浏览器兼容性  (2011-03-29 17:17:29)

2月份全球浏览器排行：

2月份，谷歌Chrome在全球的市场份额达到了5.61%，比一月份的5.22%增长了0.39个百分点。这意味着到目前为止，Chrome在2010年已经增加了0.96%的市场份额。

其他的浏览器却表现不佳：

IE：市场份额为61.58%，自1月份以来下降了0.54%；

火狐：市场份额为24.23%，自1月份以来下降了0.20%；

苹果Safari：市场份额为4.45%，自1月份以来下降了0.08%；

Opera：市场份额为2.35%，自1月份以来下降了0.03%。

       文章来着新浪：http://tech.sina.com.cn/i/ 2010-03-02/13553894312.shtml

**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<meta http-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

**CSS Hack**

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 1.0 Transitional//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;

       \*color:#0000FF;

       \_color:#66CCCC;

}

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

{ .t1{color:#9900FF}}

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

{

.t1{color:#336600}

}

</style>

</head>

<body>

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

.t1{

       color:#000000; <br>

       \*color:#0000FF; <br>

       \_color:#66CCCC; <br>

}<br>

<br>

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

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

<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 1.0 Transitional//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">

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

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

    <div style="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);

background-repeat: no-repeat;

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

\_background-image: none;

**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<div style="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**<body scroll="**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\_r(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CSS sprites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">**

**<div style="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<div style="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 solid orange;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”> <#div class=”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 1.0 Transitional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 solid orange;text-align:center;zoom:1"> position:relative

<div style="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Global CSS 中,给需要闭合的div加上 **class=”clearfix”** 即可

<style>  
  
.clearfix:after { content:".";display:block;height:0; clear:both; visibility:hidden;  
}  
.clearfix {  
    display:inline-block;  
}  
  
.clearfix {display:block;}  
  
  
</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 solid red;**

**}**

**</style>**

<script type="text/javascript">

function toBreakWord(intLen){

var obj=document.getElementByIdx\_x("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.getElementByIdx\_x  &&  !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="return false" 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\_r("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\_r("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.getElementByIdx\_x("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\_r() 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.getElementByIdx\_x("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);  
}

浏览器兼容性问题综述

——正确的认识浏览器兼容性问题

W3C对标准的推进，Firefox，Chrome，Safari，Opera的出现，结束了IE雄霸天下的日子。  
  
    然而，这对开发者来说，是好事，也是坏事。  
  
    说它是好事，是因为浏览器厂商为了取得更多的市场份额，会促使各浏览器更符合W3C标准，而得到更好的兼容性，并且，不同浏览器的扩展功能(例如 -moz，-webkit开头的样式)，对W3C标准也是个推进；说它是坏事，因为，多个浏览器同时存在，这些浏览器在处理一个相同的页面时，表现有时会有差异。这种差异可能很小，甚至不会被注意到；也可能很大，甚至造成在某个浏览器下无法正常浏览。我们把引起这些差异的问题统称为“浏览器兼容性问题”。而正是这些“浏览器兼容性问题”，无形中给我们的开发增加了不少难度。  
  
    从浏览器内核的角度 来看，浏览器兼容性问题可分为以下三类：  
        1. 渲染相关：和样式相关的问题，即体现在布局效果上的问题。  
        2. 脚本相关：和脚本相关的问题，包括JavaScript和DOM、BOM方面的问题。对于某些浏览器的功能方面的特性，也属于这一类。  
        3. 其他类别：除以上两类问题外的功能性问题，一般是浏览器自身提供的功能，在内核层之上的。  
  
    例如下面的页面，是一个渲染相关的问题：  
[img=https://lh5.googleusercontent.com/NZcN5O4UKSSaV-zCxqv9AKIiZtoJReE1F6\_r60UuKMG1Yd5Labogmbt\_93DCh0NRAn7H-A7M6IhTwzgn0uEeuwjAiqunN6pEjevWKXhhs6z3Oryu][/img]  
    在各个浏览器中都表现的不同，这就属于兼容性问题。  
  
