网址:https://blog.csdn.net/weixin\_38536027/article/details/79375411

**常见浏览器兼容性问题与解决方案**

　　所谓的浏览器兼容性问题，是指因为不同的浏览器对同一段代码有不同的解析，造成页面显示效果不统一的情况。在大多数情况下，我们的需求是，无论用户用什么浏览器来查看我们的网站或者登陆我们的系统，都应该是统一的显示效果。所以浏览器的兼容性问题是前端开发人员经常会碰到和必须要解决的问题。

　　在学习浏览器兼容性之前，我想把前端开发人员划分为两类：

　　第一类是精确按照设计图开发的前端开发人员，可以说是精确到1px的，他们很容易就会发现设计图的不足，并且在很少的情况下会碰到浏览器的兼容性问题，而这些问题往往都死浏览器的bug，并且他们制作的页面后期易维护，代码重用问题少，可以说是比较牢固放心的代码。

　　第二类是基本按照设计图来开发的前端开发人员，很多细枝末节差距很大，不如间距，行高，图片位置等等经常会差几px。某种效果的实现也是反复调试得到，具体为什么出现这种效果还模模糊糊，整体布局十分脆弱。稍有改动就乱七八糟。代码为什么这么写还不知所以然。这类开发人员往往经常为兼容性问题所困。修改好了这个浏览器又乱了另一个浏览器。改来改去也毫无头绪。其实他们碰到的兼容性问题大部分不应该归咎于浏览器，而是他们的技术本身了。

　　文章主要针对的是第一类，严谨型的开发人员，因此这里主要从浏览器解析差异的角度来分析兼容性问题。

**浏览器兼容问题一：不同浏览器的标签默认的外补丁和内补丁不同**  
W.SVP.ZU%9

　　问题症状：随便写几个标签，不加样式控制的情况下，各自的margin 和padding差异较大。

　　碰到频率:100%

　　解决方案：CSS里    \*

　　备注：这个是最常见的也是最易解决的一个浏览器兼容性问题，几乎所有的CSS文件开头都会用通配符\*来设置各个标签的内外补丁是0。

**浏览器兼容问题二：块属性标签float后，又有横行的margin情况下，在**[IE6](http://xiazai.zol.com.cn/detail/11/102978.shtml)**显示margin比设置的大**

　　问题症状:常见症状是IE6中后面的一块被顶到下一行

　　碰到频率：90%（稍微复杂点的页面都会碰到，float布局最常见的浏览器兼容问题）

　　解决方案：在float的标签样式控制中加入 display:inline;将其转化为行内属性

　　备注：我们最常用的就是div+CSS布局了，而div就是一个典型的块属性标签，横向布局的时候我们通常都是用div float实现的，横向的间距设置如果用margin实现，这就是一个必然会碰到的兼容性问题。

**浏览器兼容问题三：设置较小高度标签（一般小于10px），在IE6，**[IE7](http://xiazai.zol.com.cn/detail/14/130213.shtml)**，遨游中高度超出自己设置高度**

　　问题症状：IE6、7和遨游里这个标签的高度不受控制，超出自己设置的高度

　　碰到频率：60%

　　解决方案：给超出高度的标签设置overflow:hidden;或者设置行高line-height 小于你设置的高度。

　　备注：这种情况一般出现在我们设置小圆角背景的标签里。出现这个问题的原因是[IE8](http://xiazai.zol.com.cn/detail/6/51973.shtml)之前的浏览器都会给标签一个最小默认的行高的高度。即使你的标签是空的，这个标签的高度还是会达到默认的行高。

**浏览器兼容问题四：行内属性标签，设置display:block后采用float布局，又有横行的margin的情况，IE6间距bug**

　　问题症状：IE6里的间距比超过设置的间距

　　碰到几率：20%

　　解决方案：在display:block;后面加入display:inline;display:table;

　　备注：行内属性标签，为了设置宽高，我们需要设置display:block;(除了input标签比较特殊)。在用float布局并有横向的margin后，在IE6下，他就具有了块属性float后的横向margin的bug。不过因为它本身就是行内属性标签，所以我们再加上display:inline的话，它的高宽就不可设了。这时候我们还需要在display:inline后面加入display:talbe。

**浏览器兼容问题五：图片默认有间距**

　　问题症状：几个img标签放在一起的时候，有些浏览器会有默认的间距，加了问题一中提到的通配符也不起作用。

　　碰到几率：20%

　　解决方案：使用float属性为img布局

　　备注：因为img标签是行内属性标签，所以只要不超出容器宽度，img标签都会排在一行里，但是部分浏览器的img标签之间会有个间距。去掉这个间距使用float是正道。（我的一个学生使用负margin，虽然能解决，但负margin本身就是容易引起浏览器兼容问题的用法，所以我禁止他们使用）

**浏览器兼容问题六：标签最低高度设置min-height不兼容**

　　问题症状：因为min-height本身就是一个不兼容的CSS属性，所以设置min-height时不能很好的被各个浏览器兼容

　　碰到几率：5%

　　解决方案：如果我们要设置一个标签的最小高度200px，需要进行的设置为：{min-height:200px; height:auto ![ImportAnt](http://xiazai.zol.com.cn/detail/34/330423.shtml); height:200px; overflow:visible;}

　　备注：在B/S系统前端开时，有很多情况下我们又这种需求。当内容小于一个值（如300px）时。容器的高度为300px；当内容高度大于这个值时，容器高度被撑高，而不是出现滚动条。这时候我们就会面临这个兼容性问题。

**浏览器兼容问题七：透明度的兼容CSS设置**

　　做兼容页面的方法是：每写一小段代码（布局中的一行或者一块）我们都要在不同的浏览器中看是否兼容，当然熟练到一定的程度就没这么麻烦了。建议经常会碰到兼容性问题的新手使用。很多兼容性问题都是因为浏览器对标签的默认属性解析不同造成的，只要我们稍加设置都能轻松地解决这些兼容问题。如果我们熟悉标签的默认属性的话，就能很好的理解为什么会出现兼容问题以及怎么去解决这些兼容问题。

1.     /\* CSS hack\*/

我很少使用hacker的，可能是个人习惯吧，我不喜欢写的代码[IE](http://xiazai.zol.com.cn/detail/13/123616.shtml)不兼容，然后用hack来解决。不过hacker还是非常好用的。使用hacker我可以把浏览器分为3类：IE6 ；IE7和遨游；[其他](http://xiazai.zol.com.cn/others%20mobilesoft_soft_index/others%20mobilesoft_page_1.html)（IE8 [Chrome](http://xiazai.zol.com.cn/detail/33/327560.shtml) ff [Safari](http://xiazai.zol.com.cn/detail/37/366660.shtml) [opera](http://xiazai.zol.com.cn/detail/34/339345.shtml)等）

　　◆IE6认识的hacker 是下划线\_ 和星号 \*

　　◆IE7 遨游认识的hacker是星号 \*

　　比如这样一个CSS设置：

1.     height:300px;\*height:200px;\_height:100px;

　　IE6浏览器在读到height:300px的时候会认为高时300px；继[续](http://xiazai.zol.com.cn/detail/6/54750.shtml)往下读，他也认识\*heihgt， 所以当IE6读到\*height:200px的时候会覆盖掉前一条的相冲突设置，认为高度是200px。继续往下读，IE6还认识\_height,所以他又会覆盖掉200px高的设置，把高度设置为100px；

IE7和遨游也是一样的从高度300px的设置往下读。当它们读到\*height200px的时候就停下了，因为它们不认识\_height。所以它们会把高度解析为200px，剩下的浏览器只认识第一个height:300px;所以他们会把高度解析为300px。因为优先级相同且想冲突的属性设置后一个会覆盖掉前一个，所以书写的次序是很重要的。

在网站设计的时候，应该注意css样式兼容不同浏览器问题，特别是对完全使用DIV CSS设计的网，就应该更注意IE6 IE7 FF对CSS样式的兼容，不然，你的网乱可能出去不想出现的效果！  
  
  
所有浏览器 通用   
height: 100px;   
  
IE6 专用   
\_height: 100px;   
  
IE6 专用   
\*height: 100px;   
  
IE7 专用   
\*+height: 100px;   
  
IE7、FF 共用   
height: 100px !important;

一、CSS 兼容   
以下两种方法几乎能解决现今所有兼容.   
  
1, !important (不是很推荐，用下面的一种感觉最安全)   
  
随着IE7对!important的支持, !important 方法现在只针对IE6的兼容.(注意写法.记得该声明位置需要提前.)  
  
代码:   
<style>   
#wrapper {   
width: 100px!important; /\* IE7+FF \*/   
width: 80px; /\* IE6 \*/   
}   
</style>   
  
2, IE6/IE77对FireFox <from 针对firefoxie6 ie7的css样式>   
  
\*+html 与 \*html 是IE特有的标签, firefox 暂不支持.而\*+html又为 IE7特有标签.   
  
代码:   
<style>   
#wrapper { width: 120px; } /\* FireFox \*/   
\*html #wrapper { width: 80px;} /\* ie6 fixed \*/   
\*+html #wrapper { width: 60px;} /\* ie7 fixed, 注意顺序 \*/   
</style>   
  
注意:   
\*+html 对IE7的兼容 必须保证HTML顶部有如下声明：   
  
代码:   
<!DOCTYPE HTML PUBLIC "-//W3C//DTDHTML 4.01 Transitional//EN"　"http://www.w3.org/TR/html4/loose.dtd">  
  
二、万能 float 闭合(非常重要!) 可以用这个解决多个div对齐时的间距不对，   
  
关于 clear float 的原理可参见 [How ToClear Floats Without Structural Markup]   
将以下代码加入Global CSS 中,给需要闭合的div加上 class=”clearfix” 即可,屡试不爽.   
  
