js调class要写成className。 操作属性：.或[‘property’], .都可以用[‘...’]代替

Js调用style是行间style 样式优先级:\*<标签<class<ID<行间，这几个最好别混着调用

匿名函数 函数定义直接使用，相当于闭包，闭包中的this表明当前元素

Window.onload = function (){...} window加载完后执行里面的内容

获取数组oDiv.getElementsByTagName(‘div’) .length获取数量

自定义属性，通过js直接加，不能通过html加

可向div的innerHTML加html代码

ECMAScript几乎没有兼容问题 DOM有一些操作不兼容 BOM完全不兼容

查看变量类型 typeof a；

parseInt(a)字符串转整数 parseFloat(a)转小数 NaN(Not a Number) isNaN(a)

== - 会调用隐式类型转换

命名规范: 数组a、布尔值b、浮点数f、函数fn、整数i、对象o、正则表达式re、字符串s、变体变量v

假：false、数字零、空字符串、空对象、undefined

json： var json = {a:12, b:5, c:’abc’}; json.a json[‘a’] json没有length

json的遍历：for(var i in json){…}

function sum(){…arguments…}不定参

定时器:timer=setInterval(func, 1000)间隔型 setTimeout(func, 1000)延时型 关闭clearInterval(timer)

var oDate=new Date(); getHours(oDate); getMinutes getSeconds getFullYear getMonth()+1 getDate getDay()星期几，周日为0 …

获取字符串某一位str.charAt(0)

div: {overflow:hidden;}截边 li: {list-style:none;}横排

<a href=”javascript:;”>a</a>

子节点oUl.childNodes，空白也是节点 nodeType==3文本节点 nodeType==1元素节点

oUl.children兼容、通用子节点、只识别元素

父节点parentNode offsetParent获取元素在页面上的实际位置

firstElementChild、nextElementSibling高版本 firstChild、低版本

getAttribute(名称) setAttribute(名称, 值) removeAttribute(名称)

数组末尾加元素A.push(a)

DOM操作：var oLi=document.createElement(‘li’); 移动oUl.appendChild(oLi)

在前面插入元素：oUl.insertBefore(oLi, aLi[0]) 删除:oUl.removeChild(this.parentNode)

碎片集:var oFrag=document.createDocumentFragment(); oFrag.appendChild(oLi) oUl.appendChild(oFrag) 高级浏览器上用处不大

表格：<thead><td>…</td></thead> <tbody><tr><td>…</td></tr><tbody>

oTab.tBodies[0].rows[1].cells[1] 转小写toLowerCase()

str.search(‘c’);返回字符在字符串中的位置，否则返回-1 str.split(‘ ’);

运动oDiv.style.left=oDiv.offsetLeft+10+’px’;

oDiv.style.filter=’alpha(opacity:’+alpha+’)’; (兼容性firefox和chrome) oDiv.style.opacity= alpha/100;

Math.ceil(0.1)向上取整 Math.floor(4.9)向下取整 Math.round(3,5)四舍五入

网页滚动时偏移的距离var scrollTop=document.documentElement.scrollTop||document.body.scrollTop; 浮动oDiv.style.top=document.documentElement.clientHeight-oDiv.offsetHeight+scrollTop+’px’;

有border的时候offsetWidth会是width加border和padding，这时需要用行间样式

获取行间样式函数function getStyle(obj, name){ if(obj.currentStyle[name]){ return obj.currentStyle[name]; }else{ return getComputedStyle(obj, false)[name]; } }

var oBtnPrev = getByClass(oDiv, ‘prev’)[0];

style.zIndex层级

点击的坐标:event.clientX event.clientY ev.clientX ev.clientY

var oEvent=ev||event;

取消冒泡eEvent.cancelBubble=true

var scrollTop=document.documentElement.scrollTop||document.body.scrollTop

键盘按键:document.onkeydown eEvent.keyCode

同时按下ctrl和回车(eEvent.ctrlKey && eEvent.keyCode==13)

Document.oncontextmenu=function(){ return false//阻止右键菜单 } onkeydown中return false就是阻止输入

火狐拖拽bug：return false

oBtn.attachEvent(‘onclick’ , function…)可加多个函数到同一个事件上(IE)

火狐和chrome： oBtn.addEventListener(‘click’, function…, false)

oDiv.setCapture事件捕获(解决文字选中、鼠标脱离div问题) oDiv.releaseCapture取消捕获IE7

var oAjax=new XMLHttpRequest();//非IE6

var oAjax=new ActiveXObject(“Microsoft.XMLHTTP”);//IE6

if(window.XMLHttpRequest){ var oAjax=new XMLHttpRequest(); }else{ var oAjax=new ActiveXObject(“Microsoft.XMLHttp”) }//创建ajax对象

oAjax.open(‘GET’, ‘a.txt’, true)//连接服务器,第三个参数为true表示异步

oAjax.send();//发送请求

aAjax.onreadystatechange=function(){ if (aAjax.readyState==4//4表示交互完成,0表示还未调用open()方法;1表示已经调用send方法,正在发送请求;2表示send方法完成,已收到全部响应内容;3正在解析响应内容4响应内容解析完成){ if(oAjax.status==200){ oAjax.responseTxt//成功 } } }

