**Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?**

1. <!DOCTYPE> 声明位于文档中的最前面，处于 <html> 标签之前。告知浏览器以何种模式来渲染文档。

2. 严格模式的排版和 JS 运作模式是 以该浏览器支持的最高标准运行。

3. 在混杂模式中，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。

4. DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

**你知道多少种Doctype文档类型？**

1.该标签可声明三种 DTD 类型，分别表示严格版本、过渡版本以及基于框架的 HTML 文档。

2. HTML 4.01 规定了三种文档类型：Strict、Transitional 以及 Frameset。

3. XHTML 1.0 规定了三种 XML 文档类型：Strict、Transitional 以及 Frameset。

4. Standards （标准）模式（也就是严格呈现模式）用于呈现遵循最新标准的网页，而 Quirks（包容）模式（也就是松散呈现模式或者兼容模式）用于呈现为传统浏览器而设计的网页。

**XHTML和HTML的区别？**

1. XHTML 元素必须被正确地嵌套。
2. XHTML 元素必须被关闭。
3. 标签名必须用小写字母。
4. XHTML 文档必须拥有根元素。

**html常见兼容性问题？**

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

2. 浏览器默认的margin和padding不同 解决方案：加一个全局的 \*{margin:0;padding:0;} 来统一。

1. IE6双边距bug：在IE6下，如果对元素设置了浮动，同时又设置了margin-left或margin-right，margin值会加倍。这种情况之下IE会产生20px的距离

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。( \_ 这个符号只有ie6会识别)

4. 渐进识别的方式，从总体中逐渐排除局部。

首先，巧妙的使用“\9”这一标记，将IE游览器从所有情况中分离出来。

接着，再次使用 "+" 将IE8和IE7、IE6分离开来，这样IE8已经独立识别。

**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？**

新特性：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。

1. 拖拽释放(Drag and drop) API

2. 语义化更好的内容标签（header,nav,footer,aside,article,section）

3. 音频、视频API(audio,video)

4. 画布(Canvas) API

5. 地理(Geolocation) API

6. 本地离线存储 localStorage 长期存储数据，浏览器关闭后数据不丢失；

7. sessionStorage 的数据在浏览器关闭后自动删除

8. 表单控件，calendar、date、time、email、url、search

9. 新的技术webworker, websocket, Geolocation

移除的元素：

1. 纯表现的元素：basefont，big，center，font, s，strike，tt，u；

2. 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（当然最好的方式是直接使用成熟的框架、使用最多的是html5shiv框架）：

<!--[if lt IE 9]>

<script> src="http://html5shiv.googlecode.com/svn/trunk/html5.js"</script>

<![endif]-->

如何区分：

DOCTYPE声明新增的结构元素、功能元素/通过DOCTYPE声明来区分

**iframe的优缺点？**

优点：

1. 解决加载缓慢的第三方内容如图标和广告等的加载问题

2. Security sandbox

3. 并行加载脚本

缺点：

1. iframe会阻塞主页面的Onload事件

2. 即时内容为空，加载也需要时间

3. 没有语义

**什么是 "use strict"; ? 使用它的好处和坏处分别是什么？**

ECMAscript 5添加了第二种运行模式："严格模式"（strict mode）。顾名思义，这种模式使得Javascript在更严格的条件下运行。

设立"严格模式"的目的，主要有以下几个：

1. 消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为;

2. 消除代码运行的一些不安全之处，保证代码运行的安全；

3. 提高编译器效率，增加运行速度；

4. 为未来新版本的Javascript做好铺垫。

注：经过测试 IE6,7,8,9 均不支持严格模式。

缺点：

现在网站的 JS 都会进行压缩，一些文件用了严格模式，而另一些没有。这时这些本来是严格模式的文件，被 merge 后，这个串就到了文件的中间，不仅没有指示严格模式，反而在压缩后浪费了字节。

**document.write和 innerHTML 的区别**

document.write 只能重绘整个页面

innerHTML 可以重绘页面的一部分

**如何实现浏览器内多个标签页之间的通信?**

调用 localstorge、cookies 等本地存储方式

**webSocket 如何兼容低浏览器？**

Adobe Flash Socket 、 ActiveX HTMLFile (IE) 、 基于 multipart 编码发送 XHR 、 基于长轮询的 XHR

**web storage和cookie的区别**

web storage的容量更大，拥有setItem removeItem clear等方法，cookie需要自己封装这些函数

cookie是HTTP规范的一部分，不可或缺。web storage是为了本地存储数据而生

部分浏览器不支持web storage

**cache-control网页缓存**

网页的缓存是由HTTP消息头中的“Cache-control”来控制的，常见的取值有private、no-\*\*\*、max-age、must-revalidate等，默认为private。

