**1、[Doctype作用，标准模式与兼容模式的区别](http://www.cnblogs.com/sunhe/p/4994058.html)**

<!DOCTYPE>声明位于位于HTML文档中的第一行，处于 <html> 标签之前。告知浏览器的解析器用什么文档标准解析这个文档。DOCTYPE不存在或格式不正确会导致文档以兼容模式呈现。

标准模式的排版 和JS运作模式都是以该浏览器支持的最高标准运行。在兼容模式中，页面以宽松的向后兼容的方式显示,模拟老式浏览器的行为以防止站点无法工作。

**2、行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？**

首先：CSS规范规定，每个元素都有display属性，确定该元素的类型，每个元素都有默认的display值，如div的display默认值为“block”，则为“块级”元素；span默认display属性值为“inline”，是“行内”元素。

（1）行内元素有：a b span img input select strong（强调的语气）

（2）块级元素有：div ul ol li dl dt dd h1 h2 h3 h4…p

（3）常见的空元素：

<br> <hr> <img> <input> <link> <meta>

鲜为人知的是：

<area> <base> <col> <command> <embed> <keygen> <param> <source> <track> <wbr>

**3、页面导入样式时，使用link和@import有什么区别？**

（1）link属于XHTML标签，除了加载CSS外，还能用于定义RSS, 定义rel连接属性等作用；而@import是CSS提供的，只能用于加载CSS;

（2）页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载;  
（3）import是CSS2.1 提出的，只在IE5以上才能被识别，而link是XHTML标签，无兼容问题;

**4、介绍一下你对浏览器内核的理解？**

主要分成两部分：渲染引擎(layout engineer或Rendering Engine)和JS引擎。  
渲染引擎：负责取得网页的内容（HTML、XML、图像等等）、整理讯息（例如加入CSS等），以及计算网页的显示方式，然后会输出至显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同，所以渲染的效果也不相同。所有网页浏览器、电子邮件客户端以及其它需要编辑、显示网络内容的应用程序都需要内核。  
JS引擎则：解析和执行javascript来实现网页的动态效果。

最开始渲染引擎和JS引擎并没有区分的很明确，后来JS引擎越来越独立，内核就倾向于只指渲染引擎。

**5、常见的浏览器内核有哪些？**  
Trident内核：IE,MaxThon,TT,The World,360,搜狗浏览器等。[又称MSHTML]  
Gecko内核：Netscape6及以上版本，FF,MozillaSuite/SeaMonkey等  
Presto内核：Opera7及以上。 [Opera内核原为：Presto，现为：Blink;]  
Webkit内核：Safari,Chrome等。 [ Chrome的：Blink（WebKit的分支）]

**6、html5有哪些新特性、移除了那些元素？如何处理HTML5新标签的浏览器兼容问题？如何区分 HTML 和 HTML5？**\* HTML5 现在已经不是 SGML 的子集，主要是关于图像，位置，存储，多任务等功能的增加。  
绘画 canvas;  
用于媒介回放的 video 和 audio 元素;  
本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失;  
sessionStorage 的数据在浏览器关闭后自动删除;  
语意化更好的内容元素，比如 article、footer、header、nav、section;  
表单控件，calendar、date、time、email、url、search;  
新的技术webworker, websockt, Geolocation;

移除的元素：

纯表现的元素：basefont，big，center，font, s，strike，tt，u;  
对可用性产生负面影响的元素：frame，frameset，noframes；

**7、CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？ CSS3新增伪类有那些？**

1.id选择器（ # myid）  
2.类选择器（.myclassname）  
3.标签选择器（div, h1, p）  
4.相邻选择器（h1 + p）  
5.子选择器（ul > li）  
6.后代选择器（li a）  
7.通配符选择器（ \* ）  
8.属性选择器（a[rel = "external"]）  
9.伪类选择器（a: hover, li: nth - child）

\* 可继承的样式： font-size font-family color, UL LI DL DD DT;

