**前端开发面试题总结--（四）前端综合**

[初夏\_summer](http://www.w3cfuns.com/house/27029.html) 发布于 前天 18:22 *浏览：*423 *类型：*[原创](http://www.w3cfuns.com/notes/id/all:original:all:all.html" \t "_blank) - [随笔](http://www.w3cfuns.com/notes/id/all:all:b415fdeb2a2577cf642bb78c4672419f:all.html) *分类：*[面试经验](http://www.w3cfuns.com/notes/id/all:all:all:a45f6cf55f1ba771db57a8507edc0309.html) - [面试](http://www.w3cfuns.com/house/27029/note/class/id/e1ad3c7a846ab9adf5d587822ee2698d.html) *二维码：* *作者原创 版权保护*

这里综合是指除了HTML、CSS、JS之外的部分，比如浏览器相关、资源文件优化等等，HTML5、JS、CSS的部分还没有写完，就先发这篇吧~  
  
**一. 浏览器内核（渲染引擎）**  
**先说一下浏览器的结构：**  
①、用户界面（UI） - 包括菜单栏、工具栏、地址栏、后退/前进按钮、书签目录等，也就是能看到的除了显示页面的主窗口之外的部分；  
②、浏览器引擎（Rendering engine） - 也被称为浏览器内核、渲染引擎，主要负责取得页面内容、整理信息（应用CSS）、计算页面的显示方式，然后会输出到显示器或者打印机；  
③、JS解释器 - 也可以称为JS内核，主要负责处理javascript脚本程序，一般都会附带在浏览器之中，例如chrome的V8引擎；  
④、网络部分 - 主要用于网络调用，例如：HTTP请求，其接口与平台无关，并为所有的平台提供底层实现；  
⑤、UI后端 - 用于绘制基本的窗口部件，比如组合框和窗口等。  
⑥、数据存储 - 保存类似于cookie、storage等数据部分，HTML5新增了web database技术，一种完整的轻量级客户端存储技术。  
注：  
IE只为每个浏览器窗口启用单独的进程，而chrome浏览器却为每个tab页面单独启用进程，也就是每个tab都有独立的渲染引擎实例。  
**现在的主要浏览器：**  
IE、Firefox、Safari、Chrome、Opera。  
它们的浏览器内核（渲染引擎）：  
IE--Trident、FF(Mozilla)--Gecko、Safari--Webkit、Chrome--Blink（WebKit的分支）、Opera--原为Presto，现为Blink。  
因此在开发中，兼容IE、FF、Opera（Presto引擎是逐步放弃的）、Chrome和Safari中的一种即可，Safari、Chrome的引擎是相似的 。  
 **二. 浏览器访问url机制**  
当用户访问一个网页时，浏览器将页面显示出来，经过了哪些步骤呢？  
详细说的话有很多，简单来说：  
(1)浏览器需要查找IP地址  
DNS（域名系统）查找IP地址的顺序：浏览器缓存、系统缓存、互联网服务提供商（ISP）的DNS缓存、递归搜索（从浏览器缓存开始，如果没找到就继续往下一个找。）找到后，浏览器会获得一个IP地址。  
注：  
通过主机名，最终得到该主机名对应的IP地址的过程叫做域名解析（或主机名解析）  
(2)浏览器向该IP地址的Web服务器发送HTTP请求，服务器返回一个HTTP响应，如果该地址被301重定向，则浏览器再向重定向的地址（包含在返回的HTTP响应中）发送请求，继续上面的步骤。  
(3)服务器返回的HTTP响应状态码为200，浏览器将内容显示出来，如果是页面，则重复上面的步骤去请求页面中需要的图片、CSS、JS等文件。  
  
