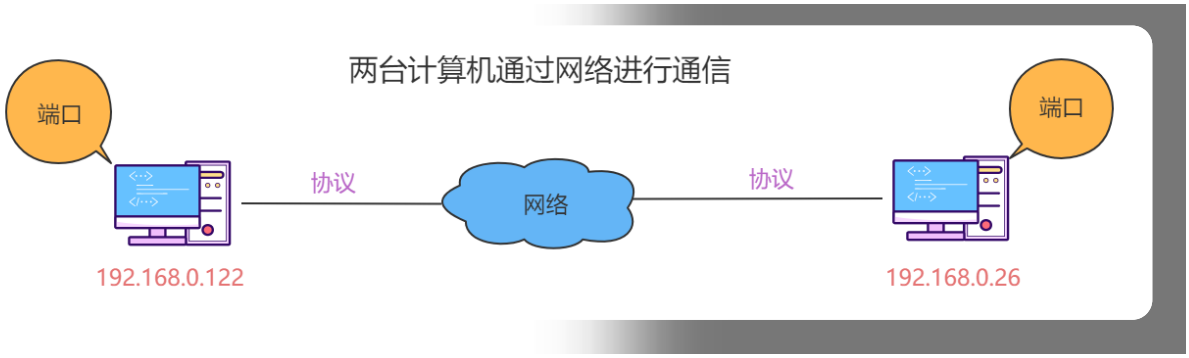
计算机网络通信



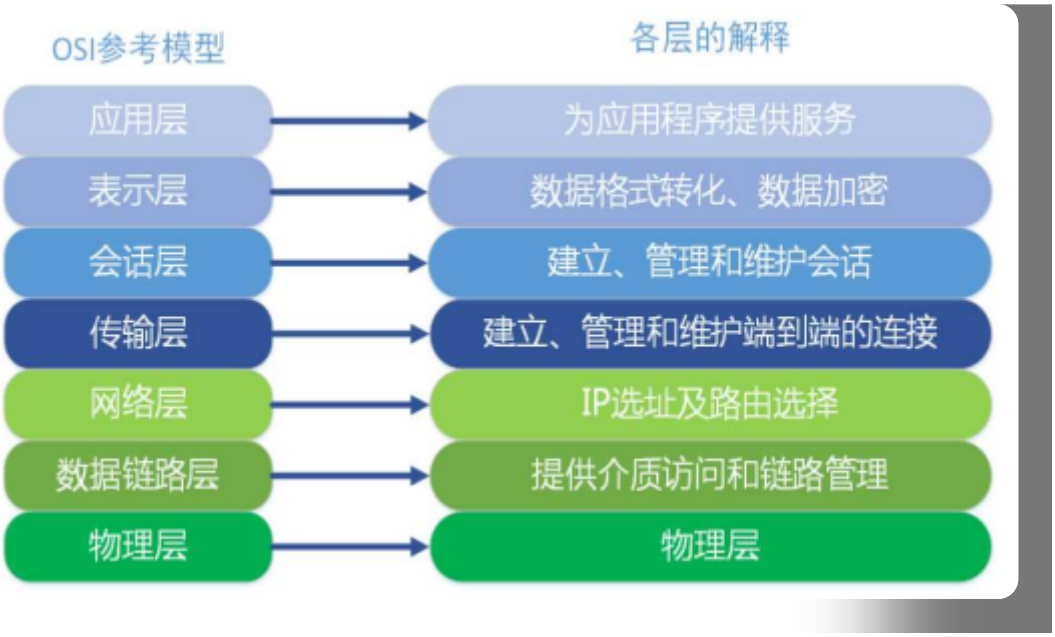
什么是通信协议

简单来说，通信协议就是计算机之间通过网络实现通信时事先达成的一种“约定”；这种“约定”使那些由不同厂商的设备，不同CPU及不同操作系统组成的计算机之间，只要遵循相同的协议就可以实现通信。

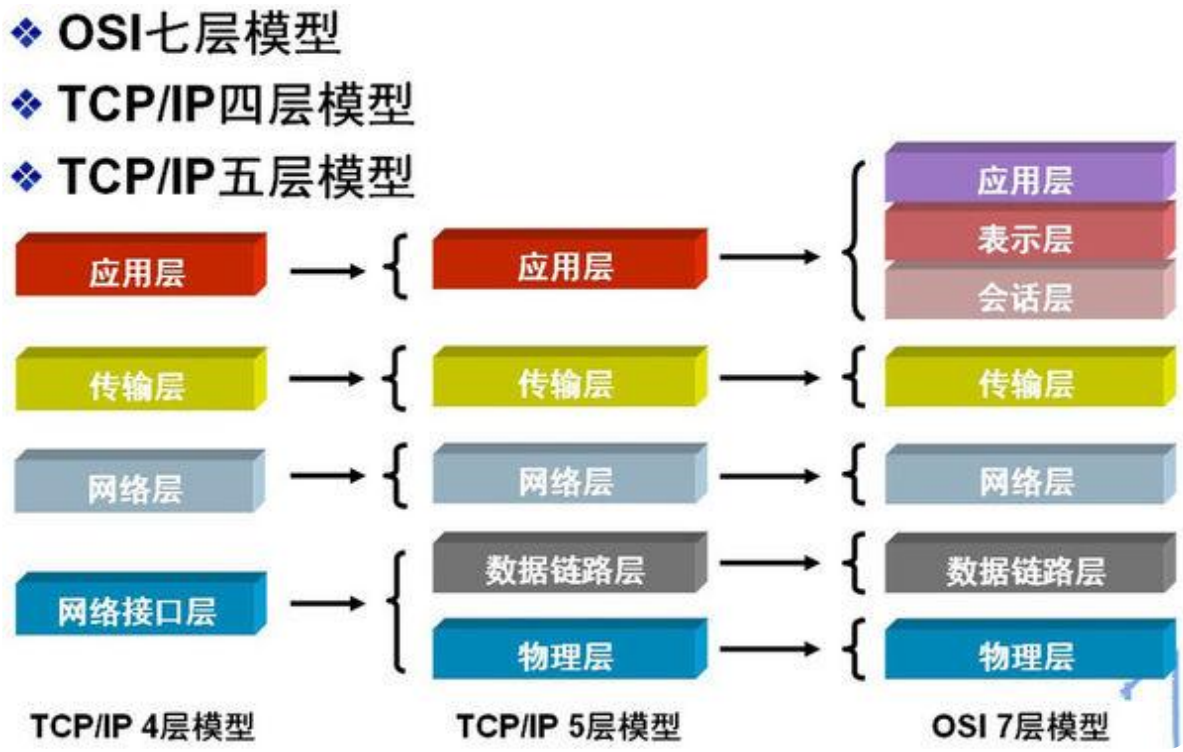
协议可以分很多种，每一种协议都明确界定了它的行为规范：2台计算机之间必须能够支持相同的协议，并且遵循相同的协议进行处理，才能实现相互通信。

协议的标准化

计算机通信诞生之初，系统化与标准化未收到重视，不同厂商只出产各自的网络来实现通信，这样就造成了对用户使用计算机网络造成了很大障碍，缺乏灵活性和可扩展性。为解决该问题，国际标准化组织（ISO）在1978年提出了“开放系统互联参考模型”，即著名的OSI/RM模型（Open System Interconnection/Reference Model）。它将计算机网络体系结构的通信协议划分为七层，自下而上依次为：物理层（Physics Layer）、数据链路层（Data Link Layer）、网络层（Network Layer）、传输层（Transport Layer）、会话层（Session Layer）、表示层（Presentation Layer）、应用层（Application Layer）。