Expires 头部字段提供一个日期和时间，响应在该日期和时间后被认为失效。允许客户端在这个时间之前不去检查（发请求），等同max-age的效果。但是如果同时存在，则被Cache-Control的max-age覆盖。

Expires = "Expires" ":" HTTP-date

例如：Expires: Thu, 01 Dec 1994 16:00:00 GMT （必须是GMT格式）。如果把它设置为-1，则表示立即过期

Expires 和 max-age 都可以用来指定文档的过期时间，但是二者有一些细微差别

1. Expires在HTTP/1.0中已经定义，Cache-Control:max-age在HTTP/1.1中才有定义，为了向下兼容，仅使用max-age不够

2. Expires指定一个绝对的过期时间(GMT格式),这么做会导致至少2个问题：

2.1客户端和服务器时间不同步导致Expires的配置出现问题。

2.2很容易在配置后忘记具体的过期时间，导致过期来临出现浪涌现象

3. max-age 指定的是从文档被访问后的存活时间，这个时间是个相对值(比如:3600s)，相对的是文档第一次被请求时服务器记录的Request\_time(请求时间)

4. Expires 指定的时间可以是相对文件的最后访问时间(Atime)或者修改时间(MTime)，而max-age相对对的是文档的请求时间(Atime)

5. 如果值为 no-\*\*\*,那么每次都会访问服务器。如果值为max-age，则在过期之前不会重复访问服务器。

**HTTP状态码**

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）

**一个页面从输入 URL 到页面加载显示完成，这个过程中都发生了什么？**

分为4个步骤：

1. 当发送一个 URL 请求时，不管这个 URL 是 Web 页面的 URL 还是 Web 页面上每个资源的 URL，浏览器都会开启一个线程来处理这个请求，同时在远程 DNS 服务器上启动一个 DNS 查询。这能使浏览器获得请求对应的 IP 地址。

2. 浏览器与远程 Web 服务器通过 TCP 三次握手协商来建立一个 TCP/IP 连接。该握手包括一个同步报文，一个同步-应答报文和一个应答报文，这三个报文在 浏览器和服务器之间传递。该握手首先由客户端尝试建立起通信，而后服务器应答并接受客户端的请求，最后由客户端发出该请求已经被接受的报文。

3. 一旦 TCP/IP 连接建立，浏览器会通过该连接向远程服务器发送 HTTP 的 GET 请求。远程服务器找到资源并使用 HTTP 响应返回该资源，值为 200 的 HTTP 响应状态表示一个正确的响应。

4. 此时，Web 服务器提供资源服务，客户端开始下载资源。

请求返回后，便进入了我们关注的前端模块

简单来说，浏览器会解析 HTML 生成 DOM Tree，其次会根据 CSS 生成 CSS Rule Tree，而 javascript 又可以根据 DOM API 操作 DOM

**WEB应用从服务器主动推送Data到客户端有那些方式？**

1. html5 websocket

2. WebSocket 通过 Flash

3. XHR长时间连接

4. XHR Multipart Streaming

5. 不可见的Iframe

6. <script>标签的长时间连接(可跨域)

**GET和POST的区别，何时使用POST？**

GET：一般用于信息获取，使用URL传递参数，对所发送信息的数量也有限制，一般在2000个字符

POST：一般用于修改服务器上的资源，对所发送的信息没有限制

GET方式需要使用 Request.QueryString 来取得变量的值， POST方式通过 Request.Form 来获取变量的值。也就是说 GET是通过地址栏来传值，而 POST 是通过提交表单来传值。

在以下情况中，请使用 POST 请求：

1. 无法使用缓存文件（更新服务器上的文件或数据库）

2. 向服务器发送大量数据（POST 没有数据量限制）

3. 发送包含未知字符的用户输入时，POST 比 GET 更稳定也更可靠

**请解释一下 JavaScript 的同源策略。**

概念：

同源策略是客户端脚本（尤其是Javascript）的重要的安全度量标准。它最早出自Netscape Navigator2.0，其目的是防止某个文档或脚本从多个不同源装载。

这里的同源策略指的是：协议，域名，端口相同，同源策略是一种安全协议，指一段脚本只能读取来自同一来源的窗口和文档的属性。

为什么要有同源限制：

我们举例说明：比如一个黑客程序，他利用Iframe把真正的银行登录页面嵌到他的页面上，当你使用真实的用户名，密码登录时，他的页面就可以通过Javascript读取到你的表单中input中的内容，这样用户名，密码就轻松到手了。

**如何解决跨域问题**

1. jsonp（jsonp 的原理是动态插入 script 标签）
2. document.domain + iframe
3. window.name、window.postMessage
4. 服务器上设置代理页面