代码:   
<style>   
/\* Clear Fix \*/   
.clearfix:after {   
content:".";   
display:block;   
height:0;   
clear:both;   
visibility:hidden;   
}   
.clearfix {   
display:inline-block;   
}   
/\* Hide from IE Mac \\*/   
.clearfix {display:block;}   
/\* End hide from IE Mac \*/   
/\* end of clearfix \*/   
</style>  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
三、其他兼容技巧(相当有用)   
  
1, FF下给 div 设置 padding 后会导致 width 和 height 增加, 但IE不会.(可用!important解决)   
2, 居中问题.   
1).垂直居中.将 line-height 设置为当前 div 相同的高度, 再通过 vetical-align: middle.( 注意内容不要换行.)   
2).水平居中. margin: 0 auto;(当然不是万能)  
3, 若需给 a 标签内内容加上 样式, 需要设置 display: block;(常见于导航标签)   
4, FF 和 IE 对 BOX 理解的差异导致相差 2px 的还有设为 float的div在ie下 margin加倍等问题.   
5, ul 标签在 FF 下面默认有 list-style 和 padding . 最好事先声明, 以避免不必要的麻烦. (常见于导航标签和内容列表)   
6, 作为外部 wrapper 的 div 不要定死高度, 最好还加上 overflow: hidden.以达到高度自适应.   
7, 关于手形光标. cursor: pointer. 而hand只适用于 IE.贴上代码:   
  
兼容代码:兼容最推荐的模式。   
/\* FF \*/   
.submitbutton {   
float:left;   
width: 40px;   
height: 57px;   
margin-top: 24px;   
margin-right: 12px;   
}   
/\* IE6 \*/   
\*html .submitbutton {   
margin-top: 21px;   
}   
/\* IE7 \*/   
\*+html .submitbutton {   
margin-top: 21px;   
}   
  
  
  
  
  
什么是浏览器兼容：当我们使用不同的浏览器（Firefox IE7 IE6）访问同一个网站，或者页面的时候，会出现一些不兼容的问题，有的显示出来正常，有的显示出来不正常，我们在编写CSS的时候会很恼火，刚修复了这个浏览器的问题，结果另外一个浏览器却出了新问题。而兼容就是一种办法，能让你在一个CSS里面独立的写支持不同浏览器的样式。这下就和谐了。呵呵！   
  
最近微软发布的IE7浏览器的兼容性确实给一些网页制 作人员添加了一个沉重的负担，虽然IE7已经走向标准化，但还是有许多和FF不同的地方，所以需要用到IE7的兼容，有许多朋友问过IE7的兼容是什么， 其实我也不知道。暂时还没找到IE7专用的兼容。除了前面那片文章，《针对firefox ie6 ie7的css样式》中的兼容方式也是很好用的。   
  
有一点逻辑思想的人都会知道可以用IE和FF的兼容结合起来使用，下面介绍三个兼容，例如：（适合新手，呵呵，高手就在这里路过吧。）   
  
程序代码   
  
第一个兼容，IE FF 所有浏览器 公用（其实也不算是兼容）   
height:100px;   
第二个兼容 IE6专用   
\_height:100px;   
第三个兼容 IE6 IE7公用   
\*height:100px;   
  
介绍完了这三个兼容了，下面我们再来看看如何在一个样式里分别给一个属性定义IE6 IE7 FF专用的兼容，看下面的代码，顺序不能错哦：   
  
程序代码   
  
height:100px;   
\*height:120px;   
\_height:150px;   
  
下面我简单解释一下各浏览器怎样理解这三个属性：   
  
在FF下，第2、3个属性FF不认识，所以它读的是height:100px;   
  
在IE7下，第三个属性IE7不认识，所以它读第1、2个属性，又因为第二个属性覆盖了第一个属性，所以IE7最终读出的是第2个属性\*height:120px;   
  
在IE6下，三个属性IE6都认识，所以三个属性都可以读取，又因为第三个属性覆盖掉前2个属性，所以IE6最终读取的是第三个属性。   
  
  
  
  
  
1 针对firefox ie6 ie7的css样式   
  
现在大部分都是用!important来兼容，对于ie6和firefox测试可以正常显示，但是ie7对!important可以正确解释，会导致页面 没按要求显示！找到一个针对IE7不错的兼容方式就是使用“\*+html”，现在用IE7浏览一下，应该没有问题了现在写一个CSS可以这样：   
  
#1 { color: #333; } /\* Moz \*/   
\* html #1 { color: #666; } /\* IE6 \*/   
\*+html #1 { color: #999; } /\* IE\*/   
  
那么在firefox下字体颜色显示为#333，IE6下字体颜色显示为#666，IE7下字体颜色显示为#999。   
  
2 css布局中的居中问题   
  
主要的样式定义如下：   
  
body {TEXT-ALIGN: center;}   
#center { MARGIN-RIGHT: auto; MARGIN-LEFT: auto; }   
  
说明：   
  
首先在父级元素定义TEXT-ALIGN: center;这个的意思就是在父级元素内的内容居中；对于IE这样设定就已经可以了。   
  
但在mozilla中不能居中。解决办法就是在子元素定义时候设定时再加上“MARGIN-RIGHT: auto;MARGIN-LEFT: auto; ”   
  
需要说明的是，如果你想用这个方法使整个页面要居中，建议不要套在一个DIV里，你可以依次拆出多个div，只要在每个拆出的div里定义MARGIN-RIGHT:auto;MARGIN-LEFT: auto; 就可以了。   
  
3 盒模型不同解释.   
  
#box{   
width:600px;   
//for ie6.0- w\idth:500px;   
//for ff+ie6.0   
}   
#box{   
width:600px!important   
//for ff   
width:600px;   
//for ff+ie6.0   
width /\*\*/:500px;   
//for ie6.0-   
}   
  
4 浮动ie产生的双倍距离   
  
#box{ float:left; width:100px; margin:0 0 0 100px; //这种情况之下IE会产生200px的距离display:inline; //使浮动忽略}   
  
这里细说一下block,inline两个元素,Block元素的特点是:总是在新行上开始,高度,宽度,行高,边距都可以控制(块元素);Inline元素的特点是:和其他元素在同一行上,…不可控制(内嵌元素);   
  
#box{ display:block; //可以为内嵌元素模拟为块元素 display:inline; //实现同一行排列的的效果 diplay:table;   
  
5 IE与宽度和高度的问题   
  
IE不认得min-这个定义，但实际上它把正常的width和height当作有min的情况来使。这样问题就大了，如果只用宽度和高度，正常的浏览器里这两个值就不会变，如果只用min-width和min-height的话，IE下面根本等于没有设置宽度和高度。比如要设置背景图片，这个宽度是比较重 要的。要解决这个问题，可以这样：   
  
#box{ width: 80px; height: 35px;}html>body #box{width: auto; height: auto; min-width: 80px; min-height: 35px;}   
  
6 页面的最小宽度   
  
min-width是个非常方便的CSS命令，它可以指定元素最小也不能小于某个宽度，这样就能保证排版一直正确。但IE不认得这个，而它实际上把 width当做最小宽度来使。为了让这一命令在IE上也能用，可以把一个<div> 放到 <body> 标签下，然后为div指定一个类：   
然后CSS这样设计：   
  
#container{   
min-width: 600px;   
width:e-xpression(document.body.clientWidth < 600? “600px”: “auto” );   
}   
  
第一个min-width是正常的；但第2行的width使用了Javascript，这只有IE才认得，这也会让你的HTML文档不太正规。它实际上通过Javascript的判断来实现最小宽度。   
  
7 清除浮动   
  
.兼容box{   
display:table;   
//将对象作为块元素级的表格显示   
}   
  
或者   
  
.兼容box{   
clear:both;   
}   
  
或者加入:after（伪对象）,设置在对象后发生的内容，通常和content配合使用，IE不支持此伪对象，非Ie 浏览器支持，所以并不影响到IE/WIN浏览器。这种的最麻烦的   
  
……#box:after{   
content: “.”;   
display: block;   
height: 0;   
clear: both;   
visibility: hidden;   
}   
  
8&nbp;DIV浮动IE文本产生3象素的bug   
  
左边对象浮动，右边采用外补丁的左边距来定位，右边对象内的文本会离左边有3px的间距.   
  
#box{   
float:left;   
width:800px;}   
#left{   
float:left;   
width:50%;}   
#right{   
width:50%;   
}   
\*html #left{   
margin-right:-3px;   
//这句是关键   
}   
HTML代码   
<DIV id=box>   
<DIV id=left></DIV>   
<DIV id=right></DIV>   
</DIV>   
  
9 属性选择器(这个不能算是兼容,是隐藏css的一个bug)   
  
p[id]{}div[id]{}   
p[id]{}div[id]{}   
  
这个对于IE6.0和IE6.0以下的版本都隐藏,FF和OPera作用   
  
属性选择器和子选择器还是有区别的,子选择器的范围从形式来说缩小了,属性选择器的范围比较大,如p[id]中,所有p标签中有id的都是同样式的.   
  
