javascript语句：注意事项  
1、遇到“-”,要去掉然后把后面字母的第一位变大写  
如：font-size=>fontSize  
2、遇到class要在后面加上Name  
如：className  
3、js里尽量不要使用相对路径来做判断  
如：img.src、href=’1.css’……  
4、js里尽量默认的颜色值也不要拿来做判断(颜色值的表示方式比较多)  
5、js里innerHTML的值也尽量不要拿来做判断  
6、.后面的值无法修改，[]里面的值可以随便写，js中允许”.”替换成”[]”，但是记得加’’  
7、js可以为任何HTML元素添加任意个自定义属性，只要是对象都可以添加，也可以添加函数  
如: \*\*\*.\*\*=\*;-给\*\*\*元素自定义一个\*\*属性，值为\*  
8、只要是从元素中拿到的内容都是字符串(string)  
9、在正式运行代码之前所有的变量都是未定义，而函数则为函数块，而这样的过程被称为域解析  
tp:在正式运行代码之前如果变量和函数重名了就只会留下函数，在运行代码时表达式是可以改变变量的值的(是函数块也能改)  
10、只要能改变变量值的都是表达式  
tp:js标签就是一个域多个标签里如果有同名的变量会互相影响，除了js标签以外函数也是一个域，结构体也是一个域，但是函数与结构体里的变量是跟js标签里的变量是不一样的，函数与结构体里的变量会遵循就近原则，如果他的作用域里没有那么就会触发做用域链  
做用域链:当变量在函数内没有声明时，那么会使用其父级同名的变量值   
tp:全局变量-从上而下 局部变量-从里到外

json-结构体：  
读取整个json键值的方法-for(var attr in json){alert(attr);}  
读取整个json键值内容的方法-for(var attr in json){alert(json[attr]);}

javascript语句：  
<script type="text/javascript" charset=”utf-8” src="js/index.js">中间代码无效</script>外部样式

事件格式：(内部样式)  
<script>

var oBtn=document.getElementById(‘btn1’);

oBtn.onload=function(){

alert(oBtn.value);

};

Onload 加载完成后再执行一般与Window连用

例如 window.onload=function(){}

</script>

js数组  
var \*\*=[‘\*\*\*’, ‘\*\*\*’] 定义一个数组  
tp:var后面跟数组名字，括号内是数组内容用，隔开区分个数，数组是从0开始计数的  
\*\*[\*] 访问数组名为\*\*中的第\*个元素  
\*\*\*.length 统计数组名为\*\*\*中的元素的个数  
tp:也可以修改数组的长度用来达到删除或清空数组的目的  
\*\*1.concat(\*\*2,\*\*3); 合并数组  
\*\*\*.reverse(); 颠倒数组  
\*\*\*.unshift(\*\*) 把数组名为\*\*\*的前面添加一个元素为\*\*  
\*\*\*.push(‘\*\*’) 把数组名为\*\*\*的最后添加一个元素为\*\*  
tp:push();有返回值其值为新数组的长度  
\*\*\*.pop(); 移除数组名为\*\*\*的最后一个元素  
\*\*\*.shift() 移除数组名为\*\*\*的最前一个元素  
\*\*\*.splice(); 删除、添加、替换具体请看实例  
\*\*\*.splice(0,1); 从下标为零的元素开始删除一个  
\*\*\*.splice(0,1,’\*’); 从下标为零的元素开始替换一个,替换为\*  
\*\*\*.splice(1,0,’\*’); 从下标为零的元素后面添加一个\*元素  
tp:返回值都是被操作的数  
\*\*\*.sort(); 为数组的每个数排序，用字符串的方式  
tp:可以给它添加排序方法  
Math:

Math.random(); 随机函数，随机生成0~1之间的数字  
Math.round(); 四舍五入  
Math.ceil(); 向上取整  
鼠标事件：

onclick 鼠标单击时触发事件  
onmouseover 鼠标移入时触发事件  
onmouseout 鼠标移开时触发事件  
onmousemove 鼠标长时间操作事件  
onmousedown 鼠标下移时触发事件  
onmouseup 鼠标上移时触发事件

