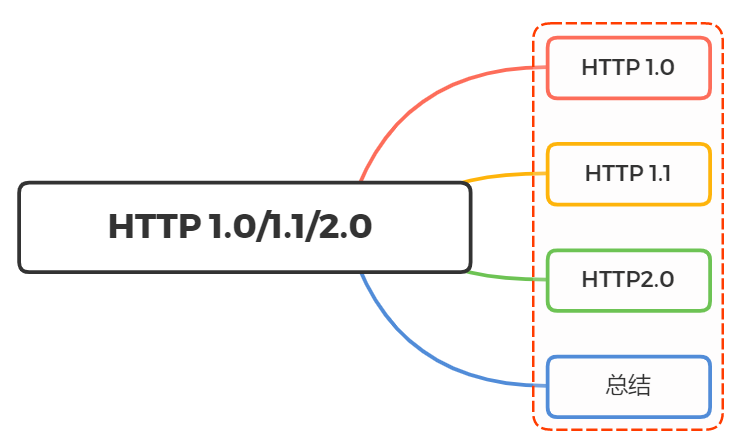
# 面试官：说说 HTTP1.0/1.1/2.0 的区别?



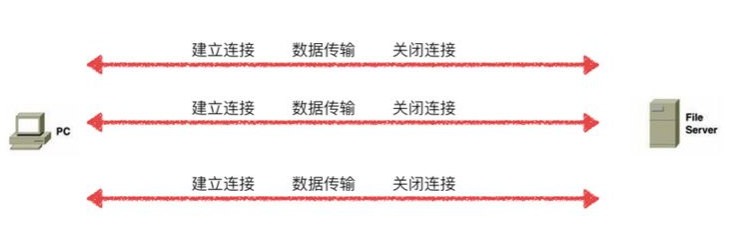
## 一、HTTP1.0

HTTP协议的第二个版本，第一个在通讯中指定版本号的HTTP协议版本

HTTP 1.0 浏览器与服务器只保持短暂的连接，每次请求都需要与服务器建立一个TCP连接

服务器完成请求处理后立即断开TCP连接，服务器不跟踪每个客户也不记录过去的请求

简单来讲，每次与服务器交互，都需要新开一个连接



例如，解析html文件，当发现文件中存在资源文件的时候，这时候又创建单独的链接

最终导致，一个html文件的访问包含了多次的请求和响应，每次请求都需要创建连接、关系连接

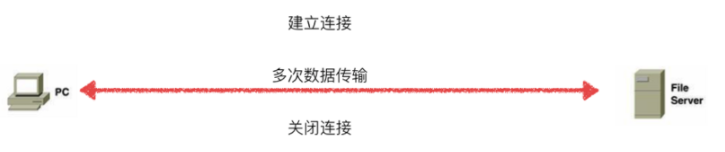
这种形式明显造成了性能上的缺陷

如果需要建立长连接，需要设置一个非标准的Connection字段 Connection: keep-alive

## 二、HTTP1.1

在HTTP1.1中，默认支持长连接（Connection: keep-alive），即在一个TCP连接上可以传送多个HTTP请求和响应，减少了建立和关闭连接的消耗和延迟

建立一次连接，多次请求均由这个连接完成



这样，在加载html文件的时候，文件中多个请求和响应就可以在一个连接中传输

同时，HTTP 1.1还允许客户端不用等待上一次请求结果返回，就可以发出下一次请求，但服务器端必须按照接收到客户端请求的先后顺序依次回送响应结果，以保证客户端能够区分出每次请求的响应内容，这样也显著地减少了整个下载过程所需要的时间

同时，HTTP1.1在HTTP1.0的基础上，增加更多的请求头和响应头来完善的功能，如下：

* 引入了更多的缓存控制策略，如If-Unmodified-Since, If-Match, If-None-Match等缓存头来控制缓存策略
* 引入range，允许值请求资源某个部分
* 引入host，实现了在一台WEB服务器上可以在同一个IP地址和端口号上使用不同的主机名来创建多个虚拟WEB站点

并且还添加了其他的请求方法：put、delete、options...

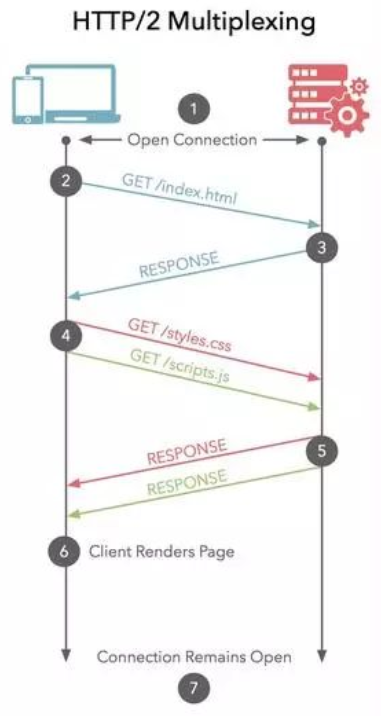
## 三、HTTP2.0

而HTTP2.0在相比之前版本，性能上有很大的提升，如添加了一个特性：

* 多路复用
* 二进制分帧
* 首部压缩
* 服务器推送

### 多路复用

HTTP/2 复用TCP连接，在一个连接里，客户端和浏览器都可以**同时**发送多个请求或回应，而且不用按照顺序一一对应，这样就避免了”队头堵塞”



