**f声明变量 var**  注：可声明的数据类型：

字符串(var flute=”橘子”)、

数字(var i=10;)、

布尔(var b=true;)、

数组(var arr1=[1,2,3,4];var arr2=new Array(1,2,3,4);**动态数组**var arr3=new Array();arr3[0]=1;arr3[1]=2;......)、

对象、

空(var n=null;注：空类型可继续赋值为其他类型)、

未定义(var r;)、

通过赋值为空清除变量(i=null;)、

函数变量(var f=sayHello();f();)。

**声明全局变量 可在函数中用不加var的方式声明全局变量** 如：x=10;

**比较运算符 ==，===，！=，！==较特殊** 注：==不一定类型相同，===一定类型相同。如：“10”==10为true;“10”===10为false。!=与!==同理。

**For/In循环 for(属性名变量 in 对象名)**

**标签名:{break 标签名;} 跳出标签名**

**定义函数 function 函数名([参数列表]){函数体;[return xx;]}** 注：函数调用可在js中调用，也可以在html中表单事件来调用。

**异常捕获 try{}catch(err){错误信息处理;}**

**抛出自定义异常 throw 内容;**

**信息封装 (function(){})();** 将信息封装在一个函数中，可将其中的属性或方法添加到window对象，以访问。 注：函数后加一小括号以立即执行。

**创建对象 直接创建1 对象名=new Object();对象.属性=属性值;****对象.方法=方法定义;......**

**直接创建2 对象名={属性名:属性值,方法名:function(){}......};[对象.属性=属性值;对象.方法=方法定义;......]​​**

**通过构造器创建 function 构造器函数名(参数列表){this.属性名=参数;......};[原型链（如下）]对象名=new 构造器函数(参数列表);[对象.属性=属性值;对象.方法=方法定义;......]** 注：声明构造其器函数后，会自动生成一个同名对象。 注：构造函数如果有返回值：当返回值不是对象，该返回值无用；当返回值是对象，将新创建的对象设为该对象，原来的原型链和this设置失效。其实相当于不写this时使用new有一个默认的return this。

**原型链 构造器函数名.prototype={属性名:属性值,方法名:function(){}......};**

**构造器函数名.prototype.属性/方法=属性值/方法定义;** 注：原型链中赋的属性值为构造器生成的对象的默认值。 注：若构造器和原型链中指明了同名属性的值，则按构造器。

**构造器的继承 子构造器函数名.prototype=new 父构造其函数名();** 注：若子对象中有与父对象同名方法，父类方法将会覆盖。

**调用父类方法 声明子类同名方法前，声明一函数变量”var 名A=构造器名.prototype.方法名;”，调用时使用”名A.call(this)“。**

**创建对象2 可通过赋值的方式事件对象的创建、继承等。** 注：对象直接赋值后仍占用同一内存，即改变一个都改变。

**对象操作 delete 对象.属性名** 删除对象属性

**Numbuer toFixed(位数)**将字符串四舍五入为指定位数

**字符串** 注：字符串既可以用双引号，也可以用单引号。

**length;**

**indexOf(子串);**查询子串在父串中的位置,返回位置或-1

**match(子串);**测试子串是否在父串中存在，返回子串或null

**replace(子串,新子串);**替代字符串中的字串，成功返回新串，失败返回原串

**toUpperCase()/toLowerCase();**转换大小写

**split(“分隔符”);**将字符串分割为字符串数组，返回此数组。

**substr(start,[length]);**截取字符串

**Object Object.keys(obj)**得到包含对象所有key的数组（不检测原型链）

**obj hasOwnProperty(key)**检测对象中是否有此属性（不检测原型链）

**isPrototypeOf(obj)**检查传入对象是否是当前对象的原形

**数组 concat(数组);** 合并数组

**sort([函数]);** 排序 注：函数书写方法function(a,b){return a-b;}此为升序，b-a为降序。

**push(元素);** 追加，为数组添加一个元素

**reverse();** 反转

**forEach(function);** 为每个数组元素执行函数，函数有三个参数：元素值，[索引]，[数组本身]

**日期对象 var 对象名=new Date();** 注：创建后为当前时间。

**getFullYear();getDay();getTime();** 获取年份/星期/毫秒（从1970.1.1至今）

**setFullYear(年,[月],[日]);** 设置日期

**Math对象 round();**四舍五入 **ceil()**向上取整 **floor()**向下取整 **random();** 0-1的随机数 **max(任意个数值);**最大值 **min(任意个数值);**最小值 **abs(数值);**绝对值

