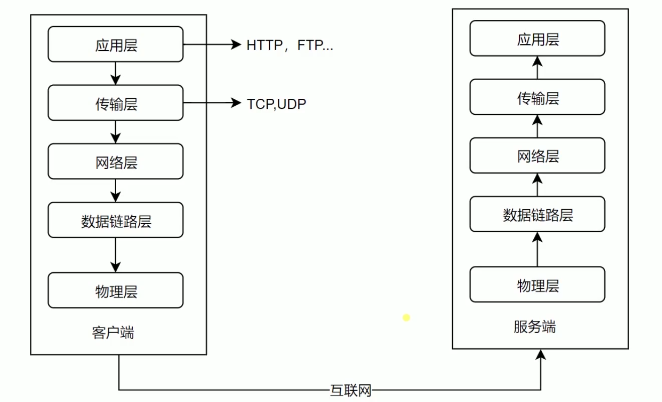


**网络协议分层**



物理层主要作用是定义物理设备如何传输数据。

数据链路层在通讯的实体间建立数据链路连接。

网络层为数据在结点之间传输创建逻辑链路。

传输层向用户提供可靠的端到端（End-to-End）服务；传输层向高层屏蔽了下层数据通信细节。

应用层为应用软件提供服务；构建于TCP协议之上；屏蔽了网络传输的相关协议。

**HTTP发展历史**

0.9

只有一个命令GET

没有HEADER等描述数据的信息

服务器发送完毕，就关闭TCP连接

1.0

增加了命令

增加了status code和header

多字符集支持、多部分发送、权限、缓存等

1.1

**持久连接**

**pipeline**，同时发送多个链接，但是服务器会按顺序处理

增加host和其他一些命令（在同一个物理服务器中可以部署不同的Web服务）

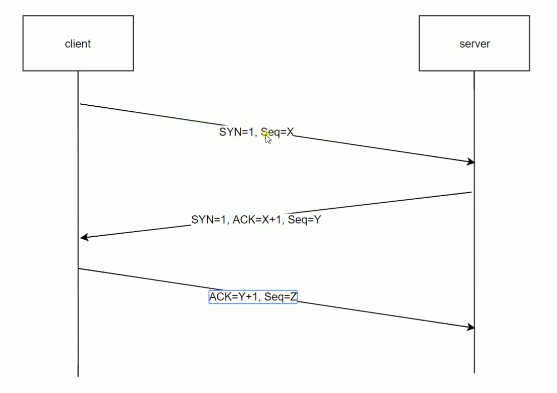
HTTP2

**所有数据以二进制传输**，帧

同一个连接里面发送多个请求**不再需要按照顺序**来

**头信息压缩**以及**推送**等提高效率的功能（请求HTML的同时由服务器主动推送CSS、JS，达到并行的效果）

HTTP三次握手



三次握手避免了无效连接的产生。

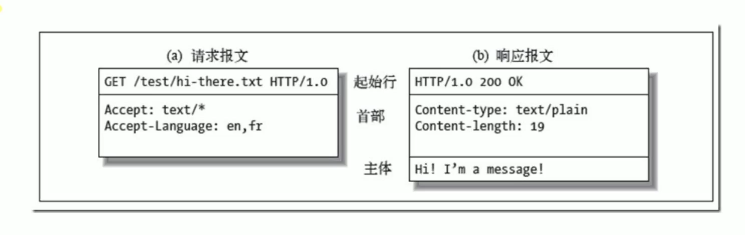
**URI Uniform Resource Identifier 统一资源标识符**

用来唯一标识互联网上的信息资源，包含URL和URN

**URL Uniform Resource Locator 统一资源定位器**，包含schema、认证信息、IP地址、port（不带端口，默认是80）、path（路由）、参数、hash（片段）

URN 永久统一资源定位符

**HTTP报文格式**



HTTP方法，用来定义对于资源的操作；

HTTP Code，定义服务器对请求的处理结果，好的HTTP服务可以通过Code判断结果；

**HTTP客户端**

浏览器、Postman工具、curl等

curl使用方式：**curl -v** [**www.baidu.com**](http://www.baidu.com)

**跨域请求**。服务器端接受到请求，返回时会检查Header是否允许操作，如果要正确返回，需要设置

response. writeHeader(200, {

**‘Access-Control-Allow-Origin’:’\*’** （这种方式不安全）

})

