## 面试题整理

## HTML部分

**HTML**

**HTML5**

**HTML5新特性有有哪些，移除了哪些元素？如何处理HTML5新标签的浏览器兼容性问题？**

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, websocket, 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；

支持HTML5新标签：

IE8/IE7/IE6支持通过document.createElement方法产生的标签，

可以利用这一特性让这些浏览器支持HTML5新标签，

浏览器支持新标签后，还需要添加标签默认的样式。、

当然也可以直接使用成熟的框架、比如html5shim;

<!--[if lt IE 9]>

<script> src="http://html5shim.googlecode.com/svn/trunk/html5.js"</script>

<![endif]-->

## CSS部分

**CSS**

**请解释一下浮动和它的工作原理？清除浮动的技巧？各自的优缺点？并说出他们应用的场景？**  
1、父级div定义height。  
2、结尾处加空div标签clear:both。  
3、父级div定义伪类:after和zoom。

Div:after{

Content:””;

Display:table;

Clear:both;

}  
4、父级div定义overflow:hidden。  
5、父级div定义overflow:auto。  
6、父级div也浮动，需要定义宽度。  
7、父级div定义display:table。  
8、结尾处加br标签clear:both。  
比较好的是第3种方式，好多网站都这么用。

**CSS3**

**如何用CSS定义一个动画？**

**JS部分**

### 原生JS

**请列举出JS对象的几种创建方式？**

javascript创建对象简单的说,无非就是使用内置对象或各种自定义对象，当然还可以用JSON；但写法有很多种，也能混合使用。

1、对象字面量的方式

var person={firstname:"Mark",lastname:"Yun",age:25,eyecolor:"black"};

2、用function来模拟无参的构造函数

function Person(){}

var person=new Person();//定义一个function，如果使用new"实例化",该function可以看作是一个Class

person.name="Mark";

person.age="25";

person.work=function(){

alert(person.name+" hello...");

}

person.work();

3、用function来模拟参构造函数来实现（用this关键字定义构造的上下文属性）

function Pet(name,age,hobby){

this.name=name;//this作用域：当前对象

this.age=age;

this.hobby=hobby;

this.eat=function(){

alert("我叫"+this.name+",我喜欢"+this.hobby+",是个程序员");

}

}

var maidou =new Pet("麦兜",25,"coding");//实例化、创建对象

maidou.eat();//调用eat方法

4、用工厂方式来创建（内置对象）

var wcDog =new Object();

wcDog.name="旺财";

wcDog.age=3;

wcDog.work=function(){

alert("我是"+wcDog.name+",汪汪汪......");

}

wcDog.work();

5、用原型方式来创建

function Dog(){

}

Dog.prototype.name="旺财";

Dog.prototype.eat=function(){

alert(this.name+"是个吃货");

}

var wangcai =new Dog();

wangcai.eat();

5、用混合方式来创建

function Car(name,price){

this.name=name;

this.price=price;

}

Car.prototype.sell=function(){

alert("我是"+this.name+"，我现在卖"+this.price+"万元");

}

var camry =new Car("凯美瑞",27);

camry.sell();

**请写出一个验证邮箱的正则表达式？**

/^([a-z0-9\_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/  
/^[a-z\d]+(\.[a-z\d]+)\*@([\da-z](-[\da-z])?)+(\.{1,2}[a-z]+)+$/

### Jquery

### Bootstrap

### Angular

### React

### Nodejs

**说说你都知道nodejs的哪些核心模块？**

1. Stdio

Console模块是Node提供的核心模块，提供基本的输出功能；

1. URL与Query String

两个模块是Node提供的核心模块，用于处理URL及URL中的查询串。通常情况下处理querystring，因为url模块的支持

### 模块化

## 其他部分

**GIT、SVN**

**浏览器兼容性**

**你都遇见过哪些常见的浏览器兼容性的问题？**

1. 不同浏览器的标签默认的padding和margin不同

在CSS样式中用通配符\*来设置各个标签的padding和margin为0

1. 块属性标签float后，又设置了行内的margin情况下，IE6显示margin比设置大

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

1. 图片默认之间有间距

使用float属性为img布局

**性能优化**

**你如何对网站的文件和资源进行优化？**

**1、Minimize HTTP Requests 减少HTTP请求**

图片、css、script、flash等等这些都会增加http请求数，减少这些元素的数量就能减少响应时间。把多个JS、CSS在可能的情况下写进一个文件，页面里直接写入图片也是不好的做法，应该写进CSS里，利用 CSS sprites 将小图拼合后利用background来定位。

