1. **Q：javascript的typeof返回哪些数据类型**

答案：基本数据类型：string,boolean,number,undefined,function, object

**------------------------------------------------------------------------------**

1. **Q：如何判断某变量是否为数组数据类型？**

答案：方法一.判断其是否具有“数组性质”，如push()方法。由于自己也可以给该变量定义push方法，所以这种方式并不准确

方法二.obj instanceof Array 在某些IE版本中不正确

方法三.方法一二皆有漏洞，在ECMA Script5中定义了新方法Array.isArray(), 保证其兼容性，最好的方法如下：

if(typeof Array.isArray==="undefined"){

Array.isArray = function(arg){

return Object.prototype.toString.call(arg)==="[object Array]"} }

**------------------------------------------------------------------------------**

1. **Q.例举3种强制类型转换和2种隐式类型转换？**

**------------------------------------------------------------------------------**

答案：强制（parseInt,parseFloat,Number）

隐式（== - \* / + > < >= <= ）

1. **Q.split() join() 的区别**

答案：前者是切割成数组的形式，后者是将数组转换成字符串

**------------------------------------------------------------------------------**

1. **Q.数组方法pop() push() unshift() shift()**

答案：Push()尾部添加 pop()尾部删除 Unshift()头部添加 shift()头部删除

**------------------------------------------------------------------------------**

1. **Q.****IE和标准下有哪些兼容性的写法**

答案：var ev = ev || window.event

document.body || document.documentElement

xxx.currentStyle || window.getComputedStyle(xxx, null)

event.cancelBubble = true; event.stopPropagation();

**------------------------------------------------------------------------------**

1. **Q.****ajax请求的时候get 和post方式的区别**

答案：1.一个在url后面 一个放在虚拟载体里面

2.get有大小限制 4KB, post理论上没有限制

3.安全问题

4.POST字符集更容易处理，get请求需要服务器对URL进行转码

5.需求不同，请求数据和提交数据

**------------------------------------------------------------------------------**

1. **ajax请求时，如何解析json数据**

答案：使用eval JSON.parse 鉴于安全性考虑 使用JSON.parse更靠谱

**------------------------------------------------------------------------------**

1. **写一个获取非行间样式的函数**

答案：

function getStyle(obj, attr){

if(obj.currentStyle){

return obj.currentStyle[attr];

} else {

return getComputedStyle(obj,null)[attr];

}

}

**------------------------------------------------------------------------------**

1. **事件委托是什么**

**------------------------------------------------------------------------------**

答案：利用事件冒泡或捕获的原理，让自己的所触发的事件，让他的父元素代替执行！

1. **闭包是什么，有什么特性，对页面有什么影响**

答案：闭包就是能够读取其他函数内部变量的函数。 闭包的存在延长了局部变量的生命周期，闭包的形式很多变，但本质一样，即局部函数但全局执行，要注意识别。

举例：

obj.onclick = function(){

var num = 99;

setInterval(function(){ //定时器的匿名函数即为闭包

console.log(num);

},1000);

}

**------------------------------------------------------------------------------**

1. **添加、移除、移动、复制、创建和查找节点的方法**

答案：1）创建新节点

createDocumentFragment() //创建一个DOM片段

createElement() //创建一个具体的元素

createTextNode() //创建一个文本节点

2）添加、移除、替换、插入

appendChild() //添加

removeChild() //移除

replaceChild() //替换

insertBefore(newnode, exsitnode) //插入

3）查找

getElementsByTagName() //通过标签名称

getElementsByName() //通过元素的Name属性的值

getElementById() //通过元素Id，唯一性

**------------------------------------------------------------------------------**

1. **jsonp为什么不是真正的ajax**

答案：jsonp动态创建script标签，在回调函数中包裹数据，没有通过XMLHttpRequest对象

Ajax是页面无刷新请求数据操作

**------------------------------------------------------------------------------**

1. **window load 和document ready的区别**

**------------------------------------------------------------------------------**

答案： document.ready 当DOM结构加载完成即执行

window.onload 当页面所有资源加载完成即执行

1. **”==”和“===”的不同**

**------------------------------------------------------------------------------**

答案：前者会自动转换类型 后者不会

答案：一段脚本只能读取来自于同一来源的窗口和文档的属性，这里的同一来源指的是主机名、协议和端口号的组合

1. **javascript的同源策略**

**------------------------------------------------------------------------------**

1. **排序算法**

答案：

快速排序function oSort(arr){

if(arr.length<=1){

return arr }

var left=[];

var right=[];

var oNum = Math.floor(arr.length/2);

var oNumVode = arr.splice(oNum,1);

