html部分的兼容问题

2017年10月26日

11:48

* 1. H5阶段标签语义的兼容性
     1. 问题描述 新标签在低版本浏览器中不识别
     2. 相关的标签
        + <header>定义页面或区段的头部；
        + <footer>定义页面或区段的尾部；
        + <nav>定义页面或区段的导航区域；
        + <section>页面的逻辑区域或内容组合；
        + <article>定义正文或一篇完整的内容；
        + <aside>定义补充或相关内容；
     3. 兼容问题浏览器
        + 在ie8及以下不行
     4. 解决方案
        + 原理

<!-- <script type="text/javascript">

document.createElement("nav");

</script> -->

* 1. H5 hack

<!--[if lte IE 8]>

<script type="text/javascript" src="html5shiv.min.js"></script>

<![endif]-->

* 1. img的alt属性 百度的/没接触到
     1. 问题描述 在图片不存在的情况下，各浏览器的解析不一致
     2. 在chrome下显示的是一张破损的图片，在ff下显示的是alt的文字，而在IE中显示的是破损的图片加文字
  2. ul标签内外边距问题
     1. ul标签在IE6\IE7中，有个默认的外边距，但是在IE8以上及其他浏览器中有个默认的内边距

CSS 部分的兼容问题

2017年10月27日

0:28

1. Css3兼容问题

低版本的chrome:-webkit-box-shadow:10px 10px 5px #888;

低版本的firefox:-moz-box-shadow:10px 10px 5px #888;

低版本的ie

1. 不同浏览器标签间的默认外间距内间距不同
   1. 解决方法{margin:0 ; padding:0}
2. css的hack问题：
   1. 主要针对IE的不同版本，不同的浏览器的写法不同
   2. IE的条件注释hack:
      1. <!--[if IE 6]>此处内容只有IE6.0可见<![endif]-->
      2. <!--[if IE 7]>此处内容只有IE7.0可见<![endif]-->

1. IE6双边距问题
   1. IE6在浮动后，又有横向的margin，此时，该元素的外边距是其值的2倍
   2. 解决办法：display:block;
2. IE6下图片的下方有空隙
   1. 给img设置display:block
3. IE6下两个float之间会有个3px的bug
   1. 给右边的元素也设置float:left;
4. IE6下没有min-width的概念，其默认的width就是min-width
5. IE6下在使用margin:0 auto;无法使其居中
   1. 解决办法：为其父容器设置text-align:center;
6. 被点击过后的超链接不再具有hover和active属性
   1. 解决办法:按lvha的顺序书写css样式
7. 在使用绝对定位或者相对定位后，IE中设置z-index失效，
   1. 原因是因为其元素依赖于父元素的z-index，但是父元素默认为0， 子高父低，所以不会改变显示的顺序
8. IE6下无法设置1px的行高，原因是由其默认行高引起的
   1. 解决办法：为期设置overflow:hidden;或者line-height:1px;

来自 <<http://www.qdfuns.com/notes/21128/db10afdde2abe5f0b72d5945358987e3.html>>

javascritp部分兼容问题

2017年10月27日

0:37

* 1. Array.isArray()
     1. *ie9以下浏览器不支持*
  2. 数组indexOf() lastIndexOf()
     1. *ie9以下不支持*
     2. Object.prototype.toString.call(value)
  3. 文本操作
     1. *textContent 标准属性，ie9以下不支持 ;*
     2. *innerText ie提出的属性，具有兼容性问题*
     3. *解决方式*
        1. *内容获取*

*//内容获取函数封装:*

**function** *getText*(element){

**if**(**typeof** element.**innerText**==**"undefined"**){

*//说明不支持,使用textContent*

**return** element.**textContent**;

}**else**{

*//说明支持，使用innerText*

**return** element.**innerText**;

}

}

* 1. 内容设置
     1. 同上

* 1. js中的样式操作
     1. *getComputedStyle(参数1,参数2) ie9以下不支持*
        1. *参数1为属性名,参数2用于选择参数1的伪元素 , 通常为null,*
        2. *返回对象形式*
     2. *标签.currentStyle ie中提供了的属性的访问方式 无法获取伪元素*
     3. *能力检测***typeof** *window*.getComputedStyle==**"function"**
     4. *函数封装：*