除了标准的OSI七层模型以外，常见的网络层次划分还有TCP/IP四层协议以及TCP/IP五层协议，它们之间的对应关系如下图所示：



实时效果反馈

1.如下对通信协议描述正确的是。

A 通信协议是计算机之间通信时达成的一种“约定”。

B 通信协议是计算机处理数据的一种“约定”。

C 通信协议是计算机存储数据的一种“约定”。

D 通信协议是计算机管理硬件设备的一种“约定”。

2.OSI模型将计算机网络体系结构的通信协议划分为”。

A 4层

B 5层

C 6层

D 7层

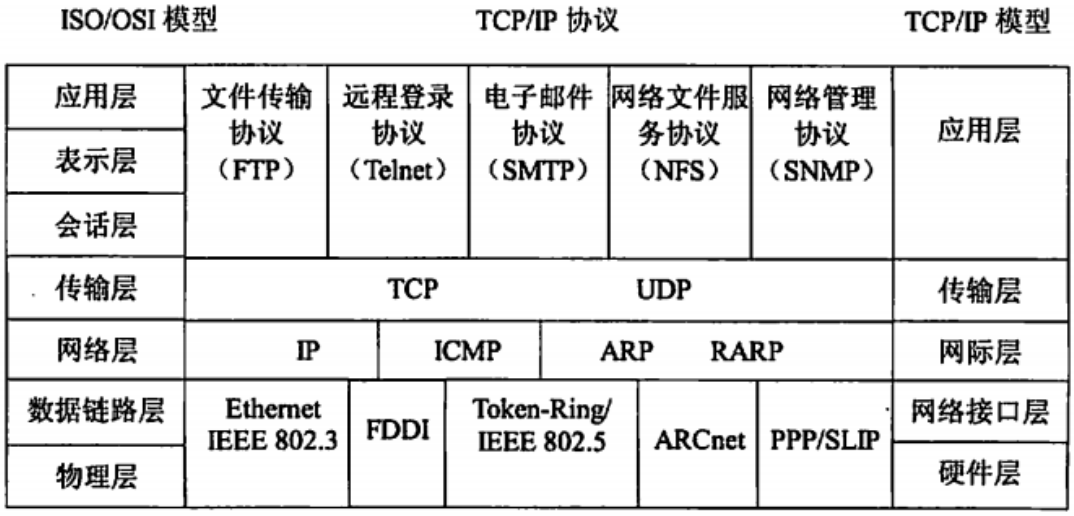
答案

1=>A　2=>D

TCP/IP协议群

什么是TCP/IP协议群

从字面意义上讲，有人可能会认为 TCP/IP 是指 TCP 和 IP 两种协议。实际生活当中有时也确实就是指这两种协议。然而在很多情况下，它只是利用 IP 进行通信时所必须用到的协议群的统称。具体来说，IP 或 ICMP、TCP 或 UDP、TELNET 或 FTP、以及 HTTP 等都属于 TCP/IP 协议。他们与 TCP 或 IP 的关系紧密，是互联网必不可少的组成部分。TCP/IP 一词泛指这些协议，因此，有时也称 TCP/IP 为网络协议群。



ISO/OSI 模型	TCP/IP 协议					TCP/IP 模型
应用层	文件传输协议（FTP）	远程登录协议（Telnet）	电子邮件协议（SMTP）	网络文件服务协议（NFS）	网络管理协议（SNMP）	应用层
表示层						
会话层						
传输层	TCP		UDP			传输层
网络层	IP	ICMP	ARP	RARP		网际层
数据链路层	Ethernet IEEE 802.3	FDDI	Token-Ring/ IEEE 802.5	ARCnet	PPP/SLIP	网络接口层
物理层						硬件层

TCP/IP 模型与 OSI 模型的对比

什么是应用协议

应用协议是定义了运行在不同系统上的应用程序进程如何相互传递报文的协议。

常见的应用协议：

1. FTP协议
   文件传输协议（File Transfer Protocol）。是用于在网络上进行文件传输的一套标准协议，它工作在 OSI 模型的第七层， TCP 模型的第四层， 即应用层，FTP允许用户以文件操作的方式（如文件的增、删、改、查、传送等）与另一主机相互通信。

2. HTTP协议
   一个简单的请求-响应协议，它通常运行在TCP之上。它指定了客户端可能发送给服务器什么样的消息以及得到什么样的响应。

3. DNS协议
   DNS（Domain Name System，域名系统），万维网上作为域名和IP地址相互映射的一个分布式数据库，能够使用户更方便的访问互联网，而不用去记住能够被机器直接知读取的IP地址。

4. SNMP协议
   简单网络管理协议。专门设计用于在 IP 网络管理网络节点（服务器、工作站、路由器、交换机及HUBS等）的一种标准协议，它是一种应用层协议。

什么是传输协议

传输层提供了进程间的逻辑通信，传输层向高层用户屏蔽了下面网络层的核心细节，使应用程序看起来像是在两个传输层实体之间有一条端到端的逻辑通信信道。

1. TCP协议
   TCP：(Transmission Control Protocol)传输控制协议，TCP是一种面向连接的、可靠的、基于字节流的传输层（Transport layer）通信协议。TCP 为提供可靠性传输，实行“顺序控制”或“重发控制”机制。此外还具备“流控制（流量控制）”、“拥塞控制”、提高网络利用率等众多功能。

2. UDP协议
   UDP：(User Datagram Protocol)的简称， 是不具有可靠性的数据报文协议。虽然可以确保发送消息的大小，却不能保证消息一定会到达。

TCP与UDP比较

TCP 和 UDP 的优缺点无法简单地、绝对地去做比较：TCP 用于在传输层有必要实现可靠传输的情况；UDP 主要用于那些对高速传输和实时性有较高要求的通信或广播通信。TCP 和 UDP 应该根据应用的目的按需使用。

什么是网际协议

网际协议是一个网络层协议，它包含寻址信息和控制信息，可使数据包在网络中路由。

1. IP协议
   IP协议（Internet Protocol网际互连协议），它主要是完成两个任务，一个是寻找地址，第二个是管理分割数据片。

2. ICMP协议
   ICMP 的主要功能包括，确认 IP 包是否成功送达目标地址，通知在发送过程当中 IP 包被废弃的具体原因，改善网络设置等。

3. ARP协议
   ARP协议(Address Resolution Protocol，地址解析协议)是一个位于TCP/IP协议群中的网络层，负责将某个IP地址解析成对应的MAC地址。主机将包含目标IP地址信息的ARP请求广播到网络中的所有主机，并接收返回消息，以此确定目标IP地址的物理地址。

实时效果反馈

1.如下说法正确的是。

A 应用协议是操作系统上的应用程序进程如何相互传递报文的协议。

B 应用协议是计算机之间通信时传输报文的协议。

C 应用协议是计算机运算时处理数据的协议。

D 应用协议是计算机通信时相互连接的协议。

2.如下哪个协议不属于应用协议。

A FTP

B HTTP

C TCP

D DNS

答案

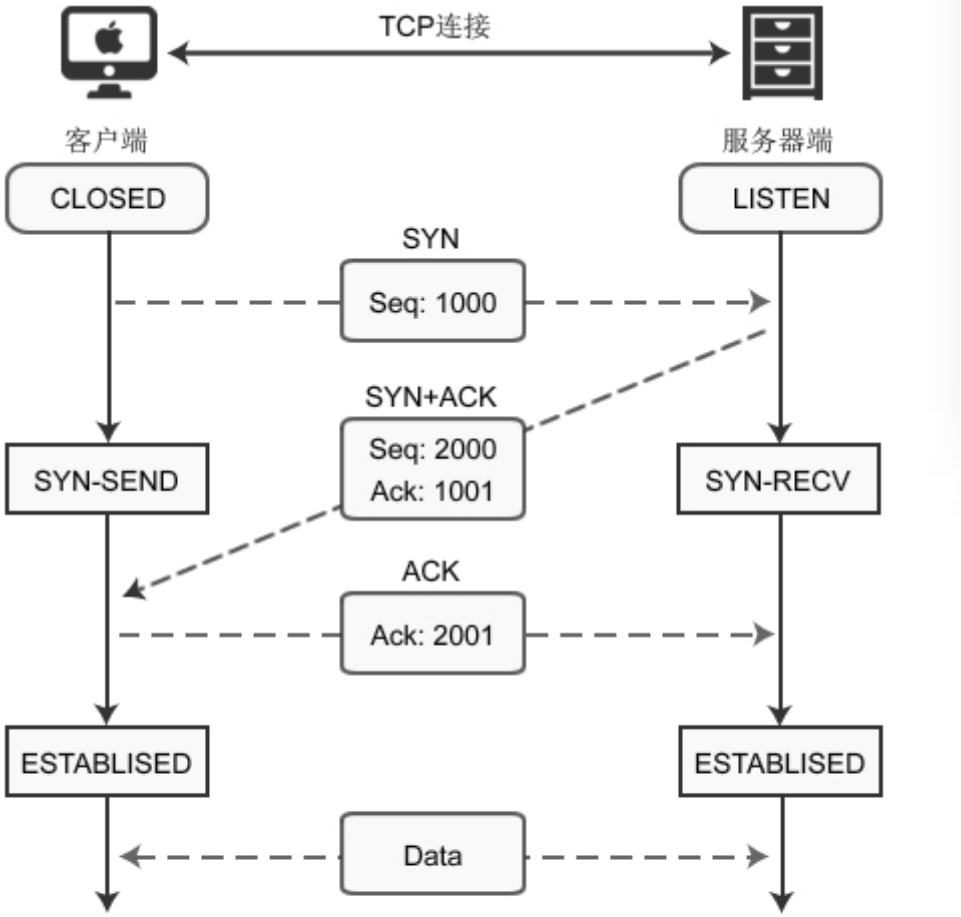
1=>A　2=>C

TCP协议传输特点

TCP是一个可靠的传输协议，在创建连接时会经历三次握手，在断开连接时会经历四次挥手。TCP 三次握手，其实就是 TCP 应用在发送数据前，通过 TCP 协议跟通信对方协商好连接信息，建立起TCP的连接关系。

建立连接的三次握手

所谓三次握手是指建立一个 TCP 连接时需要客户端和服务器端总共发送三个包以确认连接的建立。



三次握手建立连接

TCP建立连接时要传输三个数据包，俗称三次握手（Three-way Handshaking）。可以形象的比喻为下面的对话：

设备A：“你好，设备B，我这里有数据要传送给你，建立连接吧。”

设备B：“好的，我这边已准备就绪。”

设备A：“谢谢你受理我的请求。”