**‘Access-Control-Allow-Origin’:’http://127.0.0.1:8888’** （更安全的方式）

使用**jsonp**实现跨域的原理：在script标签里添加了一个链接，去访问服务器的内容。（link、img、script等标签中设置src属性是允许跨域的）

**CORS跨域限制：**

方法：**GET POST HEADER** 如果要开放其他Method，需要设置

**‘Access-Control-Allow-Header’:’POST, PUT, DELETE’**

Content-Type：**text/plain multipart/form-data application/x-www-form-unlencoded**

请求头限制：自定义头不允许、其他具体参照<https://fetch.spec.whatwg.org/#cors-safelisted-request-header>。如果要开放某些头，使之可以跨域，需要设置

**‘Access-Control-Allow-Header’:’xxxxxx’**

跨域时，在发送实际的请求之前，要发一个OPTION的域请求来获取许可。如果不想每次都发送该请求，设置一个跨域许可最长保留时间：

**‘Access-Control-Max-Age’:’1000’**

缓存头**Cache-Control**

**可缓存性：**

public 所经过的任何中间过程，都能进行缓存，

private 只有发起请求的浏览器都能进行缓存

no-cache 每次请求需要服务器验证才能使用缓存

**到期：**

max-age=<seconds> 缓存过期时间

s-maxage=<seconds> 如果设置，在代理服务器会执行该设置

max-stale=<seconds> 发起方设置，在max-stale的时间内仍可以使用过期缓存（浏览器用不到）

must-revalidate 过期后，必须重新获取数据

proxy-revalidate 针对缓存服务器

其他：

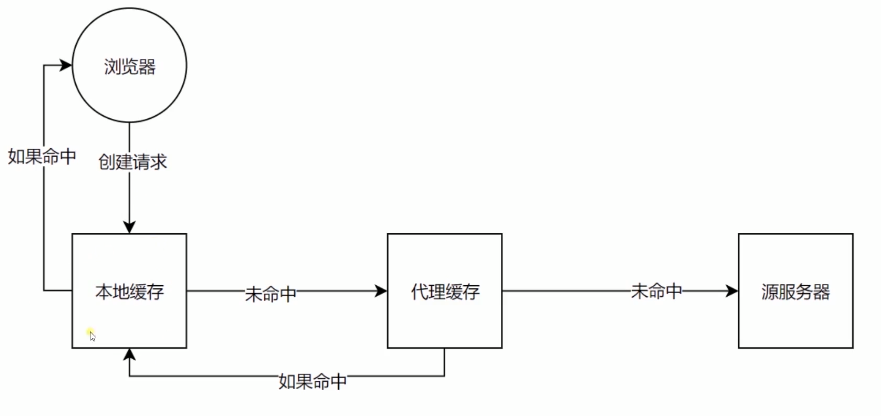
no-store 不缓存

no-transform 针对代理服务器，不能对数据进行压缩等处理

缓存一般会设置为长缓存，如何将服务器端静态资源变化反应到客户端：

在静态文件尾部，加上根据文件内容哈希的**哈希码**，这样如果内容有变化，哈希码也会不同，URL就会相应改变，就相当于请求一个新的资源。

缓存应用流程：



**缓存验证：**

**Last-Modified** 配合If-Modified-Since对应上次修改时间验证是否有更新

**Etag** 配合If-None-Match通过数据签名是否一致验证

具体实现方式：

1. 设置Header的Cache-Control时，配置no-cache，这样每次请求都需要验证

‘Cache-Control’:’max-age=2000000, **no-cache**’

1. 添加Last-Modified或者Etag配置，浏览器在下次请求时就会把对应的验证信息带上If-Modified-Since、If-None-Match
2. 在服务器读取头信息，判断是否匹配，如果匹配，返回304（Not Modified），就不会再去读取新信息