**function***getStyle*(element,styleName){

*//styleName="width"*

**if**(**typeof** *window*.getComputedStyle==**"function"**){

*//支持getComputedStyle方法*

**return** getComputedStyle(element,**null**)[styleName];

}**else**{

*//支持*

**return** element.**currentStyle**[styleName]

}

}

* 1. 根据类名获取元素 document.getElementsByClassName("box")
     1. *返回伪数组*
     2. *此方法在ie9以下不支持*
     3. *同时ie9以下也没有提供对应的方式 getByClass*(**"box"**)
     4. 能力检测**typeof** element.getElementsByClassName==**"function"**
     5. 封装函数 web-API-day3
  2. 获取第元素节点
     1. *获取第一个和最后一个子元素节点*
        1. *节点.firstElementChild 节点.astElementChild*
        2. *此属性ie9以下不支持*
        3. *解决:*
           1. *box.children[0]*
           2. *box.children[box.children.length-1]*
     2. *获取相邻元素节点*
        1. *获取上一个同级元素节点 节点.previousElementSibling*
        2. *获取下一个同级元素节点 节点.nextElementSibling*
        3. *ie9以下不支持*
        4. *兼容方案 函数封装*

functiongetNext(item){

//2获取下一个同级节点

var nextSib=item.nextSibling;

//想要继续执行下一次循环，需要满足两个条件:1有节点，2这个节点不是元素节点

while(nextSib && nextSib.nodeType!=1){

//每次从当前取到的这个节点再往后进行获取

nextSib=nextSib.nextSibling;

}

return nextSib;

}

* 1. 标准的事件绑定方法函数为
     1. addEventListener*此方法ie11以下支持*
     2. IE下是 attachEvent；*标签.attachEvent("事件类型",事件处理程序)*
  2. *事件对象e相关*
     1. *事件对象e 在ie低版本中的获取方法为window.event;*
        1. *解决方式* **var** e=e||*window*.**event**;
     2. *target属性在ie低版本不支持，*
        1. *ie低版本中提供了srcElement属性*
        2. **var *target***=***e***.**target**||***e***.**srcElement**;
  3. *获取页面可视区域的尺寸(宽高)* 
     1. *window*.**innerWidth**
     2. *document*.**body**.clientWidth
     3. *document*.**documentElement**.clientWidth
     4. <http://www.cnblogs.com/liujin0505/p/4125289.html>
     5. 补充

*在没有声明DOCTYPE的IE中，浏览器显示窗口大小只能以下获取*

|  |  |
| --- | --- |
| 1  2 | document.body.offsetWidth  document.body.offsetHeight |

*在声明了DOCTYPE的浏览器中，可以用以下来获取浏览器显示窗口大小：*

|  |  |
| --- | --- |
| 1  2 | document.documentElement.clientWidth  document.documentElement.clientHeight |

        IE，FF，Safari皆支持该方法，opera虽支持该属性，但是返回的是页面尺寸；

*同时，除了IE以外的所有浏览器都将此信息保存在window对象中，可以用以下获取：*

|  |  |
| --- | --- |
| 1  2 | window.innerWidth  window.innerHeight |

* 1. 获取页面卷曲的尺寸
     1. *以下三种写法分别具有不同的兼容性问题，没有一个可以支持所有的浏览器*

*document*.**body**.**scrollTop**;

*document*.**documentElement**.**scrollTop**;

*window*.**pageYOffset**;

* 1. 解决

**function***myScroll*(){

**return**{

**scrollTop**:*window*.**pageYOffset**||*document*.**documentElement**.**scrollTop**||*document*.**body**.**scrollTop**,

scrollLeft:*window*.**pageXOffset**||*document*.**documentElement**.scrollLeft||*document*.**body**.scrollLeft

};

}

* 1. *获取鼠标针对页面可视区域的坐标的封装函数,包括能力检测问题*
     1. **检测条件 : typeof** e.**pageX**==**"number"**
     2. **函数**

**function** *getPageX*(e){

**if**(**typeof**e.**pageX**==**"number"**){

**return** e.**pageX**;

}**else**{

**return** e.**clientX**+*myScroll*().**scrollLeft**;

}

}

* 1. 拓展 <http://blog.csdn.net/u013063153/article/details/52677821>
  2. 阻止事件冒泡

***e***.stopPropagation()*//阻止冒泡和捕获 ie9以下不支持window*.**event**.**cancelBubble**=**true**;*//ie低版本使用*

* 1. 事件的捕获方式不一致，标准浏览器是由外至内，而IE是由内到外，但是最后的结果是将IE的标准定为标准
  2. 我们常说的事件处理时的event属性，在标准浏览器其是传入的，IE下由window.event获取的。并且获取目标元素的方法也不同，标准浏览器是event.target，而IE下是event.srcElement
  3. 在低版本的IE中获取的日期处理函数的值不是与1900的差值，但是在高版本的IE中和标准浏览器保持了一致，获取的值也是与1900的差值。

比如：var year= new Date().getYear();

* 1. ajax的实现方式不同，这个我所理解的是获取XMLHttpRequest的不同，IE下是activeXObject
  2. IE中不能操作tr的innerHtml
  3. 获得DOM节点的父节点、子节点的方式不同

