在 SegmentFault，学习技能、解决问题

每个月，我们帮助 1000 万的开发者解决各种各样的技术问题。并助力他们在技术能力、职业生涯、影响力上获得提升。

IE条件注释

条件注释简介

1. IE中的条件注释（Conditional comments）对IE的版本和IE非IE有优秀的区分能力，是WEB设计中常用的hack方法。

条件注释只能用于IE5以上，IE10以上不支持。

2. 如果你安装了多个IE，条件注释将会以最高版本的IE为标准。
3. 条件注释的基本结构和HTML的注释(<!- ->)是一样的。因此IE以外的浏览器将会把它们看作是普通的注释而完全忽略它们。
4. IE将会根据if条件来判断是否如解析普通的页面内容一样解析条件注释里的内容。

条件注释使用方法示例

```
<!--[if IE 5]>仅IE5.5可见<![endif]-->
<!--[if gt IE 5.5]>仅IE 5.5以上可见<![endif]-->
<!--[if lt IE 5.5]>仅IE 5.5以下可见<![endif]-->
<!--[if gte IE 5.5]>IE 5.5及以上可见<![endif]-->
<!--[if lte IE 5.5]>IE 5.5及以下可见<![endif]-->
<!--[if !IE 5.5]>非IE 5.5的IE可见<![endif]-->
```

html代码用js动态加载进页面

```
<script type="text/html" id="T-pcList">
//这里面是你要放的html代码，例如放一个div的内容
</script>
把上面的js动态加入到页面中
$(function(){
var s=$("#T-pcList").html();//获得js的html内容
$(".picScroll-left .bd").html(s);//把s的内容放到class为bd内
thisstyle();//执行某个函数
```

```
});
```

js判断用户访问的是PC还是mobile

```
var browser={
versions:function(){
var u = navigator.userAgent, app = navigator.appVersion;
var sUserAgent = navigator.userAgent;
return {
trident: u.indexOf('Trident') > -1,
presto: u.indexOf('Presto') > -1,
isChrome: u.indexOf("chrome") > -1,
isSafari: !u.indexOf("chrome") > -1 && (/webkit|khtml/).test(u),
isSafari3: !u.indexOf("chrome") > -1 && (/webkit|khtml/).test(u) &&
u.indexOf('webkit/5') != -1,
webKit: u.indexOf('AppleWebKit') > -1,
gecko: u.indexOf('Gecko') > -1 && u.indexOf('KHTML') == -1,
mobile: !!u.match(/AppleWebKit.*Mobile.*/),
ios: !!u.match(/\(i[^;]+;( U;)? CPU.+Mac OS X/),
android: u.indexOf('Android') > -1 || u.indexOf('Linux') > -1,
iPhone: u.indexOf('iPhone') > -1,
iPad: u.indexOf('iPad') > -1,
iWinPhone: u.indexOf('Windows Phone') > -1
};
}()
}
if(browser.versions.mobile || browser.versions.iWinPhone){
window.location = "http:/www.baidu.com/m/";
}
```

js如何判断用户是否是用微信浏览器

根据关键字 MicroMessenger 来判断是否是微信内置的浏览器。判断函数如下：

```
function isWeiXin(){
var ua = window.navigator.userAgent.toLowerCase();
if(ua.match(/MicroMessenger/i) == 'micromessenger'){
return true;
```

```
}else{
return false;
}
}
```

JS,Jquery获取各种屏幕的宽度和高度

Javascript:

文档可视区域宽： document.documentElement.clientWidth

文档可视区域高： document.documentElement.clientHeight

网页可见区域宽： document.body.clientWidth

网页可见区域高： document.body.clientHeight

网页可见区域宽： document.body.offsetWidth (包括边线的宽)

网页可见区域高： document.body.offsetHeight (包括边线的高)

