get（post）请求（Http请求报文头）由哪几部分组成?

1）请求行（General）：

1.请求的资源（url）

2.请求的方式

3.协议/版本

2）请求头

是一种key-value的形式

1.Host：请求的主机

2.User-agent：浏览器内核

3.Cookie

4.Connection：连接状态（）

3）请求体

也是一种key-value的形式

里面主要是一些请求的参数（表单数据）：

比如用户名，密码等

Http响应报文

1）响应行：

1.报文协议与版本

2.状态码与状态描述

2）响应头

1.Set-Cookie：设置和页面关联的Cookie

2.Content-Encoding：文档的编码方法

3.Content-Length：内容长度

4.Content-Type：表示后面文档的类型

3）响应体：服务器返回给客户端的文本信息

HTTP1.0默认使用短链接，HTTP1.1开始默认使用长链接；使用长连接的HTTP协议，会在响应头有加入这行代码：Connection:keep-alive

TCP与UDP的区别

1）TCP是一种面向连接的全双工的可靠信道，UDP是一种无连接的不可靠信道
2）TCP提供可靠的服务，UDP只能尽最大努力交互，不能保证交付的可靠性
3）TCP传输效率相对要低些，UDP传输效率高
4）TCP是点对点，一对一的通信，而UDP的话支持一对一，一对多，多对一，多对多等多种交互通信

get与post的区别

1）get是不安全的，它将传输的数据存放在请求的url中，明文可见；post则将数据放在request body中
2）get传送的数据受限于它的url长度，而post传送的数据一般是无限制的
3）对参数的数据类型，GET只接受ASCII字符，而POST没有限制。

GET和POST本质上就是TCP链接，并无差别。但是由于HTTP的规定和浏览器/服务器的限制，导致他们在应用过程中体现出一些不同。
重大区别：GET产生一个TCP数据包；POST产生两个TCP数据包。
1.对于GET方式的请求，浏览器会把http header和data一并发送出去，服务器响应200（返回数据）；
2.而对于POST，浏览器先发送header，服务器响应100 continue，浏览器再发送data，服务器响应200 ok（返回数据）。

tcp为什么要三次握手，四次挥手

为啥需要三次握手：
为了防止已失效的连接请求报文段突然又传送到了服务端，因而产生错误

为啥需要四次挥手：
因为tcp是全双工模式，接收到FIN时意味将没有数据再发送过来，但是本方可能还有数据未发送完毕，需要继续发送，所以需要先发送ACK包确认收到之前client发送过来的FIN包，之后等服务端发送完数据后在发送FIN包准备断开连接

握手与挥手的区别：挥手有application的介入，需要等待application的回应，在等待的时间里，何尝不将ACK包先发送过去了？
挥手可以3次（启用延迟确认的情况的下可以将ACK包与FIN包合并发送，就是3次挥手）

MTU与IP分片

在IP层下面的每一种数据链路层协议都规定了一个数据帧中的数据字段的最大长度，这称为最大传送单元MTU，最常用的以太网就规定其MTU值是1500字节，如果超过这个长度，就必须对数据报进行分片处理

IP数据报首部固定长度最小20字节，最大60字节，其中：
1）标识：占16位，每产生一个数据报，计数器就+1，并将值付给标识字段
2）标志：占3位，目前只有2位有意义，最低位为MF，MF=1表示后面“还有分片”的数据报，MF=0表示这是若干数据报片中的最后一个
　　标志字段中间一位记为DF，意思是“不能分片”，只有当DF=0时才允许分片

P129

重定向和请求转发的区别 https://www.jianshu.com/p/29822c2c1ec0
请求转发：
request.getRequestDispatcher(URL地址).forward(request, response)
重定向：
response.sendRedirect(URL地址)