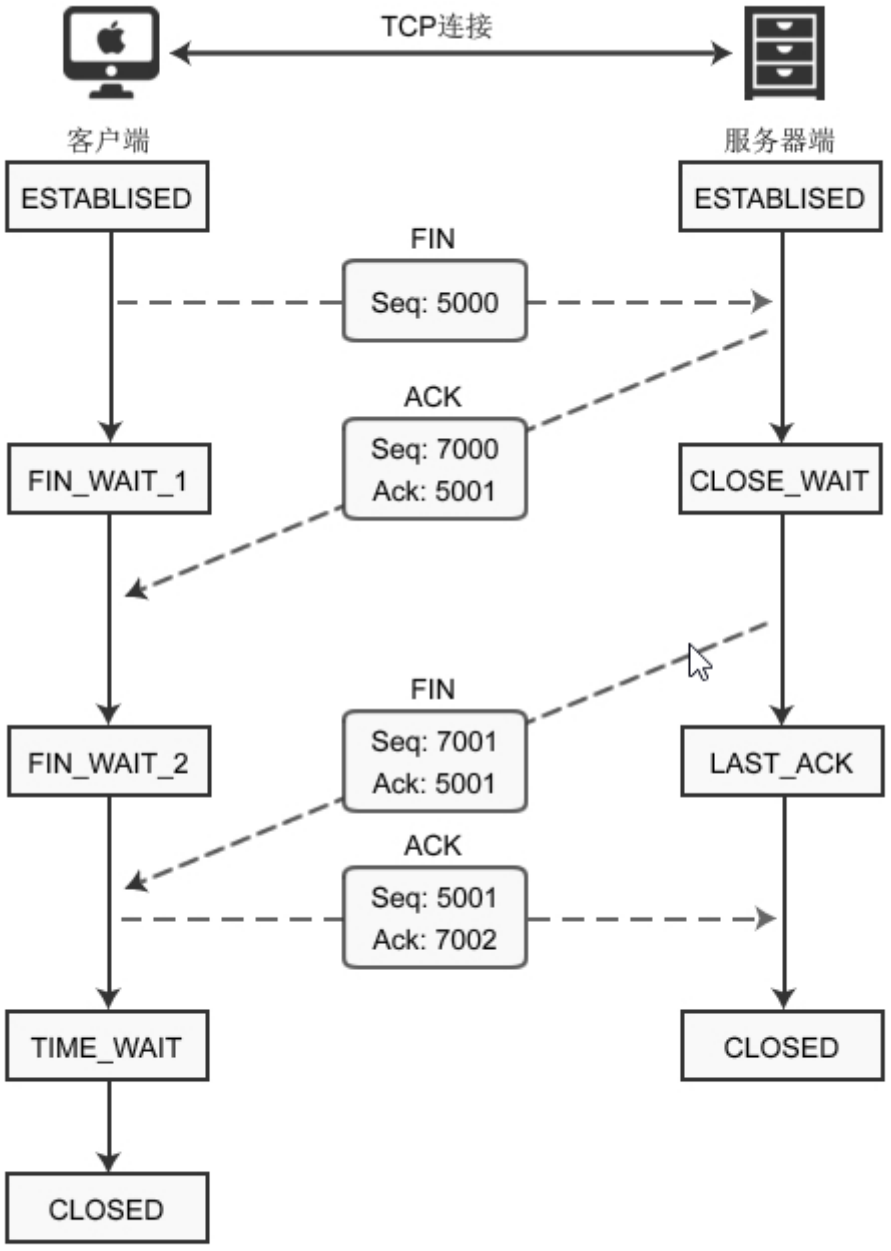
- 第一次握手：客户端发送 SYN报文，并进入 SYN_SENT状态，等待服务器的确认；

- 第二次握手：服务器收到 SYN报文，需要给客户端发送 ACK 确认报文，同时服务器也要向客户端发送一个 SYN 报文，所以也就是向客户端发送 SYN + ACK 报文，此时服务器进入 SYN_RCVD状态；

- 第三次握手：客户端收到 SYN + ACK报文，向服务器发送确认包，客户端进入 ESTABLISHED 状态。待服务器收到客户端发送的 ACK 包也会进入 ESTABLISHED状态，完成三次握手。

断开连接的四次挥手

四次挥手即终止TCP连接，就是指断开一个TCP连接时，需要客户端和服务端总共发送4个包以确认连接的断开。当我们的应用程序不需要数据通信了，就会发起断开 TCP 连接。建立一个连接需要三次握手，而终止一个连接需要经过四次挥手。



四次挥手断开连接

断开连接需要四次握手，可以形象的比喻为下面的对话：

设备A：“任务处理完毕，我希望断开连接。”

设备B：“哦，是吗？请稍等，我准备一下。”

等待片刻后……

设备B：“我准备好了，可以断开连接了。”

设备A：“好的，谢谢合作。”

- 第一次挥手。客户端发起 FIN包（FIN = 1）,客户端进入 FIN_WAIT_1 状态。TCP 规定，即使 FIN包不携带数据，也要消耗一个序号。

- 第二次挥手。服务器端收到 FIN包，发出确认包 ACK（ack = u + 1），并带上自己的序号 seq=v，服务器端进入了 CLOSE_WAIT 状态。这个时候客户端已经没有数据要发送了，不过服务器端有数据发送的话，客户端依然需要接收。客户端接收到服务器端发送的 ACK后，进入了FIN_WAIT_2状态。

- 第三次挥手。服务器端数据发送完毕后，向客户端发送FIN包（seq=w ack=u+1），半连接状态下服务器可能又发送了一些数据，假设发送 seq 为 w。服务器此时进入了 LAST_ACK状态。

- 第四次挥手。客户端收到服务器的 FIN包后，发出确认包（ACK=1，ack=w+1），此时客户端就进入了TIME_WAIT状态。注意此时 TCP 连接还没有释放，必须经过2*MSL后，才进入CLOSED状态。而服务器端收到客户端的确认包ACK后就进入了CLOSED状态，可以看出服务器端结束 TCP 连接的时间要比客户端早一些。

实时效果反馈

1.TCP协议在建立连接时需要握手。

A 一次。

B 两次。

C 三次。

D 四次。

2.TCP协议在断开连接时需要挥手。

A 一次。

B 两次。

C 三次。

D 四次。

答案

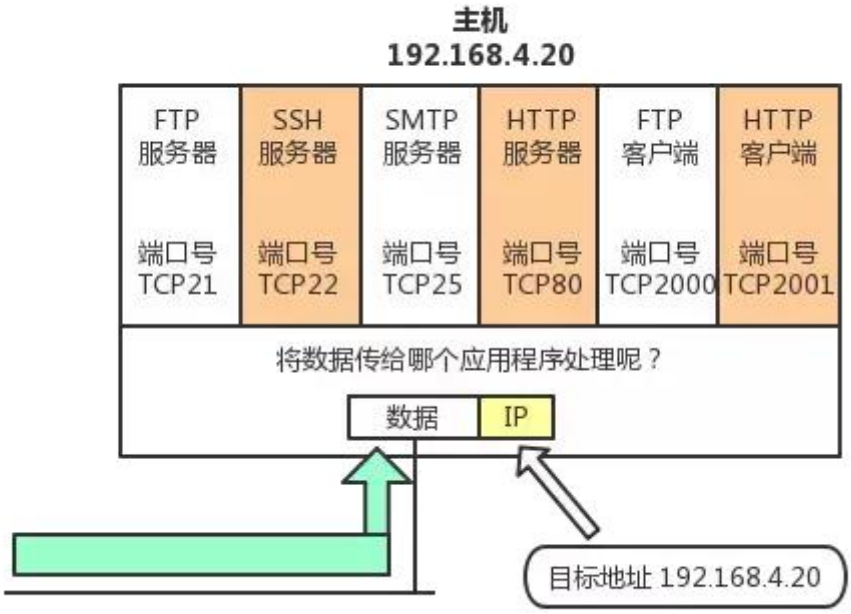
1=>C　2=>D

服务端口

端口作用

端口号用来识别计算机中进行通信的应用程序。因此，它也被称为程序地址。

一台计算机上同时可以运行多个程序。传输层协议正是利用这些端口号识别本机中正在进行通信的应用程序，并准确地进行数据传输。



端口分配

操作系统中一共提供了0~65535可用端口范围。

按端口号分类：

公认端口（Well Known Ports）：从0到1023，它们紧密绑定（binding）于一些服务。通常这些端口的通讯明确表明了某种服务的协议。例如：80端口实际上总是HTTP通讯。

注册端口（Registered Ports）：从1024到65535。它们松散地绑定于一些服务。也就是说有许多服务绑定于这些端口，这些端口同样用于许多其它目的。例如：许多系统处理动态端口从1024左右开始。

常见的应用层协议与端口分配

常用服务	协议	端口号
POP3	TCP	110
IMAP	TCP	143
SMTP	TCP	25
Telnet	TCP	23
终端服务	TCP	3389
PPTP	TCP	1723
HTTP	TCP	80
FTP(控制)	TCP	21
FTP(数据)	TCP	20
HTTPS	TCP	443
NTP	UDP	123
RADIUS	UDP	1645
DHCP	UDP	67
DNS	UDP	53
DNS	TCP	53
SNMP	UDP	161
ipsec	UDP	500
TFTP	UDP	69
L2TP	UDP	1701