网页正文全文宽： document.body.scrollWidth

网页正文全文高： document.body.scrollHeight

网页被卷去的高： document.body.scrollTop

网页被卷去的左： document.body.scrollLeft

网页正文部分上： window.screenTop

网页正文部分左： window.screenLeft

屏幕分辨率的高： window.screen.height

屏幕分辨率的宽： window.screen.width

屏幕可用工作区高度： window.screen.availHeight

屏幕可用工作区宽度： window.screen.availWidth

JQuery:

```
$(document).ready(function(){
alert($(window).height()); //浏览器当前窗口可视区域高度
alert($(document).height()); //浏览器当前窗口文档的高度
alert($(document.body).height());//浏览器当前窗口文档body的高度
alert($(document.body).outerHeight(true));//浏览器当前窗口文档body的总高度 包括
border padding margin
alert($(window).width()); //浏览器当前窗口可视区域宽度
alert($(document).width());//浏览器当前窗口文档对象宽度
alert($(document.body).width());//浏览器当前窗口文档body的宽度
```

```
alert($(document.body).outerWidth(true));//浏览器当前窗口文档body的总宽度 包括 border padding margin
})
```

jquery对文本框只读状态与可读状态的相互转化

```
$('input').removeAttr('Readonly');
$('input').attr('Readonly','true');
```

js/jquery实现密码框输入聚焦，失焦问题

js实现方法：

html代码：

```
<input id="i_input" type="text" value='会员卡号/手机号' />
```

js代码：

```
window.onload = function(){
var oIpt = document.getElementById("i_input");
if(oIpt.value == "会员卡号/手机号"){
oIpt.style.color = "#888";
}else{
oIpt.style.color = "#000";
}
oIpt.onfocus = function(){
if(this.value == "会员卡号/手机号"){
this.value="";
this.style.color = "#000";
this.type = "password";
}else{
this.style.color = "#000";
}
};
oIpt.onblur = function(){
if(this.value == ""){
this.value="会员卡号/手机号";
this.style.color = "#888";
```

```
this.type = "text";
}
};
}
```

jquery实现方法:

html代码：

```
<input type="text" class="oldpsw" id="showPwd" value="请输入您的注册密码"/>
<input type="password" name="psw" class="oldpsw" id="password" value=""
style="display:none;"/>
```

jquery代码：

```
$("#showPwd").focus(function() {
var text_value=$(this).val();
if (text_value ==’请输入您的注册密码’) {
$("#showPwd").hide();
$("#password").show().focus();
}
});
$("#password").blur(function() {
var text_value = $(this).val();
if (text_value == "") {
$("#showPwd").show();
$("#password").hide();
}
});
```

获取当前日期

```
var calculateDate = function(){

var date = new Date();

var weeks = ["日","一","二","三","四","五","六"];

return date.getFullYear()+"年"+(date.getMonth()+1)+"月"+

date.getDate()+"日 星期"+weeks[date.getDay()];
```

```
}

$(function(){

$("#dateSpan").html(calculateDate());

})
```

时间倒计时（固定倒计时的结束时间）

```
function countdown() {

var endtime = new Date("Jan 18, 2015 23:50:00");

var nowtime = new Date();

if (nowtime >= endtime) {

document.getElementById("_lefttime").innerHTML = "倒计时间结束";

return;

}

var leftsecond = parseInt((endtime.getTime() - nowtime.getTime()) / 1000);

if (leftsecond < 0) {

leftsecond = 0;

}

__d = parseInt(leftsecond / 3600 / 24);

__h = parseInt((leftsecond / 3600) % 24);
```

```
__m = parseInt((leftsecond / 60) % 60);

__s = parseInt(leftsecond % 60);

document.getElementById("_lefttime").innerHTML = __d + "天" + __h + "小时" + __m + "分" + __s + "秒";

}

countdown();

setInterval(countdown, 1000);
```

10秒倒计时跳转

html代码：

```
<div id="showtimes"></div>
```

js代码：

```
//设定倒数秒数
var t = 10;

//显示倒数秒数
function showTime(){

t -= 1;

document.getElementById('showtimes').innerHTML= t;

if(t==0){

location.href='error404.asp';

}
```

```
//每秒执行一次,showTime()

setTimeout("showTime()",1000);

}

//执行showTime()
showTime();
```

