面试中遇到的问题

PC端问题：

1. 作PC端项目时，需要考虑哪些兼容性的问题？

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

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。

/\* 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：

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。

Pc端的项目兼容问题主要集中ie6和ie7，而他们的市场份额已经非常低了，而且选择兼容他们会让我们代码变多，维护成本升高，不能使用css2.1或者3的高级命令等等。所以最好不要兼容ie6,7.

而如果要兼容ie8，那么在使用css3的时候一定要小心，因为ie8也是不兼容css3和html5的，所以建议在使用css3的时候要避免使用影响布局的css3属性.

在js中的兼容问题：

1)通过class获取标签：

function byClass(parents,className){

if (parents.getElementsByClassName){

return parents.getElementsByClassName(className);

}

var el=parents.getElementsByTagName("\*");

var arr=[];

var reg=new RegExp("\\b"+className+"\\b");

// console.log(reg);

var arr=[];

for (var i=0; i<el.length; i++){

if (reg.test(el[i].className)){

arr.push(el[i]);

}

}

return arr;

}

2)ready函数

function ready(fn){

if(document.addEventListener)

{

document.addEventListener('DOMContentLoaded', fn, false);

}

else

{

var oS=document.createElement('script');

oS.defer=true;

oS.onreadystatechange=function ()

{

if(oS.readyState=='complete')

{

fn();

}

};

var oHead=document.getElementsByTagName('head')[0];

oHead.appendChild(oS);

}

}

3) 事件绑定

function addEvent(obj,event,fn){

//现代：addEventLister 不用加on

//ie：touchEvent 需要加on

if (window.addEventListener){

obj.addEventListener(event,fn);

}else{

obj.attachEvent("on"+event, fn);

}

}

4）获取样式

function getStyle(el,styleName){

if (el.currentStyle){

return el.currentStyle[styleName];

}else{

return getComputedStyle(el, null)[styleName];

}

}

5）ajax

function ajax(url,succFn,faultFn){

if (window.XMLHttpRequest){

var ajax=new XMLHttpRequest();

}else{

var ajax=new ActiveXObject("Msxml2.XMLHTTP");

}

ajax.open("get",url,true);

ajax.send();

ajax.onreadystatechange=function (){

if (ajax.readyState=="4"){

if (ajax.status>=200&&ajax.status<300||ajax.status==304){

if (succFn){

succFn(ajax.responseText);

}

}else{

if (faultFn){

faultFn(ajax.status);

}

}

}

}

}

1. PC端项目涉及到跨域，那么什么叫跨域？为什么需要跨域？跨域的方法有几种，都是什么原理？
   1. 跨域简单的理解即为当前服务器目录下的前端需求要请求非本服务器(本域名)下的后台服务就称为跨域请求。跨域请求我们使用普通的ajax请求是无法进行的
   2. 由于同源策略，一般来说位于 server1.example.com 的网页无法与不是 server2.example.com的服务器沟通，或者说如果在server1.example.com下想获取server2.example.com的话就得用跨域请求
   3. 跨域的解决方法及解决：http://blog.csdn.net/joyhen/article/details/21631833
2. Ajax的原理
   1. Ajax的原理简单来说通过XmlHttpRequest对象来向服务器发异步请求，从服务器获得数据，然后用javascript来操作DOM而更新页面
   2. XMLHttpRequest是ajax的核心机制，它是在IE5中首先引入的，是一种支持异步请求的技术。简单的说，也就是javascript可以及时向服务器提出请求和处理响应，而不阻塞用户。达到无刷新的效果。
   3. 兼容代码在兼容题目中
3. 什么是HTTP协议，在打开网页时，HTTP协议作了哪些工作，有哪些步骤？
   1. HTTP 是基于 TCP/IP 协议的应用层协议。它不涉及数据包（packet）传输，主要规定了客户端和服务器之间的通信格式，默认使用80端口。
   2. http使得互联网不仅可以传输文字，还能传输图像、视频、二进制文件,其次是规定了许多动词方法(命令)：get,post,head,put,patch,options,delete.
   3. 在http1.1中还采用了分块传输编码的方式进行数据传输，在node中就是stream流
   4. http做的工作是：
      1. 接收客户端的请求，允许的请求命令是：get,post,head,put,patch,options,delete
      2. 对客户端的请求做出回应，可以发送文字、图像、视频、二进制文件
      3. 通过头信息，描述了一些元数据，比如：状态码（status code）、多字符集支持、多部分发送（multi-part type）、权限（authorization）、缓存（cache）、内容编码（content encoding）等，例子：
         1. Content-Encoding: gzip
         2. Content-Encoding: compress
         3. Content-Encoding: deflate
      4. 而且可以将数据压缩后再发送Content-Encoding字段说明数据的压缩方法，在客户端中，我们也可以通过Accept-Encoding字段说明自己可以接受哪些压缩方法：例子：Accept-Encoding: gzip, deflate
      5. 允许持久连接，对于同一个域名，大多数浏览器允许同时建立6个持久连接
   5. 步骤：
      1. 1.打开HTTP连接。一定要记住HTTP是一种无状态协议。正因为如此，对于每一个请求你都要建立一个新的连接。
      2. 3.设置HTTP请求头。这里面包含要传送的数据类型(二进制)和数据的总长。
      3. 4.发送请求。将二进制流写到服务器。
      4. 5.读取请求。目标servlet程序将被调用并接受HTTP请求数据。servlet程序就调用所有必要的参数选择相应的方法。注意，如果这是这个客户端的第一次请求，一个服务器对象的新的实例就会被创建。
      5. 6.调用方法。方法将会被服务器端的对象调用。
      6. 7.初始化方法响应。如果调用的方法抛出一个异常，客户将接收到出错信息。否则，返回的类型(如果有)将会被发送。
      7. 8.设置HTTP响应头。在响应头中，一定会设置待发送数据的类型和长度。
      8. 9.发送响应。二进制数据流将从Web服务器发送并返回给客户端。
4. 后台语言会哪些，到什么程度
   1. node,php
   2. 常见的后台操作，数据库(mongodb,mysql)的基础操作，接口的编写，简易网站后台的编写
5. 对于面向对象的理解
   1. Javascript是一种基于对象的语言，在程序中遇到的所有东西几乎都是对象。但是，它又不是一种真正的面向对象编程（OOP）语言，因为它的语法中没有class类
   2. 在js中使用面向对象可以通过构造函数，所谓"构造函数"，其实就是一个普通函数，但是内部使用了this变量。对构造函数使用new运算符，就能生成实例，并且this变量会绑定在实例对象上
   3. 使编程的注意力重新从应用程序的逻辑回到其数据上来。换句话说，OOP将焦点从过程式编程转向最终建模的真实实体。这使得应用程序更接近我们周围的现实世界。
   4. OOP达到了软件工程的三个目标：重用性、灵活性和扩展性。 采用面向对象方法可以使系统各部分各司其职、各尽所能; 使其编程的代码更简洁、更易于维护，并且具有更强的可重用性
6. 对于Angular.js与node.js的认识，在工作中用过到吗？应用在哪里
   1. CRUD的单页应用
   2. 后台，企业应用网站
   3. 交互密集型的应用
7. 在开发中遇到了哪些问题？，如何解决的？