实时效果反馈

1.如下说法正确的是。

A 端口号用来识别计算机通信地址的。

B 端口号用来识别计算机中进行通信的不同应用程序。

C 端口号用来识别计算机中传输协议的。

D 端口号用来识别计算机中连接协议的。

答案

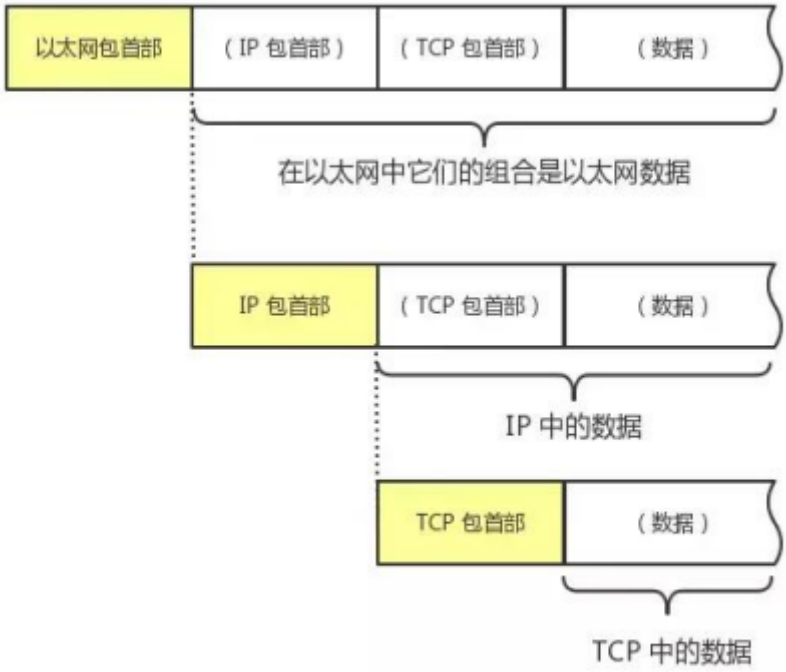
1=>B

数据包与处理流程

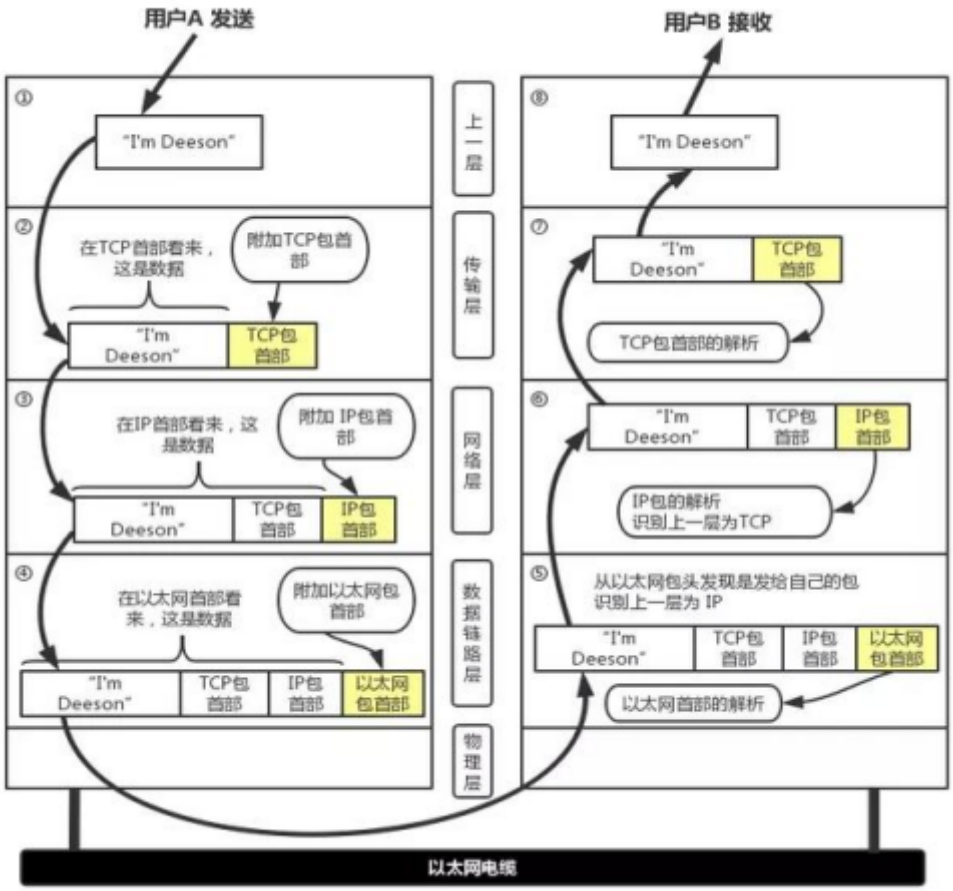
什么是数据包

通信传输中的数据单位，一般也称“数据包”。在数据包中包括：包、帧、数据包、段、消息。

网络中传输的数据包由两部分组成：一部分是协议所要用到的首部，另一部分是上一层传过来的数据。首部的结构由协议的具体规范详细定义。在数据包的首部，明确标明了协议应该如何读取数据。反过来说，看到首部，也就能够了解该协议必要的信息以及所要处理的数据。包首部就像协议的脸。



数据包处理流程



实时效果反馈

1.通信传输中的数据单位，一般也称。

A 数据流。

B 数据项。

C 数据帧。

D 数据包。

答案

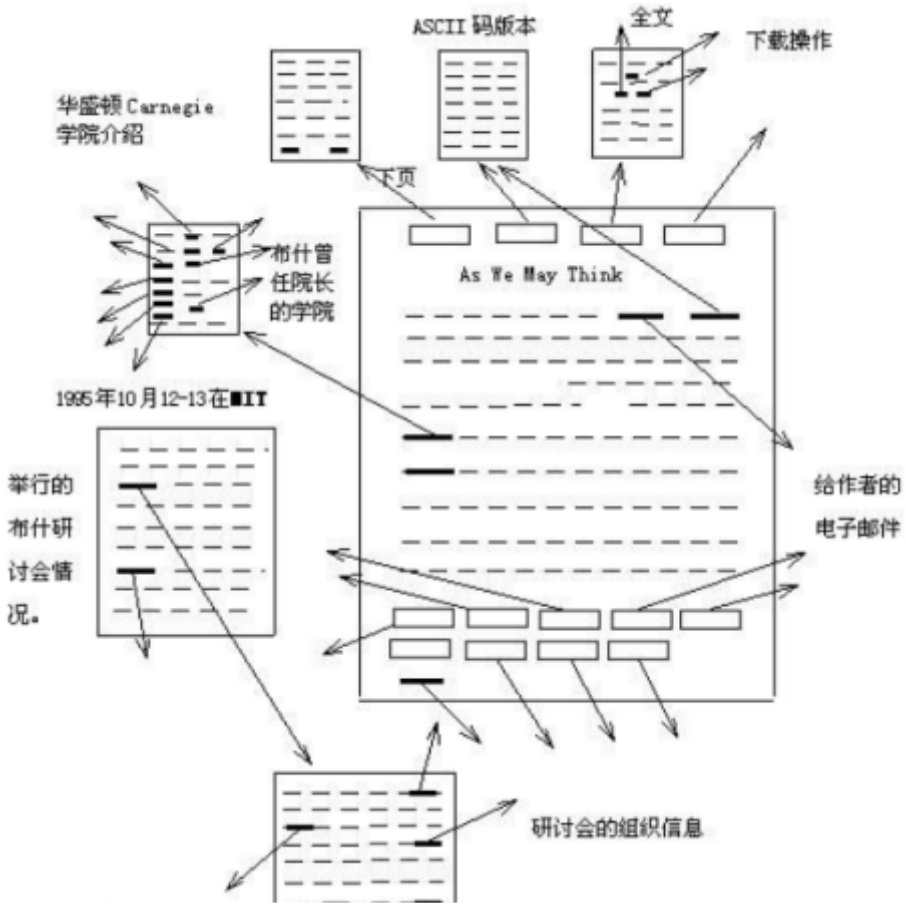
1=>D

HTTP协议简介

HTTP协议介绍

什么是超文本

超文本是用超链接的方法，将各种不同空间的文字信息组织在一起的网状文本。超文本更是一种用户界面范式，用以显示文本及与文本之间相关的内容。

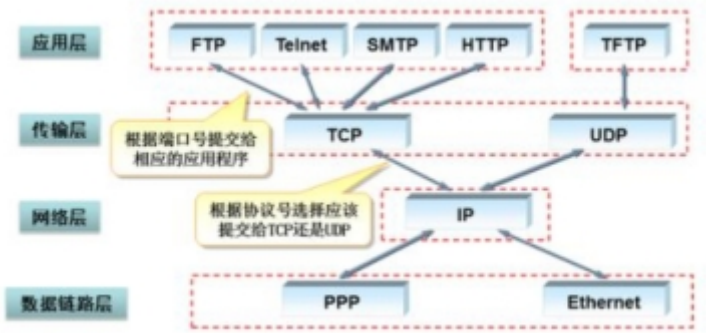


什么是HTTP协议

HTTP协议是Hyper Text Transfer Protocol（超文本传输协议）的缩写, HTTP是万维网（WWW:World Wide Web）的数据通信的基础。