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

if(arr[i]<oNumVode) {

left.push(arr[i]) }

else { right.push(arr[i]) } }

return oSort(left).concat([oNumVode],oSort(right))}

冒泡排序 var array = [5, 4, 3, 2, 1];

var temp = 0;

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

for (var j = 0; j < array.length - i; j++){

if (array[j] > array[j + 1]){

temp = array[j + 1];

array[j + 1] = array[j];

array[j] = temp; } } }

**------------------------------------------------------------------------------**

1. **在Javascript中什么是伪数组？如何将伪数组转化为标准数组？**

答案：伪数组（类数组）：无法直接调用数组方法或期望length属性有什么特殊的行为，但仍可以对真正数组遍历方法来遍历它们。典型的是函数的 argument参数，还有像调用getElementsByTagName,document.childNodes之类的,它们都返回 NodeList对象都属于伪数组。可以使用Array.prototype.slice.call(fakeArray)将数组转化为真正的Array 对象。

**------------------------------------------------------------------------------**

1. **在javascript中如何检测客户端的浏览器和操作系统类型**

答案：var resolution = "分辨率:" + window.screen.width + "\*" + window.screen.height;,

var ua = window.navigator.userAgent

**------------------------------------------------------------------------------**

1. **Cookie在客户机上是如何存储的**

答案：Cookies就是服务器暂存放在你的电脑里的文本文件，好让服务器用来辨认你的计算机。当你在浏览网站的时候，Web服务器会先送一小小资料放在你的计算机上，Cookies 会帮你在网站上所打的文字或是一些选择都记录下来。当下次你再访问同一个网站，Web服务器会先看看有没有它上次留下的Cookies资料，有的话，就会依据Cookie里的内容来判断使用者，送出特定的网页内容给你。

**------------------------------------------------------------------------------**

1. **Prompt box是什么？它的返回值有什么用？**

答案：prompt() 方法用于显示可提示用户进行输入的对话框。如果用户单击提示框的取消按钮，则返回 null。如果用户单击确认按钮，则返回输入字段当前显示的文本。

在用户点击确定按钮或取消按钮把对话框关闭之前，它将阻止用户对浏览器的所有输入。在调用 prompt() 时，将暂停对 JavaScript 代码的执行，在用户作出响应之前，不会执行下一条语句。

**------------------------------------------------------------------------------**

1. **如何获取javascript数组中的最大值和最小值？**

答案：alert(Math.max(1,23,56,88,56));//最大值

alert(Math.min.apply(null, [3,12,56,22,83,67]));//最小值

**------------------------------------------------------------------------------**

1. **判断字符串以字母开头，后面可以是数字，下划线，字母，长度为6-30**

答案：vasr reg=/^[a-zA-Z]\w{5,29}$/;

1. **请举例使用callee属性实现函数的递归使用**
2. **Ajax同步和异步的区别，如何解决跨域问题?**

答案：

(function(num){

if(num <= 1) return 1;

return num\*arguments.callee(num-1);

})(5)

解释：callee代表函数本身的引用

1. **apply和call方法有形似之处，请说出二者的区别**

答案：同步即按步骤顺序执行，异步即同时执行

jsonp，即动态执行json数据处理的方法，使用JS在浏览器页面上动态生成一个<script>标签，并加载js文件来执行。

需要服务器支持。

1. **请解读一下javascript代码，并指出问题所在**

答案：

apply固定是两个参数，第一个是调用的对象，第二个是一个数组  
call接受的参数是不固定的，第一个是对象 ，第二个参数数量不确定

|  |
| --- |
| var Obj=function(msg){  this.msg=msg;  this.shout=function(){  alert(this.msg);  }  this.waitAndShout=function(){  setTimeout(this.shout, 2000);  }  }  var aa=new Obj("abc");  aa.waitAndShout(); |