10 IE捉迷藏的问题   
  
当div应用复杂的时候每个栏中又有一些链接，DIV等这个时候容易发生捉迷藏的问题。   
有些内容显示不出来，当鼠标选择这个区域是发现内容确实在页面。   
解决办法：对#layout使用line-height属性或者给#layout使用固定高和宽。页面结构尽量简单。   
  
11 高度不适应   
  
高度不适应是当内层对象的高度发生变化时外层高度不能自动进行调节，特别是当内层对象使用   
margin 或paddign 时。例：   
  
<div id=”box”>   
<p>p对象中的内容</p>  
</div>   
  
CSS：   
  
#box {background-color:#eee; }   
#box p {margin-top: 20px;margin-bottom: 20px; text-align:center; }   
  
解决方法：在P对象上下各加2个空的div对象CSS代码：.1{height:0px;overflow:hidden;}或者为DIV加上border属性。   
  
  
  
  
  
屏蔽IE浏览器（也就是IE下不显示）   
\*:lang(zh) select {font:12px !important;} /\*FF,OP可见\*/   
select:empty {font:12px !important;} /\*safari可见\*/   
这里select是选择符，根据情况更换。第二句是MAC上safari浏览器独有的。   
  
仅IE7识别   
\*+html {…}   
当面临需要只针对IE7做样式的时候就可以采用这个兼容。   
  
IE6及IE6以下识别   
\* html {…}   
这个地方要特别注意很多地主都写了是IE6的兼容其实IE5.x同样可以识别这个兼容。其它浏览器不识别。   
html/\*\*/ >body select {……}   
这句与上一句的作用相同。   
  
仅IE6不识别   
select { display /\*IE6不识别\*/:none;}   
这里主要是通过CSS注释分开一个属性与值，流释在冒号前。   
  
仅IE6与IE5不识别   
select/\*\*/ { display /\*IE6,IE5不识别\*/:none;}   
这里与上面一句不同的是在选择符与花括号之间多了一个CSS注释。  
  
仅IE5不识别   
select/\*IE5不识别\*/ { display:none;}   
这一句是在上一句中去掉了属性区的注释。只有IE5不识别   
  
盒模型解决方法   
selct {width:IE5.x宽度; voice-family:""}""; voice-family:inherit; width:正确宽度;}   
盒模型的清除方法不是通过!important来处理的。这点要明确。   
  
清除浮动   
select:after {content:"."; display:block; height:0; clear:both;visibility:hidden;}   
在Firefox中，当子级都为浮动时，那么父级的高度就无法完全的包住整个子级，那么这时用这个清除浮动的兼容来对父级做一次定义，那么就可以解决这个问题。   
  
截字省略号   
select { -o-text-overflow:ellipsis; text-overflow:ellipsis;white-space:nowrapoverflow:hidden; }   
这个是在越出长度后会自行的截掉多出部分的文字，并以省略号结尾，很好的一个技术。只是目前Firefox并不支持。   
  
只有Opera识别   
@media all and (min-width: 0px){ select {……} }   
针对Opera浏览器做单独的设定。   
  
以上都是写CSS中的一些兼容，建议遵循正确的标签嵌套(divul li 嵌套结构关系)，这样可以减少你使用兼容的频率，不要进入理解误区，并不是一个页面就需要很多的兼容来保持多浏览器兼容)，很多情况下也许一个兼容都不用 也可以让浏览器工作得非常好，这些都是用来解决局部的兼容性问题，如果希望把兼容性的内容也分离出来，不妨试一下下面的几种过滤器。这些过滤器有的是写在 CSS中通过过滤器导入特别的样式，也有的是写在HTML中的通过条件来链接或是导入需要的补丁样式。   
  
IE5.x的过滤器，只有IE5.x可见   
@media tty {   
i{content:"";/\*" "\*/}} @import ’ie5win.css’; /\*";}   
}/\* \*/   
  
IE5/MAC的过滤器，一般用不着   
/\*\*//\*/   
@import "ie5mac.css";   
/\*\*/   
  
下面是IE的条件注释，个人觉得用条件注释调用相应 兼容是比较完美的多浏览器兼容的解决办法。把需要兼容的地方单独放到一个文件里面，当浏览器版本符合的时候就可以调用那个被兼容的样式，这样不仅使用起来非常方便，而且对于制作这个CSS本身来讲，可以更严格的观察到是否有必要使用兼容，很多情况下，当我本人写CSS如果把全部代码包括兼容都写到一个 CSS文件的时候的时候会很随意，想怎么兼容就怎么兼容，而你独立出来写的时候，你就会不自觉的考虑是否有必要兼容，是先兼容 CSS？还是先把主CSS里面的东西调整到尽可能的不需要兼容？当你仅用很少的兼容就让很多浏览器很乖很听话的时候，你是不是很有成就感呢？你知道怎么选择了吧～～呵呵   
  
IE的if条件兼容 自己可以灵活使用参看这篇IE条件注释   
Only IE   
所有的IE可识别   
  
只有IE5.0可以识别   
Only IE 5.0+   
IE5.0包换IE5.5都可以识别   
  
仅IE6可识别   
Only IE 7/-   
IE6以及IE6以下的IE5.x都可识别   
Only IE 7/-   
仅IE7可识别   
  
  
  
  
  
Css 当中有许多的东西不不按照某些规律来的话，会让你很心烦，虽然你可以通过很多的兼容，很多的!important来控制它，但是你会发现长此以往你会很不甘心，看看许多优秀的网站，他们的CSS让IE6,Ie7,Firefox,甚至Safari,Opera运行起来完美无缺是不是很羡慕？而他们看似复杂的模版下面使用的兼容少得可怜。其实你要知道IE 和 Firefox 并不不是那么的不和谐，我们找到一定的方法，是完全可以让他们和谐共处的。不要你认为发现了兼容的办法，你就掌握了一切，我们并不是兼容的奴隶。   
  
div ul li 的嵌套顺序   
  
今天只讲一个规则。就是<div><ul><li>的三角关系。我的经验就是<div>在最外面，里面 是<ul>，然后再是<li>，当然<li>里面又可以嵌套<div>什么的，但是并不建议你嵌套很多 东西。当你符合这样的规则的时候，那些倒霉的，不听话的间隙就不会在里面出现了，当你仅仅是<div>里面放<li>，而不 用<ul>的时候，你会发现你的间隙十分难控制，一般情况下，IE6和IE7会凭空多一些间距。但很多情况你来到下一行，间隙就没了，但是前 面的内容又空了很大一块，出现这种情况虽然你可以改变IE的Margin，然后调整Firefox下面的Padding，以便使得两者显示起来得效果很相 似，但是你得CSS将变得臭长无比，你不得不多考虑更多可能出现这种问题补救措施，虽然你知道千篇一律来兼容它们，但是你会烦得要命。   
  
具体嵌套写法   
  
遵循上面得嵌套方式，<div><ul><li></li></ul></div>然后在CSS 里面告诉 ul {Margin:0pxadding:0px;list- style:none;}，其中list-style:none是不让<li>标记的最前方显示圆点或者数字等目录类型的标记，因为IE和 Firefox显示出来默认效果有些不一样。因此这样不需要做任何手脚，你的IE6、和IE7、Firefox显示出来的东西(外距，间距，高度，宽度) 就几乎没什么区别了，也许细心的你会在某一个时刻发现一、两个象素的差别，但那已经很完美了，不需要你通过调整大片的CSS来控制它们的显示了，你愿意， 你可以仅仅兼容一两个地方，而且通常这种兼容可以适应各种地方，不需要你重复在不同的地方调试不同的兼容方式–减轻你的烦 overflow:hidden; }   
这个是在越出长度后会自行的截掉多出部分的文字，并以省略号结尾，很好的一个技术。只是目前Firefox并不支持。   
  
只有Opera识别   
@media all and (min-width: 0px){ select {……} }   
针对Opera浏览器做单独的设定。   
  
以上都是写CSS中的一些兼容，建议遵循正确的标签嵌套(divul li 嵌套结构关系)，这样可以减少你使用兼容的频率，不要进入理解误区，并不是一个页面就需要很多的兼容来保持多浏览器兼容)，很多情况下也许一个兼容都不用 也可以让浏览器工作得非常好，这些都是用来解决局部的兼容性问题，如果希望把兼容性的内容也分离出来，不妨试一下下面的几种过滤器。这些过滤器有的是写在 CSS中通过过滤器导入特别的样式，也有的是写在HTML中的通过条件来链接或是导入需要的补丁样式。   
  
IE5.x的过滤器，只有IE5.x可见   
@media tty {   
i{content:"";/\*" "\*/}} @import ’ie5win.css’; /\*";}   
}/\* \*/   
  
IE5/MAC的过滤器，一般用不着   
/\*\*//\*/   
@import "ie5mac.css";   
/\*\*/   
  
下面是IE的条件注释，个人觉得用条件注释调用相应 兼容是比较完美的多浏览器兼容的解决办法。把需要兼容的地方单独放到一个文件里面，当浏览器版本符合的时候就可以调用那个被兼容的样式，这样不仅使用起来非常方便，而且对于制作这个CSS本身来讲，可以更严格的观察到是否有必要使用兼容，很多情况下，当我本人写CSS如果把全部代码包括兼容都写到一个 CSS文件的时候的时候会很随意，想怎么兼容就怎么兼容，而你独立出来写的时候，你就会不自觉的考虑是否有必要兼容，是先兼容 CSS？还是先把主CSS里面的东西调整到尽可能的不需要兼容？当你仅用很少的兼容就让很多浏览器很乖很听话的时候，你是不是很有成就感呢？你知道怎么选择了吧～～呵呵   
  
