<https://blog.csdn.net/weiyuefei/article/details/52067466>

TCP报文段首部格式：

序号：本报文段所**发送的数据的第一个字节**的序号。

**确认号ack**：**期待收到**对方下一个报文段的第一个数据字节的序号

**确认ACK**：占1位，仅当ACK=1时，确认号字段才有效。ACK=0时，确认号无效

**同步SYN**：连接建立时用于同步序号。当SYN=1，ACK=0时表示：这是一个连接请求报文段。

              若同意连接，则在响应报文段中使得SYN=1，ACK=1。因此，SYN=1表示这是一个连接请求，或连接接受报文。

**终止FIN**：用来释放一个连接。FIN=1表示：此报文段的发送方的数据已经发送完毕，并要求释放运输连接。

**三次握手（三次联络）**

**还要再发送一次确认是为了，防止已失效的连接请求报文段突然又传到了B，因而产生错误。**

已失效的报文段：正常情况下：A发出连接请求，但因为丢失了，故而不能收到B的确认。于是A重新发出请求，然后收到确认，建立连接，数据传输完毕后，释放连接，A发了2个，一个丢掉，一个到达，没有“已失效的报文段”

但是，某种情况下，A的第一个在某个节点滞留了，延误到达，本来这是一个早已失效的报文段，但是在A发送第二个，并且得到B的回应，建立了连接以后，这个报文段竟然到达了，于是B就认为，A又发送了一个新的请求，于是发送确认报文段，同意建立连接，假若没有三次的握手，那么这个连接就建立起来了（有一个请求和一个回应），此时，A收到B的确认，但A知道自己并没有发送建立连接的请求，因为不会理睬B的这个确认，于是呢，A也不会发送任何数据，而B呢却以为新的连接建立了起来，一直等待A发送数据给自己，此时B的资源就被白白浪费了。但是采用三次握手的话，A就不发送确认，那么B由于收不到确认，也就知道并没有要求建立连接。

**四次握手（两个二次握手）**

B收到连接释放报文段后就立即发送确认，然后就进入close-wait状态，**此时TCP服务器进程就通知高层应用进程，因而从A到B的连接就释放了**。此时是“半关闭”状态。即A不可以发送给B，但是B可以发送给A。

此时，若B没有数据报要发送给A了，其应用进程就通知TCP释放连接，然后发送给A连接释放报文段，并等待确认。

A发送确认后，进入time-wait，注意，**此时TCP连接还没有释放掉**，然后**经过时间等待计时器设置的2MSL后，A才进入到close状态。**

为什么要等待呢?

**①、为了保证A发送的最后一个ACK报文段能够到达B**。即最后这个确认报文段很有可能丢失，那么B会超时重传，然后A再一次确认，同时启动2MSL计时器，如此下去。如果没有等待时间，发送完确认报文段就立即释放连接的话，B就无法重传了（连接已被释放，任何数据都不能出传了），因而也就收不到确认，就无法按照步骤进入CLOSE状态，即必须收到确认才能close。

**②、防止“已失效的连接请求报文段”出现在连接中**。经过2MSL，那些在这个连接持续的时间内，产生的所有报文段就可以都从网络中消失。即在这个连接释放的过程中会有一些无效的报文段滞留在楼阁结点，但是呢，经过2MSL这些无效报文段就肯定可以发送到目的地，不会滞留在网络中。这样的话，在下一个连接中就不会出现上一个连接遗留下来的请求报文段了。

**可以看出：B结束TCP连接的时间比A早一点，因为B收到确认就断开连接了，而A还得等待2MSL.**

**原文见：http://blog.sina.com.cn/s/blog\_878940b301014nge.html**