**三. HTTP状态码**  
状态代码        状态信息        含义  
100        Continue        初始的请求已经接受，客户应当继续发送请求的其余部分。（HTTP 1.1新）  
101        Switching Protocols        服务器将遵从客户的请求转换到另外一种协议（HTTP 1.1新）  
200        OK        一切正常，对GET和POST请求的应答文档跟在后面。  
201        Created        服务器已经创建了文档，Location头给出了它的URL。  
202        Accepted        已经接受请求，但处理尚未完成。  
203        Non-Authoritative Information        文档已经正常地返回，但一些应答头可能不正确，因为使用的是文档的拷贝（HTTP 1.1新）。  
204        No Content        没有新文档，浏览器应该继续显示原来的文档。如果用户定期地刷新页面，而Servlet可以确定用户文档足够新，这个状态代码是很有用的。  
205        Reset Content        没有新的内容，但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容（HTTP 1.1新）。  
206        Partial Content        客户发送了一个带有Range头的GET请求，服务器完成了它（HTTP 1.1新）。  
300        Multiple Choices        客户请求的文档可以在多个位置找到，这些位置已经在返回的文档内列出。如果服务器要提出优先选择，则应该在Location应答头指明。  
301        Moved Permanently        客户请求的文档在其他地方，新的URL在Location头中给出，浏览器应该自动地访问新的URL。  
302        Found        类似于301，但新的URL应该被视为临时性的替代，而不是永久性的。注意，在HTTP1.0中对应的状态信息是“Moved Temporatily”。出现该状态代码时，浏览器能够自动访问新的URL，因此它是一个很有用的状态代码。注意这个状态代码有时候可以和301替换使 用。例如，如果浏览器错误地请求http://host/~user（缺少了后面的斜杠），有的服务器 返回301，有的则返回302。严格地说，我们只能假定只有当原来的请求是GET时浏览器才会自动重定向。请参见307。  
303        See Other        类似于301/302，不同之处在于，如果原来的请求是POST，Location头指定的重定向目标文档应该通过GET提取（HTTP 1.1新）。  
304        Not Modified        客户端有缓冲的文档并发出了一个条件性的请求（一般是提供If-Modified-Since头表示客户只想比指定日期更新的文档）。服务器告 诉客户，原来缓冲的文档还可以继续使用。  
305        Use Proxy        客户请求的文档应该通过Location头所指明的代理服务器提取（HTTP 1.1新）。  
307        Temporary Redirect        和302 （Found）相同。许多浏览器会错误地响应302应答进行重定向，即使原来的请求是POST，即使它实际上只能在POST请求的应答是303时才能重定 向。由于这个原因，HTTP 1.1新增了307，以便更加清除地区分几个状态代码：当出现303应答时，浏览器可以跟随重定向的GET和POST请求；如果是307应答，则浏览器只 能跟随对GET请求的重定向。（HTTP 1.1新）  
400        Bad Request        请求出现语法错误。  
401        Unauthorized        客户试图未经授权访问受密码保护的页面。应答中会包含一个WWW-Authenticate头，浏览器据此显示用户名字/密码对话框，然后在填写合适的Authorization头后再次发出请求。  
403        Forbidden        资源不可用。服务器理解客户的请求，但拒绝处理它。通常由于服务器上文件或目录的权限设置导致。  
404        Not Found        无法找到指定位置的资源。这也是一个常用的应答。  
405        Method Not Allowed        请求方法（GET、POST、HEAD、DELETE、PUT、TRACE等）对指定的资源不适用。（HTTP 1.1新）  
406        Not Acceptable        指定的资源已经找到，但它的MIME类型和客户在Accpet头中所指定的不兼容（HTTP 1.1新）。  
407        Proxy Authentication Required        类似于401，表示客户必须先经过代理服务器的授权。（HTTP 1.1新）  
408        Request Timeout        在服务器许可的等待时间内，客户一直没有发出任何请求。客户可以在以后重复同一请求。（HTTP 1.1新）  
409        Conflict        通常和PUT请求有关。由于请求和资源的当前状态相冲突，因此请求不能成功。（HTTP 1.1新）  
410        Gone        所请求的文档已经不再可用，而且服务器不知道应该重定向到哪一个地址。它和404的不同在于，返回407表示文档永久地离开了指定的位置，而 404表示由于未知的原因文档不可用。（HTTP 1.1新）  
411        Length Required        服务器不能处理请求，除非客户发送一个Content-Length头。（HTTP 1.1新）  
412        Precondition Failed        请求头中指定的一些前提条件失败（HTTP 1.1新）。  
413        Request Entity Too Large        目标文档的大小超过服务器当前愿意处理的大小。如果服务器认为自己能够稍后再处理该请求，则应该提供一个Retry-After头（HTTP 1.1新）。  
414        Request URI Too Long        URI太长（HTTP 1.1新）。  
416        Requested Range Not Satisfiable        服务器不能满足客户在请求中指定的Range头。（HTTP 1.1新）  
500        Internal Server Error        服务器遇到了意料不到的情况，不能完成客户的请求。  
501        Not Implemented        服务器不支持实现请求所需要的功能。例如，客户发出了一个服务器不支持的PUT请求。  
502        Bad Gateway        服务器作为网关或者代理时，为了完成请求访问下一个服务器，但该服务器返回了非法的应答。  
503        Service Unavailable        服务器由于维护或者负载过重未能应答。例如，Servlet可能在数据库连接池已满的情况下返回503。服务器返回503时可以提供一个 Retry-After头。  
504        Gateway Timeout        由作为代理或网关的服务器使用，表示不能及时地从远程服务器获得应答。（HTTP 1.1新）  
505        HTTP Version Not Supported        服务器不支持请求中所指明的HTTP版本。（HTTP 1.1新）  
**HTTP状态码：**  
(1)1xx的响应码代表请求被接收，需要继续处理。  
(2)2xx的响应码代表请求已被服务器成功接受处理。  
常见：200----OK/正常，请求已成功，请求内容将同时返回。  
(3)3xx的响应码代表客户端需要进一步操作，通常用来重定向。  
常见：  
301: Moved Permanently/永久移动，被请求的资源已永久移到到新位置。  
304: Not Modified/未修改，客户端请求的内容并未改变，服务器将返回空内容。  
(4)4xx的响应码代表请求错误---客户端可能发生错误，服务器无法正确处理。  
常见：  
404--- Not Found/未找到，客户端所请求的资源未找到。  
403---Forbidden，资源不可用，服务器拒绝执行客户端的请求。  
(5)5xx的响应码代表服务器端错误---服务器有错误或异常发生。  
常见：  
500: Internal Server Error/内部服务器错误，服务器遇到未知错误导致请求无法完成。  
502: Bad Gateway/网关错误，网关或代理服务器接收到远端服务器的无效响应。  
503: Service Unavailable/服务无法获得，服务器已超载或维护中导致请求无法完成，服务器超时。  
  