**Cookie和Session**

**Cookie**通过**Set-Cookie**设置；下次请求会自动带上；以键值对的形式保存，可以设置多个。

max-age（保留时间，常用）和expires（过期时间点）设置过期时间

Secure只在https的时候发送

设置HttpOnly属性，无法通过document.cookie访问

同一个主域名下，二级域名下Cookie可以共享，通过设置domain实现。

HTTP长连接

**Connection:keep-alive**

默认生成长连接，TCP连接可以复用，chrome默认允许最大并发的TCP连接数为6个

如果不使用长连接，设置

**‘Connection’:’close’**

HTTP2中引入信道复用，只需要创建一个TCP连接即可完成并发

**数据协商**

**客户端**在头信息中声明想要什么样的数据Accept

Accept 数据类型text/html

Accept-Encoding 编码方式，服务器如何进行压缩

Accept-Language 语言

User-Agent 系统、浏览器相关信息，如返回的是PC端的页面还是移动端的页面

**服务器**返回的数据信息Content

Content-Type 实际返回的数据格式

Content-Encoding 具体使用了哪种编码

Content-Language 实际返回的语言

在html中的form表单中，通过enctype指定数据格式Content-Type

默认是**application/x-www-form-urlencoded**，键值对形式，拼接字符串；

**multipart/form-data**，分为多个部分（文件传输时必须采用此方式，因为文件只能用二进制的形式传输），boundary指定分割每个部分的字符串；

**text/plain**

**Redirect**

在response的Header中设置

‘Location’:’\new’

code使用302，只有302才能跳转，临时跳转；如果确定永久跳转，使用301，下次再请求浏览器自己完成变更。使用301的缓存，服务器端就无法控制，取决于用户端浏览器的缓存方式，所以使用301要慎重。

**CSP Content-Security-Policy**

作用：限制资源获取；报告资源获取越权

限制方式：default-src限制全局，也可以指定资源类型（如connect-src img-src script-src font-src等）

在html中写script，称之为inline script，如果想限制掉这种方式，在response中设置Header

‘Content-Security-Policy’:‘**default-src** http: https:’

如果想限制外链的script，在response中设置Header

‘Content-Security-Policy’:’default-src \’self\’’

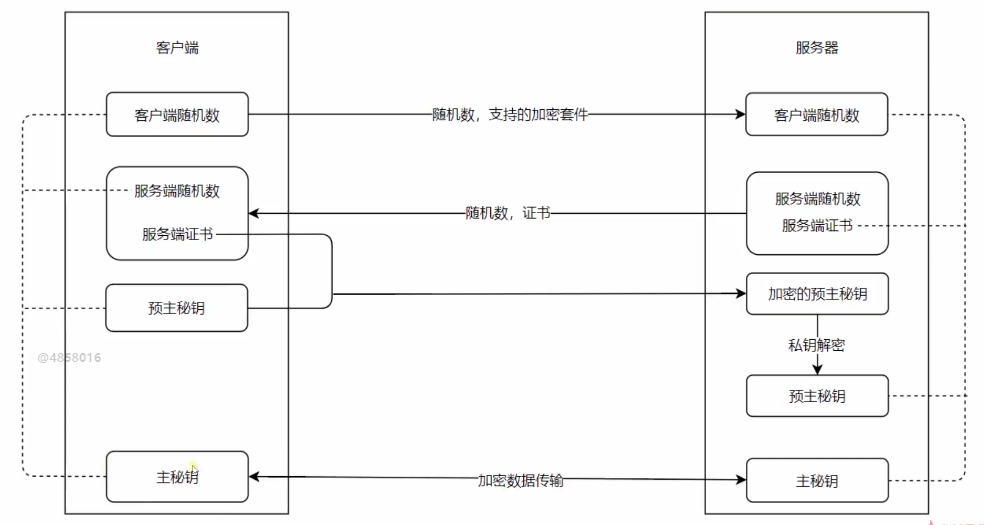
如果想限制form表单不能跳转到其他链接，设置

‘Content-Security-Policy’:’default-src \’self\’;form-action \’self\’’

CSP还可以在meta中指定，但是不推荐这么做。

**HTTPS**

加密 – 服务器保留私钥，客户端用公钥加密，发生在握手阶段



抓取工具只能拿到加密的数据，而主密钥只有客户端和服务器持有，所以无法破解。

HTTPS默认的端口是443

**HTTP2**

优势：信道复用、分帧传输（并发）、Server Push（服务器推送）

服务端推送的配置方法，在response的header中通过link配置需要推送的内容

**‘link’:’</xxx.jpg>; as=image; rel=preload’**

通过Nginx配置HTTP2，即可享用HTTP2的特性。