判断浏览器的简单有效方法

```
function getInternet(){
if(navigator.userAgent.indexOf("MSIE")>0) {
return "MSIE"; //IE浏览器
}

if(isFirefox=navigator.userAgent.indexOf("Firefox")>0){
return "Firefox"; //Firefox浏览器
}

if(isSafari=navigator.userAgent.indexOf("Safari")>0) {
return "Safari"; //Safan浏览器
}

if(isCamino=navigator.userAgent.indexOf("Camino")>0){
return "Camino"; //Camino浏览器
}
if(isMozilla=navigator.userAgent.indexOf("Gecko/")>0){
return "Gecko"; //Gecko浏览器
}
}
```

每隔0.1s改变图片的路径

```
<div id="tt"><img src="images/1.jpg" alt=""/></div>
```

js代码：

```
(function(){
function chagesimagesrc(t){
document.getElementById("tt").childNodes[0].src="images/"+t+".jpg";
}

setInterval(function(){

for(var i=0;i<2;i++){

setTimeout((function(t){

return function(){

chagesimagesrc(t);

}

})(i+1),i*100)

}

},1000);

})()
```

点击某个div区域之外，隐藏该div

一般写法：

```
$(document).on("click",function(e){
var target = $(e.target);
if(target.closest(".city_box,#city_select a.selected").length == 0){
$(".city_box").hide();
}
})
```

更全的方式：

```
$(document).click(function(e){
var _con = $(' 目标区域 '); // 设置目标区域
if(!_con.is(e.target) && _con.has(e.target).length === 0){ // Mark 1
some code... // 功能代码
}
});
/* Mark 1 的原理：
判断点击事件发生在区域外的条件是：
1. 点击事件的对象不是目标区域本身
2. 事件对象同时也不是目标区域的子元素
*/
```

js获取某年某月的哪些天是周六和周日

```
<p id="text"></p>
<script type="text/javascript">
function time(y,m){
var tempTime = new Date(y,m,0);
var time = new Date();
var saturday = new Array();
var sunday = new Array();
for(var i=1;i<=tempTime.getDate();i++){
time.setFullYear(y,m-1,i);
var day = time.getDay();
if(day == 6){
saturday.push(i);
}else if(day == 0){
sunday.push(i);
}
}
var text = y+"年"+m+"月份"+"<br />"
+"周六："+saturday.toString()+"<br />"
+"周日："+sunday.toString();
document.getElementById("text").innerHTML = text;
}
time(2014,7);
```

```
</script>
```

如何在手机上禁止浏览器的网页滚动

方法一：

```
<body ontouchmove="event.preventDefault()" >
```

方法二：

```
<script type="text/javascript">

document.addEventListener('touchmove', function(event) {

event.preventDefault();

})

</script>
```

改变type=file默认样式，"浏览"等字体

```
<input type="file" id="browsefile" style="visibility:hidden"
onchange="filepath.value=this.value">

<input type="button" id="filebutton" value="" onclick="browsefile.click()">

<input type="textfield" id="filepath">
```

js判断变量是否未定义的代码

一般如果变量通过var声明，但是并未初始化的时候，变量的值为undefined，而未定义的变量则需要通过 "typeof 变量"的形式来判断，否则会发生错误。

实际应用：

variable有的页面我们不定义,但有的页面定义了，就可以需要这样的判断方法，没有定义的就不执行。

```
if("undefined" != typeof variable){
if(variable=="abc"){
console.log('成功');
```

```
}
}
```

针对IE6，7的hack，该怎么写

你可能会这么回答：使用 “>”，“_”，“*”等各种各样的符号来写hack。是的，这样做没错，但是需要记住每个符号分别被哪些浏览器识别，并且如果写的太乱将造成代码 阅读起来十分困难。学习CSS必须抱有一种质疑精神，有没有一种hack方法可以不写这些乱七八糟的符号，并且代码易维护易读呢？我们可以看看好搜首页是怎么做的：在页面顶端有这样一句话：

```
<!DOCTYPE html>
<html>
<meta charset="utf-8"/>
<head>
<!--[if lt IE 7 ]><html class="ie6"><![endif]-->
<!--[if IE 7 ]><html class="ie7"><![endif]-->
<!--[if IE 8 ]><html class="ie8"><![endif]-->
<!--[if IE 9 ]><html class="ie9"><![endif]-->
<!--[if (gt IE 9)|!(IE)]><!--><html class="w3c"><!--<![endif]-->
</head>
```