**四.浏览器加载文件**  
**文件加载顺序**  
浏览器加载页面上引用的CSS、JS文件、图片时，是按顺序从上到下加载的，每个浏览器可以同时下载文件的个数不同，因此经常有网站将静态文件放在不同的域名下，这样可以加快网站打开的速度。  
**reflow**  
在加载JS文件时，浏览器会阻止页面的渲染，因此将js放在页面底部比较好。  
因为如果JS文件中有修改DOM的地方，浏览器会倒回去把已经渲染过的元素重新渲染一遍，这个回退的过程叫reflow。  
CSS文件虽然不影响js文件的加载，但是却影响js文件的执行，即使js文件内只有一行代码，也会造成阻塞。因为可能会有var width = $('#id').width()这样的语句，这意味着，js代码执行前，浏览器必须保证css文件已下载和解析完成。  
办法：当js文件不需要依赖css文件时，可以将js文件放在头部css的前面。  
**常见的能引起reflow的行为：**  
1.改变窗囗大小  
2.改变文字大小  
3.添加/删除样式表  
4.脚本操作DOM  
5.设置style属性  
等等……  
reflow是不可避免的，只能尽量减小，常用的方法有：  
1.尽量不用行内样式style属性，操作元素样式的时候用添加去掉class类的方式  
2.给元素加动画的时候，可以把该元素的定位设置成absolute或者fixed，这样不会影响其他元素  
**repaint**  
另外，有个和reflow相似的术语，叫做repaint（重绘），在元素改变样式的时候触发，这个比reflow造成的影响要小，所以能触发repaint解决的时候就不要触发reflow……  
  
**五.Google浏览器不支持12px以下字体大小**  
谷歌浏览器不支持12px以下小字体，就算给设置成10px，11px也不起作用，看起来还是12px那么大。如何解决呢？  
办法一：  
-webkit-text-size-adjust:none;  
但是，自chrome 27之后，就取消了对这个属性的支持，所以不能用。  
办法二：  
-webkit-transform: scale(0.5);  
既然最小支持到12px，那么就以12px为基点进行缩放，同时可以使用-webkit-transform-origin-x: 0;来解决缩放后margin-left的位移问题，scale里面的值可以修改。  
但是这个办法会导致如果字体太小，看起来会不清楚。  
所以就用图片，或者干脆就不要采用12px以下字体……  
  