其他浏览器：parentNode  parentNode.childNodes

IE：parentElement parentElement.children

来自 <<http://www.qdfuns.com/notes/21128/db10afdde2abe5f0b72d5945358987e3.html>>



[H5项目常见问题汇总及解决方案](http://www.open-open.com/lib/view/open1449325854077.html)

2017年10月27日

20:47

H5项目常见问题及注意事项

Meta基础知识：

* + H5页面窗口自动调整到设备宽度，并禁止用户缩放页面

//一、HTML页面结构  
<meta name="viewport" content="width=device-width,initial-scale=1.0,minimum-scale=1.0,maximum-scale=1.0,user-scalable=no" />  
// width 设置viewport宽度，为一个正整数，或字符串‘device-width’  
// height 设置viewport高度，一般设置了宽度，会自动解析出高度，可以不用设置  
// initial-scale 默认缩放比例，为一个数字，可以带小数  
// minimum-scale 允许用户最小缩放比例，为一个数字，可以带小数  
// maximum-scale 允许用户最大缩放比例，为一个数字，可以带小数  
// user-scalable 是否允许手动缩放

//二、JS动态判断  
var phoneWidth = parseInt(window.screen.width);  
var phoneScale = phoneWidth/640;  
var ua = navigator.userAgent;  
if (/Android (\d+\.\d+)/.test(ua)){  
 var version = parseFloat(RegExp.$1);  
 if(version>2.3){  
 document.write('<meta name="viewport" content="width=640, minimum-scale = '+phoneScale+', maximum-scale = '+phoneScale+', target-densitydpi=device-dpi">');  
 }else{  
 document.write('<meta name="viewport" content="width=640, target-densitydpi=device-dpi">');  
 }  
} else {  
 document.write('<meta name="viewport" content="width=640, user-scalable=no, target-densitydpi=device-dpi">');  
}

空白页基本meta标签

<!-- 设置缩放 -->  
<meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=no, minimal-ui" />  
<!-- 可隐藏地址栏，仅针对IOS的Safari（注：IOS7.0版本以后，safari上已看不到效果） -->  
<meta name="apple-mobile-web-app-capable" content="yes" />  
<!-- 仅针对IOS的Safari顶端状态条的样式（可选default/black/black-translucent ） -->  
<meta name="apple-mobile-web-app-status-bar-style" content="black" />  
<!-- IOS中禁用将数字识别为电话号码/忽略Android平台中对邮箱地址的识别 -->  
<meta name="format-detection"content="telephone=no, email=no" />

其他meta标签

<!-- 启用360浏览器的极速模式(webkit) -->  
<meta name="renderer" content="webkit">  
<!-- 避免IE使用兼容模式 -->  
<meta http-equiv="X-UA-Compatible" content="IE=edge">  
<!-- 针对手持设备优化，主要是针对一些老的不识别viewport的浏览器，比如黑莓 -->  
<meta name="HandheldFriendly" content="true">  
<!-- 微软的老式浏览器 -->  
<meta name="MobileOptimized" content="320">  
<!-- uc强制竖屏 -->  
<meta name="screen-orientation" content="portrait">  
<!-- QQ强制竖屏 -->  
<meta name="x5-orientation" content="portrait">  
<!-- UC强制全屏 -->  
<meta name="full-screen" content="yes">  
<!-- QQ强制全屏 -->  
<meta name="x5-fullscreen" content="true">  
<!-- UC应用模式 -->  
<meta name="browsermode" content="application">  
<!-- QQ应用模式 -->  
<meta name="x5-page-mode" content="app">  
<!-- windows phone 点击无高光 -->  
<meta name="msapplication-tap-highlight" content="no">

常见问题：

* + 移动端如何定义字体font-family

@ --------------------------------------中文字体的英文名称  
@ 宋体 SimSun  
@ 黑体 SimHei  
@ 微信雅黑 Microsoft Yahei  
@ 微软正黑体 Microsoft JhengHei  
@ 新宋体 NSimSun  
@ 新细明体 MingLiU  
@ 细明体 MingLiU  
@ 标楷体 DFKai-SB  
@ 仿宋 FangSong  
@ 楷体 KaiTi  
@ 仿宋\_GB2312 FangSong\_GB2312  
@ 楷体\_GB2312 KaiTi\_GB2312   
@  
@ 说明：中文字体多数使用宋体、雅黑，英文用Helvetica