在页面的CSS中，会看到这样的规则：

```
.ie7 #hd_usernav:before, .ie8 #hd_usernav:before {
display: none
}
.ie6 .skin_no #hd_nav li, .ie7 .skin_no #hd_nav li, .ie8 .skin_no #hd_nav li {
border-right-color: #c5c5c5
}
.ie6 .skin_no #hd_nav a, .ie7 .skin_no #hd_nav a, .ie8 .skin_no #hd_nav a {
color: #c5c5c5
}
```

行内级元素可以设置宽高吗？有哪些？

在面试时，当被问到行内级元素可否设置宽高时，根据我们的经验往往会回答不能。但是这样往往着了面试官的道，因为有一些特殊的行内元素，比如img，input，select等等，是可以被设置宽高的。一个内容不受CSS视觉格式化模型控制，CSS渲染模型并不考虑对此内容的渲染，且元素本身一般拥有固有尺寸（宽度，高度，宽高比）的元素，被称之为置换元素。

比如img是一个置换元素，当不对它设置宽高时，它会按照本身的宽高进行显示。所以这个问题的正确答案应该是置换元素可以，非置换元素不可以。

js动态创建css样式添加到head内

```
function addCSS(cssText){
var style = document.createElement('style');
var head = document.head || document.getElementsByTagName('head')[0];
style.type = 'text/css';
if(style.styleSheet){ //IE
var func = function(){
try{
//防止IE中stylesheet数量超过限制而发生错误
style.styleSheet.cssText = cssText;
}catch(e){

}
}
//如果当前styleSheet还不能用，则放到异步中则行
if(style.styleSheet.disabled){
setTimeout(func,10);
}else{
func();
}
}else{ //w3c
//w3c浏览器中只要创建文本节点插入到style元素中就行了
var textNode = document.createTextNode(cssText);
style.appendChild(textNode);
}
//把创建的style元素插入到head中
head.appendChild(style);
}

//使用
addCSS('#demo{ height: 30px; background:#f00;}');
```

在IE8以及其低版本浏览器下，IE独有属性styleSheet.cssText。所以一般的兼容简单写法：

```
var style = document.createElement('style');
style.type = "text/css";
if (style.styleSheet) { //IE
style.styleSheet.cssText = '/*..css content here..*/';
} else { //w3c
style.innerHTML = '/*..css content here..*/';
}
document.getElementsByTagName('head')[0].appendChild(style);
```

form表单提交时设置编码格式

```
<form name="form" method="post" action="XXXX" accept-charset="utf-8"
onsubmit="document.charset='utf-8';">
//内容
</form>
```

js 加入收藏代码

```
function addFavorite(title, url) {
url = encodeURI(url);
try {
window.external.addFavorite(url, title);
}
catch (e) {
try {
window.sidebar.addPanel(title, url, "");
}
catch (e) {
alert("加入收藏失败,Ctrl+D进行添加");
}
}
}
addFavorite(document.title,window.location);
```

打印方法：（整个页面 window.print()）

```
function Printpart(id_str)//id-str 内容中的id{
var el = document.getElementById(id_str);
var iframe = document.createElement('IFRAME');
var doc = null;
iframe.setAttribute('style',
'position:absolute;width:0px;height:0px;left:-500px;top:-500px;');
document.body.appendChild(iframe);
doc = iframe.contentWindow.document;
doc.write('<div>' + el.innerHTML + '</div>');
doc.close();
iframe.contentWindow.focus();
iframe.contentWindow.print();
if (navigator.userAgent.indexOf("MSIE") > 0)
{
document.body.removeChild(iframe);
}
}
```

参考地址：

http://www.cnblogs.com/yeming...

http://www.cnblogs.com/jikey/...