    造成浏览器兼容性问题的**根本原因就是浏览器各浏览器使用了不同的内核，并且它们处理同一件事情的时候思路不同。**  
  
    现今常见的浏览器及其排版引擎（又称渲染引擎）及脚本引擎，如下：  
[img=https://lh3.googleusercontent.com/3c8gVfVOJYaUCyLIXV32bSbMyTLnUWznJSzx6NCsaaDPXFP8XGvFcubkS1xK0NQ9oziSfUIHwbltRzK3F-s0FhANSgcqQIs00mAusat37F59Q-ez][/img]  
  
    而造成浏览器兼容性问题的**常见原因则是设计师写出了不规范的代码，不规范的代码会使不兼容现象更加突出。**  
  
    例如：  
    不规则的嵌套：

XML/HTML code

[?](http://bbs.csdn.net/topics/340167672#clipboardWindow)

|  |  |
| --- | --- |
| 1  2  3  4  5 | <div>     <li>新闻标题一</li>     <li>新闻标题一</li>     <li>新闻标题一</li>  </div> |

DIV 中直接嵌套 LI 元素是不合标准的，LI 应该处于 UL 内。此类问题常见的还有 P 中嵌套 DIV， TABLE等元素。  
  
    不规范的DOM接口和属性设置：

JavaScript code

[?](http://bbs.csdn.net/topics/340167672#clipboardWindow)

|  |  |
| --- | --- |
| 1 | document.all.a\_name.style.top=35; |

上面代码中top的值，其实应该是一个字符串值，需有单位。例如：35px。  
  
    总之，人为的原因也占很大一部分。而人为造成兼容性问题的原因，除了粗心之外，大都源于浏览器bug的存在，和开发者对标准的不了解。  
  
    比如，如果要做一个功能，功能是想让鼠标悬停在 IMG 元素上方时，可以出现提示信息，经常针对 IE 做开发的人，可能会使用 IMG 元素的 “alt” 属性，但其他浏览器中就是不给 ‘alt’ 属性面子。因为 W3C 标准中规定要去做这件事的属性是 ”title“，大多浏览器符合标准，IE 不符合，这是 IE 浏览器内核的问题；开发者不知道 ”title“ 才是正解，不遵循标准去写代码，是开发者的问题。所以，一个问题分两半，浏览器和开发者都有责任。  
  
    既然都有责任，就都有义务去解决兼容性问题。那么，从浏览器的角度来讲，它的厂商应该修复浏览器的bug和不合标准的地方，当某一天 IE 的 ”alt“ 不能用于提示了，还有人用这个错误的属性去显示提示么？ 从开发者角度来讲，多了解标准，了解浏览器兼容性问题，就可以在开发的过程中，有效的避开兼容性问题，让你的页面在所有浏览器中畅通无阻。  
  
    废话少说，下面就讲讲如何有效的避免一些兼容性问题。  
  
    概括来说，避免兼容性问题的有效途径就是：**正确的使用 DTD，书写符合 W3C 规范的代码。**  
  
    那什么叫做 DTD 呢，DTD是做什么的和为什么要写DTD呢？DTD 是  Document Type Definition 的缩写，是文档类型定义。定义上说“此标签可告知浏览器文档使用哪种 HTML 或 XHTML 规范”。  
  
    那其中所说的规范又是什么呢？我们平时所说的规范大部分情况下就指的是 W3C 制定的规范，做WEB前端开发的时候主要接触其中的HTML、CSS、DOM、XML部分。顺便说一下，W3C的网站现在已经提供可校验HTML和CSS 代码的工具了，不知道的同学可以看一下：<http://validator.w3.org/> 这个是校验HTML和XHTML的，<http://jigsaw.w3.org/css-validator/> 这是校验CSS样式表的。  
  