1）转发是容器中控制权的转向，在客服端地址栏中不会显示出转向后的地址；由于是服务器内部转发，整个过程处于同一个请求当中。
重定向则是完全的跳转，浏览器将会得到跳转后的地址。重定向，实际上客服端会发送两个请求给服务端。
2）转发中数据的存取可以用request作用域，而重定向不能使用，只能用session
3）一般来说转发更加高效，尽量使用转发（站内资源）而非重定向（可能访问站外资源）
4）重定向可以跨域访问，而转发是在web服务器内部进行的，不能跨域访问

TCP的数据报结构
P217

TCP拥塞控制的四种算法
详情：P229
慢开始：
拥塞窗口cwnd每次的增长量=min（N,SMSS）
每经过一个传输轮次，拥塞窗口cwnd就加倍（一个传输轮次所经历的时间其实就是往返时间RTT）（RTT的得到可以通过TCP报文段的选项部分的时间戳实现）

慢开始门限（ssthresh）

cwnd<ssthresh时，使用上述的慢开始算法

cwnd>ssthresh时，停止使用慢开始算法，改为使用拥塞避免算法

cwnd=ssthresh时，既可以用慢开始算法，又可以用拥塞避免算法

拥塞避免算法：

让拥塞窗口缓慢地增大，即每经过一个往返时间RTT就把发送方的拥塞窗口cwnd加1，而不是像慢开始阶段那样加倍增长。因此在拥塞避免阶段就有“加法增大”AI的特点。这表明在拥塞避免阶段，拥塞窗口cwnd按线性规律缓慢增长，比慢开始算法的拥塞窗口增长速率缓慢得多。

TCP流量控制

所谓流量控制就是让发送方的发送速率不要太快，要让接收方来得及接受，根本目的就是防止分组丢失的问题

利用滑动窗口实现流量控制（主要的方式就是接收方返回的 ACK 中会包含自己的接收窗口的大小，并且利用大小来控制发送方的数据发送）

持续计时器（打破死锁的僵局）

TCP超时重传

原理是在发送某一个数据以后就开启一个计时器，在一定时间内如果没有得到发送的数据报的ACK报文，那么就重新发送数据，直到发送成功为止。

快重传

快速重传(Fast retransmit)要求接收方在收到一个失序的报文段后就立即发出重复确认（为的是使发送方及早知道有报文段没有到达对方），而不要等到自己发送数据时捎带确认。

快重传算法规定，发送方只要一连收到3个重复确认就应当立即重传对方尚未收到的报文段，而不必继续等待设置的重传计数器时间到期。

https的连接过程

http的问题：

容易被监听、被伪装、被篡改

https改变了通信方式，它由以前的http----->tcp，改为http------>SSL----->tcp；https采用了共享密钥加密+公开密钥加密的方式

- 客户端发送请求到服务器端
- 服务器端返回证书和公开密钥，公开密钥作为证书的一部分而存在
- 客户端验证证书和公开密钥的有效性，如果有效，则生成共享密钥并使用公开密钥加密发送到服务器端
- 服务器端使用私有密钥解密数据，并使用收到的共享密钥加密数据，发送到客户端
- 客户端使用共享密钥解密数据
- SSL加密建立...

- 怎样保证公开密钥的有效性
    - 你也许会想到，怎么保证客户端收到的公开密钥是合法的，不是伪造的，证书很好的完成了这个任务。证书由权威的第三方机构颁发，并且对公开密钥做了签名。
- https的缺点
    - https保证了通信的安全，但带来了加密解密消耗计算机cpu资源的问题，不过，有专门的https加解密硬件服务器

什么是阻塞IO？什么是非阻塞IO？

在了解阻塞IO和非阻塞IO之前，先看下一个具体的IO操作过程是怎么进行的。

通常来说，IO操作包括：对硬盘的读写、对socket的读写以及外设的读写。

当用户线程发起一个IO请求操作（本文以读请求操作为例），内核会去查看要读取的数据是否就绪，对于阻塞IO来说，如果数据没有就绪，则会一直在那等待，直到数据就绪；对于非阻塞IO来说，如果数据没有就绪，则会返回一个标志信息告知用户线程当前要读的数据没有就绪。当数据就绪之后，便将数据拷贝到用户线程，这样才完成了一个完整的IO读请求操作，也就是说一个完整的IO读请求操作包括两个阶段：