js强制手机页面横屏显示

```
$( window ).on( "orientationchange", function( event ) {
if (event.orientation=='portrait') {
$('body').css('transform', 'rotate(90deg)');
} else {
$('body').css('transform', 'rotate(0deg)');
}
});
$( window ).orientationchange();
```

jquery获得select中option的索引

html代码：

```
<select class="select-green">
<option value="0">高级客户经理</option>
```

```
<option value="1">中级客户经理</option>
</select>
```

jquery代码：

```
$(".select-green").change(function(){
var _indx = $(this).get(0).selectedIndex;
$(".selectall .selectCon").hide();
$(".selectall .selectCon").eq(_indx).fadeIn();
});
```

注：其中(this).get(0)与(this)[0]等价

获取上传文件的大小

html代码：

```
<input type="file" id="filePath" onchange="getFileSize(this)"/>
```

js代码：

```
//兼容IE9低版本获取文件的大小
function getFileSize(obj){
var filesize;
if(obj.files){
filesize = obj.files[0].size;
}else{
try{
var path,fso;
path = document.getElementById('filePath').value;
fso = new ActiveXObject("Scripting.FileSystemObject");
filesize = fso.GetFile(path).size;
}
catch(e){
//在IE9及低版本浏览器，如果不容许ActiveX控件与页面交互，点击了否，就无法获取size
console.log(e.message); //Automation 服务器不能创建对象
filesize = 'error'; //无法获取
}
}
return filesize;
}
```

限制上传文件的类型

如果是高版本浏览器，一般在HTML代码中写就能实现，如：

```
<input type="file" name="filePath" accept=".jpg,.jpeg,.doc,.docxs,.pdf">
```

如果限制上传文件为图片类型，如下：

```
<input type="file" class="file" value="上传" accept="image/*"/>
```

但是在其它低版本浏览器就不管用了，需要js来判断。

html代码：

```
<input type="file" id="filePath" onchange="limitTypes()"/>
```

js代码：

```
/* 通过扩展名，检验文件格式。
*@parma filePath{string} 文件路径
*@parma acceptFormat{Array} 允许的文件类型
*@result 返回值{Boolen}：true or false
*/
function checkFormat(filePath,acceptFormat){
var resultBool= false,
ex = filePath.substring(filePath.lastIndexOf('.') + 1);
ex = ex.toLowerCase();
for(var i = 0; i < acceptFormat.length; i++){
    if(acceptFormat[i] == ex){
resultBool = true;
break;
    }
}
return resultBool;
};
function limitTypes(){
var obj = document.getElementById('filePath');
var path = obj.value;
var result = checkFormat(path,['bmp','jpg','jpeg','png']);
if(!result){
alert('上传类型错误，请重新上传');
obj.value = '';
}
```

```
}
```

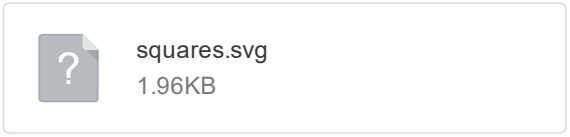
随机产生lowwer - upper之间的随机数

```
function selectFrom(lower, upper) {
var sum = upper - lower + 1; //总数-第一个数+1
return Math.floor(Math.random() * sum + lower);
};
```

保留后端传递到前端页面的空格

```
var objt = {
name:’ aaaa 这是一个空格多的标签 这是一个空格多的标签’
}
objt.name = objt.name.replace(/\s/g,’ ’);
console.log(objt.name);
```

用firebug查看结果：



为什么Image对象的src属性要写在onload事件后面？

```
var image=new Image();
imgae.onload = funtion;
imgae.src = ’url’
```

js内部是按顺序逐行执行的，可以认为是同步的

给imgae赋值src时，去加载图片这个过程是异步的，这个异步过程完成后，如果有onload，则执行onload

如果先赋值src，那么这个异步过程可能在你赋值onload之前就完成了（比如图片缓存，或者是js由于某些原因被阻塞了），那么onload就不会执行

反之，js同步执行确定onload赋值完成后才会赋值src,可以保证这个异步过程在onload赋值完成后才开始进行，也就保证了onload一定会被执行到

跨浏览器添加事件

```
//跨浏览器添加事件
function addEvent(obj,type,fn){
if(obj.addEventListener){
```

```
obj.addEventListener(type,fn,false);
}else if(obj.attachEvent){//IE
obj.attchEvent('on'+type,fn);
}
}
```