IE的if条件兼容 自己可以灵活使用参看这篇IE条件注释   
Only IE   
所有的IE可识别   
  
只有IE5.0可以识别   
Only IE 5.0+   
IE5.0包换IE5.5都可以识别   
  
仅IE6可识别   
Only IE 7/-   
IE6以及IE6以下的IE5.x都可识别   
Only IE 7/-   
仅IE7可识别   
  
  
  
  
  
Css 当中有许多的东西不不按照某些规律来的话，会让你很心烦，虽然你可以通过很多的兼容，很多的!important来控制它，但是你会发现长此以往你会很不甘心，看看许多优秀的网站，他们的CSS让IE6,Ie7,Firefox,甚至Safari,Opera运行起 来完美无缺是不是很羡慕？而他们看似复杂的模版下面使用的兼容少得可怜。其实你要知道IE 和 Firefox 并不不是那么的不和谐，我们找到一定的方法，是完全可以让他们和谐共处的。不要你认为发现了兼容的办法，你就掌握了一切，我们并不是兼容的奴隶。   
  
div ul li 的嵌套顺序   
  
今天只讲一个规则。就是<div><ul><li>的三角关系。我的经验就是<div>在最外面，里面 是<ul>，然后再是<li>，当然<li>里面又可以嵌套<div>什么的，但是并不建议你嵌套很多 东西。当你符合这样的规则的时候，那些倒霉的，不听话的间隙就不会在里面出现了，当你仅仅是<div>里面放<li>，而不 用<ul>的时候，你会发现你的间隙十分难控制，一般情况下，IE6和IE7会凭空多一些间距。但很多情况你来到下一行，间隙就没了，但是前 面的内容又空了很大一块，出现这种情况虽然你可以改变IE的Margin，然后调整Firefox下面的Padding，以便使得两者显示起来得效果很相 似，但是你得CSS将变得臭长无比，你不得不多考虑更多可能出现这种问题补救措施，虽然你知道千篇一律来兼容它们，但是你会烦得要命。   
  
具体嵌套写法   
  
遵循上面得嵌套方式，<div><ul><li></li></ul></div>然后在CSS 里面告诉 ul{Margin:0pxadding:0px;list- style:none;}，其中list-style:none是不让<li>标记的最前方显示圆点或者数字等目录类型的标记，因为IE和 Firefox显示出来默认效果有些不一样。因此这样不需要做任何手脚，你的IE6、和IE7、Firefox显示出来的东西(外距，间距，高度，宽度) 就几乎没什么区别了，也许细心的你会在某一个时刻发现一、两个象素的差别，但那已经很完美了，不需要你通过调整大片的CSS来控制它们的显示了，你愿意， 你可以仅仅兼容一两个地方，而且通常这种兼容可以适应各种地方，不需要你重复在不同的地方调试不同的兼容方式–减轻你的烦 overflow:hidden; }   
这个是在越出长度后会自行的截掉多出部分的文字，并以省略号结尾，很好的一个技术。只是目前Firefox并不支持。   
  
只有Opera识别   
@media all and (min-width: 0px){ select {……} }   
针对Opera浏览器做单独的设定。   
  
以上都是写CSS中的一些兼容，建议遵循正确的标签嵌套(divul li 嵌套结构关系)，这样可以减少你使用兼容的频率，不要进入理解误区，并不是一个页面就需要很多的兼容来保持多浏览器兼容)，很多情况下也许一个兼容都不用 也可以让浏览器工作得非常好，这些都是用来解决局部的兼容性问题，如果希望把兼容性的内容也分离出来，不妨试一下下面的几种过滤器。这些过滤器有的是写在 CSS中通过过滤器导入特别的样式，也有的是写在HTML中的通过条件来链接或是导入需要的补丁样式。   
  
IE5.x的过滤器，只有IE5.x可见   
@media tty {   
i{content:"";/\*" "\*/}} @import ’ie5win.css’; /\*";}   
}/\* \*/   
  
IE5/MAC的过滤器，一般用不着   
/\*\*//\*/   
@import "ie5mac.css";   
/\*\*/   
  
下面是IE的条件注释，个人觉得用条件注释调用相应 兼容是比较完美的多浏览器兼容的解决办法。把需要兼容的地方单独放到一个文件里面，当浏览器版本符合的时候就可以调用那个被兼容的样式，这样不仅使用起来非常方便，而且对于制作这个CSS本身来讲，可以更严格的观察到是否有必要使用兼容，很多情况下，当我本人写CSS如果把全部代码包括兼容都写到一个 CSS文件的时候的时候会很随意，想怎么兼容就怎么兼容，而你独立出来写的时候，你就会不自觉的考虑是否有必要兼容，是先兼容 CSS？还是先把主CSS里面的东西调整到尽可能的不需要兼容？当你仅用很少的兼容就让很多浏览器很乖很听话的时候，你是不是很有成就感呢？你知道怎么选择了吧～～呵呵   
  
IE的if条件兼容 自己可以灵活使用参看这篇IE条件注释  
Only IE   
所有的IE可识别   
  
只有IE5.0可以识别   
Only IE 5.0+   
IE5.0包换IE5.5都可以识别   
  
仅IE6可识别   
Only IE 7/-   
IE6以及IE6以下的IE5.x都可识别   
Only IE 7/-   
仅IE7可识别   
  
  
  
  
  
Css 当中有许多的东西不不按照某些规律来的话，会让你很心烦，虽然你可以通过很多的兼容，很多的!important来控制它，但是你会发现长此以往你会很不甘心，看看许多优秀的网站，他们的CSS让IE6,Ie7,Firefox,甚至Safari,Opera运行起 来完美无缺是不是很羡慕？而他们看似复杂的模版下面使用的兼容少得可怜。其实你要知道IE 和 Firefox 并不不是那么的不和谐，我们找到一定的方法，是完全可以让他们和谐共处的。不要你认为发现了兼容的办法，你就掌握了一切，我们并不是兼容的奴隶。   
  
div ul li 的嵌套顺序   
  
今天只讲一个规则。就是<div><ul><li>的三角关系。我的经验就是<div>在最外面，里面 是<ul>，然后再是<li>，当然<li>里面又可以嵌套<div>什么的，但是并不建议你嵌套很多 东西。当你符合这样的规则的时候，那些倒霉的，不听话的间隙就不会在里面出现了，当你仅仅是<div>里面放<li>，而不 用<ul>的时候，你会发现你的间隙十分难控制，一般情况下，IE6和IE7会凭空多一些间距。但很多情况你来到下一行，间隙就没了，但是前 面的内容又空了很大一块，出现这种情况虽然你可以改变IE的Margin，然后调整Firefox下面的Padding，以便使得两者显示起来得效果很相 似，但是你得CSS将变得臭长无比，你不得不多考虑更多可能出现这种问题补救措施，虽然你知道千篇一律来兼容它们，但是你会烦得要命。   
  
具体嵌套写法   
  
遵循上面得嵌套方式，<div><ul><li></li></ul></div>然后在CSS 里面告诉 ul {Margin:0pxadding:0px;list- style:none;}，其中list-style:none是不让<li>标记的最前方显示圆点或者数字等目录类型的标记，因为IE和 Firefox显示出来默认效果有些不一样。因此这样不需要做任何手脚，你的IE6、和IE7、Firefox显示出来的东西(外距，间距，高度，宽度) 就几乎没什么区别了，也许细心的你会在某一个时刻发现一、两个象素的差别，但那已经很完美了，不需要你通过调整大片的CSS来控制它们的显示了，你愿意， 你可以仅仅兼容一两个地方，而且通常这种兼容可以适应各种地方，不需要你重复在不同的地方调试不同的兼容方式–减轻你的烦。你可以ul.class1, ul.class2, ul.class3{xxx:xxxx}的方式方便的整理出你要兼容的地方，而统一兼容。尝试一下吧，再也不要乱嵌套了，虽然在Div+CSS的方式下你几乎可以想怎么嵌套就怎么嵌套，但是按照上面的规律你将轻松很多，从而事半功倍！  
  
去掉ie有默认最低高度

<div style="height:2px;background:red;overflow:hidden;"></div>

其中height:2px为你要设的高度，overflow:hidden最为关键，他就是帮你去掉默认高度

随着最新CSS的不断完善，越来越多的网站采用DIV+CSS布局。而原来使用table套 table的网页布局模式也逐渐应该淘汰了。由于目前IE6不能支持有些标准的CSS，需要用微软特有的CSS来修复这些BUG.而且现在随着浏览器层出不穷，要是页面能够适应尽量多的浏览器成为一个课题。 但是随着CSS标准的进一步完善，浏览器将最终都会遵循这个标准，到时候写DIV+CSS布局的页 面就不那么麻烦了。

但是现在，我们还是需要处理CSS在不同浏览器下的兼容性。一下是一个网友写的CSS兼容技巧，值得大家参考。

CSS兼容技巧

　　1 FF下给 div 设置 padding 后会导致 width 和 height 增加, 但IE不会.

            可用important解决

　　2 居中问题.

　　  1).垂直居中.将 line-height 设置为 当前 div 相同的高度, 再通过vertical-align: middle.( 注意内容不要换行.)