**在控制台输出 console.log(“内容”);** 注：此方法只在部分浏览器支持。

**弹出对话框 alert(内容);**

**解析值 parseInt();** 将其他类型值解析为int型。不是整数返回NaN

**document 通过id document.getElementById(“元素id”);**

**通过name document.getElementsByName(“元素name”);**

**通过class document.getElementsByClass(“元素class”);** 注：带s系列方法返回nodelist，可通过下标访问，也可通过item(下标)访问。

**通过标签 document.getElementsByTagName(“标签名”);**

**直接得到body元素 document.body;**

**直接得到文档根元素 document.documentElement;**

**输出 document.write(“”)**  注：输出中可以带html的标签。 注：字符串可用+连接。 注：不要在文档加载完成后使用document.write(),这回覆盖该文档。

**document.domain**返回现在当前文档的服务器域名 注：可将其设置为当前域的父域，以解决跨域问题

**document.referrer**返回载入当前文档的文档的url

**节点 nodeName;**节点名，如：p，div

**nodeType;**节点类型

**节点尺寸 clientHeight;/clientWidth;**元素宽高（不包括border）

**offsetHeight;/offsetWidght;** 元素宽高（包括border）

**scrollHeight;/scrollWidth;**元素的正文全文宽高（滚动条的总长度）

**clientTop;/clientLeft;**此节点border的实际宽度

**offsetTop;/offsetLeft;**此节点相对于根节点滚动条顶部的位置（若滚动条中有滚动条，按滚动条高度叠加计算）

**scrollTop;/scrollLeft;**此节点当前滚动条的位置（滚动条被卷起多少）

**访问父节点 parentNode;**

**访问子节点 childNodes;** 注：注意空白项（某些浏览器将换行的空白看为一个节点）。

**访问最后一个子节点 lastchild**

**创建节点 document.createElement(“节点名”);** 注：可对此节点进行更改属性，插入等操作。

**复制节点 node.cloneNode(bool)** 注：参数true为复制属性

**添加子节点 父节点.appendChild(子节点);** 注：插入到最后。

**父节点.insertBefore(新节点,此节点之前);** 插入到节点之前

**移除子节点 父节点.removeChild(节点);**

**更改元素内容 元素.innerHTML=”内容”;** 注：元素的或取方法在上面。

**元素属性 获取 元素名.getAttribute(“属性名”);**

**修改 元素名.setAttribute(“属性名”,”属性值”);**

**获取或修改 元素.属性名=”属性值”;**

**更改元素CSS属性 元素.style.CSS属性名=”属性值”;** 注：使用此方式获取css属性值时，只能获取内嵌式定义的css属性

**在html标签中添加事件 <事件名=”处理函数(参数)”>** 注：若参数为this，this为所在元素对象。

**添加DOM0级事件处理 元素.事件名=匿名函数;**

**移除DOM0级事件处理 元素.事件名=null;**

**添加DOM2级事件句柄 元素.addEventListener(“事件名”,函数名[,bool]);** 注：事件名不带on。 注：第二个参数函数名不加括号；也可以在此处声明匿名函数。 注：可给一个元素的相同事件添加多个句柄。 注:第三个参数，true为事件捕获,false为事件冒泡，现一般不予操作，默认为事件冒泡。 注：添加的函数有一默认参数为事件对象。

