

从浏览器地址栏输入 url 到页面展示的过程 中发生了什么?——网络进程

作者: limanting

原文链接: https://ld246.com/article/1629388704705

来源网站:链滴

许可协议: 署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

从浏览器地址栏输入url到页面展示的过程中发生了什么?其实这个过程可以分为网络进程和渲染进程。

弄明白这两个进程,能帮助我们在开发时提高效率。本篇文章详细讲述了网络进程,渲染进程会在下 篇中详细讲述(链接:)。

网络进程

网络进程可以分成 解析URL、封装http报文、DNS寻址、传输层TCP传输报文、网络层IP协议查询Mc地址、数据达到数据链路层、服务器接受数据、服务器响应请求、服务器返回相应文件。

解析url

当url在浏览器的地址栏输入,并按下回车键后。浏览器会解析该url,获取url中的协议、域名、资源径。

协议 :// 域名(服务器名) / 资源路径名

封装HTTP报文

用于 HTTP 协议交互的信息被称为 HTTP 报文,本身是由多行数据构成的字符串文本。 报文又可以分为**请求报文**和**响应报文**。

请求方式 请求URI 协议版本

"GET /list HTTP/1.1\r\nHost: payment-admin.com\r\nConnection: keep-alive\r\nCache-Control: maxage=0\r\nUpgrade-Insecure-Requests: 1\r\nUser-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3345.0 Safari/537.36\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8\r\nAccept-Encoding: gzip, deflate\r\nAccept-Language: en-US,en;q=0.9\r\nIf-None-Match: W/\"68104920-260-\"2018-02-13T14:16:35.000Z\"\"\r\nIf-Modified-Since: Tue, 13 Feb 2018 14:16:35 GMT\r\n\r\n\r\n"

请求头

请求报文

请求行

请求端(客户端)发送的 HTTP 报文叫做请求报文。请求 报文大致可分为请求行、请求头、(空行)、求主体。

请求行: 请求方法 (Method) + 空格 + 统一资源标识符 (URI) + 空格 + HTTP版本 + CR LF;

请求头:字段名 + 冒号 + 值 + CR LF ; (一般包含host、类型、大小、缓存等等)

空行: 回车符 (CR) + 换行符 (LF) ;

请求体: 由用户自定义添加,如post的body等;

请求行	方法	URI	协议版本
请求头	Host、类型、大小、缓存、等等		
请求体		数据	

响应报文

响应端(服务器端)发送的报文叫做响应报文。响应报文结构与请求报文结构唯一的区别在于第一行用状态信息代替了请求信息,这个状态码说明所请求的资源情况。

状态行: HTTP版本 + 空格 + 状态码 + 空格 + 状态码描述 + CR LF;

响应头: 字段名 + 冒号 + 值 + CR LF;

空行: 回车符 (CR) + 换行符 (LF) ;

响应体: 由用户自定义添加,如post的body等;

响应状态码

状态代码由服务器发出,以响应客户端对服务器的请求。

1xx (信息): 收到请求,继续处理

2xx (成功): 请求已成功接收, 理解和接受

3xx (重定向):需要采取进一步措施才能完成请求 4xx (客户端错误):请求包含错误的语法或无法满足 5xx (服务器错误):服务器无法满足明显有效的请求

HTTP协议与HTTPS协议的区别

上面我们说了HTTP报文,http协议和https协议的主要区别就是是否对HTTP报文加密。

HTTP协议(超文本传输协议)是以明文的形式发送内容,不提供任何加密。明文数据会经过中间代服务器、路由器、wifi热点、通信服务运营商等多个物理节点,如果信息在传输过程中被劫持,传输内容就完全暴露了。劫持者还可以篡改传输的信息且不被双方察觉,这就是中间人攻击。

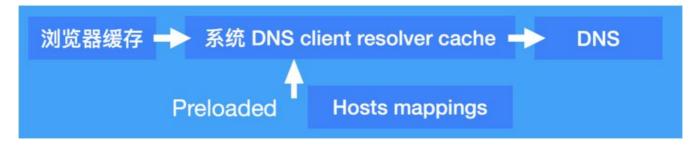
HTTPS协议(安全套接字层超文本传输协议)。

https://zhuanlan.zhihu.com/p/43789231

DNS寻址

域名并不是真正意义上的地址,互联网上每一台计算机的唯一标识是它的IP地。将一个域名转化到IP地址的过程就是DNS解析。

但是在DNS寻址之前,浏览器会根据url查找浏览器的缓存,看看是否有该请的缓存结果和缓存标识,只有当前没有该请求的缓存结果和缓存标识时或者需要协商缓存时,浏览器会开始DNS寻址。所以这里会先详细讲一下浏览器缓存,再讲述DNS。