**2、Use a Content Delivery Network 利用CDN技术**

CDN的全称是Content Delivery Network，即**内容分发网络**。其基本思路是尽可能避开互联网上有可能影响数据传输速度和稳定性的瓶颈和环节，使内容传输的更快、更稳定。通过在网络各处放置节点服务器所构成的在现有的互联网基础之上的一层智能虚拟网络，CDN系统能够实时地根据网络流量和各节点的连接、负载状况以及到用户的距离和响应时间等综合信息将用户的请求重新导向离用户最近的服务节点上。其目的是使用户可就近取得所需内容，解决 Internet网络拥挤的状况，提高用户访问网站的响应速度。

**3、Add an Expires or a Cache-Control Header 设置头文件过期或者静态缓存**

浏览器会用缓存来减少http请求数来加快页面加载的时间，如果页面头部加一个很长的过期时间，浏览器就会一直缓存页面里的元素。不过这样如果 页面里的东西变动的话就要改名字了，否则用户端不会主动刷新，看自己衡量了~ 这项可以通过修改.htaccess文件来实现。

**4、Gzip Components Gzip压缩**

Gzip格式是一种很普遍的压缩技术，几乎所有的浏览器都有解压Gzip格式的能力，而且它可以压缩的比例非常大，一般压缩率为85%。压缩没压缩，可以到 这里 做下测试。

**5、Put Stylesheets at the Top 把CSS放顶部**

让浏览者能尽早的看到网站的完整样式。

**6、Put Scripts at the Bottom 把JS放底部**

网站呈现完毕后再进行功能设置，当然这些JS要在你的加载过程中不影响内容表现。

**7、Reduce DNS Lookups 减少DNS查找**

貌似是要减少网站从外部调用资源。

**8、Minify JavaScript and CSS 减小JS和CSS的体积**

写JS和CSS都是有技巧的，用最少的代码实现同样的功能，减少空白，增强逻辑性，用缩写方式等等，当然也有不少工具也能够帮你实现这一点。

**GET和POST的区别，如何使用POST？**

1. Get是从服务器上获取数据，POST是向服务器传送数据；
2. Get是把参数数据队列加到URL中，值和表单内数据一一对应，在URL中可以看到，

Post是将表单各字段与其内容放置在HTML HEADER内一起传送到指定的URL地址，用户看不到这个过程；

1. Get传送的数据较小，不能大于2kb。Post方式，一般被默认为不受限制；
2. Get安全性较低，Post安全性高，但执行效率Get更高。

**如何解决跨域问题？**

**写一个通用的事件监听函数？**

**JS有哪几种数据类型？**

原始类型和引用类型（2大类）

原始数据类型：值直接保存在变量本地的数据类型

包括：undefined null string number boolean

引用类型：值无法直接在变量本地保存的数据

包括：Object Array Function等

**AJAX是什么？AJAX的交互模型？同步和异步的区别？如何解决跨域问题？**

ajax的全称：Asynchronous Javascript And XML。

异步传输+js+xml：是一种创建交互式网页应用的技术，可以实现页面数据的异步请求和局部加载，达到不用刷新页面也可显示内容的技术，提高可用户的体验。

AJAX用到的技术：

1. 基于web标准XHTML+CSS表示；
2. 使用DOM进行动态显示及交互
3. 使用XML和XSLT进行数据交换及相关操作；
4. 使用XMLHttpRequest进行异步数据查询、检索

AJAX的交互模型：

其交互模型是，ajax在浏览器端引入一个之行引擎，它一边接受user的请求，一边传送数据给服务器，并把服务器返回的结果展示给user

同步：脚本会停留并等待服务器发送请求然后得到回复才继续进行

异步：脚本不必停留并处理可能的回复

**this对象的理解**  
this总是指向函数的直接调用者（而非间接调用者）；  
如果有new关键字，this指向new出来的那个对象；  
在事件中，this指向触发这个事件的对象，特殊的是，IE中的attachEvent中的this总是指向全局对象Window。