HTTP是一个简单的请求-响应协议，它通常运行在TCP之上。它指定了客户端可能发送给服务器什么样的消息以及得到什么样的响应。

HTTP是一个基于TCP/IP通信协议来传递数据（HTML 文件, 图片文件, 查询结果等）。



实时效果反馈

1.HTTP协议是指。

A 数据流传输协议。

B 对象传输协议。

C 文本传输协议。

D 超文本传输协议。

答案

1=>D

HTTP协议特点

支持客户端/服务端模式：

HTTP协议支持客户端服务端模式，需要使用浏览器作为客户端来访问服务端。

简单快速：

客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、POST等。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快

灵活：

HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type（Content-Type是HTTP包中用来表示内容类型的标识）加以标记。

无连接：

每次请求一次，释放一次连接。所以无连接表示每次连接只能处理一个请求。优点就是节省传输时间，实现简单。我们有时称这种无连接为短连接。对应的就有了长链接，长连接专门解决效率问题。当建立好了一个连接之后，可以多次请求。但是缺点就是容易造成占用资源不释放的问题。当HTTP协议头部中字段Connection：keep-alive表示支持长链接

单向性：

服务端永远是被动的等待客户端的请求。

无状态：

HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。为了解决HTTP协议无状态，于是，两种用于保持HTTP连接状态的技术就应运而生了，一个是Cookie，而另一个则是Session。

实时效果反馈

1.如下那个不是HTTP协议的特点。

A 无连接。

B 单向性。

C 断点续传。

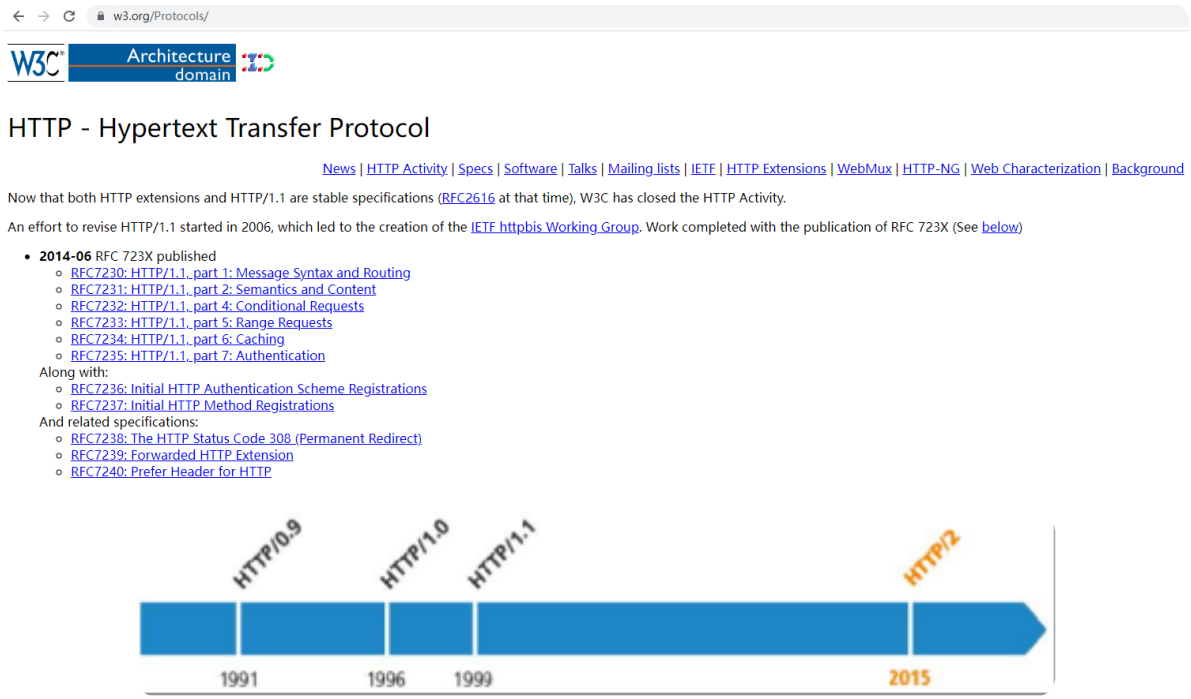
D 无状态。

答案

1=>C

HTTP协议发展和版本

http协议在1991年发布第一个版本版本号为0.9。随后WWW联盟（WWW Consortium-W3C）于1994年成立，http协议被纳入到W3C组织中进行维护和管理。

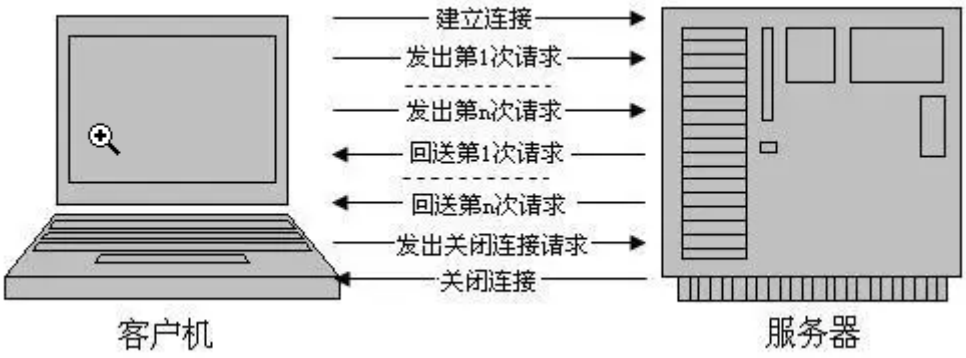


HTTP1.0

最早在1996年在网页中使用，内容简单，所以浏览器的每次请求都需要与服务器建立一个TCP连接，服务器处理完成后立即断开TCP连接（无连接），服务器不跟踪每个客户端也不记录过去的请求（无状态）,请求只能由客户端发起（单向性）。

HTTP1.1

到1999年广泛在各大浏览器网络请求中使用，HTTP/1.0中默认使用Connection: close。在HTTP/1.1中已经默认使用Connection: keep-alive（长连接），避免了连接建立和释放的开销，但服务器必须按照客户端请求的先后顺序依次回送相应的结果，以保证客户端能够区分出每次请求的响应内容。通过Content-Length字段来判断当前请求的数据是否已经全部接收。不允许同时存在两个并行的响应。1.1中最重要的一个特点是支持“长连接”，即“一次连接可以多次请求”。

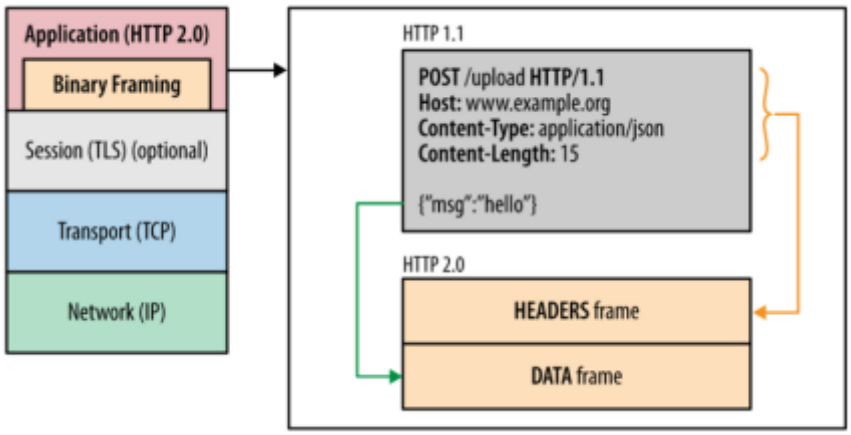


HTTP2.0

- 长连接
  在HTTP/2中，客户端向某个域名的服务器请求页面的过程中，只会创建一条TCP连接，即使这页面可能包含上百个资源。 单一的连接应该是HTTP2的主要优势，单一的连接能减少TCP握手带来的时延 。HTTP2中用一条单一的长连接，避免了创建多个TCP连接带来的网络开销，提高了吞吐量。

- 多路复用 (Multiplexing)

  HTTP2.0中所有加强性能的核心是二进制传输，在HTTP1.x中，我们是通过文本的方式传输数据。在HTTP2.0中引入了新的编码机制，所有传输的数据都会被分割，并采用二进制格式编码。