body { font-family: Microsoft Yahei,SimSun,Helvetica; }

打电话发短信写邮件怎么实现

// 一、打电话  
<a href="tel:0755-10086">打电话给:0755-10086</a>

// 二、发短信，winphone系统无效  
<a href="sms:10086">发短信给: 10086</a>

// 三、写邮件  
//注：在添加这些功能时，第一个功能以"?"开头，后面的以"&"开头  
//1.普通邮件  
<a href="mailto:863139978@qq.com">点击我发邮件</a>  
//2.收件地址后添加?cc=开头，可添加抄送地址（Android存在兼容问题）  
<a href="mailto:863139978@qq.com?cc=zhangqian0406@yeah.net">点击我发邮件</a>  
//3.跟着抄送地址后，写上&bcc=,可添加密件抄送地址（Android存在兼容问题）  
<a href="mailto:863139978@qq.com?cc=zhangqian0406@yeah.net&bcc=384900096@qq.com">点击我发邮件</a>  
//4.包含多个收件人、抄送、密件抄送人，用分号(;)隔开多个邮件人的地址  
<a href="mailto:863139978@qq.com;384900096@qq.com">点击我发邮件</a>  
//5.包含主题，用?subject=  
<a href="mailto:863139978@qq.com?subject=邮件主题">点击我发邮件</a>  
//6.包含内容，用?body=;如内容包含文本，使用%0A给文本换行   
<a href="mailto:863139978@qq.com?body=邮件主题内容%0A腾讯诚信%0A期待您的到来">点击我发邮件</a>  
//7.内容包含链接，含http(s)://等的文本自动转化为链接  
<a href="mailto:863139978@qq.com?body=http://www.baidu.com">点击我发邮件</a>  
//8.内容包含图片（PC不支持）  
<a href="mailto:863139978@qq.com?body=<img src='images/1.jpg' />">点击我发邮件</a>  
//9.完整示例  
<a href="mailto:863139978@qq.com;384900096@qq.com?cc=zhangqian0406@yeah.net&bcc=993233461@qq.com&subject=[邮件主题]&body=腾讯诚邀您参与%0A%0Ahttp://www.baidu.com%0A%0A<img src='images/1.jpg' />">点击我发邮件</a>

移动端touch事件（区分webkit和winphone）

/\* 当用户手指放在移动设备在屏幕上滑动会触发的touch事件 \*/  
// 以下支持webkit  
touchstart——当手指触碰屏幕时候发生。不管当前有多少只手指  
touchmove——当手指在屏幕上滑动时连续触发。通常我们再滑屏页面，会调用event的preventDefault()可以阻止默认情况的发生：阻止页面滚动  
touchend——当手指离开屏幕时触发  
touchcancel——系统停止跟踪触摸时候会触发。例如在触摸过程中突然页面alert()一个提示框，此时会触发该事件，这个事件比较少用

//TouchEvent说明：  
touches：屏幕上所有手指的信息  
targetTouches：手指在目标区域的手指信息  
changedTouches：最近一次触发该事件的手指信息  
touchend时，touches与targetTouches信息会被删除，changedTouches保存的最后一次的信息，最好用于计算手指信息

//参数信息(changedTouches[0])  
clientX、clientY在显示区的坐标  
target：当前元素

//事件响应顺序  
ontouchstart > ontouchmove > ontouchend > onclick

// 以下支持winphone 8  
MSPointerDown——当手指触碰屏幕时候发生。不管当前有多少只手指  
MSPointerMove——当手指在屏幕上滑动时连续触发。通常我们再滑屏页面，会调用css的html{-ms-touch-action: none;}可以阻止默认情况的发生：阻止页面滚动  
MSPointerUp——当手指离开屏幕时触发

移动端click屏幕产生200-300ms的延时响应

说明：移动设备上的web网页是有300ms延迟的，玩玩会造成按钮点击延迟甚至是点击失效。

以下是历史原因，来源一个公司内一个同事的分享：  
2007年苹果发布首款iphone上IOS系统搭载的safari为了将适用于PC端上大屏幕的网页能比较好的展示在手机端上，使用了双击缩放(double tap to zoom)的方案，比如你在手机上用浏览器打开一个PC上的网页，你可能在看到页面内容虽然可以撑满整个屏幕，但是字体、图片都很小看不清，此时可以快速双击屏幕上的某一部分，你就能看清该部分放大后的内容，再次双击后能回到原始状态。

双击缩放是指用手指在屏幕上快速点击两次，iOS 自带的 Safari 浏览器会将网页缩放至原始比例。

原因就出在浏览器需要如何判断快速点击上，当用户在屏幕上单击某一个元素时候，例如跳转链接<a href="#"></a>，此处浏览器会先捕获该次单击，但浏览器不能决定用户是单纯要点击链接还是要双击该部分区域进行缩放操作，所以，捕获第一次单击后，浏览器会先Hold一段时间t，如果在t时间区间里用户未进行下一次点击，则浏览器会做单击跳转链接的处理，如果t时间里用户进行了第二次单击操作，则浏览器会禁止跳转，转而进行对该部分区域页面的缩放操作。那么这个时间区间t有多少呢？在IOS safari下，大概为300毫秒。这就是延迟的由来。造成的后果用户纯粹单击页面，页面需要过一段时间才响应，给用户慢体验感觉，对于web开发者来说是，页面js捕获click事件的回调函数处理，需要300ms后才生效，也就间接导致影响其他业务逻辑的处理。