**解释jsonp的原理，以及为什么不是真正的ajax？**

**请使用bootstrap的栅格系统写一个简单的布局，要求左边占总宽度的40%右边占总宽度的60%？**

**什么是MVC，简单说说你对MVC的理解，为什么使用MVC？**

MVC是一种设计模式，包括三类对象。

1. 模型（Model）

模型是应用的主体部分，模型表示业务数据，或者业务逻辑

1. 视图（View）

视图是应用的中用户体验交互的主要界面，数据呈现的界面

1. 控制器（Controller）

控制器主要负责获取模型的数据，选择视图加以呈现

降低代码的耦合性，将后端和前台分离开来，是使代码更易维护

**写一个简单的渐变效果，颜色从#FFF到#000透明度0.3？**

**如何一张图片正常是配屏幕，要求所有屏幕中都能满屏显示？**

img{

width:100%;

height:auto;

}

**用Jquery怎么给一个页面对象添加和删除class？**

$(“选择器”).addClass(“class名”); $(“选择器”).removeClass(“class名”);

**用Jquery怎么隐藏和显示页面对象？**

$(“选择器”).show(); $(“选择器”).hide();

**一个id为a的对象，怎么用Jquery找到它的父对象的上一个兄弟对象的第一个子对象？**

$(“#a”).parent().prev().first();

**一个select对象，怎么删除它的所有option并重新添加一个值为a,显示为test的option？**

$(“select”).html(“<option value=’a’>test</option>”);

**对WEB标准以及W3C的理解和认识？**

**XHTML和HTML有什么区别？**

HTML是一种基本的web网页标记语言，XHTML是XML向HTML的过渡，在语法上XHTML更为严格；

XHTML要求正确嵌套

XHTML所有元素必须关闭

XHTML区分大小写

XHTML属性值要用双引号

XHTML用id属性代替name属性

XHTML特殊字符的处理

**CSS引入方式有哪些？Link和@import的区别是？**

外部样式表，内部样式表，内联样式

Link和@import都是外部样式标引入方式

Link方式：

<link rel=”stylesheet” type=”text/css” href=”XX.css”>

@import方式：

<style>

@import “XX.css”;

</style>

两种方式的主要区别：

1. link属于XHTML标签，除了加载CSS外，还能用于定义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;

**CSS选择器有哪些？哪些属性可以继承？CSS新增伪类有哪些？优先级算发如何计算？内联和important哪个优先级高？**

通用选择器、元素选择器、类选择器、ID选择器、群组选择器、后代选择器、子代选择器、伪类选择器

可以继承的属性有：font-size font-family color

不可以继承的：border padding margin background-color width height等

新增伪类：

目标伪类： :target

元素状态伪类： :enabled :disabled :checked

结构伪类： :first-child :last-child :empty :only-child

否定伪类： :not(selector) 匹配非指定元素

优先级算法： 选择器类型 权值

元素选择器 0001

类选择器 0010

伪类选择器 0010

ID选择器 0100

内联选择器 1000

注意：选择器权值相加比较大小，权值相同时，采用就近原则。

相对于内联样式 !important的优先级更高

**前端页面有哪三层结构，分别是什么？作用是什么？**

分成：结构层、表示层、行为层

结构层：

由HTML或XHTML之类的标记语言负责创建。负责标记页面内容，提供页面内容的基本框架；

表示层：

由CSS负责创建。负责为页面内容提供样式布局，渲染页面元素；

行为层：

由Javascript创建。负责对页面元素进行操，DOM、BOM以及数据交互。

**CSS的基本语句构成是？**

选择器{  
 属性名：属性值；  
}

**你做的网页在那些浏览器上测试过？这些浏览器的内核分别是什么？**

IE Firefox Safari Chrome Opera

浏览器的内核负责对网页语法的解释并显示网页，它决定了浏览器如何显示网页的内容以及页面的格式信息。

IE内核为：trident

Firefox内核为：Gecko

Safari内核为：Webkit

Chrome内核为：Blink

Opera内核为：Presto 2013年后也用webkit

**标签上title与alt属性的区别是什么？**

Alt主要是为不能显示图像、窗体或applets的用户代理，用来指定文字替换，主要用于图片无法显示时的替代文字；

Title属性为改元素属性的元素提供解释性文字，常用于对图片信息的文字性补充

**描述CSS reset的作用和用途？**

CSS reset,又叫CSS重写或者CSS重置，用于改写HTML标签的默认样式。

因为浏览器的兼容问题，不同浏览器对有些标签的默认值是不同的，如果没对CSS初始化往往会出现浏览器之间的页面显示差异。

最简单的初始化方法： \* {padding: 0; margin: 0;} （强烈不建议）

**什么是HTML语义化？、**

直观的认识标签和属性的用途和作用，可概括为：用正确的标签做正确的事情。

html语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析,提高代码的可维护和可重用性;