　　   2).水平居中. margin: 0 auto;(当然不是万能)

　　3 若需给 a 标签内内容加上样式, 需要设置 display: block;(常见于导航标签)

　　4 FF 和 IE 对 BOX 理解的差异导致相差 2px 的还有设为 float的div在ie下 margin加倍等问题.

　　5 ul 标签在 FF 下面默认有list-style 和 padding . 最好事先声明, 以避免不必要的麻烦. (常见于导航标签和内容列表)

　　6 作为外部 wrapper 的 div 不要定死高度, 最好还加上 overflow: hidden.以达到高度自适应.

　　7 关于手形光标. cursor: pointer. 而hand 只适用于 IE.

针对firefox ie6 ie7的css样式

　　现在大部分都是用!important来hack，对于ie6和firefox测试可以正常显示，

　　但是ie7对!important可以正确解释，会导致页面没按要求显示!找到一个针

　　对IE7不错的hack方式就是使用“\*+html”，现在用IE7浏览一下，应该没有问题了。

　　现在写一个CSS可以这样：

　　#1 { color: #333; } /\* Moz \*/

　　\* html #1 { color: #666; } /\* IE6\*/

　　\*+html #1 { color: #999; } /\* IE7\*/

　　那么在firefox下字体颜色显示为#333，IE6下字体颜色显示为#666，IE7下字体颜色显示为#999。

css布局中的居中问题

　　主要的样式定义如下：

　　body {TEXT-ALIGN: center;}

　　#center { MARGIN-RIGHT: auto;MARGIN-LEFT: auto; }

　　说明：

　　首先在父级元素定义TEXT-ALIGN: center;这个的意思就是在父级元素内的内容居中;对于IE这样设定就已经可以了。

　　但在mozilla中不能居中。解决办法就是在子元素定义时候设定时再加上“MARGIN-RIGHT: auto;MARGIN-LEFT: auto; ”

　　需要说明的是，如果你想用这个方法使整个页面要居中，建议不要套在一个DIV里，你可以依次拆出多个div，

　　只要在每个拆出的div里定义MARGIN-RIGHT: auto;MARGIN-LEFT: auto; 就可以了。

盒模型不同解释

　　#box{ width:600px; //for ie6.0-w/idth:500px; //for ff+ie6.0}

　　#box{ width:600px!important //forff width:600px; //for ff+ie6.0 width /\*\*/:500px; //for ie6.0-}

浮动ie产生的双倍距离

　　#box{ float:left; width:100px;margin:0 0 0 100px; //这种情况之下IE会产生200px的距离 display:inline; //使浮动忽略}

　　这里细说一下block,inline两个元素,Block元素的特点是:总是在新行上开始,高度,宽度,行高,边距都可以控制(块元素);Inline元素的特点是:和其他元素在同一行上,…不可控制(内嵌元素);

　　#box{ display:block; //可以为内嵌元素模拟为块元素 display:inline; //实现同一行排列的的效果 diplay:table;

　　IE不认得min-这个定义，但实际上它把正常的width和height当作有min的情况来使。这样问题就大了，如果只用宽度和高度，

　　正常的浏览器里这两个值就不会变，如果只用min-width和min-height的话，IE下面根本等于没有设置宽度和高度。

　　比如要设置背景图片，这个宽度是比较重要的。要解决这个问题，可以这样：

　　#box{ width: 80px; height:35px;}html>body #box{ width: auto; height: auto; min-width: 80px;min-height: 35px;}

 页面的最小宽度

　　min-width是个非常方便的CSS命令，它可以指定元素最小也不能小于某个宽度，这样就能保证排版一直正确。但IE不认得这个，

　　而它实际上把width当做最小宽度来使。为了让这一命令在IE上也能用，可以把一个

放到 标签下，然后为div指定一个类：

　　然后CSS这样设计：

　　#container{ min-width: 600px;width:expression(document.body.clientWidth < 600? “600px”: “auto” );}

　　第一个min-width是正常的;但第2行的width使用了Javascript，这只有IE才认得，这也会让你的HTML文档不太正规。它实际上通过Javascript的判断来实现最小宽度。

清除浮动

　　.hackbox{ display:table; //将对象作为块元素级的表格显示}或者.hackbox{ clear:both;}

　　或者加入:after(伪对象),设置在对象后发生的内容，通常和content配合使用，IE不支持此伪对象，非Ie 浏览器支持，

　　所 以并不影响到IE/WIN浏览器。这种的最麻烦的……#box:after{ content: “.”; display: block; height: 0; clear: both; visibility: hidden;}

DIV浮动IE文本产生3象素的bug

　　左边对象浮动，右边采用外补丁的左边距来定位，右边对象内的文本会离左边有3px的间距.

　　#box{ float:left;width:800px;}#left{ float:left; width:50%;}#right{ width:50%;}\*html #left{margin-right:-3px; //这句是关键}

　　HTML代码

属性选择器(这个不能算是兼容,是隐藏css的一个bug)

　　p[id]{}div[id]{}

　　这个对于IE6.0和IE6.0以下的版本都隐藏,FF和OPera作用

　　属性选择器和子选择器还是有区别的,子选择器的范围从形式来说缩小了,属性选择器的范围比较大,如p[id]中,所有p标签中有id的都是同样式的.

IE捉迷藏的问题

　　当div应用复杂的时候每个栏中又有一些链接，DIV等这个时候容易发生捉迷藏的问题。

　　有些内容显示不出来，当鼠标选择这个区域是发现内容确实在页面。

　　解决办法：对#layout使用line-height属性或者给#layout使用固定高和宽。页面结构尽量简单。

高度不适应

　　高度不适应是当内层对象的高度发生变化时外层高度不能自动进行调节，特别是当内层对象使用

　　margin 或paddign时。

　　例：

       p对象中的内容

　　CSS：#box{background-color:#eee; }

　　#box p {margin-top:20px;margin-bottom: 20px; text-align:center; }

　　解决方法：在P对象上下各加2个空的div对象CSS代码：.1{height:0px;overflow:hidden;}或者为DIV加上border属性。

一、超链接点击过后hover**样式就不出现的问题？**

被点击访问过的超链接样式不再具有hover和active样式了,解决方法是改变CSS属性的排列顺序: L-V-H-A

二、IE6**的margin双倍边距bug问题**

例如：

|  |
| --- |
| <style type="text/css">      body {margin:0;}      div {float:left; margin-left:10px; width:200px; height:200px; border:1px solid red;}  </style> |

浮动后本来外边距10px,但IE解释为20px,解决办法是加上display:inline;

三、为什么中火狐浏览器下文本无法撑开容器的高度？

标准浏览器中固定高度值的容器是不会象IE6里那样被撑开的,那我又想固定高度，又想能被撑开需要怎样设置呢？办法就是去掉height设置min-height:200px; 这里为了照顾不认识min-height的IE6 可以这样定义：

|  |
| --- |
| div { height:auto!important; height:200px; min-height:200px; } |

四、为什么web**标准中无法设置IE浏览器滚动条颜色了？**

原来样式设置：

|  |
| --- |
| <style type="text/css">      body { scrollbar-face-color:#f6f6f6; scrollbar-highlight-color:#fff; scrollbar-shadow-color:#eeeeee; scrollbar-3dlight-color:#eeeeee; scrollbar-arrow-color:#000; scrollbar-track-color:#fff; scrollbar-darkshadow-color:#fff; }  </style> |

解决办法是将body换成html

五、如何定义1px**左右高度的容器？**

IE6下这个问题是因为默认的行高造成的，解决的方法也有很多，例如：overflow:hidden | zoom:0.08 | line-height:1px

六、怎么样才能让层显示在FLASH**之上呢？**

解决的办法是给FLASH设置透明:

|  |
| --- |
| <a href="http://www.chinaz.com/">:</a>  <pre lang="html" line="1">  <param name="wmode" value="transparent" /> |

七、怎样使一个div**层居中于浏览器中？**

|  |
| --- |
| <style type="text/css">  <!--  div {  position:absolute;  top:50%;  left:50%;  margin:-100px 0 0 -100px;  width:200px;  height:200px;  border:1px solid red;  }  -->  </style> |

这里使用百分比绝对定位，与外补丁负值的方法，负值的大小为其自身宽度高度除以二

八、firefox**浏览器中嵌套div标签的居中问题的解决方法**

假定有如下情况：

|  |
| --- |
| <div id="a">  <div id="b"> </div>  </div> |

如果要实现b在a中居中放置，一般只需用CSS设置a的text-align属性为center。这样的方法在IE里看起来一切正常；但是在Firefox中b却会是居左的。

解决办法就是设置b的横向margin为auto。例如设置b的CSS样式为：margin: 0 auto;

**浏览器的内核**

Mozilla Firefox ( Gecko )

Internet Explorer ( Trident )

Opera ( Presto )

Safari ( WebKit )

Google Chrome ( WebKit )

腾讯TT、世界之窗、360浏览器、遨游浏览器都是给IE加了个外壳，不过如果电脑上装的是ie8的话，这些浏览器还是调用ie7的内核。搜狗浏览器比较特殊，它有两种浏览模式：一是兼容模式，该模式使用IE内核；二是高速模式，该模式使用WebKit内核。解决ie7、ie8兼容性最好的办法是在head标签中加入meta 类型<metahttp-equiv="X-UA-Compatible" content="IE=EmulateIE7" />，只要IE8一读到这个标签,它就会自动启动IE7兼容模式

