



链滴

# 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、资源路径。



### 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**。



### 请求报文

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)、请求主体。

**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；





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接从内存缓存中读取(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是否使用缓存的过程**，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，通过此字段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，若服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，通过此字段告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-Match，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，代表资源无更新，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，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服务器。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录，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DNS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，你可以到域服务器上去继续查询，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域服务器收到请求之后，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

&nbsp;&nbsp; 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务器端后（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，服务端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，如果正确则连接建立成功，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之间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、服务器返回相应文件。

### 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状态。

**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，一个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收数据，）

**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状态。

**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，确认序号为收到序号+1，服务端进入CLOSED状态，完成四次挥手。