比如，减少无使用意义的div标签，使用结构化标签<header> <section> <footer>

**JS的typeof返回哪些数据类型？**

Undefined类型。当声明变量未初始化时，该变量的默认值是undefined

Boolean

Number

String

Object

**列举三种强制类型转换和两种隐式类型转换？**

**Split（）和join（）的区别？**

**事件绑定和普通事件有什么区别？**

**IE和DOM事件流的区别？**

**IE和标准下有哪些兼容性的写法？**

**Call和apply的区别？**  
apply()函数有两个参数：第一个参数是上下文，第二个参数是参数组成的数组。如果上下文是null，则使用全局对象代替。  
如：function.apply(this,[1,2,3]);  
call()的第一个参数是上下文，后续是实例传入的参数序列。  
如：function.call(this,1,2,3);

**AJAX请求时，如何解释json数据？**

JS语言中：

JSON字符串 => JS对象/数组： var obj = JSON.parse( str )

JS对象/数组 => JSON字符串： var str = JSON.stringify( obj )

PHP语言中：

PHP数组 => JSON字符串： $str = json\_encode( $arr );

JSON字符串 => PHP数组： $arr = json\_decode( $str );

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。它是基于JavaScript的一个子集。数据格式简单, 易于读写, 占用带宽小。  
  
格式：采用键值对，例如：{'age':'12', 'name':'back'}

**什么是JS的本地对象、内置对象和宿主对象？**

本地对象为array、object等可以new实例化

内置对象为gload Math 等不可以实例化的

宿主对象为浏览器自带的document、window等

**Document.load和Document.ready的去别？**

Document.onload()是在页面所有关联资源（包括图像、音频）加载完毕才会调用，document.ready()是在页面DOM元素加载完成后调用，且也面中可以使用多个document.ready()，而document.on;load()只能使用一次。

**编写一个数组去重的方法？**

var arr=[1,5,9,6,4,6,8,4,5,7];

Function quChong(arr){

for(var i=0 hash={};i<arr.length;i++){

if(hash[arr[i]]==undefined){

hash[arr[i]]=1;

}

}

var num=0;

for(arr[num++] in hash){};

return arr;

}

Var arr=quChong(arr);

**display：none和visibility:hidden 的区别？**

Display:none属性后，HTML元素（对象）的宽度、高度等各种属性都将”丢失”，不会占据页面空间；而使用visibility：hidden属性后，HTML元素（对象）仅仅是在视觉上看不见，而它所占据的空间仍然存在，也就是它仍然具有高度、宽度等属性值。

**Overflow-x是CSS2还是CSS3的属性？**

CSS3

**请缩写一下代码：**

**.box{**

**background-position:10px 20px;**

**background-repeat:no-repeat;**

**background-attachment:fixed;**

**background-color:red;**

**background-image:url(box.png);**

**}**

.box{

background:red url(box.png) no-repeat fixed 10px 20px ;

//background：color url() repeat attachment position;

}

**如何让一段文本中的所有英文单词的首字母大写？**

**如何居中一个浮动的元素？**

1、父级div设置固定宽度后 设置样式margin：0 auto; 然后子级div添加浮动

2、在table中插入div,div设置浮动 而在table样式设置margin：0 auto 不用设置table的宽度

行内元素水平居中 那么在div中设置样式text-align:center;

块级元素水平居中 那么设置元素样式margin：0 auto；width：100px；

行内元素垂直居中 那么需要设置div的高和行高

块级元素垂直居中 结合块级元素水平居中，设置margin-top的相应数值

**写出你目前所了解的主流浏览器（ie6,ie7,ie8,ie9,firefox,chrome）的CSS hash？**

**JS中表单中弹出对话框函数是？获得输入焦点函数是？**

**列举一下常用JS字符串操作和数组函数，以及它们的功能？**

**JS中如何定义class,如何扩展prototype？**

Element.className = “XXX”; //在css中定义，形式如下：.XXX {属性名:属性值;}   
A.prototype.B = C;   
A是某个构造函数的名字   
B是这个构造函数的属性（方法）  
C是想要定义的属性的值（方法）