函数传参的要素：

1、尽量保证函数结构一致，可以通过父级选子集  
2、核心主程序要能实现，然后用函数包裹  
3、把每组中不同的值找出来，通过传参实现

document.createElement(“\*\*\*”) 自定义、自创建标签(默认为内嵌元素)

多选框：  
<select id=”select1”>

<option value=”上海”>上海</option>

<option value=”杭州”>杭州</option>

<option value=”北京”>北京</option>

(前面的是后端工程师设置的后面的是给用户看的，通常保持一致)

(选中的是哪个城市那么弹出的value也是对应的城市名称)

</select>

+ 可以表示数字相加也可以表示字符串的连接

innerHTML-读取元素内的所有HTML内容(添加代码也可以起到作用)一般用在p身上

属性读写注意事项：  
直接调用函数：函数名后+;  
属性读操作语法(获取、找到):元素.属性名;  
属性写操作语法(替换、修改):元素.属性名 = 新的值;  
var oDiv = document.getElementById('div1'); 用ID获取元素

getElementsByTagName 获取一类元素  
tp:TagName是一种动态获取元素的方法所以它前面的document是可以改变的，而且信息事实更新  
doucument.body 获取元素(但是只能获取页面上只有一个的元素)

alert( oText.value); 文本中用户输入的内容会保存在value里

alert(a); 弹出系统消息

var a = 123; 定义语句

defer=”defer” 延迟执行该程序(当页面加载完毕时)

innerHTML div里的文字

Label 打包(分组)元素与字

Function 函数

Length 求某一个元素的长度

Index 引索一个标签

\*document.write(‘我是第一个程序’); 文档对象

数据类型：  
typeof 判断数据类型  
tp:number-数字类型 string-字符串类型 booleam-布尔值类型(true false) 函数名不加括号就是函数里的内容 object-对象(null属于特殊对象) undefined-未定义  
数据类型的转换:  
Number(\*,10); 转换为数字型  
tp:如果是空的或是空格就传成0,true-1 false-0，传不了的话就是NaN(number-false,与自己都不相等),10就是10进制  
parseInt 转换为数字类型  
tp:从左到右碰到非数字就结束转换,+- 不会影响  
parseFloat 转换为浮点型(小数)  
+-\*/……也可以进行转换，类型为最后的数据的类型,!-取反  
isNaN(\*) 先用Number(\*)进行转换，如果是数字就是false，不是数字就是true

console.log() 在控制台输出

backgroundColor 背景颜色

函数包裹代码(方便起名)  
function i()

{

alert(“”);

}

参数i:函数名

定时器：

开启  
var \*=setInterval(函数名，时间(毫秒)) 间隔性—重复执行  
tp:时间尽量写14以上的

setTimeout(函数名，时间(毫秒)) 延时性—只有一次

关闭(给定时器命名timer=null)  
clearInterval(变量名) 停止变量名称的定时器  
clearTimeout(变量名) 停止变量名称的定时器

var oDiv=new Date(\*,\*,\*,\*,\*,\*) 获取系统时间  
参数(不写的话就是系统时间):从左到右为年月日时分秒

getFullYear() 获取系统时间-年

getMonth() 获取系统时间-月(从零开始的)

getDear() 获取系统时间-日

getDay() 获取系统时间-星期(星期日=0)

getHours() 获取系统时间-时

getMinutes() 获取系统时间-分

getSeconds() 获取系统时间-秒

tp:除了获取系统时间是obj，其他均为数字

offsetWidth 某个元素的宽度的数值

获取窗口到顶部的垂直距离  
window.onscroll = function() {   
console.info(window.scrollY);   
}