略

移动端：

1. H5界面的响应式
   1. 就是一个网站能够兼容多个终端——而不是为每个终端做一个特定的版本
   2. 主要使用css3中的media媒体查询，以及百分比布局
2. 移动端的框架都用到过哪些，在作移动端时，都用到过哪些，如何选择的？是否查看过某个框架的源码
   1. bootstrap->响应式样式框架
   2. mui->移动端app开发框架
   3. ionic->使用JavaScript MVVM 框架和 AngularJS 来增强应用
3. 微信端的支付与地图功能
   1. 要使用微信支付功能需要公众号类型为服务号，并获得微信支付权限api
      1. 需要设置测试的目录，在微信支付-》开发配置下，添加白名单
      2. 设置正式支付目录
      3. 在微信浏览器里面打开H5网页中执行JS调起支付。接口输入输出数据格式为JSON
      4. 实例代码：

function onBridgeReady(){

WeixinJSBridge.invoke(

'getBrandWCPayRequest', {

"appId" ： "wx2421b1c4370ec43b", //公众号名称，由商户传入

"timeStamp"：" 1395712654", //时间戳，自1970年以来的秒数

"nonceStr" ： "e61463f8efa94090b1f366cccfbbb444", //随机串

"package" ： "prepay\_id=u802345jgfjsdfgsdg888",

"signType" ： "MD5", //微信签名方式：

"paySign" ： "70EA570631E4BB79628FBCA90534C63FF7FADD89" //微信签名

},

function(res){

if(res.err\_msg == "get\_brand\_wcpay\_request：ok" ) {} // 使用以上方式判断前端返回,微信团队郑重提示：res.err\_msg将在用户支付成功后返回 ok，但并不保证它绝对可靠。

}

);

}

if (typeof WeixinJSBridge == "undefined"){

if( document.addEventListener ){

document.addEventListener('WeixinJSBridgeReady', onBridgeReady, false);

}else if (document.attachEvent){

document.attachEvent('WeixinJSBridgeReady', onBridgeReady);

document.attachEvent('onWeixinJSBridgeReady', onBridgeReady);

}

}else{

onBridgeReady();

}

* 1. 地图功能

使用微信内置地图查看位置接口

wx.openLocation({

latitude: 0, // 纬度，浮点数，范围为90 ~ -90

longitude: 0, // 经度，浮点数，范围为180 ~ -180。

name: '', // 位置名

address: '', // 地址详情说明

scale: 1, // 地图缩放级别,整形值,范围从1~28。默认为最大

infoUrl: '' // 在查看位置界面底部显示的超链接,可点击跳转

});