**Jquery中如何获取和设置CSS样式？**

获取：$(“选择器”).attr(“样式名”);

设置：$(“选择器”).attr(“样式名”,”样式值”);

$(“选择器”).attr({样式名:”样式值”,样式名:”样式值”})

**请用Jquery为所有的input添加属性class=”btn”;**

**Jquery中如何使用AJAX？**

$.ajax({

url:””; //请求地址

type:””; //发送方式

data:{}; //发送数据

dataType:”json”; //返回数据

success:fun, //请求成功的回调函数

error:fun //请求失败的回调函数

});

**怎样添加、移除、复制、创建和查找节点？**

1、创建新节点  
createDocumentFragment() //创建一个DOM片段  
createElement() //创建一个具体的元素  
createTextNode() //创建一个文本节点  
  
2、添加、移除、替换、插入  
appendChild()  
removeChild()  
replaceChild()  
insertBefore() //在已有的子节点前插入一个新的子节点  
  
3、查找  
getElementsByTagName() //通过标签名称  
getElementsByName() //通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)  
getElementById() //通过元素Id，唯一性

**分别写出JS中如何显示/隐藏一个DOM元素？**

设置dom元素的样式 display:none 隐藏 display：inline

设置dom元素的掩饰 visibility:hidden 隐藏 visibility:visible;隐藏

**添加事件的几种方法？以及IE和DOM事件模型之间存在哪些主要区别？**

1、直接为事件处理函数赋值一个函数对象:

elem.onXXX=function(){

//this->elem

}

事件触发时: elem.onXXX();

问题: 1. 一个事件处理函数，只能绑定一个函数对象

2. 无法删除，只能替换。

原理: 每个事件处理函数的绑定都是一个底层对象

绑定的事件处理函数越多，对象越多，直接影响网页执行效率。

事件绑定对象即使在删除元素后，依然存在

建议: 在删除一个带事件处理函数的元素前，先删除其绑定的事件处理函数。

可用elem.onXXX=null，模拟删除绑定的事件

2、使用API绑定处理函数:

elem.addEventListener("事件名",函数对象,false/true);

"事件名": 事件处理函数名去掉on的部分

函数对象: 不带()

第三个参数: 是否在捕获阶段触发?

elem.removeEventListener("事件名",函数对象,false/true);

优点: 1. 可为一个事件处理函数，绑定多个任务函数对象

2. 可使用removexxx移除任意一个绑定的函数对象

1. \*\*\*事件周期:

DOM: 3个阶段:

1. 捕获: 由最外层元素向内，逐级记录绑定的事件处理函数

只是记录，而不触发

2. 目标触发: 首先触发实际发生事件的元素上绑定的处理函数

目标元素: 实际发生事件的元素

3. 冒泡: 由内层向外层依次执行各级绑定的处理函数

鄙视题:能否修改事件触发的顺序: ——能！

elem.addEventListener("事件名",函数对象,false/true);

第三个参数: 是否在捕获阶段提前触发，

默认false，只能在冒泡阶段顺序触发

改为true，可在捕获阶段提前触发！

在捕获阶段提前触发的处理函数，冒泡阶段不会重复触发。

IE8:( 2个阶段)没有捕获！——不能修改事件触发的顺序

elem.attachEvent("on事件名",函数对象)

2. 事件对象event: 事件发生时自动创建的，封装事件信息的对象

获得事件对象:

1. 在html中绑定: <elem on事件名="fun(this,event)"

function(elem,e){

其中，e得到的就是事件对象...