1. **请给Array本地对象增加一个原型方法，他的用途是删除数组中重复的条目，并将新的数组返回**

Array.prototype.distinct = function(){

var \_arr = new Array();

var obj = {};

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

var val = this[i];

if(!obj[val]){

obj[val]= 1;

\_arr.push(val);

}

}

return \_arr;

}

定时器函数的执行，都是处在全局的环境中，2秒后执行的shout函数，在获取this.msg时已经取不到内容了，因为此刻的this已经代表了window对象。

var Obj=function(msg){

this.msg=msg;

**var that = this;**

this.shout=function(){

alert(this.msg);

}

this.waitAndShout=function(){

setTimeout(function(){

**that**.shout();

}, 2000);

}

}

var aa=new Obj("abc");

aa.waitAndShout();

使用变量that将当前引用保存起来，执行结果就正确了

1. **JavaScript是一门什么样的语言，它有哪些特点？**(无标准答案)
2. **希望获取到页面中所有的checkbox怎么做？(不使用第三方框架)**

var arr = new Array();

var boxs = document.getElementsByTagName("input");

for(var index in boxs) {

if(boxs[index].type == "checkbox"){

arr.push(boxs[index]);

}

}

return arr;

1. **当一个DOM节点被点击时候，我们希望能够执行一个函数，应该怎么做？**
2. **看下列代码输出为何？解释原因。**

box.onclick = function(){}

box.addEventListener("click",function(){},false);

function xxx(){}

<button onclick="xxx()"></button>

|  |  |
| --- | --- |
| 1  2  3 | var a;  alert(typeof a); // undefined  alert(b); // 报错 |

1. **看下列代码,输出什么？解释原因。**

变量a未赋值，初始值为undefined

变量b未定义，所以报错

|  |  |
| --- | --- |
| 1  2 | var a = null;  alert(typeof a); //object |

1. **看下列代码,输出什么？解释原因。**

所有对象的默认值都是null

|  |  |
| --- | --- |
| 1  2  3 | var foo = "11"+2-"1";  console.log(foo);  console.log(typeof foo); |

111 number

1. **将字符串”<tr><td>{$id}</td><td>{$name}</td></tr>”中的{$id}替换成10，{$name}替换成Tony （使用正则表达式）**
2. **为了保证页面输出安全，我们经常需要对一些特殊的字符进行转义，请写一个函数escapeHtml，将<, >, &, “进行转义 &lt; &gt; &amp; &quot;**

function escapeHtml(str) {

return str.replace(/[<>"&]/g, function(match) {

switch (match) {

case "<": return "&lt;";

case ">": return "&gt;";

case "&": return "&amp;";

case "\"": return "&quot;";

}

});

}

str.replace(/\{\$id\}/g,"10").replace(/\{\$name\}/g,"Tony");

1. **看下列代码，将会输出什么?(变量声明提升)**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | var foo = 1;  (function(){      var foo;  console.log(foo);      foo = 2;      console.log(foo);  })(); |

var foo = 1;

(function(){

var foo;

  console.log(foo); //两种写法是等价的

  foo = 2;

  console.log(foo);

})();

1. **有这样一个URL：**<http://item.taobao.com/item.htm?a=1&b=2&c=&d=xxx&e> **请写一段JS程序提取URL中的各个GET参数(参数名和参数个数不确定)，将其按key-value形式返回到一个json结构中，如{a:'1', b:'2', c:'', d:'xxx', e:undefined}。**

var url = "http://item.taobao.com/item.htm?a=1&b=2&c=&d=xxx&e";

var paramstr = url.substring(url.indexOf("?")+1);

var pms = paramstr.split("&");

var obj = {};

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

if(pms[i].indexOf("=") < 0){

obj[pms[i]] = "undefined";

}else{

var pa = pms[i].split("=");

obj[pa[0]] = pa[1];

}

}

alert(obj.toSource());

1. **看下面代码，给出输出结果。**

|  |  |
| --- | --- |
|  | **for(var i=1;i<=3;i++){**  **setTimeout(function(){**  **console.log(i);**  **},0);**  **};** |