\* 不可继承的样式：border padding margin width height ;

\* 优先级就近原则，同权重情况下样式定义最近者为准;

\* 载入样式以最后载入的定位为准;

优先级为:

!important > id > class > tag

important 比 内联优先级高

CSS3新增伪类举例：

p:first-of-type 选择属于其父元素的首个 <p> 元素的每个 <p> 元素。  
p:last-of-type 选择属于其父元素的最后 <p> 元素的每个 <p> 元素。  
p:only-of-type 选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。  
p:only-child 选择属于其父元素的唯一子元素的每个 <p> 元素。  
p:nth-child(2) 选择属于其父元素的第二个子元素的每个 <p> 元素。  
:enabled :disabled 控制表单控件的禁用状态。

checked 单选框或复选框被选中。

**8、列出display的值，说明他们的作用。position的值， relative和absolute定位原点是？**

1.  
block 象块类型元素一样显示。  
none 缺省值。象行内元素类型一样显示。  
inline-block 象行内元素一样显示，但其内容象块类型元素一样显示。  
list-item 象块类型元素一样显示，并添加样式列表标记。

2.  
\*absolute  
生成绝对定位的元素，相对于 static 定位以外的第一个父元素进行定位。

\*fixed （老IE不支持）  
生成绝对定位的元素，相对于浏览器窗口进行定位。

\*relative  
生成相对定位的元素，相对于其正常位置进行定位。

\* static 默认值。没有定位，元素出现在正常的流中  
\*（忽略 top, bottom, left, right z-index 声明）。

\* inherit 规定从父元素继承 position 属性的值。

**9、常见的浏览器兼容问题，原因，及解决方案？**

**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**

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    \*{margin:0;padding:0;}

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，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; 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（IE8 chrome ff safari opera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。

**10、sessionStorage 、localStorage 和 cookie 之间的区别**

共同点：都是保存在浏览器端，且同源的。

区别：

1. cookie数据始终在同源的http请求中携带（即使不需要），即cookie在浏览器和服务器间来回传递。而sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存。cookie数据还有路径（path）的概念，可以限制cookie只属于某个路径下。
2. 存储大小限制也不同，cookie数据不能超过4k，同时因为每次http请求都会携带cookie，所以cookie只适合保存很小的数据，如会话标识。sessionStorage和localStorage 虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。
3. 数据有效期不同，sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数据；cookie只在设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭。
4. 作用域不同，sessionStorage**不在**不同的浏览器窗口中共享，即使是同一个页面；localStorage 在所有同源窗口中都是共享的；cookie也是在所有同源窗口中都是共享的。
5. Web Storage 支持事件通知机制，可以将数据更新的通知发送给监听者。
6. Web Storage 的 api 接口使用更方便。

**11、js中的基本数据类型**

[js](https://www.baidu.com/s?wd=js&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y1ujbYn1u9m1RsP179Pvw90ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHD3rj6YPjmk)中有5种数据类型：Undefined、Null、Boolean、Number和String。  
还有一种复杂的数据类型[Object](https://www.baidu.com/s?wd=Object&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y1ujbYn1u9m1RsP179Pvw90ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHD3rj6YPjmk)，[Object](https://www.baidu.com/s?wd=Object&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1Y1ujbYn1u9m1RsP179Pvw90ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHD3rj6YPjmk" \t "_blank)本质是一组无序的名值对组成的。

**12.Null与Undefined的区别？**

Undefined类型只有一个值，即undefined。当声明的变量还未被初始化时，变量的默认值为undefined。  
Null类型也只有一个值，即null。null用来表示尚未存在的对象，常用来表示函数企图返回一个不存在的对象。

**13、手写一种JavaScript的继承方式**

**原型链继承方式**

function Person(){

this.name = ‘zhangsan’

}

Person.prototype.hello = "hello";

Person.prototype.sayHello = function(){

  alert(this.hello);

}

function Child(){

}

Child.prototype = new Person();//这行的作用是：将Parent中将所有通过prototype追加的属性和方法都追加到Child，从而实现了继承