**六. 如何对网站的文件和资源进行优化**  
期待的解决方案包括：  
文件合并、文件最小化、文件压缩、使用CDN托管、缓存的使用等……  
为什么要用这些方法呢？跟下面这些常见问题有关：  
**浏览器同一时间可以从一个域名下载多少资源**  
这个专业的说法叫“浏览器并发请求数”。  
意即，同一时间针对同一域名下的请求有一定数量限制。超过限制数目的请求会被阻止。不同浏览器这个限制的数目不一样。  
为了用更少的时间下载更多的文件，提高网站加载速度，提高用户体验，可以使用以下方法：  
1.css sprites----将小图片合并为一张大图片，使用background-position等css属性取得图片位置  
2.将资源放在多个域名下-----打开控制台，可以看到很多网站都是这么做的~  
3.图片延迟加载-----很多电商网站、新闻网站，尤其是用到瀑布流展示图片的时候，很多都这么做了，这个技术已经很普遍~  
**合并压缩文件**  
js和css文件在百度上搜一个压缩网站就能压缩，但是在实际开发的项目中，使用gulp、webpack等工具可以打包出合并压缩后的文件，小图片可以在打包时转换成base64方式引入，大图片可以被压缩，html文件也是可以被压缩的。  
**书写代码的时候注意优化**  
1.css  
将可以合并的样式合并起来，比如margin-top、margin-bottom等。  
给img图片设置宽高，因为浏览器渲染元素的时候没有找到这两个参数，需要一边下载图片一边计算大小，如果图片很多，浏览器需要不断地调整页面。这不但影 响速度，也影响浏览体验。当浏览器知道了高度和宽度参数后，即使图片暂时无法显示，页面上也会腾出图片的空位，然后继续加载后面的内容。从而加载时间快 了，浏览体验也更好了。  
2.js  
少改变DOM元素，少触发reflow，可以复用的代码提出来写成公共的等等……  
3.img  
优化图片，不失真的情况下尽量减少图片大小，使用iconfont等  
  
**七. 渐进增强和优雅降级**  
渐进增强 progressive enhancement：针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。  
优雅降级 graceful degradation：一开始就构建完整的功能，然后再针对低版本浏览器进行兼容。  
区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带。  
  
**八. 说说你喜欢的开发环境**  
其实也就是说说你的操作系统、编辑器、常用的浏览器，控制台等等  
 **九. 用过什么版本控制系统**  
现在的版本控制系统么，基本都是svn或者Git，git要更高大上一点，还有github  
 **十. 线程与进程的区别**  
一个程序至少有一个进程,一个进程至少有一个线程.  
线程的划分尺度小于进程，使得多线程程序的并发性高。  
另外，进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大地提高了程序的运行效率。  
线程在执行过程中与进程还是有区别的。每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。  
从逻辑角度来看，多线程的意义在于一个应用程序中，有多个执行部分可以同时执行。但操作系统并没有将多个线程看做多个独立的应用，来实现进程的调度和管理以及资源分配。这就是进程和线程的重要区别。  
  
**十一. CORS是什么，解决什么问题的？**  
CORS（跨域资源共享）  
  
**十二. css动画和js动画的优缺点  
CSS3的动画**  
优点：  
1.在性能上会稍微好一些，浏览器会对CSS3的动画做一些优化（比如专门新建一个图层用来跑动画）  
2.代码相对简单  
缺点：  
1.在动画控制上不够灵活  
2.兼容性不好  
3.部分动画功能无法实现（如滚动动画，视差滚动等）  
**JavaScript的动画**  
优点：  
1.控制能力很强，可以单帧的控制、变换  
2.兼容性好，写得好完全可以兼容IE6，且功能强大。  
缺点：  
计算没有css快，另外经常需要依赖其他的库。  
**结论**  
所以，不复杂的动画完全可以用css实现，复杂一些的，或者需要交互的时候，用js会靠谱一些~  
  
**十三. data-\*属性的作用是什么？**  
data-\*属性是HTML5出来的自定义属性，这里附上一篇链接：  
http://www.vaikan.com/html-5-data-attributes-how-to-use-them-and-why/  
其实不使用这个属性也没关系，但是写上的话，代码看起来会更清晰的。  
  
**十四. 有哪些的隐藏内容的方法 (同时还要保证屏幕阅读器可用)？**  
答：display:none或者visibility:hidden，overflow:hidden。  
1.display:none;的缺陷  
搜索引擎可能认为被隐藏的文字属于垃圾信息而被忽略  
屏幕阅读器（是为视觉上有障碍的人设计的读取屏幕内容的程序）会忽略被隐藏的文字。  
2.visibility: hidden ;的缺陷  
这个大家应该比较熟悉就是隐藏的内容会占据他所应该占据物理空间  
3.overflow:hidden;一个比较合理的方法  
例：  
.texthidden { display:block; overflow: hidden; width: 0; height: 0; }  
将宽度和高度设定为0，然后超过部分隐藏，就会弥补上述一、二方法中的缺陷，也达到了隐藏内容的目的。  
 **十五. 你用过栅格系统 (grid system) 吗？如果使用过，你最喜欢哪种？**  
比如Bootstrap~