跨浏览器移除事件

```
//跨浏览器移除事件
function removeEvent(obj,type,fn){
if(obj.removeEventListener){
obj.removeEventListener(type,fn,false);
}else if(obj.detachEvent){//兼容IE
obj.detachEvent('on'+type,fn);
}
}
```

跨浏览器阻止默认行为

```
//跨浏览器阻止默认行为
function preDef(ev){
var e = ev || window.event;
if(e.preventDefault){
e.preventDefault();
}else{
e.returnValue =false;
}
}
```

跨浏览器获取目标对象

```
//跨浏览器获取目标对象
function getTarget(ev){
if(ev.target){//w3c
return ev.target;
}else if(window.event.srcElement){//IE
return window.event.srcElement;
}
}
```

跨浏览器获取滚动条位置

```
//跨浏览器获取滚动条位置，sp == scroll position
function getSP(){
return{
top: document.documentElement.scrollTop || document.body.scrollTop,
left : document.documentElement.scrollLeft || document.body.scrollLeft;
}
}
```

跨浏览器获取可视窗口大小

```
//跨浏览器获取可视窗口大小
function getWindow () {
if(typeof window.innerWidth !='undefined') {
return{
width : window.innerWidth,
height : window.innerHeight
}

} else{
return {
width : document.documentElement.clientWidth,
height : document.documentElement.clientHeight
}
}
}
```

js 对象冒充

```
<script type = 'text/javascript'>
function Person(name , age){
this.name = name ;
this.age = age ;
this.say = function (){
return "name : "+ this.name + " age: "+this.age ;
} ;
}
```

```
var o = new Object() ;//可以简化为Object()
Person.call(o , "zhangsan" , 20) ;
console.log(o.say() );//name : zhangsan age: 20
</script>
```

js 异步加载和同步加载

异步加载也叫非阻塞模式加载，浏览器在下载js的同时，同时还会执行后续的页面处理。

在script标签内，用js创建一个script元素并插入到document中，这种就是异步加载js文件了：

```
(function() {
var s = document.createElement('script');
s.type = 'text/javascript';
s.async = true;
s.src = 'http://yourdomain.com/script.js';
var x = document.getElementsByTagName('script')[0];
x.parentNode.insertBefore(s, x);
})();
```

同步加载

平常默认用的都是同步加载。如：

```
<script src="http://yourdomain.com/script.js"></script>
```

同步模式又称阻塞模式，会阻止流览器的后续处理。停止了后续的文件的解析，执行，如图像的渲染。浏览器之所以会采用同步模式，是因为加载的js文件中有对dom的操作，重定向，输出document等默认行为，所以同步才是最安全的。

通常会把要加载的js放到body结束标签之前，使得js可在页面最后加载，尽量减少阻塞页面的渲染。这样可以先让页面显示出来。

同步加载流程是瀑布模型，异步加载流程是并发模型。

js获取屏幕坐标

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7"/>
<meta name="auther" content="fq" />
<title>获取鼠标坐标</title>
```