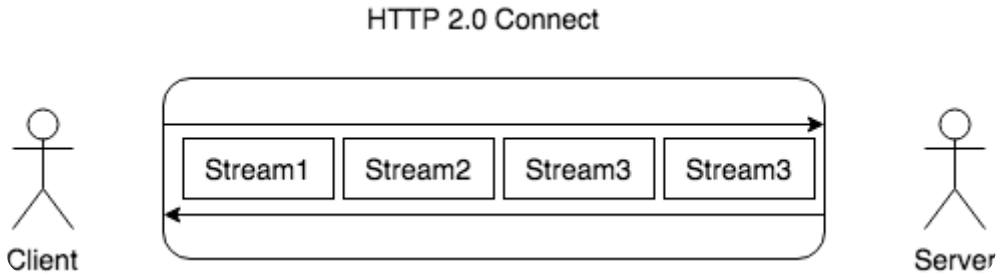


  多路复用，连接共享。不同的request可以使用同一个连接传输（最后根据每个request上的id号组合成正常的请求）。

  HTTP2.0中，有两个概念非常重要：帧（frame）和流（stream）。

  帧是最小的数据单位，每个帧会标识出该帧属于哪个流，流是多个帧组成的数据流。

  所谓多路复用，即在一个TCP连接中存在多个流，即可以同时发送多个请求，对端可以通过帧中的表示知道该帧属于哪个请求。在客户端，这些帧乱序发送，到对端后再根据每个帧首部的流标识符重新组装。通过该技术，可以避免HTTP旧版本的队头阻塞问题，极大提高传输性能。



- 首部压缩（Header Compression）

由于1.1中header带有大量的信息，并且得重复传输，2.0使用encoder来减少需要传输的hearder大小。

- 服务端推送（Server Push）
  在HTTP2.0中，服务端可以在客户端某个请求后，主动推送其他资源。
  可以想象一下，某些资源客户端是一定会请求的，这时就可以采取服务端push的技术，提前给客户端推送必要的资源，就可以相对减少一点延迟时间。在浏览器兼容的情况下也可以使用prefetch。

- 更安全
  HTTP2.0使用了tls的拓展ALPN做为协议升级，除此之外，HTTP2.0对tls的安全性做了近一步加强，通过黑名单机制禁用了几百种不再安全的加密算法。

http1.1和http2.0区别

协议	传输格式	多路复用	首部压缩	服务器推送	请求优先级
http1.1	文本	×	×	×	×
http2.0	二进制帧	√	√	√	√

实时效果反馈

1.如下那个不是HTTP2.0协议的新特性。

A 短连接。

B 多路复用。

C 首部压缩。

D 服务推送。

答案

1=>A

HTTP协议中URI、URL、URN

URI

URI：（Uniform Resource Identifier），统一资源标识符，是一个用于标识互联网某个唯一资源的字符串名称。

URL和URN都是URI的子集。

URI是个纯粹的语法结构，用于指定标识Web资源的字符串的各个不同部分。他不属于定位符，因为根据该标识符无法定位任何资源。

URL

URL(Uniform Resource Location)，统一资源定位符，可以帮助我们定位互联网上的某一个唯一资源，相当于是互联网资源的身份证号。

URL的五个元素包括在一个简单的地址中：

- 传送协议。
- 服务器。（通常为域名或者IP地址）
- 端口号。（以数字方式表示，若为HTTP的默认值":80"可省略）
- 请求资源路径。
- 传递数据。（在URL中传递数据是以key=value的结构进行数据绑定，以"?"字符为起点，每个参数以"&"隔开，再以"="分开参数名称与数据，通常以UTF8的URL编码，避开字符冲突的问题）

举个栗子：

```
http://www.itbaizhan.cn:80/course/id/18.html?a=3&b=4
```

其中：

1. http, 是协议；
2. www.itbaizhan.cn，是服务器域名；
3. 80，是服务器上的默认网络端口号，默认不显示；
4. /course/id/18.html，是路径(URI：直接定位到对应的资源)；
5. ?a=3&b=4，请求时传递的数据；

URN

URN（Uniform Resource Name，）统一资源名称，其目的是通过提供一种途径，用于在特定的命名空间资源的标识，以补充网址。

举个栗子：

```
www.itbaizhan.cn:80/course/id/18.html?a=3&b=4
```

URN是URI的子集，包括名字(给定的命名空间内)，但是不包括访问方式来。

实时效果反馈

1.URI是。

A 统一资源定位符。

B 统一资源标识符。

C 统一资源名称。

D 统一资源路径。

2.URL是。

A 统一资源定位符。

B 统一资源标识符。

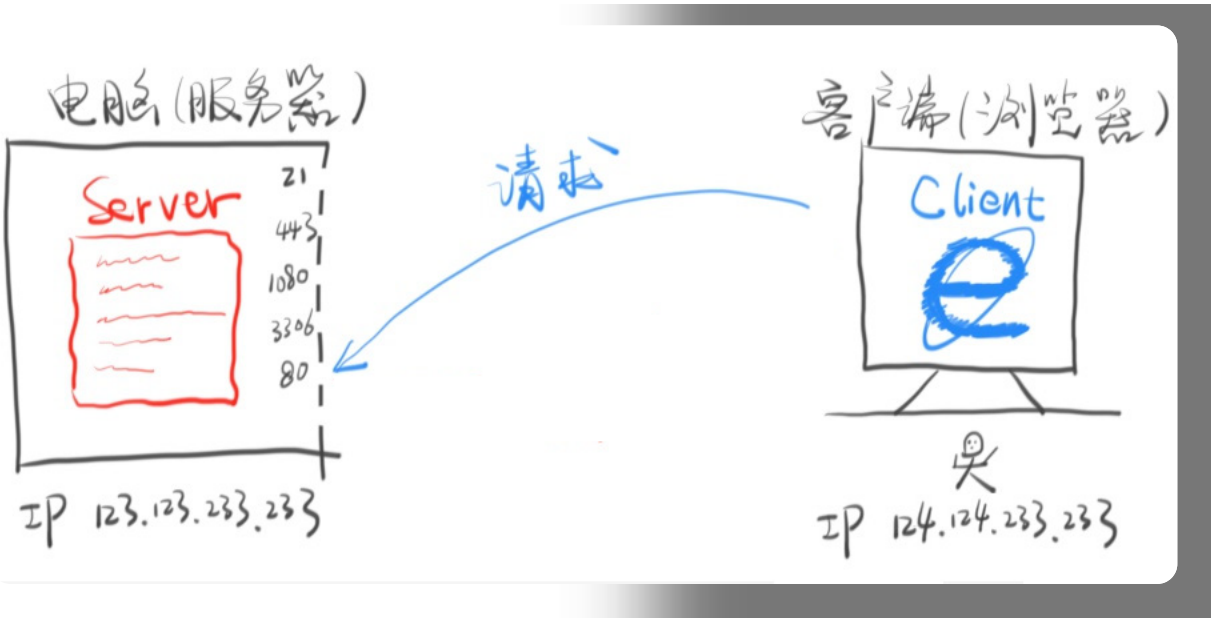
C 统一资源名称。

D 统一资源路径。

答案

1=>B　2=>A

HTTP协议的请求



HTTP协议中的请求信息

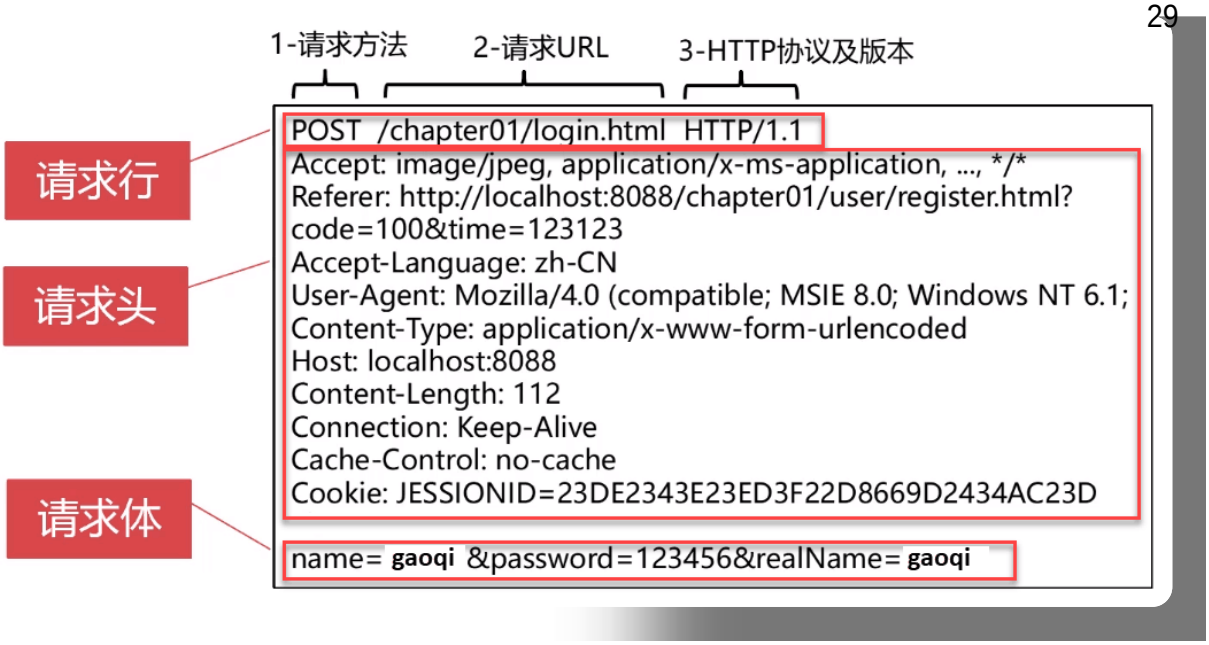
打开一个网页需要浏览器发送很多次Request

- 当你在浏览器输入URL http://www.itbaizhan.cn 的时候，浏览器发送一个Request去获取 http://www.itbaizhan.cn 的html. 服务器把Response发送回给浏览器。
- 浏览器分析Response中的 HTML，发现其中引用了很多其他文件，比如图片，CSS文件，JS文件。
- 浏览器会自动再次发送Request去获取图片，CSS文件，或者JS文件。
- 等所有的文件都下载成功后。 网页就被显示出来了。