上图中，可以看到第四步中css、js资源是同时发送到服务端

### 二进制分帧

帧是HTTP2通信中最小单位信息

HTTP/2 采用二进制格式传输数据，而非 HTTP 1.x的文本格式，解析起来更高效

将请求和响应数据分割为更小的帧，并且它们采用二进制编码

HTTP2中，同域名下所有通信都在单个连接上完成，该连接可以承载任意数量的双向数据流

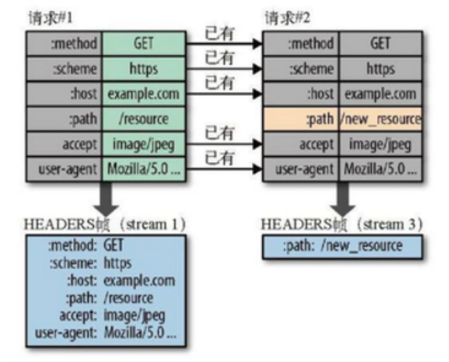
每个数据流都以消息的形式发送，而消息又由一个或多个帧组成。多个帧之间可以乱序发送，根据帧首部的流标识可以重新组装，这也是多路复用同时发送数据的实现条件

### 首部压缩

HTTP/2在客户端和服务器端使用“首部表”来跟踪和存储之前发送的键值对，对于相同的数据，不再通过每次请求和响应发送

首部表在HTTP/2的连接存续期内始终存在，由客户端和服务器共同渐进地更新

例如：下图中的两个请求， 请求一发送了所有的头部字段，第二个请求则只需要发送差异数据，这样可以减少冗余数据，降低开销



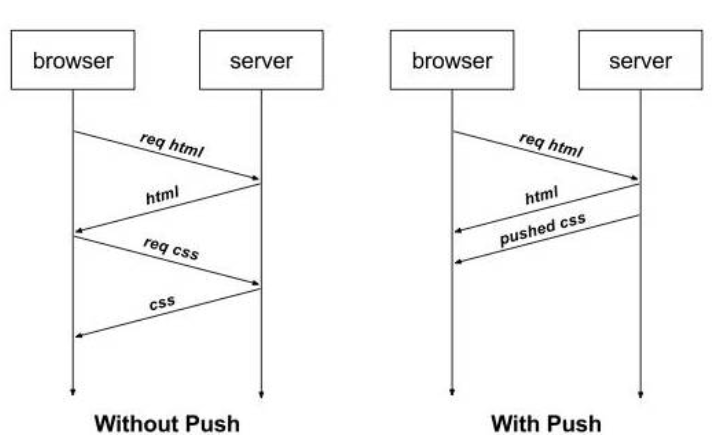
### 服务器推送

HTTP2引入服务器推送，允许服务端推送资源给客户端

服务器会顺便把一些客户端需要的资源一起推送到客户端，如在响应一个页面请求中，就可以随同页面的其它资源

免得客户端再次创建连接发送请求到服务器端获取

这种方式非常合适加载静态资源



## 四、总结

HTTP1.0：

* 浏览器与服务器只保持短暂的连接，浏览器的每次请求都需要与服务器建立一个TCP连接

HTTP1.1：

* 引入了持久连接，即TCP连接默认不关闭，可以被多个请求复用
* 在同一个TCP连接里面，客户端可以同时发送多个请求
* 虽然允许复用TCP连接，但是同一个TCP连接里面，所有的数据通信是按次序进行的，服务器只有处理完一个请求，才会接着处理下一个请求。如果前面的处理特别慢，后面就会有许多请求排队等着
* 新增了一些请求方法
* 新增了一些请求头和响应头

HTTP2.0：

* 采用二进制格式而非文本格式
* 完全多路复用，而非有序并阻塞的、只需一个连接即可实现并行
* 使用报头压缩，降低开销
* 服务器推送

## 参考文献

* https://zh.wikipedia.org/wiki/%E8%B6%85%E6%96%87%E6%9C%AC%E4%BC%A0%E8%BE%93%E5%8D%8F%E8%AE%AE#HTTP/1.0
* https://www.jianshu.com/p/52d86558ca57
* https://segmentfault.com/a/1190000016496448
* https://zhuanlan.zhihu.com/p/26559480