1、ajax加载的页面，跳转到另外一个页面再跳转回来，内容相同，如何节约读取请求?

答：后台做缓存，读取缓存里面的数据。CDN

2、javascript数据类型?

答：

简单类型：

Number 数字类型

String 字符串类型

Boolean 布尔类型

Null 空类型

Undefined 未定义类型

复合：

Object 对象类型

Array 数组类型

Function 函数类型

Undefined类型只有一个值，即undefined。当声明的变量还未被初始化时，变量的默认值为undefined。

Null类型也只有一个值，即null。null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象,

alert(null==undefined) // true; alert(null===undefined) // false;

null==null 不成立

3、什么是闭包，闭包什么情况会导致内存溢出?

答：

直接来讲，闭包就是能够读取其他函数内部变量的函数，在Javascript语言中，只有函数内部的子函数才能读取局部变量，因此可以把闭包简单理解成"定义在一个函数内部的函数"，所以，在本质上，闭包就是将函数内部和函数外部连接起来的一座桥梁。闭包最大用处有两个，一个是可以读取函数内部的变量，另一个就是让这些变量的值始终保持在内存中。

闭包引起内存泄露的原因有两种：

1. 循环引用导致了内存泄漏

2. 由外部函数调用引起的内存泄漏

避免内存泄漏

1. 打破循环引用

2. 添加另一个闭包

3. 避免闭包自身

解决闭包：用完直接==null;

4、事件委托

答：

function f2(d) {

document.onclick = function(e) {

var e = e || window.event;

var target = e.srcElement || e.target;

if (target.tagName == “LI”) {

……

}

}

}

事件委托：利用事件冒泡机制，在最顶层触发事件的dom对象上绑定一个处理函数。在当有需要很多dom对象要绑定事件的情况下，可以使用事件委托。

e.srcElement || e.target: 获取事件发生时，最初的那个dom元素。

target.tagName：最初的那个dom元素的标签名字，例如：LI;

通过对元素属性的判断，例如标签名，ID ,CLASS等等，来执行不同的处理函数。

这样做比起每一个元素都绑定一个事件，然后为每一个事件再绑定一个函数效率要高很多。

上例：点击任何一个LI都会触发，document的onclick的事件，(注意这里的LI并没有绑定任何的onclick事

件)，这样就可以理解为一种“委托”。把事件交给document.让它去处理li点击后应该执行的语句。

5.常用验证正则：

答：

1、邮箱：/^([a-z0-9][\w\.\-]\*@[a-z0-9\-]+(\.[a-z]{2,5}){1,2})$/i

2、年龄：/^(1[89]|[2-9]\d|100)$/

3、url : /^((http:\/\/)?([a-z0-9][a-z0-9\-]\*\.)+[a-z]{2,5}(\.[a-z]{2,5})?\/?)$/

4、手机：/^((\+86[\-\s]?)?1(3\d|47|[58][0-35-9])[\-\s]?\d{4}[\-\s]?\d{4})$/

6. H5的一些新标签如何在低版本浏览器上显示

答： WQ

基本原理是在使用新标签之前通过dom操作创建这些标签，那么这些标签就可以在低版本浏览器上显示了。

创建方法，比如："article"标签，可通过"document.creatElement('article')"方法创建。通过javascript创建这些标签的方法在网上有一套完整的解决方案，可以在google code的svn上下载 html5.js,然后将这个js文件插入到文件头部就可以了，插入方法如下：

<!--[if lt IE 9]>

<script src='js/html5.js'></script>

<![endif]-->

7.关于异步加载（js放在页头和body底部区别）

Js默认同步加载，加载时是阻塞的，后面的元素要等JS加载完才可以加载，放在页头加载很慢，影响用户体验

1. defer ----只支持IE

defer属性规定是否对脚本执行进行延迟

1. async 异步（仅用于脚本）

规定一旦脚本可用，则异步执行

（3）动态创建script,插入到DOM中 加载后执行callback

8.异步编程的四个方法

（1）回调函数（基本方法）

优点：简单容易理解和部署

缺点:不利于代码的阅读和维护，程序比较混乱

（2）事件监听（采用事件驱动模式）

绑“done”的事件

1. 发布/订阅（观察者模式）一对多

让多个观察者对象 同时监听某一个主题对象，这个主题对象发生变化就会通知所有观察者，使他们自动更新自己

1. Promise对象

是common.js工作提出的一种规范，每个异步任务返回一个promise对象，该对象有then的方法允许回调

9、清除浮动有哪几种方法

答：

A: 父容器内放一个： <div style=”clear:both”></div>

B: 父容器的 css:overflow:hidden;width:auto; 也行，会有问题，比如父容器弹出一个框

如果只写overflow:hidden 的话， IE6下无效，所以需要加上 width:auto 或 zoom:1

C: after进行后续控制的高度位零的伪类层清除。  
 D：采用display：table将对象变成table形式  
 E：采用div标签，以及css的clear属性  
 F：采用br标签，以及css的clear属性  
 G：采用br标签，以及其自身HTML的clear属性