1. **写一个function，清除字符串前后的空格。（兼容所有浏览器）**

String.prototype.trim = function(){

return this.replace(/^\s+/,"").replace(/\s+$/,"");

}

1. **下面这个ul，如何点击每一列的时候alert其index?（闭包）**

|  |  |
| --- | --- |
| 1  2  3  4  5 | <ul id=”test”>  <li>这是第一条</li>  <li>这是第二条</li>  <li>这是第三条</li>  </ul> |

1. **给String对象定义一个repeatify方法。该方法接收一个整数参数，作为字符串重复的次数，最后返回重复指定次数的字符串。例如：**

var arrLi = document.getElementById("test").children;

for(var k=0; k<arrLi.length; k++) {

arrLi[k].onclick = (function(num){

return function(){

console.log(num);

};

})(k);

}

|  |  |
| --- | --- |
| 1 | console.log('hello'.repeatify(3)); |

**输出应该是**

|  |  |
| --- | --- |
| 1 | hellohellohello. |

1. **以下哪条语句会产生运行错误：**

**A.var obj = ();**         B.var obj = [];        C.var obj = {};        D.var obj = //;

1. **给Array原型添加一个方法,这个方法接受一个参数,如果这个参数在数组中存在那么就把它从数组中删除。**

|  |
| --- |
| Array.prototype.remove = fucntion(obj) {  for(var i in this) {  if(typeof this[i] == "object") {  var flag = true;  for(var attr in this[i]){  if(this[i][attr] != obj[attr]){  flag = false;  break;  }  }  if(flag) {  this.splice(i,1);  }  } else {  if(this[i] == obj) {  this.splice(i,1);  }  }  }  } |

1. **对下面这个对象进行克隆.** var student = { name:”zhang3″, sno:”20080328″, age:”20″, call:function(){alert(this.name+this.sno+this.age);} };
2. **请问p.name是多少？**

|  |
| --- |
| var fun = function(){  this.name = 'peter';  return {  name: 'jack'  };  }  var p = new fun(); |

1. **请问a.info.name和b.info.name分别是多少？**

|  |
| --- |
| var fun = function(){  }  fun.prototype = {  info : {  name : 'peter',  age : 25  }  }  var a = new fun();  var b = new fun();  a.info.name = 'jack';  b.info.name = 'tom'; |

1. [**写一个traverse函数，输出页面所有宽度和高度大于50像素的节点。**](http://www.nowcoder.com/questionTerminal/448399736ff34b9399d04ca6ca94567e?orderByHotValue=0&done=0&pos=1&mutiTagIds=571&onlyReference=false)
2. **填写内容让a.name 等于“name1” 让 b.name 等于“name2”**

|  |
| --- |
| function obj(name){      1  }  obj.  2 = "name2";  var a = obj("name1");  var b = new obj(); |

1. javascript 语言特性中，有很多方面和我们接触的其他编程语言不太一样，比如说，javascript语言实现继承机制的核心就是  1  (原型)，而不是Java语言那样的类式继承。Javascript 解析引擎在读取一个Object的属性的值时，会沿着  2  (原型链)向上寻找，如果最终没有找到，则该属性值为  3  undefined； 如果最终找到该属性的值，则返回结果。与这个过程不同的是，当javascript解析引擎执行“给一个Object的某个属性赋值”的时候，如果当前Object存在该属性，则改写该属性的值，如果当前的Object本身并不存在该属性，则赋值该属性的值 。

var a = function b(){

     b = 3;

     console.log(b);

}

a();

    //function b 看做一个有名字的 匿名函数

    //一个有名字的匿名函数，其函数名  是只读的，不可更改

    //同时，该函数名只能在函数内部使用，相当于局部变量

    //b(); 报错，无法调用，找不到b函数

if (!("a" in window)) {

    var a = 1;

}

alert(a);

**结果： undefined**

var a = 1,

    b = function a(x) {

        x && a(--x);

    };

alert(a);

**结果 1**

function b(x, y, a) {

    arguments[2] = 10;

    alert(a);

}

b(1, 2, 3);

**结果 10 严格模式下 3**