}

没有兼容性问题

2. 在js中绑定时:

DOM: 事件对象作为处理函数的第一个参数传入函数对象

1. elem.on事件名=function(e){

arguments[0]->event

e->event

}

2. elem.addEventListener("事件名",function(e){

e->event

});

IE8: 将事件对象保存为一个全局变量event

window.event;

兼容: e=e||window.event;

如果e有效，就使用e，否则使用window.event

已经用过的: e.keyCode e.screenX/screenY

取消冒泡: 一般用于事件处理函数的末尾

DOM: e.stopPropagation();

IE8: e.cancelBubble=true;

兼容: if(e.stopPropagation!==undefined){

e.stopPropagation();

}else{

e.cancelBubble=true;

}

**利用冒泡:**

优化: 如果多个平级子元素都绑定了相同的事件处理函数，则只需要在父元素上绑定

一次即可,不必反复绑定！

为什么: 绑定的事件处理函数越多，页面的执行效率越低。

应尽量减少页面中绑定的事件处理函数个数。

核心问题：如何获得目标元素:

var target=e.target||e.srcElement

DOM IE8

取消事件: 在事件处理函数执行过程中，发生了异常或错误，希望事件不再继续触发。

如何取消:

1. js动态绑定事件处理函数:

DOM: e.preventDefault();

IE8: e.returnValue=false;

兼容: if(e.preventDefault!==undefined){

e.preventDefault();

}else{

e.returnValue=false;

}

**面向对象编程：b怎么继承a？**

Object.setPrototypeOf(子对象，父对象);

**什么是闭包，谈谈你对闭包的理解？**

闭包是指有权访问另一个函数作用域中变量的函数，创建闭包的最常见的方式就是在一个函数内创建另一个函数，通过另一个函数访问这个函数的局部变量,利用闭包可以突破作用链域，将函数内部的变量和方法传递到外部。

闭包的特性：

1.函数内再嵌套函数

2.内部函数可以引用外层的参数和变量

3.参数和变量不会被垃圾回收机制回收，导致内存泄漏

**HTTP状态码知道哪些？**  
100 Continue 继续，一般在发送post请求时，已发送了http header之后服务端将返回此信息，表示确认，之后发送具体参数信息  
200 OK 正常返回信息  
201 Created 请求成功并且服务器创建了新的资源  
202 Accepted 服务器已接受请求，但尚未处理  
301 Moved Permanently 请求的网页已永久移动到新位置。  
302 Found 临时性重定向。  
303 See Other 临时性重定向，且总是使用 GET 请求新的 URI。  
304 Not Modified 自从上次请求后，请求的网页未修改过。  
400 Bad Request 服务器无法理解请求的格式，客户端不应当尝试再次使用相同的内容发起请求。  
401 Unauthorized 请求未授权。  
403 Forbidden 禁止访问。  
404 Not Found 找不到如何与 URI 相匹配的资源。  
500 Internal Server Error 最常见的服务器端错误。  
503 Service Unavailable 服务器端暂时无法处理请求（可能是过载或维护）。

**JS浏览器兼容性的写法：**  
一、元素查找问题：  
1. document.all[name]

(1)现有问题：Firefox不支持document.all[name]

(2)解决方法：使用getElementsByName(name)，getElementById(id)等来替代。  
  
2. 集合类对象问题  
  (1)现有问题：IE中对许多集合类对象取用时可以用 ()，但在Firefox只能用[]。  
      如：IE中可以使用document.forms("formName")来返回名字为"formName"的Form，但 在Firefox却行不通。  
  (2)解决方法：使用[]，上例中可以改为document.forms["formName"]

3. HTML元素的ID在JavaScript可见  
  (1)现有问题：IE中HTML元素中的ID可以作为document的下属对象变量名直接使用。在 Firefox中不能。  
  (2)解决方法：使用getElementById("idName")代替idName作为对象变量使用。  
  
4. eval(idName)取得对象  
  (1)现有问题：在IE中，利用eval(idName)可以取得ID为idName的HTML对象，在Firefox中不能。  
  (2)解决方法：用 getElementById(idName) 代替 eval(idName)。  
    
5. 变量名与某HTML对象ID相同  
  (1)现有问题：在Firefox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。  
  (2)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。此外，最好不要取与HTML对象id相同的变量名，以减少错误。  
  
  注：3、4和5都属于同一类的问题。  
  
6. Frame  
  (1)现有问题：在IE中可以用window.top.frameId和window.top.frameName来得到该Frame所代表的Window，Firefox中只能用window.top.frameName。  
  (2)解决方法：将Frame的Id和Name设置成相同，使用window.top.frameName来访问Frame。  
  
二、DOM操作  
1. 设置元素的文本内容。  
  (1)现有问题：IE使用innerText，而Firefox使用textContent来设置元素文本内容。  
  (2)解决方法：如果文本内容不包含"<"和">"等特殊字符，可以使用innerHTML。否则，可以使用：  
        var child = elem.firstChild;  
            if (child != null) elem.removeChild(child);  
            elem.appendChild(document.createTextNode(content));  
  