**移除DOM2级事件句柄 元素.removeEventListener(“事件名”,函数名);**

**IEDOM2事件处理 元素.attachEvent/detacheEvent** 注：其使用方法与add/removeEventListener相同，但其事件名带on。

**多浏览器适配 元素.addEventListener/attachEvent等可返回bool值测试浏览器是否支持**

**event对象** 注：event通常作为事件处理事件的参数。 注：event=event||window.event;

**event.type;**获取事件类型 注：如，click。

**event.target**获得引发事件的元素

**event.dataTransfer**用于交换数据

**event.stopPropagation()**阻止事件冒泡

**event.preventDefault()**阻止事件的默认动作 注：如submit，拖拽等

**frame对象 contentWindow**得到frame的window对象

**contentDocument**得到frame的document对象

**window其他** 注：该对象对应一个打开的窗口，但窗口重新加载时，除name属性外其他的属性和方法会被清空

**innerHeight;/innerWidth;** 窗口尺寸注：浏览器工作区尺寸。

**screenTop;/screenLeft;/screenX;/screenY;**窗口距屏幕距离（不同浏览器支持不同）

**open(“url”[,”name”,”其他窗口属性”]);** 打开窗口 注：name除可指定窗口名外，还可指定\_blank\\_self等指定打开窗口，但其值不会被设置到name属性

**close();**关闭当前窗口

**name**窗口名称 注：窗口重新加载时该属性不会被清空

**History back();**后退 **forward();**前进  **go(数字);**跳转，数字为负后退，正前进。

**Location hostname;**主机域名 **pathname;**路径 **port;**端口号 **protocol;**web协议

**href;**当前页面的url

**assign();** 加载新的文档

**Screen availHeight;/availWidth;**可用尺寸 注：此尺寸非浏览器窗口尺寸，而是屏幕整体的可用尺寸。

**height;/width;** 屏幕分辨率的宽高

**Navigator userAgent**返回运行浏览器的user-agent头部的值 注：可使用此值判断使用何种浏览器，判断在pc端还是移动端

**计时器 延时调用函数 setTimeout(函数,毫秒数);** 注：可利用回调方法计时。

**停止 clearTiemout(值);** 注：参数为setTimeout(...)返回的值。

**不停调用函数 setInterval(函数,毫秒数);**

**停止 clearInterval(值)** 注：参数为setInterval(...)返回的值。

**cookie 设置cookie document.cookie=”key=”+value;** 注：cookie的格式为key1=value1;key2=value2。

**获取cookie document.cookie**

**localStorge setItem(key,value)**设置键值对

**getItem(key)**获取键值对

**removeItem(key)**移除键值对

**clear()**删除所有键值对

**key(i)**得到第i个键名称

**length**得到键值对的数量

**全局函数 encodeURIComponent()**把字符串编码为URI组件

**decodeURIComponent()**解码一个编码的URI组件

**正则表达式** 汉字：[\u4e00-\u9fa5]

**test(str)** 判断字符串是否符合正则表达式

**AJAX**

**创建XHR对象 var 变量名=new XMLHttpRequest()**

**var 变量名=new ActiveXObject(“Microsoft.XMLHTTP”)** IE6浏览器

**windows.XMLHttpRequest** 检测是否支持XMLHttpRequest

**XHR请求 XHR对象.open(“请求方式”,”url”,同异步)** 请求方式为GET或POST；true为异步

**XHR对象.send([“字符串”])** 注：字符串只用于POST方式，为发送的字符串

**XHR对象.setRequestHeader(“key”,”value”)** POST方式时，添加http头

**XHR响应** **XHR对象.responseText** 获取字符串响应数据

**XHR对象.responseXML** 获取XML形式的响应数据

**onreadystatechange事件** 注：当readyState改变时，该事件触发。 注：只在异步时使用。

**XHR对象.onreadystatechange=function** 注册事件触发函数

**XHR对象.readyState** XHR状态：0请求未初始化；1服务器连接已建立；2:请求已接收；3请求处理中；4:请求已完成，且响应已就绪

**XHR对象.status** 200: "OK";404: 未找到页面