10.如果扩展js中原生的String对象?string的方法？

String.prototype.name= function(){}

Slice从字符串的第一个参数提取第二个参数，也可以截取数组。返回的结果类型：string/object

Substring 从字符串的第一个参数提取第二个参数，返回的结果类型，string。

Indexof 返回短字符串在长字符串出现的位置。

Lastindexof 返回最后一个短字符串出现的位置。

Replace 字符串的替换方法，

Split 字符串分割方法，能转换为数组，数组转换字符串，用jion（）方法。

11.document.ready()和window.onload的区别?

Document.ready()是jQuery中准备出发的事件，当加载到当前元素就执行了，

Window.onload是整个页面加载之后才执行。

12.性能优化？

根据两方面回答，文件和结构

文件方面

1. CSS文件链接在head头部，CSS文件压缩合并
2. Js文件放在body最下面

结构方面

1、不要使用全局变量。  
2、不要使用with语句；  
3、尽量少使用闭包；  
4、数组中不要嵌套太多；  
5、少用for-in循环；  
6、最好用js包含文件，且尽量放在最下面；  
7、该dom节点样式时尽量用class。  
8、减少http请求

----------------------常考题-------------------

编写一个方法 求一个字符串的字节长度

<script language="javascript" type="text/javascript">  
new function(s)  
{  
if(!arguments.length||!s) return null;  
if(""==s) return 0;  
var l=0;  
for(var i=0;i<s.length;i++)  
{  
   if(s.charCodeAt(i)>255) l+=2;  
   else l++;  
}  
alert(l);  
}("hello你好，我好,大家好！world!");  
</script>

如何控制alert中的换行？

<script language="javascript" type="text/javascript">  
alert("hello\nworld");  
</script>

解释document.getElementById("ElementID").style.fontSize="1.5em" ？

em是个相对单位

编写一个方法 去掉一个数组的重复元素

方法一：

<script language="javascript" type="text/javascript">  
Array.prototype.strip=function()  
{  
if(this.length<2) return [this[0]]||[];  
var arr=[];  
for(var i=0;i<this.length;i++)  
{  
   arr.push(this.splice(i--,1));  
   for(var j=0;j<this.length;j++)  
   {  
    if(this[j]==arr[arr.length-1])  
    {  
     this.splice(j--,1);  
    }  
   }  
}  
return arr;  
}  
var arr=["abc",85,"abc",85,8,8,1,2,5,4,7,8];  
alert(arr.strip());  
</script>

方法二：

var arr=[25,30,25,88,60,99,88,9];

function inArray(v,arr){

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

if(v==arr[i]){

return i;

}

}

return -1;

}

function removeRepeat(arr){

var newArr=[];

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

if(inArray(arr[i],newArr)==-1){//如果这个值不在新数组里

newArr.push(arr[i]);

}

}

return newArr;

}

console.log(removeRepeat(arr).sort(function(a,b){return a>b})); //用了逆序

字符串反转函数，代码应尽量简短。

答：

function reverse(str) {

return str.split(",").reverse().join(",")

}

简述JSONP的原理

JS：

① 动态创建 src标签；var script=document.createElement("script");

② 设置script的src为跨域的地址？callback=动态获取的地址

document.body.appendChild(script)

JQ:

① 动态创建 src标签；var script=document.createElement("script");

② 设置script的src为跨域的地址？callback=?

随机生成一个函数名：

var cbName="callback"+Math.random().toSring().substr(2,10);

将url地址中的callback=？替换为callback=生成的函数名

var url=opt.url.replace(/callback=\?/,"callback="+cbName)

[全局，不要冲突； 函数名是变量，通过[]; 污染了window，释放，window[cbName]=null; ]

JSONP原理

只要你聊到跨域，就必须聊到JSONP，那么就必须要讲一下JSONP的实现原理，以及你在项目中那个需求使用了JSONP，这里简单讲就是HTML里面所有带src属性的标签都可以跨域，如iframe，

img，script等。所以可以把需要跨域的请求改成用script脚本加载即可，服务器返回执行字符串，但是这个字符串是在window全局作用域下执行的，你需要把他返回到你的代码的作用域内，

这里就需要临时创建一个全局的回调函数，并把到传到后台，最后再整合实际要请求的数组，返回给前端，让浏览器直接调用，用回调的形式回到你的原代码流程中。