1）查看数据是否就绪；

2）进行数据拷贝（内核将数据拷贝到用户线程）。

那么阻塞（blocking IO）和非阻塞（non-blocking IO）的区别就在于第一个阶段，如果数据没有就绪，在查看数据是否就绪的过程中是一直等待，还是直接返回一个标志信息。

什么是同步IO？什么是异步IO？

同步IO即 如果一个线程请求进行IO操作，在IO操作完成之前，该线程会被阻塞；
而异步IO为 如果一个线程请求进行IO操作，IO操作不会导致请求线程被阻塞。
事实上，同步IO和异步IO模型是针对用户线程和内核的交互来说的：

对于同步IO：当用户发出IO请求操作之后，如果数据没有就绪，需要通过用户线程或者内核不断地去轮询数据是否就绪，当数据就绪时，再将数据从内核拷贝到用户线程；

而异步IO：只有IO请求操作的发出是由用户线程来进行的，IO操作的两个阶段都是由内核自动完成，然后发送通知告知用户线程IO操作已经完成。也就是说在异步IO中，不会对用户线程产生任何阻塞。

这是同步IO和异步IO关键区别所在，同步IO和异步IO的关键区别反映在数据拷贝阶段是由用户线程完成还是内核完成。所以说异步IO必须要有操作系统的底层支持。

多路复用IO模型

在多路复用IO模型中，会有一个线程不断去轮询多个socket的状态，只有当socket真正有读写事件时，才真正调用实际的IO读写操作。因为在多路复用IO模型中，只需要使用一个线程就可以管理多个socket，系统不需要建立新的进程或者线程，也不必维护这些线程和进程，并且只有在真正有socket读写事件进行时，才会使用IO资源，所以它大大减少了资源占用。

常见HTTP状态码及含义

1** 信息。服务器收到请求，请继续执行请求
2** 成功。请求被成功接收并处理
3** 重定向。需要进一步操作来完成请求
4** 客户端错误。无法完成请求，或请求包含语法错误
5** 服务器错误。服务器在处理请求的过程中发成错误
100：服务器收到了请求的初始部分，并且请客户端继续发送
101：切换协议，服务端根据客户端的请求切换协议（只能切换到更高层的协议）
200：服务器已成功处理了请求。
300：多种选择，表示请求的资源可包括多个位置
301：被请求的资源已永久移动到新位置，即永久重定向。
302：要求客户端执行临时重定向
303：对应当前请求的响应可以在另一个 URI 上被找到，而且客户端应当采用 GET 的方式访问那个资源
400：服务器无法解析该客户端的这个请求
401：要求用户进行身份认证
403：服务器接收到请求，但是拒绝提供服务（认证失败）
404：请求失败，请求所希望得到的资源未被在服务器上发现。

500：服务器内部发生错误，无法完成请求。
503：由于超载或系统维护，服务器暂时的无法处理客户端的请求。延时的长度可包含在服务器的retry-after头信息中

http 和 https 区别

一、传输信息安全性不同
1、http协议：是超文本传输协议，信息是明文传输。如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息。
2、https协议：是具有安全性的ssl加密传输协议，为浏览器和服务器之间的通信加密，确保数据传输的安全。

二、连接方式不同
1、http协议：http的连接很简单，是无状态的。
2、https协议：是由SSL＋HTTP协议构建的可进行加密传输、身份认证的网络协议。

三、端口不同
1、http协议：使用的端口是80。
2、https协议：使用的端口是443.

四、证书申请方式不同
1、http协议：免费申请。
2、https协议：需要到ca申请证书，一般免费证书很少，需要交费。