//解决方案：  
fastclick可以解决在手机上点击事件的300ms延迟  
zepto的touch模块，tap事件也是为了解决在click的延迟问题

Rentina显示屏原理及设计方案

说明：retina屏是一种具备超高像素密度的液晶屏，同样大小的屏幕上显示的像素点由1个变为多个，如在同样带下的屏幕上，苹果设备的retina显示屏中，像素点1个变为4个。  
在高清显示屏中的位图被放大，图片会变得模糊，因此移动端的视觉稿通常会设计为传统PC的2倍。  
那么，前端的应对方案是：设计稿切出来的图片长宽保证为偶数，并使用backgroud-size把图片缩小为原来的1/2

//例如图片宽高为：200px\*200px，那么写法如下  
.css{width:100px;height:100px;background-size:100px 100px;}  
//其它元素的取值为原来的1/2，例如视觉稿40px的字体，使用样式的写法为20px  
.css{font-size:20px}

//image-set设计Rentina背景图  
image-set,webkit私有属性，也是CSS4的属性，为解决Rentina屏幕下的图像而生。  
.css {  
 background: url(images/bg.jpg) no-repeat center;  
 background: -webkit-image-set(  
 url(images/bg.jpg) 1x, //支持image-set普通屏  
 url(images/bg-2x.jpg) 2x); //支持image-set的Rentinan  
}

点击元素产生背景或边框怎么去掉

/ios用户点击一个链接，会出现一个半透明灰色遮罩, 如果想要禁用，可设置-webkit-tap-highlight-color的alpha值为0去除灰色半透明遮罩；  
//android用户点击一个链接，会出现一个边框或者半透明灰色遮罩, 不同生产商定义出来额效果不一样，可设置-webkit-tap-highlight-color的alpha值为0去除部分机器自带的效果；  
//winphone系统,点击标签产生的灰色半透明背景，能通过设置<meta name="msapplication-tap-highlight" content="no">去掉；  
//特殊说明：有些机型去除不了，如小米2。对于按钮类还有个办法，不使用a或者input标签，直接用div标签  
a,button,input,textarea {   
 -webkit-tap-highlight-color: rgba(0,0,0,0);   
 -webkit-user-modify:read-write-plaintext-only; //-webkit-user-modify有个副作用，就是输入法不再能够输入多个字符  
}   
// 也可以   
\* { -webkit-tap-highlight-color: rgba(0,0,0,0); }  
//winphone下  
<meta name="msapplication-tap-highlight" content="no">

美化表单元素

//一、使用appearance改变webkit浏览器的默认外观  
input,select { -webkit-appearance:none; appearance: none; }

//二、winphone下，使用伪元素改变表单元素默认外观  
//1.禁用select默认箭头，::-ms-expand修改表单控件下拉箭头，设置隐藏并使用背景图片来修饰  
select::-ms-expand { display:none; }

//2.禁用radio和checkbox默认样式，::-ms-check修改表单复选框或单选框默认图标，设置隐藏并使用背景图片来修饰  
input[type=radio]::-ms-check,  
input[type=checkbox]::-ms-check { display:none; }

//3.禁用pc端表单输入框默认清除按钮，::-ms-clear修改清除按钮，设置隐藏并使用背景图片来修饰  
input[type=text]::-ms-clear,  
input[type=tel]::-ms-clear,  
input[type=number]::-ms-clear { display:none; }

移动端字体单位font-size选择px还是rem

// 如需适配多种移动设备，建议使用rem。以下为参考值：  
html { font-size: 62.5%; } //10\*16 = 62.5%  
//设置12px字体 这里注意在rem前要加上对应的px值，解决不支持rem的浏览器的兼容问题，做到优雅降级  
body { font-size:12px; font-size:1.2rem; }

超实用的CSS样式

//去掉webkit的滚动条——display: none;  
//其他参数  
::-webkit-scrollba //滚动条整体部分  
::-webkit-scrollbar-thumb //滚动条内的小方块  
::-webkit-scrollbar-track //滚动条轨道  
::-webkit-scrollbar-button //滚动条轨道两端按钮  
::-webkit-scrollbar-track-piece //滚动条中间部分，内置轨道  
::-webkit-scrollbar-corner //边角，两个滚动条交汇处  
::-webkit-resizer //两个滚动条的交汇处上用于通过拖动调整元素大小的小控件

// 禁止长按链接与图片弹出菜单  
a,img { -webkit-touch-callout: none }

// 禁止ios和android用户选中文字  
html,body {-webkit-user-select:none; user-select: none; }