请求状态分析(request)

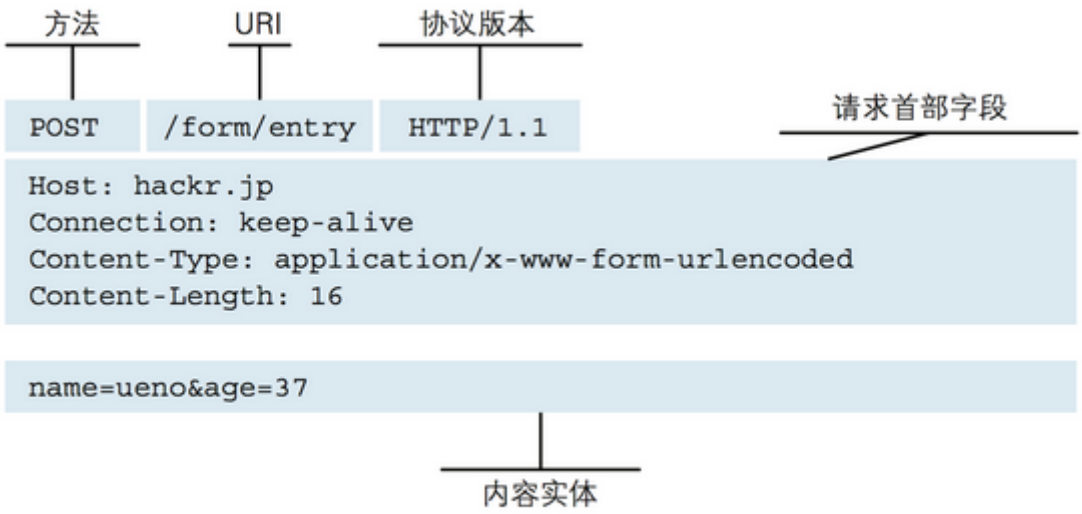
Request 消息分为3部分，第一部分叫Request line, 第二部分叫Request header, 第三部分是Request body。Request header和Request body之间有个空行。

客户端发送一个HTTP请求到服务器的请求消息包括以下格式：

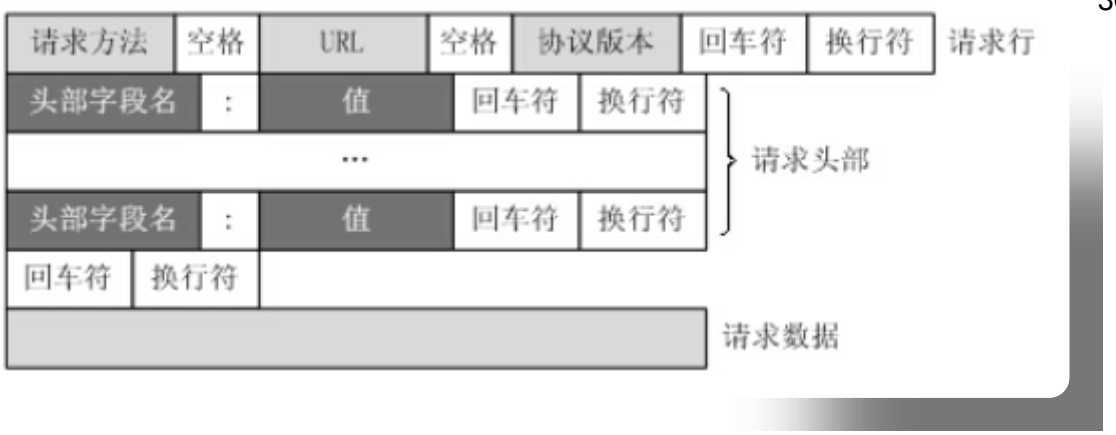


我们以访问：http://www.itbaizhan.cn/course/id/18.html为例，查看请求状态：

```
GET /course/id/18.html HTTP/1.1
Host: www.itbaizhan.com
Connection: keep-alive
sec-ch-ua: "Chromium";v="94", "Google Chrome";v="94", ";Not A Brand";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.71 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
Cookie: goeasyNode=3; UM_distinctid=17a386c56f60-0cf4cb54cc9fe4-6373267-1fa400-17a386c56f7d15; 53gid2=10728050664011; 53revisit=1624444262186; _uab_collina=162493408960900530419057;
11%3A%2215910989505%22%3B%7D; token=208D5526880A7994D424E1EED98DC4B0; goeasyNode=2; CNZZDATA1273229192=143616405-1624444213-%7C1637277126
```



图：请求报文的构成



实时效果反馈

1.当以GET方式发送请求时，请求共分为。

A 两部分。

B 三部分。

C 四部分。

D 五部分。

2.当以POST方式发送请求时，请求共分为。

A 两部分。

B 三部分。

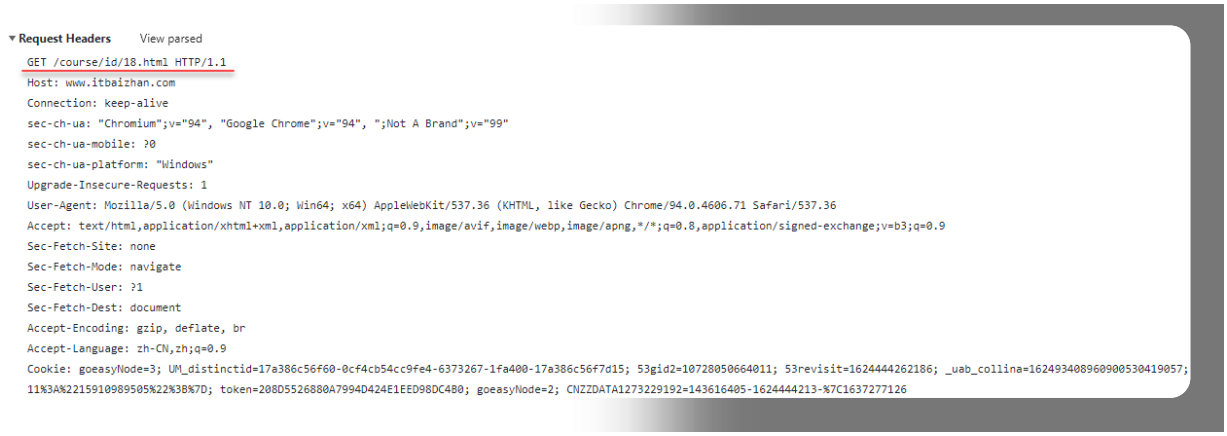
C 四部分。

D 五部分。

答案

1=>A　2=>B

请求行



GET /course/id/18.html?a=3&b=4 HTTP/1.1

POST /login HTTP/1.1

请求头

请求头用于说明是谁或什么在发送请求、请求源于何处，或者客户端的喜好及能力。服务器可以根据请求头部给出的客户端信息，试着为客户端提供更好的响应。请求头中信息的格式为key：value。

- Host
  客户端指定自己想访问的WEB服务器的域名/IP 地址和端口号。

- Connection
  连接方式。如果值是close则表示基于短连接方式，如果该值是keep-alive，网络连接就是持久的，在一定时间范围内是不会关闭，使得对同一个服务器的请求可以继续在该连接上完成。

- Upgrade-Insecure-Requests
  服务端是否支持https加密协议。

- Cache-Control

指定请求和响应遵循的缓存机制。

- User-Agent
  浏览器表明自己的身份（是哪种浏览器）。例如Chrome浏览器：Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.129 Safari/537.36。

- Accept
  告诉WEB服务器自己接受什么介质类型，/表示任何类型，type/* 表示该类型下的所有子类型。

- Accept-Encoding
  浏览器申明自己接收的编码方法，通常指定压缩方法，是否支持压缩，支持什么压缩方法（gzip，deflate）。

- Accept-Language
  浏览器申明自己接收的语言。语言跟字符集的区别：中文是语言，中文有多种字符集，比如big5，gb2312，gbk等。

- Accept-Charset
  浏览器告诉服务器自己能接收的字符集。

- Referer
  表示浏览器应该在多少时间之后刷新文档，以秒计时。

- Cookie
  可向服务端传递数据一种模型。

请求体

客户端传递给服务器的数据。比如：表单使用post方式提交的数据、上传的文件数据.等。

图 15-4　HTTP 请求报文

实时效果反馈

1.请求头中信息的格式为。

A key = value。

B key & value。

C key ~ value。

D key : value。

答案

1=>D

请求方式

- GET
  向指定的资源发出“显示”请求。GET请求中会将请求中传递的数据包含在URL中并在浏览器的地址栏中显示。GET请求传递数据时要求数据必须是字符。GET请求可以被浏览器缓存。

