**var right=[];//存放比基准点大的数组**

**//遍历数组，进行判断分配**

**for(var i=0;i<arr.length;i++){**

**if(arr[i]<midIndexVal){**

**left.push(arr[i]);//比基准点小的放在左边数组**

**}**

**else{**

**right.push(arr[i]);//比基准点大的放在右边数组**

**}**

**console.log("第"+(++times)+"次排序后："+arr);**

**}**

**//递归执行以上操作,对左右两个数组进行操作，直到数组长度为<=1；**

**return quickSort(left).concat(midIndexVal,quickSort(right));**

**};**

**console.log(quickSort(arr));**

**19.原生实现二分查找法**

**var arr = [5,0,-56,90,12];**

**var flag = false;//标志位进行优化，数组本来就是有序序列的话，无需再排序**

**//先进行大的排序**

**for(var i=0;i<arr.length-1;i++){**

**//小的排序**

**for(var j=0;j<arr.length-1-i;j++){**

**//比较**

**if(arr[j]>arr[j+1]){**

**//交换**

**var temp = arr[j];**

**arr[j] = arr[j+1];**

**arr[j+1] = temp;**

**flag = true;**

**}**

**}**

**//此部分为优化，已排序的话，无需再次排序**

**if(flag){**

**flag=false;**

**}else{**

**break;//已排序，无需交换**

**}**

**}**

**//输出新数组**

**for(var i=0;i<arr.length;i++){**

**document.write(arr[i]+' ');**

**}**

**20.正则表达式，清楚字符串前后空格**

**var str=" hello ";   
str=str.replace(/^s\*|s\*$/g,'');   
alert(str);**

**21.简述http协议中get和post方法的区别**

1. **GET主要用于从服务器查询数据，POST用于向服务器提交数据**
2. **GET通过URL传递数据，POST通过http请求体传递数据**
3. **GET传输数据量有限制，不能大于2kb，POST传递的数据量较大，一般大量的数据提交都是通过POST方式**
4. **GET安全性较低，容易在URL中暴漏数据，POST安全性较高**

**22.什么是csrf攻击，如何阻止**

**CSRF（Cross-site request forgery），中文名称：跨站请求伪造，也被称为：one click attack/session riding，缩写为：CSRF/XSRF。**



**从上图可以看出，要完成一次CSRF攻击，受害者必须依次完成两个步骤：**

**1.登录受信任网站A，并在本地生成Cookie。**

**2.在不登出A的情况下，访问危险网站B。**

**防止方式**

**(1).Cookie Hashing(所有表单都包含同一个伪随机值)**

**(2).验证码**

**(3).One-Time Tokens(不同的表单包含一个不同的伪随机值)**

**23.服务器推送数据到前端有哪些解决方案**

**一、Ajax轮询**

**用定时器不断发送请求**

**优点：实现简单。**

**缺点：这是通过模拟服务器发起的通信，不是实时通信，不顾及应用的状态改变而盲目检查更新，导致服务器资源的浪费，且会加重网络负载，拖累服务器。**

**二、comet**

**基于 HTTP 长连接的 "服务器推" 技术，能使服务器端主动以异步的方式向客户端程序推送数据，而不需要客户端显式的发出请求，目前有两种实现方式：**

**1. 基于 AJAX 的长轮询（long-polling）方式**

**优点**

**客户端很容易实现良好的错误处理系统和超时管理，实现成本与Ajax轮询的方式类似。**

**缺点**

**需要服务器端有特殊的功能来临时挂起连接。当客户端发起的连接较多时，服务器端会长期保持多个连接，具有一定的风险。**

**2. 基于 Iframe 及 htmlfile 的流（streaming）方式**

**Comet的优缺点**

**优点： 实时性好（消息延时小）；性能好（能支持大量用户）  
缺点： 长期占用连接，丧失了无状态高并发的特点。**

**三、websocket**

**WebSocket是HTML5开始提供的一种在单个 TCP 连接上进行全双工通讯的协议。WebSocket通讯协议于2011年被IETF定为标准RFC 6455，WebSocketAPI被W3C定为标准。在WebSocket API中，浏览器和服务器只需要做一个握手的动作，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。**