// 改变输入框placeholder的颜色值  
::-webkit-input-placeholder { /\* WebKit browsers \*/  
color: #999; }  
:-moz-placeholder { /\* Mozilla Firefox 4 to 18 \*/  
color: #999; }  
::-moz-placeholder { /\* Mozilla Firefox 19+ \*/  
color: #999; }  
:-ms-input-placeholder { /\* Internet Explorer 10+ \*/  
color: #999; }  
input:focus::-webkit-input-placeholder{ color:#999; }

// android上去掉语音输入按钮  
input::-webkit-input-speech-button {display: none}

// 阻止windows Phone的默认触摸事件  
/\*说明：winphone下默认触摸事件事件使用e.preventDefault是无效的，可通过样式来禁用，如：\*/  
html { -ms-touch-action:none; } //禁止winphone默认触摸事件

* + 取消input在ios下，输入的时候英文首字母的默认大写

<input autocapitalize="off" autocorrect="off" />

* + 手机拍照和上传图片

//IOS有拍照、录像、选取本地图片功能，部分Android只有选择本地图片功能。Winphone不支持  
<input type="file" accept="images/\*" />  
<input type="file" accept="video/\*" />

屏幕旋转的事件和样式

//JS处理  
function orientInit(){  
 var orientChk = document.documentElement.clientWidth > document.documentElement.clientHeight?'landscape':'portrait';  
 if(orientChk =='lapdscape'){  
 //这里是横屏下需要执行的事件  
 }else{  
 //这里是竖屏下需要执行的事件  
 }  
}

orientInit();  
window.addEventListener('onorientationchange' in window?'orientationchange':'resize', function(){  
 setTimeout(orientInit, 100);  
},false)

//CSS处理  
//竖屏时样式  
@media all and (orientation:portrait){ }  
//横屏时样式  
@media all and (orientation:landscape){ }

audio元素和video元素在ios和andriod中无法自动播放

//音频，写法一  
<audio src="music/bg.mp3" autoplay loop controls>你的浏览器还不支持哦</audio>

//音频，写法二  
<audio controls="controls">   
 <source src="music/bg.ogg" type="audio/ogg"></source>  
 <source src="music/bg.mp3" type="audio/mpeg"></source>  
 优先播放音乐bg.ogg，不支持在播放bg.mp3  
</audio>

//JS绑定自动播放（操作window时，播放音乐）  
$(window).one('touchstart', function(){  
 music.play();  
})

//微信下兼容处理  
document.addEventListener("WeixinJSBridgeReady", function () {  
 music.play();  
}, false);

//小结  
//1.audio元素的autoplay属性在IOS及Android上无法使用，在PC端正常  
//2.audio元素没有设置controls时，在IOS及Android会占据空间大小，而在PC端Chrome是不会占据任何空间

重力感应事件

// 运用HTML5的deviceMotion，调用重力感应事件  
if(window.DeviceMotionEvent){  
 document.addEventListener('devicemotion', deviceMotionHandler, false)  
}

var speed = 30;  
var x = y = z = lastX = lastY = lastZ = 0;  
function deviceMotionHandler(eventData){  
 var acceleration = event.accelerationIncludingGravity;  
 x = acceleration.x;  
 y = acceleration.y;   
 z = acceleration.z;  
 if(Math.abs(x-lastX)>speed || Math.abs(y-lastY)>speed || Math.abs(z-lastZ)>speed ){  
 //这里是摇动后要执行的方法   
 yaoAfter();  
 }  
 lastX = x;  
 lastY = y;  
 lastZ = z;  
}

function yaoAfter(){  
 //do something  
}

//说明：说见案例摇一摇效果中yao.js

微信浏览器用户调整字体大小后页面矬了，怎么阻止用户调整

//以下代码可使Android机页面不再受用户字体缩放强制改变大小，但是会有1S左右延时，期间可以考虑loading来处理  
if (typeof(WeixinJSBridge) == "undefined") {  
 document.addEventListener("WeixinJSBridgeReady", function (e) {  
 setTimeout(function(){  
 WeixinJSBridge.invoke('setFontSizeCallback', { 'fontSize':0}, function(res){  
 alert(JSON.stringify(res));  
 })  
 }, 0)  
 });  
}else{   
 setTimeout(function(){  
 WeixinJSBridge.invoke('setFontSizeCallback', { 'fontSize':0}, function(res){  
 alert(JSON.stringify(res));  
 })  
 }, 0)   
}

//IOS下可使用 -webkit-text-size-adjust禁止用户调整字体大小  
body { -webkit-text-size-adjust:100%!important; }