new function(); function中可用this 会自动return this

原型(相当于扩展方法):Array.prototype.sum=function(){…}

继承function B(){ A.call(this) } B.prototype.fn=function(){ alert(‘abc’) }; for (var i in A.prototype){ B.prototype[i]=A.prototype[i]; }

window.open(‘about:blank’, ‘\_self’);打开一个空白页, 最后一个参数\_self:覆盖本窗口；\_blank:打开一个新窗口，返回打开的窗口

window.close()关闭窗口，火狐下只能关闭脚本打开的窗口

window.location网页地址

document.documentElement.clientWidth可视区宽

position:fixed固定定位

window.onresize=function (){…}//窗口改变时重新算大小，可处理抖动

提示框confirm,有取消键 prompt(‘title’,’text’)带文字输入

cookie: var oDate=new Date(); oDate.setDate(oDate().getDate()+14)

document.cookie=‘user=blue;expires=’+oDate

删除cookie: setCookie(name, 1, -1)

str.search(‘x’) str.substring(2,5)//不包括结束位置 str.charAt(3) str.match(…) str.replace(re, ‘0’) re.test(str)//返回bool

正则表达式：

var re = RegExp(‘a’,’i’) 或 var re = /a/i

修饰符:i忽略大小写, g找出所有的

/w: 英文，数字，下划线 /s空白字符

**jQuery:**

$(document).ready(function(){ $("p").click(function(){ $(this).hide(); }); });

css选择器: $("p").css("background-color","red");

$(selector).toggle(speed,callback); toggle/show/hide speed可以是fast,slow,或具体数值

$(selector).fadeIn(speed,callback); fadeout/fadeIn/ fadeToggle $(selector).fadeTo(speed,opacity,callback);

上下滑动$(selector).slideToggle(speed,callback); slideToggle/ slideDown/ slideUp

动画: $("div").animate({ left:'250px', opacity:'0.5', height:'+=150px', width: 'toggle', fontSize:'3em' },speed,callback);

停止动画: $(selector).stop(stopAll,goToEnd); stopAll是否清除动画队列，默认false；goToEnd是否立即完成当前动画。

动画链: $("#p1").css("color","red").slideUp(2000).slideDown(2000);

获取文本: $("#test").text() html()含html标记 获取input输入框的值 $("input").val()输入参数则为写入。

获取属性值: alert($("#w3s").attr("href"));

写入文本回调: $("#test1").text(function(i,origText){ return …} 其中i：被选元素列表中当前元素的下标。属性同样有写入和回调。

在元素结尾加内容:$("p").append("Some appended text."); 开头prepend 也可用于添加元素. 除此还有after、before //区别?

删除被选元素remove()，从被选元素中删除子元素empty() $("p").remove(".italic");删除 class="italic" 的所有 <p> 元素

操作css: $("h1,h2,p").addClass("important blue"); removeClass toggleClass

获取css属性值: $("p").css("background-color"); 设置css属性值: $("p").css("background-color","yellow"); 多个css属性: css({"propertyname":"value","propertyname":"value",...});

获取尺寸: 不包括内边距、边框或外边框:width(),height()输入参数则为设置尺寸。 包括内边距:innerWidth(),innerHeight() 包括内边距和边框:outerWidth(),outerHeight() 包括内边距和边框以及外边框: outerWidth(true),outerHeight(true)

层级: 直接父元素:$("span").parent(); 所有父元素:$("span").parents(); 特定父元素:$("span").parents("ul"); 到某个父级元素之间的所有父元素:parentsUntil("div"); 子元素: 直接子元素$("div").children(); 特定类的子元素:$("div").children("p.1"); 所有后代(子、孙…)中的span元素:$("div").find("span"); 所有后代: $("div").find("\*");

同胞元素: $("h2").siblings(); siblings("p"); 下一个同胞元素: next(); 所有跟随的同胞元素: nextAll(); 到某个同胞元素之间的所有同胞元素:nextUntil("h6"); prev,prevAll,prevUntil同next，方向相反。

按索引取元素:$("div p").first() last() eq(0)

取class为filter的元素:$("p").filter(".intro");

加载文本$("#div1").load("demo\_test.txt"); 加载文本中某个标签的内容load("demo\_test.txt #p1"); 加载后的回调:$("#div1").load("demo\_test.txt",function(responseTxt,statusTxt,xhr){ … }

用jQuery代替$: $.noConflict(); jQuery(document)

加载xml: if (window.ActiveXObject) {xmlDoc = new ActiveXObject("Microsoft.XMLDOM") xmlDoc.async=false; xmlDoc.load("note.xml") displaymessage() }else if (document.implementation && document.implementation.createDocument) { xmlDoc = document.implementation.createDocument(" ","",null) xmlDoc.load("note.xml"); xmlDoc.onload=displaymessage } function displaymessage(){ document.write(xmlDoc.getElementsByTagName("body")[0].firstChild.nodeValue) }

使用E4X(还没有主流浏览器支持): var xmlDoc = new XML() xmlDoc.load("note.xml") document.write(xmlDoc.body)