浏览器缓存

浏览器是否有该请求的缓存结果和缓存标识是取决于该请求的最近一次返回响应报文中的请求结果和缓存标识。缓存标识在响应报文的HTTP头中。

强制缓存

控制强制缓存的字段分别是Expires和Cache-Control ,其中Cache-Conctrol的优先级比Expires高同时存在时,只有Cache-Control生效。

Expires是HTTP/1.0控制网页缓存的字段,其值为服务器返回该请求的结果缓存的到期时间,即再次送请求时,如果客户端的时间小于Expires的值时,直接使用缓存结果。Expires控制缓存的原理是使用户端的时间与服务端返回的时间做对比,如果客户端与服务端的时间由于某些原因(时区不同;客端和服务端有一方的时间不准确)发生误差,那么强制缓存直接失效,那么强制缓存存在的意义就毫意义。

Cache-Control是在HTTP/1.1中最重要的规则,主要用于控制网页缓存,主要取值为:

(1) public: 所有内容都将被缓存(客户端和代理服务器都可缓存)

(2) private: 所有内容只有客户端可以缓存, Cache-Control的默认取值

(3) no-cache: 客户端缓存内容, 但是是否使用缓存则需要经过协商缓存来验证决定

(4) no-store: 所有内容都不会被缓存,即不使用强制缓存,也不使用协商缓存

(5) max-age=xxx (xxx is numeric): 缓存内容将在xxx秒后失效

那么,怎么知道网页的内容是不是缓存的内容呢?

我们可以通过 chrome 的 devtool 的 network 看状态码是什么颜色,如果状态码是灰色的,说明是缓存中的取得的。请求对应的Size值则代表该缓存存放的位置,分别为**from memory cache (内存的缓存)** 和**from disk cache (硬盘中的缓存)** 。浏览器读取缓存的顺序为memory -> disk。

那内存缓存和硬盘缓存有什么区别?

- (1) **内存缓存(from memory cache)**: 会将编译解析后的文件,直接存入该进程的内存中,占据进程一定的内存资源,以方便下次运行使用时的快速读取。但一旦该进程关闭,则该进程的内存则会空。
 - (2) **硬盘缓存(from disk cache)**: 硬盘缓存则是直接将缓存写入硬盘文件中,读取缓存需要对该

存存放的硬盘文件进行I/O操作,然后重新解析该缓存内容,读取复杂,速度比内存缓存慢。

问题来了, 那什么时候有内存缓存什么时候有硬盘缓存呢?

在浏览器中,浏览器会在**js和图片等文件解析执行后直接存入内存缓存中** ,那么当刷新页面时只需直从内存缓存中读取(from memory cache);而**css文件则会存入硬盘文件** 中,所以每次渲染页面都需从硬盘读取缓存(from disk cache)。

协商缓存

协商缓存就是强制缓存失效后,浏览器携带缓存标识向服务器发起请求,由服务器根据缓存标识决定 否使用缓存的过程,主要有以下两种情况:

- (1) 协商缓存生效, 返回304。浏览器从缓存中获取上次的缓存结果。
- (2) 协商缓存失败,返回200和请求结果

协商缓存的报文会携带控制协商缓存的字段,分别有

Last-Modified (服务器响应请求时,返回该资源文件在服务器最后被修改的时间)

If-Modified-Since (客户端再次发起该请求时,携带上次请求返回的Last-Modified值,通过此字值告诉服务器该资源上次请求返回的最后被修改时间)服务器收到该请求,发现请求头含有If-Modified-Since字段,则会根据If-Modified-Since的字段值与该资源在服务器的最后被修改时间做对比,若务器的资源最后被修改时间大于If-Modified-Since的字段值,则重新返回资源,状态码为200;否则返回304,代表资源无更新,可继续使用缓存文件,如下。

Etag (服务器响应请求时,返回当前资源文件的一个唯一标识(由服务器生成))