//最好的解决方案：最好使用rem或百分比布局

定位的坑

//fixed定位  
//1.ios下fixed元素容易定位出错，软键盘弹出时，影响fixed元素定位  
//2.android下fixed表现要比iOS更好，软键盘弹出时，不会影响fixed元素定位  
//3.ios4下不支持position:fixed  
//解决方案：使用[Iscroll](http://cubiq.org/iscroll-5)，如：  
<div id="wrapper">  
 <ul>  
 <li></li>  
 .....  
 </ul>  
</div>  
<script src="iscroll.js"></script>  
<script>  
 var myscroll;  
 function loaded(){  
 myscroll=new iScroll("wrapper");  
 }  
 window.addEventListener("DOMContentLoaded",loaded,false);  
</script>

//position定位  
//Android下弹出软键盘弹出时，影响absolute元素定位  
//解决方案:  
var ua = navigator.userAgent.indexOf('Android');  
if(ua>-1){  
 $('.ipt').on('focus', function(){  
 $('.css').css({'visibility':'hidden'})  
 }).on('blur', function(){  
 $('.css').css({'visibility':'visible'})  
 })  
}

播放视频不全屏

<!--  
1.ios7+支持自动播放  
2.支持Airplay的设备（如：音箱、Apple TV)播放  
x-webkit-airplay="true"   
3.播放视频不全屏  
webkit-playsinline="true"   
-->  
<video x-webkit-airplay="true" webkit-playsinline="true" preload="auto" autoplay src="http://"></video>

JS判断设备

function deviceType(){  
 var ua = navigator.userAgent;  
 var agent = ["Android", "iPhone", "SymbianOS", "Windows Phone", "iPad", "iPod"];   
 for(var i=0; i<len,len = agent.length; i++){  
 if(ua.indexOf(agent[i])>0){   
 break;  
 }  
 }  
}  
deviceType();  
window.addEventListener('resize', function(){  
 deviceType();  
})

JS判断微信浏览器

function isWeixin(){  
 var ua = navigator.userAgent.toLowerCase();  
 if(ua.match(/MicroMessenger/i)=='micromessenger'){  
 return true;  
 }else{  
 return false;  
 }  
}

android 2.3 bug

//1.@-webkit-keyframes 需要以0%开始100%结束，0%的百分号不能去掉  
//2.after和before伪类无法使用动画animation  
//3.border-radius不支持%单位，如要兼容，可以给radius设置一下较大的值  
//4.translate百分比的写法和scale在一起会导致失效，例如：  
-webkit-transform: translate(-50%,-50%) scale(-0.5, 1)

android 4.x bug

//1.三星 Galaxy S4中自带浏览器不支持border-radius缩写  
//2.同时设置border-radius和背景色的时候，背景色会溢出到圆角以外部分  
//3.部分手机(如三星)，a链接支持鼠标:visited事件，也就是说链接访问后文字变为紫色  
//4.android无法同时播放多音频audio

消除transition闪屏

.css {  
 -webkit-transform-style: preserve-3d;  
 -webkit-backface-visibility: hidden;  
 -webkit-perspective: 1000;  
}

开启硬件加速

//目前，像Chrome/Filefox/Safari/IE9+以及最新版本Opera都支持硬件加速，当检测到某个DOM元素应用了某些CSS规则时就会自动开启，从而解决页面闪白，保证动画流畅。  
.css {  
 -webkit-transform: translate3d(0,0,0);  
 -moz-transform: translate3d(0,0,0);  
 -ms-transform: translate3d(0,0,0);  
 transform: translate3d(0,0,0);  
}

渲染优化

//1.禁止使用iframe（阻塞父文档onload事件）  
//2.禁止使用gif图片实现loading效果（降低CPU消耗，提升渲染性能）  
//使用CSS3代码代替JS动画；  
//开启GPU加速；  
//使用base64位编码图片(不小图而言，大图不建议使用)  
 // 对于一些小图标，可以使用base64位编码，以减少网络请求。但不建议大图使用，比较耗费CPU。小图标优势在于：  
 //1.减少HTTP请求；  
 //2.避免文件跨域；  
 //3.修改及时生效；

来自 <<http://www.open-open.com/lib/view/open1449325854077.html>>

[H5常见的兼容问题及解决方法](http://www.cnblogs.com/wxtlinlin/p/6427265.html)

2017年10月27日

20:51

最近这两天经常遇到一些麻烦的兼容问题，统一整理一下，比较简单也不是特别全面，希望大家多多交流。

**几种IE6 bug的解决方法**

  1）png24位的图片在iE6浏览器上出现背景，解决方案是做成PNG8.也可以引用一段脚本处理.