**CSSHack**

解决浏览器兼容性问题的主要方法是CSS hack。由于不同的浏览器，比如Internet Explorer 6,Internet Explorer 7,Mozilla Firefox等，对CSS的解析认识不一样，因此会导致生成的页面效果不一样，得不到我们所需要的页面效果。这个时候我们就需要针对不同的浏览器去写不同的CSS，让它能够同时兼容不同的浏览器，能在不同的浏览器中也能得到我们想要的页面效果。这个针对不同的浏览器写不同的CSS code的过程，就叫CSS hack,也叫写CSS hack。

CSS Hack的原理是什么

由于不同的浏览器对CSS的支持及解析结果不一样，还由于CSS中的优先级的关系。我们就可以根据这个来针对不同的浏览器来写不同的CSS。比如 IE6能识别下划线"\_"和星号" \* "，IE7能识别星号" \* "，但不能识别下划线"\_"，而firefox两个都不能认识。等等

各浏览器CSS hack兼容表：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | IE6 | IE7 | IE8 | Firefox | Opera | Safari |
| !important |  | Y | Y | Y | Y | Y |
| \_ | Y |  |  |  |  |  |
| \* | Y | Y |  |  |  |  |
| \*+ |  | Y |  |  |  |  |
| \9 | Y | Y | Y |  |  |  |
| \0 |  |  | Y |  |  |  |
| nth-of-type(1) |  |  |  |  | Y | Y |

**如何解决浏览器的兼容性**

在head标签中加入meta 类型<meta http-equiv="X-UA-Compatible"content="IE=EmulateIE7" />，这样就解决了ie7、ie8兼容问题。现在剩下ie6、ie7、Firefox、Chrome(Safari与Chrome使用同一内核)、Opera这几种浏览器的兼容性问题，我们需要使用CSS Hack来解决该问题。代码如下所示：

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

<html>

<head>

<meta http-equiv="Content-Type"content="text/html; charset=utf-8" />

<meta http-equiv="X-UA-Compatible"content="IE=EmulateIE7" />

<title>浏览器兼容性问题</title>

<style type="text/css">

.t1

{

       color:#000000; /\*所有浏览器都支持 此处填写Firefox的css\*/

       \*color:#0000FF;/\* ie6 id7 支持此处填写ie7的css\*/

       \_color:#66CCCC;/\* ie6支持  此处填写ie6的css\*/

}

@media all and (-webkit-min-device-pixel-ratio:10000),not all and (-webkit-min-device-pixel-ratio:0)

{ .t1{color:#9900FF}} /\* oprea支持  此处填写oprea的css\*/

@media screen and (-webkit-min-device-pixel-ratio:0)

{

.t1{color:#336600}/\* Chrome、Safari支持  此处填写Chrome的css\*/

}

</style>

</head>

<body>

<div class="t1">ff、ie8、ie7、ie6、oprea、Safari兼容性css 书写模式<br>

.t1{

       color:#000000; /\*所有浏览器都支持 此处填写Firefox的css\*\*/<br>

       \*color:#0000FF;/\* ie6 id7 支持此处填写ie7的css\*/<br>

       \_color:#66CCCC;/\* ie6支持  此处填写ie6的css\*/<br>

}<br>

/\* oprea支持此处填写oprea的css\*/<br>

@media all and (-webkit-min-device-pixel-ratio:10000),not all and (-webkit-min-device-pixel-ratio:0)<br>

{ .t1{color:#CC66FF}}<br>

/\* Chrome、Safari支持 此处填写Chrome的css\*/<br>

@media screen and (-webkit-min-device-pixel-ratio:0)

{

.t1{color:#336600}}

}

</div>

</div>

</body>

</html>

**常见的浏览器兼容问题**

Css样式是与DOCTYPE引入的W3C//DTD有关的，不同的dtd对css的解析也不同，我们现在统一使用<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

**css兼容问题：**

**1. 默认的内外边距不同**

问题：

各个浏览器默认的内外边距不同

解决：

\*{margin:0;padding:0;}

**2. 水平居中的问题**

问题：

**设置**text-align: center

ie6-7文本居中，嵌套的块元素也会居中

ff /opera /safari /ie8文本会居中，嵌套块不会居中

解决：

块元素设置

1、margin-left:auto;margin-right:auto

2、margin:0 auto;

3、<div align=”center”></div>

**3. 垂直居中的问题**

**问题：**

在浏览器中 想要垂直居中，设置vertical-align:middle; 不起作用。**例如：**ie6下**文本与文本输入框对不齐，需设置**vertical-align:middle，但是文本框的内容不会垂直居中

**解决：**

给容器设置一个与其高度相同的行高

line-height:与容器的height一样

**4. 关于高度问题**

问题：

如果是动态地添加内容，高度最好不要定义。浏览器可以自动伸缩，然而如果是静态的内容，高度最好定好。

如果设定了高度，内容过多时，ie6下会自动增加高度、其他浏览器会超出边框

解决：

1.设置overflow:hidden;

2.高度自增height:auto!important;height:100px;

**5. IE6 默认的div高度**

问题：

ie6默认div高度为一个字体显示的高度，所在ie6下div的高度大于等于一个字的高度，因此在ie6下定义高度为1px的容器，显示的是一个字体的高度

解决：

**为这个容器设置下列属性之一**

1、设置overflow:hidden;

2、设置line-height:1px;

3、设置zoom:0.08

**6. IE6 最小高度(宽度)的问题**

**问题：**

ie6不支持min-height、min-width属性，默认height是最小高度，width是最小宽度。

**解决：**

使用ie6不支持但其余浏览器支持的属性!important。

设置属性min-height:200px; height:auto !important;height:200px;

**7. td高度的问题**

问题：

table中td的宽度都不包含border的宽度，但是oprea和ff中td的高度包含了border的高度

解决：

       设置line-height和height一样。在ie中如果td中的没有内容，那么border将不会显示

**8. div嵌套p时，出现空白行**

问题：

div中显示<p>文本</p>，ff、oprea、Chrome：top和bottom都会出现空白行，但是在ie下不会出现空白行。

解决：

设置p的margin:0px，再设置div的padding-top和padding-bottom

**9. IE6-7图片下面有空隙的问题**

**问题：**

块元素中含有图片时，ie6-7中会出现图片下有空隙

**解决：**

1、在源代码中让</div>和<img>在同一行

2、**将图片转换为块级对象**display:block;

3、**设置图片的垂直对齐方式** vertical-align:top/middle/bottom

**4、改变父对象的属性，**如果父对象的宽、高固定，图片大小随父对象而定，那么可以对父元素设置：overflow:hidden;

**5、设置图片的浮动属性**float:left;

**10. IE6双倍边距的问题**

**问题：**

**ie6中设置浮动，同时又设置margin时，会出现双倍边距的问题**

**例**float:left;width:100px;margin:0 100px;

**解决：**

**设置**display:inline;

**11. IE6 weidth为奇数，右边多出1px的问题**

**问题：**

父级元素采用相对定位，且宽度设置为奇数时，子元素采用绝对定位，在ie6中会出现右侧多出1像素

**解决：**

将宽度的奇数值改成偶数

**12. IE6两个层之间3px的问题**

**问题：**

**左边层采用浮动，右边没有采用浮动，这时在ie6中两层之间就会产生3像素的间距**

**解决：**

1、右边层也采用浮动  float

2、左边层添加属性  margin-right:-3px;

**13. IE6 子元素绝对定位的问题**

**问题：**

**父级元素使用padding后，子元素使用绝对定位，不能精确定位**

**解决：**

在子元素中设置 \_left:-20px; \_top:-1px;

**14. 显示手型cursor:hand**

**问题：**

**ie6/7/8、opera      都支持  但是safari 、 ff 不支持**

**解决：**

**写成 cursor:pointer;  (所有浏览器都能识别)**

**15. IE6-7 line-height失效的问题**

**问题：**

**在ie中img与文字放一起时， line-height不起作用**

**解决：**

都设置成float

**16. td自动换行的问题**

问题：

Table宽度固定，td自动换行

解决：

设置Table的table-layout:fixed，td的word-wrap:break-word

**17. 子容器浮动后，父容器扩展问题**

问题：

子容器都float以后，父容器没有设定高度,父容器将不会扩展

解决：

只需要添加一个clear:both的div，代码如下：

<div style="border:1px solid#333;width:204px">

    <divstyle="width:100px;border:1px solid #333; float:left; ">子容器a</div>

    <divstyle="width:100px;border:1px solid #333; float:left;">子容器b</div>

    <divstyle="clear:both"></div>

</div>

**18. 透明png图片会带背景色**

问题：

在ie6下透明的png图片会带一个背景色

解决：

background-image: url(icon\_home.png);/\* 其他浏览器 \*/

background-repeat: no-repeat;

\_filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(src='icon\_home.png');/\* IE6 \*/

\_background-image: none; /\* IE6 \*/

**19. list-style-position默认值的问题**

问题：

ie下list-style-position默认为inside,  firefox默认为outside

解决：

css中指定为outside即可解决兼容性问题

**20. list-style-image准确定位的问题**

**问题：**

**li前设置图片时，图片与其后的文字对齐问题**

**解决：**

1、采用背景定位 和 字符缩进的方法

background:url() no-repeat left center;text-index:16px;

2、采用相对定位方法

li 设置list-style:url();

li的子元素position:relative;top:-5px;