If-None-Match (客户端再次发起该请求时,携带上次请求返回的唯一标识Etag值,通过此字段值 诉服务器该资源上次请求返回的唯一标识值) 服务器收到该请求后,发现该请求头中含有If-None-Mach,则会根据If-None-Match的字段值与该资源在服务器的Etag值做对比,一致则返回304,代表资 无更新,继续使用缓存文件;不一致则重新返回资源文件,状态码为200

其中Etag / If-None-Match的优先级比Last-Modified / If-Modified-Since高。

DNS寻址

- 1. 一般来说,浏览器会首先查看本地硬盘的 hosts 文件,看看其中有没有和这个域名对应的规则,如有的话就直接使用 hosts 文件里面的 ip 地址。
- 2. 如果在本地的 hosts 文件没有能够找到对应的 ip 地址,浏览器会发出一个 DNS请求到本地DNS服器。本地DNS服务器一般都是你的网络接入服务器商提供,比如中国电信,中国移动。
- 3. 查询你输入的网址的DNS请求到达本地DNS服务器之后,本地DNS服务器会首先查询它的缓存记,如果缓存中有此条记录,就可以直接返回结果,此过程是递归的方式进行查询。如果没有,本地DN服务器还要向DNS根服务器进行查询。
- 4. 根DNS服务器没有记录具体的域名和IP地址的对应关系,而是告诉本地DNS服务器,你可以到域服器上去继续查询,并给出域服务器的地址。这种过程是迭代的过程。
- 5. 本地DNS服务器继续向域服务器发出请求,在这个例子中,请求的对象是.com域服务器。.com域务器收到请求之后,也不会直接返回域名和IP地址的对应关系,而是告诉本地DNS服务器,你的域名解析服务器的地址。
- 6. 最后,本地DNS服务器向域名的解析服务器发出请求,这时就能收到一个域名和IP地址对应关系,

地DNS服务器不仅要把IP地址返回给用户电脑,还要把这个对应关系保存在缓存中,以备下次别的用查询时,可以直接返回结果,加快网络访问。

传输层TCP传输报文

拿到域名对应的IP地址之后,浏览器会以一个随机端口(1024<端口<65535向服务器的WEB程序(常用的有httpd,nginx等)80端口发起TCP的连接请求。这个连接请求到达服器端后(这中间通过各种路由设备,局域网内除外),进入到网卡,然后是进入到内核的TCP/IP协议(用于识别该连接请求,解封包,一层一层的剥开),还有可能要经过Netfilter防火墙(属于内核的块)的过滤,最终到达WEB程序,最终建立了TCP/IP的连接。

TCP三次握手

第一次握手 客户端A将标志位SYN置为1,随机产生一个值为seq=J (J的取值范围为1234567) 的数据到服务器,客户端A进入SYN SENT状态,等待服务端B确认;

第二次握手服务端B收到数据包后由标志位SYN=1知道客户端A请求建立连接,服务端B将标志位SYN ACK都置为1, ack=J+1, 随机产生一个值seq=K, 并将该数据包发送给客户端A以确认连接请求, 务端B进入SYN RCVD状态。

第三次握手客户端A收到确认后,检查ack是否为J+1,ACK是否为1,如果正确则将标志位ACK置为1 ack=K+1,并将该数据包发送给服务端B,服务端B检查ack是否为K+1,ACK是否为1,如果正确则接建立成功,客户端A和服务端B进入ESTABLISHED状态,完成三次握手,随后客户端A与服务端B之可以开始传输数据了

建立了TCP连接之后,发起一个http请求。

接着就是网络层IP协议查询Mac地址、数据达到数据链路层、服务器接受数据、服务器响应请求、服器返回相应文件。

为什么需要三次握手?

主要目的防止server端一直等待,浪费资源。

TCP四次挥手

第一次挥手: 客户端发送一个FIN,用来关闭客户端到服务端的数据传送,客户端进入FIN_WAIT_1 态。

第二次挥手: 服务端收到FIN后,发送一个ACK给客户端,确认序号为收到序号+1 (与-SYN相同,个FIN占用一个序号),服务端进入CLOSE_WAIT状态。(仅仅表示对方不再发送数据了但是还能接数据,)

第三次挥手: 服务端发送一个FIN,用来关闭服务端到客户端的数据传送,Server进入LAST_ACK状

第四次挥手: 客户端收到FIN后, Client进入TIME_WAIT状态,接着发送一个ACK给服务端,确认序为收到序号+1,服务端进入CLOSED状态,完成四次挥手。