  2）IE6双倍边距bug：在该元素中加入display:inline 或 display:block

  3）像素问题 使用多个float和注释引起的 使用dislpay:inline -3px

  4）超链接hover 点击后失效  使用正确的书写顺序 link visited hover active

  5）z-index问题 给父级添加position:relative

  6）Min-height 最小高度 ！Important 解决’ 7.select 在ie6下遮盖 使用iframe嵌套

  7）为什么没有办法定义1px左右的宽度容器（IE6默认的行高造成的，使用over:hidden,zoom:0.08 line-height:1px）

**清除浮动的几种方法以及优缺点**

**1、父级div定义伪类:after和zoom**

<style type="text/css">

 　  .div1{background:#000080;border:1px solid red;}

 　  .div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}

 　  .left{float:left;width:20%;height:200px;background:#DDD}

 　  .right{float:right;width:30%;height:80px;background:#DDD}

   /\*清除浮动代码\*/

 　  .clearfloat:after{display:block;clear:both;content:"";visibility:hidden;height:0}

 　  .clearfloat{zoom:1}

</style>

<div class="div1 clearfloat">

<div class="left">Left</div>

<div class="right">Right</div>

</div>

<div class="div2">

     div2

</div>

原理：IE8以上和非IE浏览器才支持:after，原理和方法2有点类似，zoom(IE转有属性)可解决ie6,ie7浮动问题。

优点：浏览器支持好，不容易出现怪问题（目前：大型网站都有使用，如：腾迅，网易，新浪等等）。

缺点：代码多，不少初学者不理解原理，要两句代码结合使用，才能让主流浏览器都支持。

建议：推荐使用，建议定义公共类，以减少CSS代码。

**2、父级div定义overflow:hidden**

<style type="text/css">

.div1{background:#000080;border:1px solid red;/\*解决代码\*/width:98%;overflow:hidden}

 .div2{background:#800080;border:1px solid red;height:100px;margin-top:10px;width:98%}

.left{float:left;width:20%;height:200px;background:#DDD}

.right{float:right;width:30%;height:80px;background:#DDD}

</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>

</div>

<div class="div2">

div2

</div>

原理：必须定义width或zoom:1，同时不能定义height，使用overflow:hidden时，浏览器会自动检查浮动区域的高度。

优点：简单，代码少，浏览器支持好。

缺点：不能和position配合使用，因为超出的尺寸的会被隐藏。

建议：只推荐没有使用position或对overflow:hidden理解比较深的朋友使用。

**3、结尾处加空div标签clear:both**

<style type="text/css">

   .div1{background:#000080;border:1px solid red}

   .div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}

 　  .left{float:left;width:20%;height:200px;background:#DDD}

   .right{float:right;width:30%;height:80px;background:#DDD}

   /\*清除浮动代码\*/

   .clearfloat{clear:both}

</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>

<div class="clearfloat"></div>

</div>

<div class="div2">

div2

</div>

原理：添加一个空div，利用css提高的clear:both清除浮动，让父级div能自动获取到高度。

优点：简单，代码少，浏览器支持好，不容易出现怪问题。

缺点：不少初学者不理解原理；如果页面浮动布局多，就要增加很多空div，让人感觉很不爽。

建议：不推荐使用，但此方法是以前主要使用的一种清除浮动方法。

**4、父级div定义height**

<style type="text/css">

   .div1{background:#000080;border:1px solid red;/\*解决代码\*/height:200px;}

   .div2{background:#800080;border:1px solid red;height:100px;margin-top:10px}

 　　  .left{float:left;width:20%;height:200px;background:#DDD}

   .right{float:right;width:30%;height:80px;background:#DDD}

</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>

</div>

<div class="div2">

div2

</div>

原理：父级div手动定义height，就解决了父级div无法自动获取到高度的问题。

优点：简单，代码少，容易掌握。

缺点：只适合高度固定的布局，要给出精确的高度，如果高度和父级div不一样时，会产生问题。

建议：不推荐使用，只建议高度固定的布局时使用。

**5、父级div定义overflow:auto**

<style type="text/css">

 　　  .div1{background:#000080;border:1px solid red;

         /\*解决代码\*/width:98%;overflow:auto}

 　.div2{background:#800080; border:1px solid red; height:100px; margin-top:10px;width:98%}

 　　  .left{float:left;width:20%;height:200px;background:#DDD}

   .right{float:right;width:30%;height:80px;background:#DDD}

</style>

<div class="div1">

<div class="left">Left</div>

<div class="right">Right</div>

</div>

<div class="div2">

div2

</div>

原理：必须定义width或zoom:1，同时不能定义height，使用overflow:auto时，浏览器会自动检查浮动区域的高度。

优点：简单，代码少，浏览器支持好。

缺点：内部宽高超过父级div时，会出现滚动条。

建议：不推荐使用，如果你需要出现滚动条或者确保你的代码不会出现滚动条就使用吧。

来自 <<http://www.cnblogs.com/wxtlinlin/p/6427265.html>>

[CSS3新特性，兼容性，兼容方法总结](http://www.cnblogs.com/jesse131/p/5441199.html)

2017年10月27日

21:00