Child.prototype.world = "world";

Child.prototype.sayWorld = function(){

  alert(this.world);

}

var c = new Child();

c.sayHello();

c.sayWorld();

**14、谈谈对json的理解**

 JSON([JavaScript](http://baike.baidu.com/view/16168.htm) Object Notation) 是一种轻量级的数据交换格式。它基于[ECMAScript](http://baike.baidu.com/view/810176.htm" \t "_blank)的一个子集。 JSON采用完全独立于语言的文本格式，但是也使用了类似于**[C语言](http://lib.csdn.net/base/c" \t "_blank" \o "C语言知识库)**家族的习惯（包括[C](http://baike.baidu.com/subview/10075/6770152.htm" \t "_blank)、C++、[C#](http://baike.baidu.com/view/6590.htm" \t "_blank)、[Java](http://baike.baidu.com/subview/29/12654100.htm" \t "_blank)、**[JavaScript](http://lib.csdn.net/base/javascript" \t "_blank" \o "JavaScript知识库)**、[Perl](http://baike.baidu.com/view/46614.htm" \t "_blank)、[Python](http://baike.baidu.com/view/21087.htm" \t "_blank)等）。这些特性使JSON成为理想的数据交换语言。 易于人阅读和编写，同时也易于机器解析和生成(一般用于提升网络传输速率)。

15、[Ajax 是什么? 如何创建一个Ajax？](http://blog.csdn.net/sinat_35836870/article/details/53256603)

**Ajax是什么**

我理解Ajax 是一种异步请求数据的一种技术，对于改善用户的体验和程序的性能很有帮助。

1.创建Ajax核心对象XMLHttpRequest

**[javascript]** [view plain](http://blog.csdn.net/lxcao/article/details/52745743) [copy](http://blog.csdn.net/lxcao/article/details/52745743)

[在CODE上查看代码片](https://code.csdn.net/snippets/1915044)

1. **var** xmlhttp;
2. **if** (window.XMLHttpRequest)
3. {// 兼容 IE7+, Firefox, Chrome, Opera, Safari
4. xmlhttp=**new** XMLHttpRequest();
5. }
6. **else**
7. {// 兼容 IE6, IE5
8. xmlhttp=newActiveXObject("Microsoft.XMLHTTP");
9. }

2.向服务器发送请求

**[javascript]** [view plain](http://blog.csdn.net/lxcao/article/details/52745743) [copy](http://blog.csdn.net/lxcao/article/details/52745743)

[在CODE上查看代码片](https://code.csdn.net/snippets/1915044)

1. xmlhttp.open(method,url,async);
2. send(string)

**14、如何解决跨域问题**

[**http://www.cnblogs.com/chenshishuo/p/4919224.html**](http://www.cnblogs.com/chenshishuo/p/4919224.html)

**15、谈谈你对前端模块化组件化的理解**

随着前端的不断发展，前端已经不仅仅是一个简单依据需求完成网页开发的过程，

如何提高网页开发效率，提高代码复用率，降低块与块之间的耦合性是我们需要思考的问题，

而前端的组件化、模块化让我们能解决这样的问题。

　　组件化更多关注的UI部分，页面的每个部件，比如头部，内容区，弹出框甚至确认按钮都可以成为一个组件，每个组件有独立的HTML、css、js代码。

可以根据需要把它放在页面的任意部位，也可以和其他组件一起形成新的组件。一个页面是各个组件的结合，可以根据需要进行组装。

　　而模块化侧重的功能的封装，主要是针对Javascript代码，隔离、组织复制的javascript代码，将它封装成一个个具有特定功能的的模块。

模块可以通过传递参数的不同修改这个功能的的相关配置，每个模块都是一个单独的作用域，根据需要调用。

一个模块的实现可以依赖其它模块。

# 16、web应用从服务器主动推送data到客户端有哪些方式

主要就是post和get两种传递方式