2. parentElement，parent.children  
  (1)现有问题：IE可以使用parentElement获得父结点，parent.children得到结点的所有孩子结点。Firefox不支持。  
  (2)解决方法：使用parentNode和parent.childNodes。  
  
3. 对childNodes的解释。  
  (1)现有问题：IE和Firefox中对childNodes的解释不同，IE不会包含空白文本结点，而Firefox会包含。  
  (2)解决方法：使用childNodes过滤文本结点，如下：  
      var children = elem.childNodes;  
          for (i = 0; i < children.length; i++) {  
            if (children[i].nodeType != 3) { // 过滤文本结点  
              // ...  
            }  
          }  
  
4. 对document.getElementsByName的解释。  
  (1)现有问题：IE中getElementsByName只会检查<input>和<img>元素，而在Firefox下会检查所有元素。  
  (2)解决方法：不要使用getElementsByName检查除<input>和<img>之外的元素，如果要获得单个元素，尽量使用getElementById。  
  
5. 对document.getElementById的解释。  
  (1)现有问题：IE中getElementById不仅检查Id属性，也会检查Name属性，当Name属性匹配参数时也会返回该元素。而在Firefox中只会检查Id属性。  
  (2)解决方法：尽量保持Id和Name相同，不要让一个元素name属性和另一个元素的id属性相同。  
  
  
三、事件  
1. event.x与event.y问题  
  (1)现有问题：在IE中，event对象有x,y 属性，Firefox中没有。  
  (2)解决方法：在Firefox中，与event.x 等效的是 event.pageX。可以使用：  
      mX = event.x ? event.x : event.pageX;  
        
2. window.event  
  (1)现有问题：使用window.event无法在Firefox上运行  
  (2)解决方法：  
         原代码(可在IE中运行)：  
            <input type="button" name="someButton" value="提交" onclick="javascript:gotoSubmit()"/>  
            ...  
            <script language="javascript">  
                function gotoSubmit() {  
                    ...  
                    alert(window.event);    // use window.event  
                    ...  
                }  
            </script>  
  
        新代码(可在IE和Firefox中运行)：  
            <input type="button" name="someButton" value="提交" onclick="javascript:gotoSubmit(event)"/>  
            ...  
            <script language="javascript">  
                function gotoSubmit(evt) {  
                    evt = evt ? evt : (window.event ? window.event : null);  
                    ...  
                    alert(evt);             // use evt  
                    ...  
                }  
            </script>  
  
3. attachEvent和addEventListener  
  (1)现有问题：IE中使用attachEvent来添加事件，Firefox中使用addEventListener。  
  (2)解决方法：如下，注意事件参数的区别，一个是click，一个是onclick。  
        if (document.attachEvent) document.attachEvent("click", clickHandler,false);  
        else document.addEventListener("onclick",clickHandler);  
  
四、语法  
1. const  
  (1)现有问题：在IE中不能使用const关键字。如const constVar = 32;在IE中这是语法错误。  
  (2)解决方法：不使用const，以var代替。  
  
2. 多余的逗号  
  (1)现有问题：firefox中对象文字常量容许多余的逗号，在IE中不允许。下面语句在IE中非法。  
      var obj = { 'key' : 'aaa', }  
  (2)解决方法：去掉多余逗号。  
    
五、XML  
1. 创建XMLHttpRequest  
  (1)现有问题：Firefox使用XMLHttpRequest，IE使用ActiveXObject。  
  (2)解决方法：  
      if (window.XMLHttpRequest) {  
          req = new XMLHttpRequest();  
      } else if (window.ActiveXObject) {  
          req = new ActiveXObject("Microsoft.XMLHTTP");  
      }  
  
2. 创建DOM  
  (1)现有问题：Firefox和IE创建DOM的方式不同。  
  (2)解决方法：  
        function createXmlDom() {  
          var oXmlDom;  
          if (Window.ActiveXObject) { // IE  
            oXmlDom = new ActiveXObject("Microsoft.XmlDom");  
          } else {  // Firefox  
            oXmlDom = document.implementation.createDocument("", "", null);  
          }  
        }  
  