    **DTD** 和 **W3C校验** 及 **浏览器解析** 之间有啥关系？简单地说：W3C校验将检查DTD与文档内容是否匹配；浏览器将根据DTD决定启用何种渲染模式。  
  
    后面半句就是说，写了正确的DTD，浏览器就会按照标准方式（Standards mode）渲染和解析你的页面，那如果不写或者写错呢，那浏览器就会使用怪癖模式（Quirks mode）来渲染和解析你的页面。怪癖模式的产生的主要原因大概是为了向后兼容，就是为了使那些很早之前（IE 6之前的时代）的网页看上去跟原来的样子差不多。  
  
下面有一个直观的例子。  
  
其中，牌子是DTD，水果什么的是文档真实内容，对号和差是W3C代码检测，动植物是浏览器模式。  
  
这是我们写的代码：  
[img=https://lh6.googleusercontent.com/rQJpoZRtB3Gii8vyWafB7IyxiMjpu3xSpaGQutTxe\_t49yDhgsfzfU7tuIqw4q4oZNNSinsQIZMt-nTWkuMoHWe19RcX06I\_Dtw0xDa6tsd8unom][/img]  
W3C校验将检查DTD与文档内容是否匹配，发现有的对不上号，会给出提示：  
[img=https://lh5.googleusercontent.com/oaClgM9\_SZnQfdMAQfDw1OtYsiUsZ1PkF5em9vhGwxlOeuzU0P6n6KKOFZJa-J-hjCbxtXnhGpSZHJNA21SQqgoYUROyz\_FhIROjeYm7lYH2XisV][/img]  
浏览器运行的时候，浏览器将根据DTD决定启用何种渲染模式，不会管你写的正确与否，按照你写的方式渲染：  
[img=https://lh5.googleusercontent.com/j2emEHoC2yhYhcSr42uFGzxWROWT6MSbVTMVcPzRllnNvMRkuVWAnpCeBD8pz-\_2vitoUyfDgs\_BRgvhTNZHRkLrc4nSSlO-cR4kBXbCeegtebzO][/img]  
  
    如何判断一个DTD是写得对不对？或者想知道现在的DTD在各个浏览器下到底会触发哪种渲染模式？来这里<http://hsivonen.iki.fi/doctype/> 核对一下就行。网页的靠下部分有个红红绿绿的表，自己对照一下好了。  
  
    另外补充两点个人经验：  
    1. 正确的使用DTD，并书写符合W3C规范的代码，虽然现在你即使做到这两点也不能保证所有浏览器都会表现一直，但可以最大程度上避免很多兼容性问题。   
    2. 推荐使用 HTML5 的 DTD，好写好记：**<!DOCTYPE html>**，可以触发各浏览器的 Standards Mode。  
  
    但是，问题又来了，这样做一定就能解决问题了吗？ 不一定，但是这样可以避免很多问题。比如，IE的盒模型bug等，都可以通过此手段避免。另外有一些是浏览器的bug，无法解决的。比如，在Firefox中，TABLE 里的绝对定位元素的定位问题，再比如IE早期版本对 ‘!important’ 的支持缺陷等。  
  
    这时候，就要采取一些非凡的手段了。  
  
    一种是在JS里用浏览器判断，针对不同的浏览器做不同的处理；另一种，CSS hack。  
  
    在使用CSS hack 的时候注意，CSS hack一定要是丑陋的，时刻提醒自己，那是不得已而为之的解决方式。  
   
    总而言之，正确的使用DTD、并书写符合W3C规范的代码 + 使用其他手段……，这些都不能完全避免浏览器的不兼容问题。  
  
    浏览器兼容性问题的“解决”需要浏览器开发商、W3C、开发者共同的努力才能实现。  
  
    而作为开发者，我们要做的，就是尽量了解标准，熟悉各浏览器的Bug和特性，灵活设计页面功能，才可以避免出现浏览器兼容性问题。