- POST
  向指定资源提交数据，请求服务器进行处理（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求传递数据时，数据可以是字符也可以是字节型数据，默认为字符型。POST请求默认情况下不会被浏览器所缓存。

- HEAD
  向服务器索要与GET请求相一致的响应，只不过响应体将不会被返回。这一方法可以在不必传输整个响应内容的情况下，就可以获取包含在响应消息头度中的元信息。

- PUT
  向指定资源位置上传其最新内容。

- DELETE
  请求服务器删除Request-URI所标识的资源。

GET和POST的区别(重要,面试常问)

1. GET在浏览器回退时是无害的，而POST会再次提交请求。
2. GET产生的URL地址可以被Bookmark，而POST不可以。

3. GET请求会被浏览器主动cache，而POST不会，除非手动设置。
4. GET请求参数会被完整保留在浏览器历史记录里，而POST中的参数不会被保留。
5. GET请求在URL中传送的参数是有长度限制的，而POST则没有。对参数的数据类型GET只接受ASCII字符，而POST即可是字符也可是字节。
6. GET比POST更不安全，因为参数直接暴露在URL上，所以不能用来传递敏感信息。
7. GET参数通过URL传递，POST放在Request body中。

实时效果反馈

1.如下哪个不是GET请求的特点。

A 浏览器回退时是无害的。

B 请求会被浏览器主动cache。

C 传送的参数是有长度限制的。

D 参数通过请求体传递。

2.如下哪个不是POST请求的特点。

A 浏览器回退时会再次提交请求。

B 请求不会被浏览器主动cache。

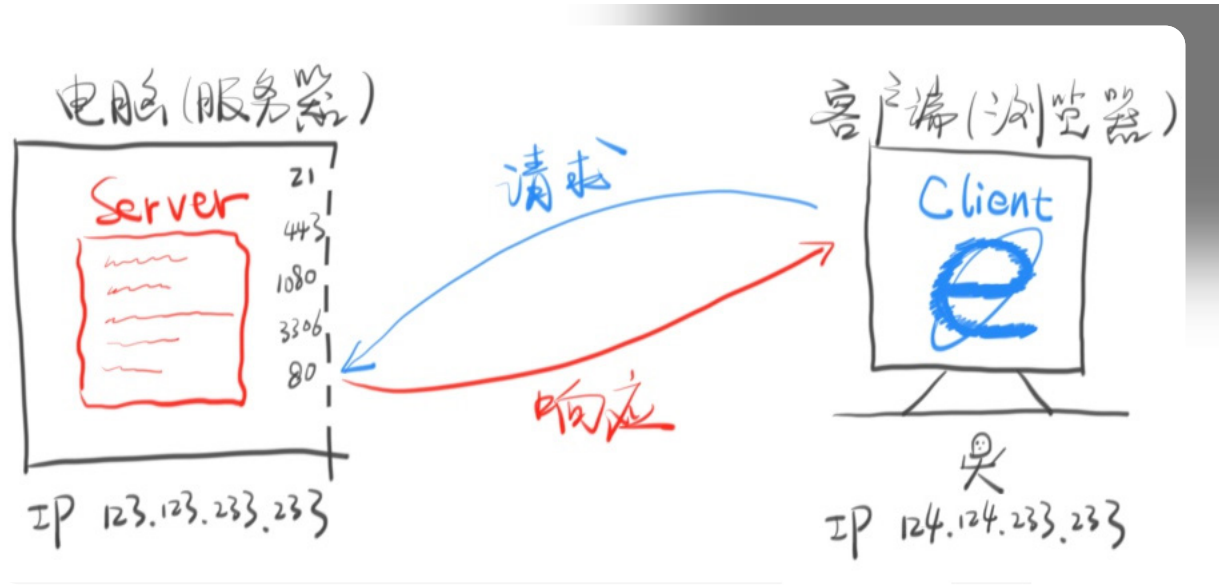
C 传送的参数是没有长度限制的。

D 参数通过URL传递。

答案

1=>D　2=>D

HTTP协议的响应



HTTP协议中的响应信息

服务端接收到客户端的请求消息，并将处理信息返回给客户端，以次来表示响应。当服务器收到浏览器的请求后，会发送响应消息给浏览器。一个完整的响应消息主要包括响应行、响应头信息和响应体。

响应状态分析（response）

Response消息也由三部分组成：第一部分叫Response line、第二部分叫Response header、第三部分叫Response body。

服务端为浏览器返回消息格式如下：

实时效果反馈

1.一个完整的响应消息主要包括。

A 响应行和响应体。

B 响应行和响应头信息。

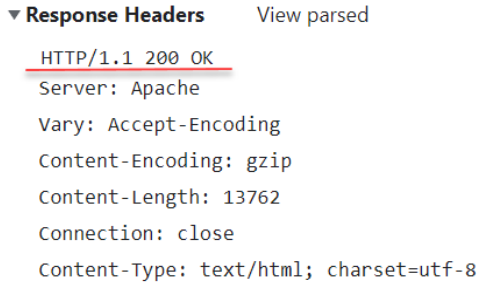
C 响应行、响应头信息和响应体。

D 响应头信息和响应体。

答案

1=>C

响应行



响应行：

HTTP/1.1 200 OK

响应行中的状态码

和请求消息相比，响应消息多了一个“响应状态码”，它以“清晰明确”的语言告诉客户端本次请求的处理结果。

http状态码分类

状态码	描述
1XX	指示信息——表示请求已接收，继续处理
2XX	成功——表示请求已被成功接收，理解，接收
3XX	重定向——要完成请求必须进行更进一步的操作
4XX	客户端错误——请求有语法错误或请求无法实现
5XX	服务端错误——服务器未能实现合法的请求

常见状态码及含义

200 - 请求成功，已经正常处理完毕

301 - 请求永久重定向，转移到其它URL

302 - 请求临时重定向

304 - 请求被重定向到客户端本地缓存

400 - 客户端请求存在语法错误

401 - 客户端请求没有经过授权

403 - 客户端的请求被服务器拒绝，一般为客户端没有访问权限

404 - 资源未找到，客户端请求的URL在服务端不存在

500 - 服务端出现异常

响应头

响应头用于告知浏览器当前响应中的详细信息，浏览器通过获取响应头中的信息可以知道应该如何处理响应结果。响应头中信息的格式为key：value。

- Date
  响应的Date使用的是GMT时间格式，表示响应消息送达时间。

- Server
  服务器通过这个Server告诉浏览器服务器的类型。

- Vary

客户端缓存机制或者是缓存服务器在做缓存操作的时候，会使用到Vary头，会读取响应头中的Vary的内容，进行一些缓存的判断。

- Content-Encoding
  文档的编码(Encode)方式。用gzip压缩文档能够显著地减少HTML文档的响应时间。

- Content-Length
  表示内容长度。只有当浏览器使用持久HTTP连接时才需要这个数据。

- Content-Type
  表示响应的文档属于什么MIME类型。

响应体

响应体就是响应的消息体，如果是纯数据就是返回纯数据，如果请求的是HTML页面，那么返回的就是HTML代码，如果是JS就是JS代码，如此之类。

实时效果反馈

1.如果请求的资源未找到返回的状态码为。

A 200

B 400

C 404

D 500

2.如果请求时服务端出现异常返回的状态码为。

A 200

B 400

C 404

D 500

答案

1=>C　2=>D

MIME类型

什么是MIME类型

MIME(Multipurpose Internet Mail Extensions)多用途互联网邮件扩展类型。是设定某种扩展名的文件用一种应用程序来打开的方式类型，当该扩展名文件被访问的时候，浏览器会自动使用指定应用程序来打开。多用于指定一些客户端自定义的文件名，以及一些媒体文件打开方式。

MIME类型的作用

HTTP协议所产生的响应中正文部分可以是任意格式的数据，那么如何保证接收方能看得懂发送方发送的正文数据呢？HTTP协议采用MIME协议来规范正文的数据格式。

MIME类型的使用

在服务端我们可以设置响应头中Content-Type的值来指定响应类型。

MIME类型对应列表

Type	Meaning
application/msword	Microsoft Word document
application/octet-stream	Unrecognized or binary data
application/pdf	Acrobat (.pdf) file
application/postscript	PostScript file
application/vnd.lotus-notes	Lotus Notes file
application/vnd.ms-excel	Excel spreadsheet
application/vnd.ms-powerpoint	PowerPoint presentation
application/x-gzip	Gzip archive
application/x-java-archive	JAR file
application/x-java-serialized-object	Serialized Java object
application/x-java-vm	Java bytecode (.class) file
application/zip	Zip archive
audio/basic	Sound file in .au or .snd format
audio/midi	MIDI sound file
audio/x-aiff	AIFF sound file
audio/x-wav	Microsoft Windows sound file
image/gif	GIF image
image/jpeg	JPEG image
image/png	PNG image
image/tiff	TIFF image
image/x-xbitmap	X Windows bitmap image
text/css	HTML cascading style sheet
text/html	HTML document
text/plain	Plain text
text/xml	XML
video/mpeg	MPEG video clip
video/quicktime	QuickTime video clip

实时效果反馈

1.我们可以设置响应头中的值来指定响应类型。

A Content

B Responsoe-Content

C Responsoe-Type

D Content-Type

答案

1=>D