3. 加载XML  
  (1)现有问题：如果要加载外部文件IE和Firefox都可以用:  
          oXmlDom.async=false;      // 这在Firefox中是必须的  
          oXmlDom.load("test.xml");  
     但是它们加载xml字符串的方式不一样，IE中直接可以使用oXmlDom.loadXML("<root><child/></root>")，而Firefox要使用DOMParser:  
        var oParser = new DOMParser();  
          var oXmlDom = oParser.parseFromString("<root/>", "text/xml");  
  (2)解决方法：比较好的方法是给Firefox产生的XMLDom加上loadXML方法：  
        if (isFirefox) { // 需要浏览器检测  
          Document.prototype.loadXML = function(sXml) {  
            var oParser = new DOMParser();  
            var oXmlDom = oParser.parseFromString(sXml, "text/xml");  
              
            while (this.firstChild) this.removeChild(this.firstChild);  
              
            for (var i = 0; i < oXmlDom.childNodes.length; i++) {  
              var oNewNode = this.importNode(oXmlDom.childNodes[i], true);  
              this.appendChild(oNewNode);  
            }  
          }  
        }  
      这样在IE和Firefox就可以调用loadXML方法了。  
        
4. XPath支持  
  (1)现有问题：IE中可以直接用XmlDOM的selectNodes来根据XPath表示式来选择结点，Firefox则比较复杂，需要使用XPathEvaluator。  
     IE:   
        var lstNodes = oXmlDom.documentElement.selectNodes("employee/name");  
        for (var i = 0; i < lstNodes.length; i++) {  
          alert(lstNodes[i].firstChild.nodeValue);  
        }  
     Firefox:   
          var oEvaluator = new XPathEvaluator();  
          var oResult = oEvaluator.evaluate("employee/name", oXmlDom.documentElement, null, XPathResult.ORDERED\_NODE\_ITERATOR\_TYPE, null);  
          var oElement = oResult.iterateNext();  
          while (oElement) {  
            alert(oElement.firstChild.nodeValue);  
            oElement = oResult.iterateNext();  
          }  
  (2)解决方法：比较好的方法给Firefox的Element添加selectNodes方法。  
        if (isFirefox) { // 需要浏览器检测  
            Element.prototype.selectNodes = function(sXPath) {  
            var oEvaluator = new XPathEvaluator();  
              var oResult = oEvaluator.evaluate(sXPath, this, null, XPathResult.ORDERED\_NODE\_ITERATOR\_TYPE, null);  
                
              var aNodes = new Array();  
                
              if (oResult != null) {  
                var oElement = oResult.iterateNext();  
                while (oElement) {  
                  aNodes.push(oElement);  
                  oElement = oResult.iterateNext();  
                }  
              }  
              return aNodes;  
           }  
      }  
   这样在IE和Firefox中就都可以调用selectNodes方法了。  
     
5. XSLT支持  
  (1)现有问题：IE中可以使用XmlDOM的transferNode方法将其转换成html，而Firefox需要使用XSLTProcessor。  
  IE:  
        oXmlDom.load("employee.xml");  
        oXslDom.load("employee.xslt");  
        var sResult=oXmlDom.transformNode(oXslDom);  
    Firefox:  
        var oProcessor = new XSLTProcessor();  
        oProcessor.importStylesheet(oXslDom);  
        var oResultDom = oProcessor.transformToDocument(oXmlDom);          
        var oSerializer = new XMLSerializer();  
        var sXml = oSerializer.serializeToString(oResultDom, "text/xml");  
        alert(sXml);  
  (2)解决方法：比较好的方法给Firefox的Node添加transferNode方法。  
        if (isFirefox) { // 需要浏览器检测  
          Node.prototype.transformNode = function(oXslDom) {  
          var oProcessor = new XSLTProcessor();  
            oProcessor.importStylesheet(oXslDom);  
            var oResultDom = oProcessor.transformToDocument(oXmlDom);  
              
            var oSerializer = new XMLSerializer();  
            var sXml = oSerializer.serializeToString(oResultDom, "text/xml");  
              
            return sXml;  
          }  
        }  
   这样在IE和Firefox中就都可以调用transferNode方法了。

**常使用的库有哪些？常用的前端工具？开发过什么应用或组件？**

**请举例几个你经常光顾的IT网站和博客？最近看的一篇JS的文章是？**