用Class来获取元素是一个数组，不能一次性定义

使用for循环的注意事项：  
1、如果for循环里有包裹其他函数那么，这时你如果想读取i每次的变化，是不可以的只能读到最后一个值。

改变元素样式的方法：  
1、\*\*\*.style.width=’200px’; 给\*\*\*元素添加行间样式  
2、\*\*\*.style.cssText=’width=200px;’ 给元素的外部css样式上添加样式  
tp:如果对同一个元素写两个外部样式的话，后写的会把前面写的给替换掉，不是添加上去。

this 执行(调用)该函数的源头(对象)

getComputedStyle(\*\*\*.width) 获取浏览器计算后的最终的样式  
tp:IE8及一下要改成\*\*\*.currentStyle.width，尽量不要用复合样式

return的应用:  
1、return一般用在函数后面，而且return后面的语句都不会执行  
2、每一个函数后面都有一个默认的return，值为未定义(…)  
3、函数名+()就是return后面的值

顺便一提:arguments-一个函数的实际参数的集合 当函数的个数无法确定的时候可以使用arguments

字符串:  
\*.charAt() 获取字符串里的单个字符  
\*.charCodeAt() 获取单个字符的编码值  
tp:默认为括号里为零,如果找不到就是空白  
String.fromCharCode(\*\*\*) 将编码转换为汉字  
\*\*\*.indexOf(‘\*’，‘\*\*’) 找出\*\*\*字符串里\*的位置  
参数:\*\*\*-字符串的名字 \*-要找到的字符 \*\*-从该字符串的第\*\*位开始寻找,默认为0  
tp:如果没有找到就为-1，而且找到之后就不会再找了,从左往右寻找  
\*\*\*.lastIndexOf(‘\*’, ‘\*\*’) 找出\*\*\*字符串里\*的位置  
tp:同上，唯一不一样的是从右往左找的而且如果\*\*是负数的话就当成零来处理  
\*\*\*.substring(\*，\*) 截取字符串从\*以后到\*的所有字符  
tp:\*，\*没写的话就截取所有，\*为负数就当0来处理，而且前面的\*比后面的\*小(交换)  
slice(\*，\*) 截取字符串从\*以后到\*的  
tp:与substring相比它不会交换位置，但是负数会从最后开始计算  
\*\*\*.toUpperCase() 将所有\*\*\*里的小写字母转换为大写  
\*\*\*.toLowerCase() 将所有\*\*\*里的大写字母转换为小写

\*\*\*.split(‘\*’，‘\*\*’) 以\*为分隔符分段\*\*\*字符串，变成一个数组  
tp:只要使用后无论是否有\*字符串都会变成一个数组，如果就写‘’就把字符串一个一个分段，也可以‘\*’，\*\*代表数组存放的最大个数  
\*\*\*.join() 把\*\*\*从一个数组变成一个字符串  
tp:括号内为连接符

隐视类型转换： - \* /% ++ --

tp:字符数字跟逻辑数字进行运算时，用以上符号可以将字符数字转真逻辑数字，进行逻辑运算。

回调函数：endFn 例：

change(1,2,3,function{alert(1);})

function change(one,two,three,endFn){

if(endFn){

endFn();

}//等同于 endFn && endFn();

tp:判断endFn不是nudefind 不为空 有定义函数 则执行；

}

tp：在函数里面 传递一个参数，这个参数是一个函数，则需要用endFn来调用，调用时还需要判断endFn是否定义；

Tp:键盘事件:

onkeypress 事件会在键盘按键被按下并释放一个键时发生

onkeydown 事件会在键盘按下一个键时发生；

onkeyup 事件会在键盘松开一个键时发生

Event 对象代表事件的状态，比如事件在其中发生的元素、键盘按键的状态、鼠标的位置、鼠标按钮的状态。

tp:事件通常与函数结合使用，函数不会在事件发生前被执行！

Clientheight/width/top..获取指定属性的值;

tp:用于position后的div无法直接style获取元素宽高的问题，用这个

document.documentElement.clientHeight 获取当前页面高度

document.documentElement.clientWidth 获取当前页面宽度

tp:贼好用！

History.back() 退回上一格,回退，退步，回格，返回。