**24.h5c3有哪些新增特性**

**h5新增的标签**

| **新增元素** | **说明** |
| --- | --- |
| **video** | **表示一段视频并提供播放的用户界面** |
| **audio** | **表示音频** |
| **canvas** | **表示位图区域** |
| **source** | **为video和audio提供数据源** |
| **track** | **为video和audio指定字母** |
| **svg** | **定义矢量图** |
| **code** | **代码段** |
| **figure** | **和文档有关的图例** |
| **figcaption** | **图例的说明** |
| **main** |  |
| **time** | **日期和时间值** |
| **mark** | **高亮的引用文字** |
| **datalist** | **提供给其他控件的预定义选项** |
| **keygen** | **秘钥对生成器控件** |
| **output** | **计算值** |
| **progress** | **进度条** |
| **menu** | **菜单** |
| **embed** | **嵌入的外部资源** |
| **menuitem** | **用户可点击的菜单项** |
| **menu** | **菜单** |
| **template** | **模板** |
| **section** | **区块** |
| **nav** | **导航** |
| **aside** | **侧边栏** |
| **article** | **文章** |
| **footer** | **底部** |
| **header** | **头部** |

**- css3   
css3被划分为模块，最重要的几个模块包括：选择器、框模型、背景和边框、文本效果、2D/3D 转换、动画、多列布局、用户界面**

* **选择器**
* **框模型**
* **背景和边框   
  border-radius、box-shadow、border-image、   
  background-size：规定背景图片的尺寸   
  background-origin：规定背景图片的定位区域   
  background-clip：规定背景的绘制区域**
* **文本效果（常用）   
  text-shadow：设置文字阴影   
  word-wrap：强制换行   
  word-break   
  css3提出@font-face规则，规则中定义了font-family、font-weight、font-style、font-stretch、src、unicode-range**
* **2/3D转换   
  transform：向元素应用2/3D转换   
  transition：过渡**
* **动画**
* **@keyframes规则：   
  animation、animation-name、animation-duration等**
* **用户界面（常用）   
  box-sizing、resize   
  css3新增伪类   
  ：nth-child()   
  ：nth-last-child()   
  ：only-child   
  ：last-child   
  ：nth-of-type()   
  ：only-of-type()   
  ：empty   
  ：target 这个伪类允许我们选择基于URL的元素，如果这个元素有一个识别器（比如跟着一个#），那么:target会对使用这个ID识别器的元素增加样式。   
  ：enabled   
  ：disabled   
  ：checked   
  ：not**

**25.正则验证邮箱**

**由于邮箱的基本格式为“名称@域名”，需要使用“^”匹配邮箱的开始部分，用“$”匹配邮箱结束部分以保证邮箱前后不能有其他字符，所以最终邮箱的正则表达式为**

**^[a-zA-Z0-9\_-]**[**+@[a-zA-Z0-9\_-]+(\.[a-zA-Z0-9\_-]+)+$**](mailto:+@[a-zA-Z0-9_-]+(\.%5ba-zA-Z0-9_-%5d+)+$)

**26.标准模式与怪异模式下盒子模型的计算方式**

**两种模式的区别：**

**标准模式会被设置的padding撑开，而怪异模式则相当于将盒子的大小固定好，再将内容装入盒子。盒子的大小并不会被padding所撑开。**

**标准模式:盒子总宽度/高度** = **设置宽度/高度+padding+border**。

**怪异模式:盒子总宽度/高度= 内容宽度/高度 + padding + border + margin;**

**27.你用到了es6中哪些新特性**

1. **默认参数**
2. **模版表达式**
3. **箭头函数**
4. **Promise**
5. **块级作用域的let和const**
6. **模块化**

**28.描述一个闭包**

**function A(){**

**var x = 1;**

**return function(){**

**return ++x;**

**}**

**}**

* + - 1. **存在一个函数A**
      2. **在函数A内部返回一个函数**
      3. **返回的函数引用A函数的私有变量**
      4. **这个返回的函数是A函数的闭包函数**

**29.一个箭头函数，如何获取传入的所有实参**

**用无限参数**

**var fn = (…args)=>{**

**console.log(args);**

**}**

**这样就可以打印传入的所有参数**

**注意：在箭头函数中不能使用arguments**

**30.通讯协议知道哪些？讲讲websocket协议。**

**http、https、websocket、tcp/ip**

**webSocket是H5的新协议，它先通过http请求的tcp层3次握手建立连接，然后通过请求带的update:webSocket信息将http协议转换成webSocket协议，再然后就可以服务端向客户端推送信息了。Websocket建立的是长连接，它是双工通信，允许服务器端主动推送信息到客户端。http建立的是短连接，无状态单工通信。**