**21. ul标签默认值的问题**

**问题：**

**ul标签在**ff中默认是有padding值的,而在ie中只有margin有值

**解决：**

定义ul{margin:0;padding:0;}就能解决大部分问题

**22. IE中li指定高度后，出现排版错误**

问题：

在ie下如果为li指定高度可能会出现排版错位

解决：

       设置line-height

**23. ul或li浮动后，显示在div外**

问题：

div中的ul或li设置float以后，都不在div中

解决：

必须在ul标签后加<divstyle="clear:both"></div>来闭合外层div

**24. ul浮动后，margin变大**

问题：

ul设置 float后，在ie中margin将变大

解决：

设置ul的display:inline，li的list-style-position:outside

**25. li浮动后，margin变大**

问题：

li设置 float后，在ie中margin将变大

解决：

设置li的display:inline

**26. 嵌套使用ul、li的问题**

问题：

ie的bug，嵌套使用ul、li时，里层的li设置float以后，外层li不设置float, 里面的ul顶部和它外面的li总是有一段间距

解决：

设置里面的ul的zoom:1

**27. IE6-7 li底部有3px的问题**

**问题：**

这个bug产生的充要条件是li的子元素浮动并且li设置了以下CSS属性之一：width、height、zoom、padding-top、padding-bottom、margin-top、margin-bottom。

**解决：**

1、div设置clear:left|both，这时li不能设置width、height、zoom。

2、li设置float:left，这时li可以设置width、height、zoom。

3、li设置clear:left|both，这时li不能设置width、height、zoom。

4、IE6/IE7的这个Bug可以通过给li中的div设置vertical-align:top|middle|bottom解决。

**28. IE6 垂直列表间隙的问题**

**问题：**

li中有a且设置display:block时，ie6中列表间会出现空隙

**解决：**

1、li中加display:inline;

2、li使用float  然后 clear:both;

3、给包含的文本末尾添加一个空格

4、设置width

**29. IE6 列表背景颜色失效的问题**

**问题：**

当父元素设置position:relative;时，在ie6中第一个ul、ol、dl的背景颜色失效

**解决：**

ul、ol、dl 都设置为position:relative;

**30. IE6-7 列表背景颜色失效的问题2**

**问题：**

做横向导航栏时，ul设置为float且有背景色，li设置为float。ie6-7背景颜色失效

**解决：**

很多ie的bug都可以通过触发layout来解决 ul添加属性

1、height:1%;

2、float:left;

3、zoom:1;

**31. 列表不能换行的问题**

**问题：**

**li设置为浮动，后面的li紧随其后，不能换行**

**解决：**

1、为这个ul定义合适的宽高

2、给包含这个ul 的父div定义合适的宽高。

**32. li中的内容以省略号显示**

**问题：**

li中内容超过长度时，想以省略号显示， 此方法适用于ie6-7-8、opera、safari浏览器

ff浏览器不支持

**解决：**

li{width:200px;white-space:nowrap;text-overflow:ellipsis;

-o-text-overflow:ellipsis; overflow: hidden; }

**33. 超链接访问过后hover样式不出现的问题**

**问题：**

**点击超链接后，hover、active样式没有效果**

**解决：**

改变CSS属性的排列顺序: L-V-H-A

**34. 禁用中文输入法的问题**

**问题：**

**不能在输入框中输入汉字**

**解决：**

只在ie系列 和ff中有效

ime-mode:disabled    **(但可以粘贴)**

禁用粘贴：

onpaste="return false"

**35. 除去滚动条的问题**

**问题：**

**隐藏滚动条**

**解决：**

**1、只有ie6-7支持**<bodyscroll="**no**">

**2、除ie6-7不支持**body{overflow:hidden}

**3、所有浏览器**html{overflow:hidden}

**36. 让层显示在FLASH之上**

**问题：**

**想让层的内容显示在flash上**

**解决：**

把FLASH设置透明

1、<param name=" wmode "value="transparent" />

2、<param name="wmode"value="opaque"/>

**37. 去除链接虚线边框的问题**

**问题：**

**当点击超链接后，ie6/7/8  ff会出现虚线边框 ,而**opera、safari没有虚线边框

**解决：**

ie6/7中 设置为a { blr:expression(this.onFocus=this.blur()) }

ie8 和 ff 都不支持expression  在ie8 、ff中设置为 :focus { outline: none; }

**38. css滤镜的问题**

**问题：**

css滤镜只在ie中有效，Firefox, Safari(WebKit), Opera只能够设置透明，它们不支持滤镜filter，无法实现图片切换中间变换的效果，只能通过透明度来设置。

**解决：**

**ff中设置透明度**  -moz-opacity:0.10; opacity:0.6;

ie中只设置filter:alpha(opacity=50); 时，ie6-7失效，还要设置

1、zoom:1;  2、width:100%;

**39. IE6背景闪烁的问题**

**问题：**

链接、按钮用CSSsprites作为背景，在ie6下会有背景图闪烁的现象。原因是:IE6没有将背景图缓存，每次触发hover的时候都会重新加载

**解决：**

可以用JavaScript设置ie6缓存这些图片：

document.execCommand("BackgroundImageCache",false,true);

**40. 出现重复文字的问题**

**问题：**

**<div style="width:400px">**

**<divstyle="float:left"></div>**

**<!– \_ –>**

**<div style="float:right;width:400px">↓这就是多出来的那只猪</div>**

**</div>**

**解决：**

**1、  改变结构，不出现【一个容器包含2两个具有“float”样式的子容器】的结构。**

**2、减小第二个容器的宽度，使父容器宽度减去第二个容器宽度的值大于3**

3、**去掉所有的注释。**

**4、**修正注释的写法。<!--[if !IE]>这里是注释内容<![endif]-->

5、在第二个容器后面加一个或者多个<divstyle="clear"></div>来解决。

**41. ff、chrome绝对定位无效**

问题：

在IE给td设置position:relative，然后给它包含的一个容器使用position:absolute进行定位是有效的，但在FF和Chrome下却不可以。

解决：

设置td的display:block。

**42. IE6 绝对定位的问题**

**问题：**

<div style="position:relative;border:1px solidorange;text-align:center;">

<div style="position:absolute;top:0;left:0;

background:#CCC;">dovapour</div>

<a href="#" title="vapour的blog">内容</a>

</div>

**解决：**

left的定位错误问题

1、给父层设置zoom:1触发layout。

2、给父层设置宽度width

bottom的定位错误问题

1、给父层设置zoom:1触发layout。

2、给父层设置高度height

**43. 子容器宽度大于父容器宽度时，内容超出**

问题：

子DIV的宽度和父DIV的宽度都已经定义，在IE6中如果其子DIV的宽度大于父DIV的宽度，父DIV的宽度将会被扩展，在其他浏览器中父DIV的宽度将不会扩展，子DIV将超出父DIV

解决：

设置overflow:hidden，子DIV将不会超出父DIV。

**44. float的div闭合的问题**

问题：

例如：<#div id=”floatA” ><#div id=”floatB”><#div id=” NOTfloatC” >这里的NOTfloatC并不希望继续平移，而是希望往下排。(其中floatA、floatB的属性已经设置为 float:left;)

这段代码在IE中毫无问题，问题出在其他浏览器中。原因是NOTfloatC并非float标签，必须将float标签 闭合。

解决：

在 <#div class=”floatB”> <#divclass=”NOTfloatC”>之间加上 < #div class=”clear”>这个div一定要注意位置，而且必须与两个具有float属性的div同级，之间不能存在嵌套关系，否则会 产生异常。并且将clear这种样式定义为为如下即可：.clear{ clear:both;}

**45. 单选框、复选框与后面的文字对不齐**

问题：

     单选框、复选框与后面的文字对不齐。

解决：

.align{font-size:12px;}

.align input{ display:block; float:left;}

.align label{ display:block; float:left;padding-top:3px; \*padding-top:5px;}

**需注意的问题：**

**1. 设置padding后高度和宽带都会增加**

说明：

       除了ie5.5，其他所有浏览器中，设置padding以后高度和宽带都会增加

**2. 使用XHTML 1.0 Transitional后，div宽度**

说明：

       在使用XHTML 1.0Transitional以后div宽度都不包含border的宽度了，设置宽度的时候需要注意下。

**3. 外层相对定位，内层绝对定位**

说明：

ie6下，外层div的postion: relative，并设置text-align，内层div的postion:absolute，这时内层的位置是相对于text-align而言的

例如：

<div style="position:relative;border:1px solidorange;text-align:center;zoom:1"> position:relative

<divstyle="position:absolute;top:0;left:0;background:#CCC;">position:absolute</div>

</div>

**4. &nbsp; 显示的大小不一致**

说明：

默认字本显示问题，导致&nbsp;显示的大小不一致，在ie下比较小一点，其他的浏览器都一致，当你使用了&nbsp;造成问题时请注意。

**5. 边框重叠说明**

说明：

为 table、td 都指定了边框后，然后使用border-collapse:collapse让边框重叠，可以看出在发生重叠时，Firefox是用 td 覆盖 table 的，而 IE 是用 table 覆盖 td 的。使用时候需要注意。

**6. 设置td padding的说明**

说明：

设置td的padding以后高度和宽带都会增加,padding-left和padding-right的效果都一样增加了td的宽带，但是padding-top和padding-bottom的效果不一样。最好不要使用td的ding-top和padding-bottom

**7. ul设置的说明**

说明：

ul一般设置：list-style-type:none;margin:0px;padding:0px；li一般设置：list-style-type:none;list-style-position:outside

