**基础题目**

**语义化的理解？**

a. 用正确的标签做正确的事情！

b. html语义化就是让页面的内容结构化，便于对浏览器、搜索引擎解析；

c. 在没有样式CCS情况下也以一种文档格式显示，并且是容易阅读的。

d. 搜索引擎的爬虫依赖于标记来确定上下文和各个关键字的权重，利于 SEO。

e. 使阅读源代码的人对网站更容易将网站分块，便于阅读维护理解。

**垂直水平居中的方法有几种？**

三四种就行：

1，绝对定位，left0，top 0 bottom0 marg auto

2，绝对定位 left50%，top50%；给负margin

3，绝对定位 left50%，top50%；给transform：translate（-50%。-50%）

4，父：display：table-cell，text-aline：center，vetiral-align：middle

子：display：inline-block

5，display：flex；justify-content：center；align-item：center；

**Javascript如何实现继承？**

答案：原型链继承，借用构造函数继承，组合继承，寄生式继承，寄生组合继承

**请你谈谈Cookie的弊端？**

缺点：

1. Cookie数量和长度的限制。部分浏览器每个domain最多只能有50条cookie，基本所有浏览器中每个cookie长度不能超过4KB，否则会被截掉。

2.安全性问题。如果cookie被人拦截了，那人就可以取得所有的session信息。即使加密也与事无补，因为拦截者并不需要知道cookie的意义，他只要原样转发cookie就可以达到目的了。

3.有些状态不可能保存在客户端。例如，为了防止重复提交表单，我们需要在服务器端保存一个计数器。如果我们把这个计数器保存在客户端，那么它起不到任何作用。

4.占用网络上传带宽。每次请求服务器资源时，都会携带 cookie 信息向服务器传递。

**如何解决跨域问题?**

理解跨域的概念：URL的协议、域名、端口都相同才是同域，否则都是跨域

出于安全考虑，浏览器不允许跨域获取数据，这是浏览器同源策略的限制，但实际开发过程中会遇到非常多的跨域问题，可以采用以下的办法解决跨域问题：

方案1：JSONP

虽然浏览器不允许跨域直接获取数据，但是可以跨域获取文件内容，所以基于这一点，可以动态创建script标签（<script>标签的使用不受同源策略限制），使用标签的src属性访问js文件的形式获取服务器的一段js脚本，这个js脚本中的内容是函数调用，该函数调用的参数是服务器返回的数据，为了获取这里的参数数据，需要事先在页面中定义全局回调函数，在回调函数中处理服务器返回的数据。JSONP只能实现GET请求的跨域，但其兼容性好，不需要使用类似 XMLHttpRequest 的对象。

方案2：CORS（跨域资源共享）

跨域资源共享定义了在必须访问跨域资源的时，浏览器与服务器应该如何沟通。他的原理是使用自定义的 HTTP 头部，让服务器与浏览器进行沟通，主要是通过设置响应头的 Access-Control-Allow-Origin 来达到目的的。CORS对GET、POST请求的跨域都能够处理，而且对客户端来说可以像使用自己域下资源一样使用ajax来跨域访问数据，非常方便。

方案3：document.domain

浏览器的同源策略使得不同域的框架是不能进行JS的交互操作的。比如：有一个页面是http://www.example.com/a.html,在这个页面中还有一个http://example.com/b.html，很显然，a.html与b.html是不同域的，所以我们无法通过在页面中书写js代码来获取iframe中的东西，但是，如果我们把这2个页面的document.domain都设置成相同的域名就可以了，需要注意的是，我们只能把document.domain设置成自身或更高一级的父域，且主域名必须相同。

方案4：window.name

window对象有个name属性，该属性有一个特征：即在一个窗口的生命周期内，窗口载入的所有页面都是共享一个window.name的，每个页面对window.name都有读写的权限，window.name是持久存在一个窗口载入过的所有页面中的。

方案5：window.postMessage

window.postMessage（message,targetOrigin）方法，可以用来向其他的window对象发送消息，无论这个window对象是属于同一个源还是不同源。

