首先介绍一下我的平台和应用环境。我使用的硬件平台是友善之臂Micro2440，软件平台是Embedded Linux，kernel版本是2.6.32，使用的java语言。

可能大家知道在使用Java Socket编程的时候经常要检测网络是不是完好连接的。比如Socket服务器检测和客户端的连接是否已经断开。首先想到socket类的方法isClosed()、isConnected()、isInputStreamShutdown()、isOutputStreamShutdown()等，但经过试验并查看相关文档，这些方法都是本地端的状态，无法判断远端是否已经断开连接。使用Socket的isconnected（）方法检测时，滞后是很大的，往往在20秒以上。这不利于系统的实时性。

然后想到是否可以通过OutputStream发送一段测试数据，如果发送失败就表示远端已经断开连接，类似ping，但是这样会影响到正常的输出数据，远端无法把正常数据和测试数据分开。发现有一个方法sendUrgentData，查看文档后得知它会往输出流发送一个字节的数据，只要对方Socket的SO\_OOBINLINE属性没有打开，就会自动舍弃这个字节，而SO\_OOBINLINE属性默认情况下就是关闭的。

于是使用的代码如下：  
public boolean isConnected(){  
        try{  
            s.sendUrgentData(0xFF);  
            return true;  
        }catch(Exception e){  
            return false;  
        }

}

通过测试发现，系统可以迅速检测到网络的状态！

2. java socket tcp连接判断客户端中断 心跳简单实现          
最近做android程序开发，当中遇到一个问题，当客户端网络异常中断时，服务器端会抛出异常，利用这个异常可以判断客户端掉线，从而做其他的逻辑处理，但是，当类似于“悄悄”拔掉网线这样的断线，服务器端却不会抛异常，我用的是java连接池，socket长连接，在read（）堵塞等待的时候也不会出问题，在网上查说是什么要两小时服务器端才会知道！

        合理的解决方案就是做心跳处理，本来嫌麻烦，后来想想其实不麻烦的，只是在客户端新开一个线程，然后while，里面sleep（5000）+向网络发送一个数据，也就是每五秒发送一个数据，这个数据最好跟正常的数据分开，简单些，然后在客户端设置socket.setSoTimeout(10000);，也就是说如果10秒钟没接收到数据的话就抛异常，这样就可以了