```
</head>
<body>
<script type="text/javascript">
function mousePosition(ev){
if(ev.pageX || ev.pageY){
return {x:ev.pageX, y:ev.pageY};
}
return {
x:ev.clientX + document.body.scrollLeft - document.body.clientLeft,
y:ev.clientY + document.body.scrollTop - document.body.clientTop
};
}
function mouseMove(ev){
ev = ev || window.event;
var mousePos = mousePosition(ev);
document.getElementById('xxx').value = mousePos.x;
document.getElementById('yyy').value = mousePos.y;
}
document.onmousemove = mouseMove;
</script>
X:<input id="xxx" type="text" /> Y:<input id="yyy" type="text" />
</body>
</html>
```

注释：

1.documentElement 属性可返回文档的根节点。

2.scrollTop() 为滚动条向下移动的距离

3.document.documentElement.scrollTop 指的是滚动条的垂直坐标

4.document.documentElement.clientHeight 指的是浏览器可见区域高度

DTD已声明的情况下：

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

如果在页面中添加这行标记的话

IE

document.body.clientWidth ==> BODY对象宽度

document.body.clientHeight ==> BODY对象高度

document.documentElement.clientWidth ==> 可见区域宽度

document.documentElement.clientHeight ==> 可见区域高度

Firefox

document.documentElement.scrollHeight ==> 浏览器所有内容高度

document.body.scrollHeight ==> 浏览器所有内容高度

document.documentElement.scrollTop ==> 浏览器滚动部分高度

document.body.scrollTop ==>始终为0

document.documentElement.clientHeight ==>浏览器可视部分高度

document.body.clientHeight ==> 浏览器所有内容高度

Chrome

document.documentElement.scrollHeight ==> 浏览器所有内容高度

document.body.scrollHeight ==> 浏览器所有内容高度

document.documentElement.scrollTop==> 始终为0

document.body.scrollTop==>浏览器滚动部分高度

document.documentElement.clientHeight ==> 浏览器可视部分高度

document.body.clientHeight ==> 浏览器所有内容高度

浏览器所有内容高度即浏览器整个框架的高度，包括滚动条卷去部分+可视部分+底部隐藏部分的高度总和

浏览器滚动部分高度即滚动条卷去部分高度即可视顶端距离整个对象顶端的高度。

综上

1、document.documentElement.scrollTop和document.body.scrollTop始终有一个为0，所以可以用这两个的和来求scrollTop

2、scrollHeight、clientHeight 在DTD已声明的情况下用documentElement，未声明的情况下用body

clientHeight在IE和FF下，该属性没什么差别，都是指浏览器的可视区域，即除去浏览器的那些工具栏状态栏剩下的页面展示空间的高度。

PageX和clientX

PageX:鼠标在页面上的位置,从页面左上角开始,即是以页面为参考点,不随滑动条移动而变化

clientX:鼠标在页面上可视区域的位置,从浏览器可视区域左上角开始,即是以浏览器滑动条此刻的滑动到的位置为参考点,随滑动条移动 而变化.

可是悲剧的是,PageX只有FF特有,IE则没有这个，所以在IE下使用这个：

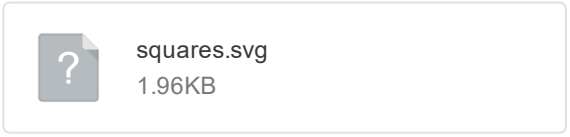
PageY=clientY+scrollTop-clientTop;(只讨论Y轴,X轴同理,下同)

scrollTop代表的是被浏览器滑动条滚过的长度

offsetX:IE特有,鼠标相比较于触发事件的元素的位置,以元素盒子模型的内容区域的左上角为参考点,如果有boder,可能出现负值

只有clientX和screenX 皆大欢喜是W3C标准.其他的,都纠结了.

最给力的是，chrome和safari一条龙通杀!完全支持所有属性



js拖拽效果

```
<!doctype html>
<html lang="zn-CN">
<head>
<meta http-equiv="Content-Type" content="text/html;charset=UTF-8" />
<title></title>
<style type="text/css">
#login{
height: 100px;
width: 100px;
border: 1px solid black;
position: relative;
top:200px;
left: 200px;
background: red;
}
</style>
</head>
<body>
<div id="login"></div>
<script type="text/javascript">
var oDiv = document.getElementById("login");
oDiv.onmousedown = function(e){
var e = e || window.event;//window.event兼容IE,当事件发生时有效

var diffX = e.clientX - oDiv.offsetLeft;//获取鼠标点击的位置到所选对象的边框的水平距离
```

```
var diffY = e.clientY - oDiv.offsetTop;

document.onmousemove = function(e){ //需设为document对象才能作用于整个文档
var e = e||window.event;
oDiv.style.left = e.clientX - diffX + 'px';//style.left表示所选对象的边框到浏览器左侧距离
oDiv.style.top = e.clientY -diffY + 'px';
};
document.onmouseup = function(){
document.onmousemove = null;//清除鼠标释放时的对象移动方法
document.onmouseup = null;
}
}
</script>
</body>
</html>
```

offsetTop 返回的是数字，而 style.top 返回的是字符串，除了数字外还带有单位：px。