当然最常用的跨域解决方案还是前2种，如果是开发时有跨域的需求，但项目上线后没有跨域需求，也可以使用浏览器插件来暂时解决跨域问题。

**原生 JS 中 call()、apply()、bind() 方法有什么区别？**

三个方法都可以改变函数运行时的 this 指向。三个方法第一个参数都是函数调用执行时this 指向的对象。call() 方法第二个参数是个可变参数，是函数调用执行时本身所需要的参数，apply() 方法第二个参数是数组或arguments。call()与apply()都是立即调用函数执行，在运行时修改this指向。bind()是返回一个新的函数，新函数的函数主体与原函数的函数主体一致，当新函数被调用时，函数体中 this 指向的是 bind() 方法第一个参数传递的对象，而bind() 方法不会影响原函数本身的 this 指向。

**什么是闭包？特点是？**

闭包，官方对闭包的解释是：一个拥有许多变量和绑定了这些变量的环境的表达式（通常是一个函数），因而这些变量也是该表达式的一部分。

闭包的特点：

（1）作为一个函数变量的一个引用，当函数返回时，其处于激活状态。

（2） 一个闭包就是当一个函数返回时，一个没有释放资源的栈区。

简单的说，Javascript允许使用内部函数---即函数定义和函数表达式位于另一个函数的函数体内。而且，这些内部函数可以访问它们所在的外部函数中声明的所有局部变量、参数和声明的其他内部函数。当其中一个这样的内部函数在包含它们的外部函数之外被调用时，就会形成闭包。

**你有哪些性能优化的方法？**

（1） 减少http请求次数：CSS Sprites, JS、CSS源码压缩、图片大小控制合适；网页Gzip，CDN托管，data缓存 ，图片服务器。

（2） 前端模板 JS+数据，减少由于HTML标签导致的带宽浪费，前端用变量保存AJAX请求结果，每次操作本地变量，不用请求，减少请求次数

（3） 用innerHTML代替DOM操作，减少DOM操作次数，优化javascript性能。

（4） 当需要设置的样式很多时设置className而不是直接操作style。

（5） 少用全局变量、缓存DOM节点查找的结果。减少IO读取操作。

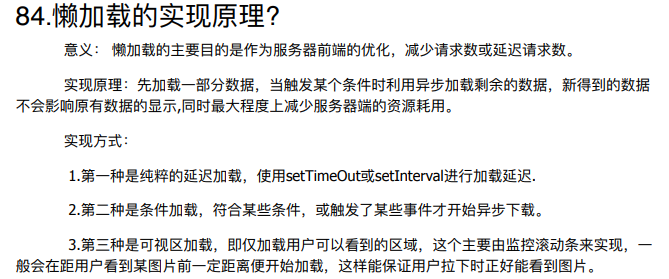
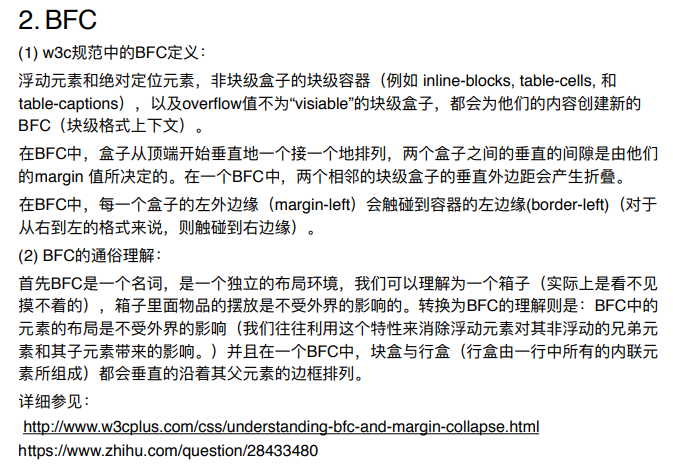
（6） 避免使用CSS Expression（css表达式)又称Dynamic properties(动态属性)。

（7） 图片预加载，将样式表放在顶部，将脚本放在底部，加上时间戳……

**什么叫优雅降级和渐进增强？**

优雅降级：Web站点在所有新式浏览器中都能正常工作，如果用户使用的是老式浏览器，则代码会检查以确认它们是否能正常工作。由于IE独特的盒模型布局问题，针对不同版本的IE的hack实践过优雅降级了,为那些无法支持功能的浏览器增加候选方案，使之在旧式浏览器上以某种形式降级体验却不至于完全失效.

渐进增强：从被所有浏览器支持的基本功能开始，逐步地添加那些只有新式浏览器才支持的功能,向页面增加无害于基础浏览器的额外样式和功能的。当浏览器支持时，它们会自动地呈现出来并发挥作用。



**框架工具类题目**