**8. 使一个层垂直居中于浏览器中**

说明：

使用百分比绝对定位,与外补丁负值的技巧,负值的大小为其自身宽度高度除以二

div {   
position:absolute; top:50%; lef:50%; margin:-100px 0 0 -100px;

width:200px; height:200px; border:1px solid red;   
}

**9. 万能 float 闭合**

说明：

可以用这个解决多个div对齐时的间距不对， 将以下代码加入GlobalCSS 中,给需要闭合的div加上 class=”clearfix” 即可

<style>  
/\* Clear Fix \*/  
.clearfix:after { content:".";display:block;height:0; clear:both;visibility:hidden;  
}  
.clearfix {  
    display:inline-block;  
}  
/\* Hide from IE Mac \\*/  
.clearfix {display:block;}  
/\* End hide from IE Mac \*/  
/\* end of clearfix \*/  
</style>

**10. 触发layout**

说明：

IE6中很多Bug都可以通过触发layout得到解决.下列的CSS属性或取值会让一个元素获得layout：

position:absolute 绝对定位元素的包含区块(containing block)就会经常在这一方面出问题

float:left|right 由于layout元素的特性，浮动模型会有很多怪异的表现

display:inline-block 当一个内联级别的元素需要layout的时候就往往符用到它，这也可能也是这个CSS属性的唯一效果----让某个元素有layout

width: 除auto外的任何值

height: 除auto外的任何值

zoom: 除auto外的任何值

**11、如何使连续长字段自动换行**

**ff最新版本 word-wrap:break-word;就可以了**

**ff旧版本 还要使用javascript完成文字换行**

**<style type="text/css">**

**div {**

**width:300px;**

**word-wrap:break-word;**

**border:1px solidred;**

**}**

**</style>**

<script type="text/javascript">

function toBreakWord(intLen){

var obj=document.getElementById("ff");

var strContent=obj.innerHTML;

var strTemp="";

while(strContent.length>intLen){

strTemp+=strContent.substr(0,intLen)+"&#10;";

strContent=strContent.substr(intLen,strContent.length);

}

strTemp+="&#10;"+strContent;

obj.innerHTML=strTemp;

}

if(document.getElementById  && !document.all)  toBreakWord(37)

**12、设置滚动条颜色 只对ie系列有效 在html中 而不是设置body**

<style type="text/css">

html {

      scrollbar-face-color:#f6f6f6;

      scrollbar-highlight-color:#fff;

      scrollbar-shadow-color:#eeeeee;

      scrollbar-3dlight-color:#eeeeee;

      scrollbar-arrow-color:#000;

      scrollbar-track-color:#fff;

      scrollbar-darkshadow-color:#fff;

       }

</style>

IE不支持float：inherit overflow:hidden有2个用法，一个是隐藏溢出，另一个是清除浮动。

<div>, <p>, <h1>, <form>, <ul>和 <li>是块元素的例子

<span>, <a>, <label>, <input>,<img>, <strong> 和<em>是inline元素

<body oncontextmenu="return false"ondragstart="return false"  tstart="returnfalse"  scroll="auto">

这行代码放在body中，去掉了页面鼠标右键快捷菜单，达到防止图片另存为的目的。

**javascript部分  
1. document.form.item 问题**  
问题：  
代码中存在 document.formName.item("itemName") 这样的语句，不能在FF下运行  
解决方法：  
改用 document.formName.elements["elementName"]  
  
**2. 集合类对象问题**  
问题：  
代码中许多集合类对象取用时使用()，IE能接受，FF不能  
解决方法：  
改用 [] 作为下标运算，例：  
document.getElementsByName("inputName")(1) 改为document.getElementsByName("inputName")[1]  
  
**3. window.event**  
问题：  
使用 window.event 无法在FF上运行  
解决方法：  
FF的 event 只能在事件发生的现场使用，此问题暂无法解决。可以把 event 传到函数里变通解决：  
onMouseMove = "functionName(event)"  
function functionName (e) {  
    e = e || window.event;  
    ......  
}  
  
**4. HTML对象的 id 作为对象名的问题**  
问题：  
在IE中，HTML对象的 ID 可以作为 document 的下属对象变量名直接使用，在FF中不能  
解决方法：  
使用对象变量时全部用标准的 getElementById("idName")  
  
**5. 用 idName 字符串取得对象的问题**  
问题：  
在IE中，利用eval("idName") 可以取得 id 为 idName 的HTML对象，在FF中不能  
解决方法：  
用 getElementById("idName") 代替 eval("idName")  
  
**6. 变量名与某HTML对象 id 相同的问题**  
问题：  
在FF中，因为对象 id 不作为HTML对象的名称，所以可以使用与HTML对象 id 相同的变量名，IE中不能  
解决方法：  
在声明变量时，一律加上 var ，以避免歧义，这样在IE中亦可正常运行  
最好不要取与HTML对象 id 相同的变量名，以减少错误  
  
**7. event.x 与 event.y 问题**  
问题：  
在IE中，event 对象有x,y属性，FF中没有  
解决方法：  
在FF中，与 event.x 等效的是 event.pageX ，但event.pageX IE中没有  
故采用 event.clientX 代替 event.x ，在IE中也有这个变量  
event.clientX 与 event.pageX 有微妙的差别，就是滚动条  
要完全一样，可以这样：  
mX = event.x ? event.x : event.pageX;  
然后用 mX 代替 event.x  
  
**8. 关于frame**  
问题：  
在IE中可以用 window.testFrame 取得该frame，FF中不行  
解决方法：  
window.top.document.getElementById("testFrame").src = 'xx.htm'  
window.top.frameName.location = 'xx.htm'  
  
**9. 取得元素的属性**  
在FF中，自己定义的属性必须 getAttribute() 取得  
  
**10. 在FF中没有parentElement，parement.children 而用 parentNode，parentNode.childNodes**  
问题：  
childNodes 的下标的含义在IE和FF中不同，FF的 childNodes 中会插入空白文本节点  
解决方法：  
可以通过 node.getElementsByTagName() 来回避这个问题  
问题：  
当html中节点缺失时，IE和FF对 parentNode 的解释不同，例如：  
<form>  
<table>  
<input/>  
</table>  
</form>  
FF中 input.parentNode 的值为form，而IE中 input.parentNode 的值为空节点  
问题：  
FF中节点自己没有 removeNode 方法  
解决方法：  
必须使用如下方法 node.parentNode.removeChild(node)  
  
**11. const 问题**  
问题：  
在IE中不能使用 const 关键字  
解决方法：  
以 var 代替  
  
**12. body 对象**  
FF的 body 在 body 标签没有被浏览器完全读入之前就存在，而IE则必须在 body 完全被读入之后才存在  
这会产生在IE下，文档没有载入完时，在body上appendChild会出现空白页面的问题  
解决方法：  
一切在body上插入节点的动作，全部在onload后进行  
  
**13. url encoding**  
问题：  
一般FF无法识别js中的&  
解决方法：  
在js中如果书写url就直接写&不要写&  
  
**14. nodeName 和 tagName 问题**  
问题：  
在FF中，所有节点均有 nodeName 值，但 textNode 没有 tagName 值，在IE中，nodeName 的使用有问题  
解决方法：  
使用 tagName，但应检测其是否为空  
  
**15. 元素属性**  
IE下 input.type 属性为只读，但是FF下可以修改  
  
**16. document.getElementsByName() 和document.all[name] 的问题**  
问题：  
在IE中，getElementsByName()、document.all[name] 均不能用来取得 div 元素  
是否还有其它不能取的元素还不知道（这个问题还有争议，还在研究中）  
  
**17. 调用子框架或者其它框架中的元素的问题**  
在IE中，可以用如下方法来取得子元素中的值  
document.getElementById("frameName").(document.)elementName  
window.frames["frameName"].elementName  
在FF中则需要改成如下形式来执行，与IE兼容：  
window.frames["frameName"].contentWindow.document.elementName  
window.frames["frameName"].document.elementName  
  
**18. 对象宽高赋值问题**  
问题：  
FireFox中类似 obj.style.height = imgObj.height 的语句无效  
解决方法：  
统一使用 obj.style.height = imgObj.height + "px";  
**19. innerText的问题**  
问题：  
innerText 在IE中能正常工作，但是innerText 在FireFox中却不行  
解决方法：  
在非IE浏览器中使用textContent代替innerText

**20. event.srcElement和event.toElement问题**  
问题：  
IE下，even对象有srcElement属性，但是没有target属性；Firefox下，even对象有target属性，但是没有srcElement属性  
解决方法：  
var source = e.target || e.srcElement;  
var target = e.relatedTarget || e.toElement;  
  
**21. 禁止选取网页内容**  
问题：  
FF需要用CSS禁止，IE用JS禁止  
解决方法：  
IE: obj.onselectstart = function() {return false;}  
FF: -moz-user-select:none;  
**22. 捕获事件**  
问题：  
FF没有setCapture()、releaseCapture()方法  
解决方法：  
IE:  
obj.setCapture();   
obj.releaseCapture();  
FF:  
window.captureEvents(Event.MOUSEMOVE|Event.MOUSEUP);  
window.releaseEvents(Event.MOUSEMOVE|Event.MOUSEUP);  
if (!window.captureEvents) {  
       o.setCapture();  
}else {  
       window.captureEvents(Event.MOUSEMOVE|Event.MOUSEUP);  
}  
if (!window.captureEvents) {  
       o.releaseCapture();  
}else {  
       window.releaseEvents(Event.MOUSEMOVE|Event.MOUSEUP);  
}