1. 出现了端口重复绑定的问题应该如何解决？

* 错误描述

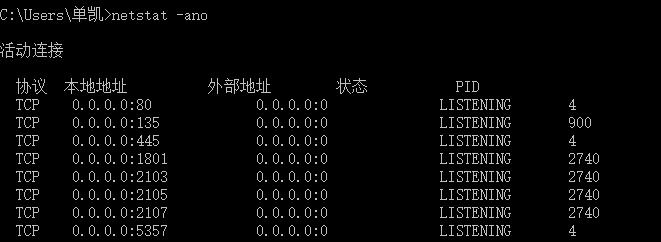
java.net.BindException: Address already in use: JVM\_Bind  
 at java.net.PlainSocketImpl.socketBind(Native Method)  
 at java.net.PlainSocketImpl.bind(PlainSocketImpl.java:359)  
 at java.net.ServerSocket.bind(ServerSocket.java:319)

* 原因分析

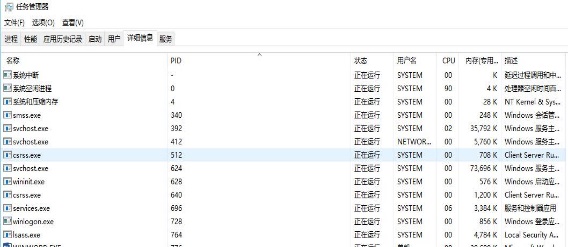
因为端口被占用了。

* 解决方案

1. Netstat –an0 查看占用端口的Pid



1. 任务管理器，根据Pid杀死进程



* Tomcat中也可能会出现这种问题，怎么解决？？

要么按上述方法解决，另一种方法就是主动改变**tomcat的端口**.

到 conf/server.xml 文件中修改

<Connector connectionTimeout="20000" port="8088" （去修给config->server.xml的端口号）protocol="org.apache.coyote.http11.Http11NioProtocol" redirectPort="8443"/>

1. **http的if-modified-since字段**

If-Modified-Since是标准的HTTP请求头标签，在发送HTTP请求时，把浏览器端缓存页面的最后修改时间一起发到服务器去，服务器会把这个时间与服务器上实际文件的最后修改时间进行比较。

如果时间一致，那么返回HTTP状态码304（不返回文件内容），客户端接到之后，就直接把本地缓存文件显示到浏览器中。

如果时间不一致，就返回HTTP状态码200和新的文件内容，客户端接到之后，会丢弃旧文件，把新文件缓存起来，并显示到浏览器中。

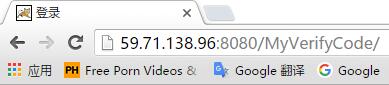
**可以做一个实验证明之：**

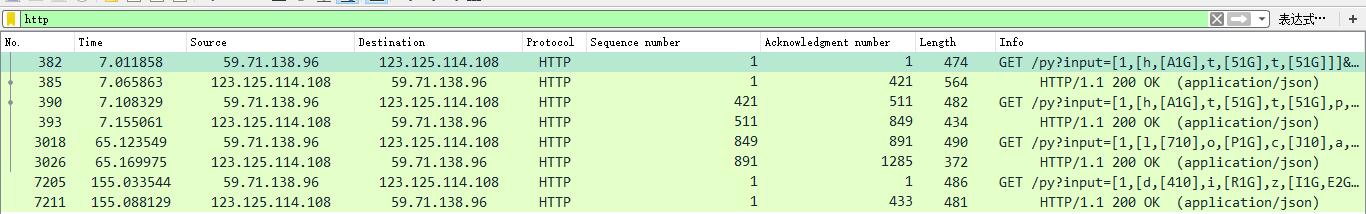
1. 首先在服务器创建一个简单的HTML文件，用浏览器访问一下，成功表示HTML页面。用wireshark捕获信息。  需要留意的是  
     （1）因为是第一次访问该页面，客户端发请求时，请求头中没有If-Modified-Since标签。  
    （2）服务器返回的HTTP状态码是200，并发送页面的全部内容。  
     （3）服务器返回的HTTP头标签中有Last-Modified，告诉客户端页面的最后修改时间。
2. 在浏览器中刷新一下页面，观察wireshark的捕获信息。  
     需要注意的是  
     （1）客户端发HTTP请求时，使用If-Modified-Since标签，把上次服务器告诉它的文件最后修改时间返回到服务器端了。  
     （2）因为文件没有改动过，所以服务器返回的HTTP状态码是304，没有发送页面的内容。
3. 用文本编辑器稍微改动一下页面文件，保存。再用浏览器访问一下，观察wireshark的捕获信息。  
   需要留意的是  
     （1）客户端发HTTP请求时，使用If-Modified-Since标签，把上次服务器告诉它的文件最后修改时间返回到服务器端了。  
     （2）因为文件被改动过，两边时间不一致，所以服务器返回的HTTP状态码是200，并发送新页面的全部内容。  
     （3）服务器返回的HTTP头标签中有Last-Modified，告诉客户端页面的新的最后修改时间。
4. **wireshark抓不到自己到自己的包**

* **问题现象**





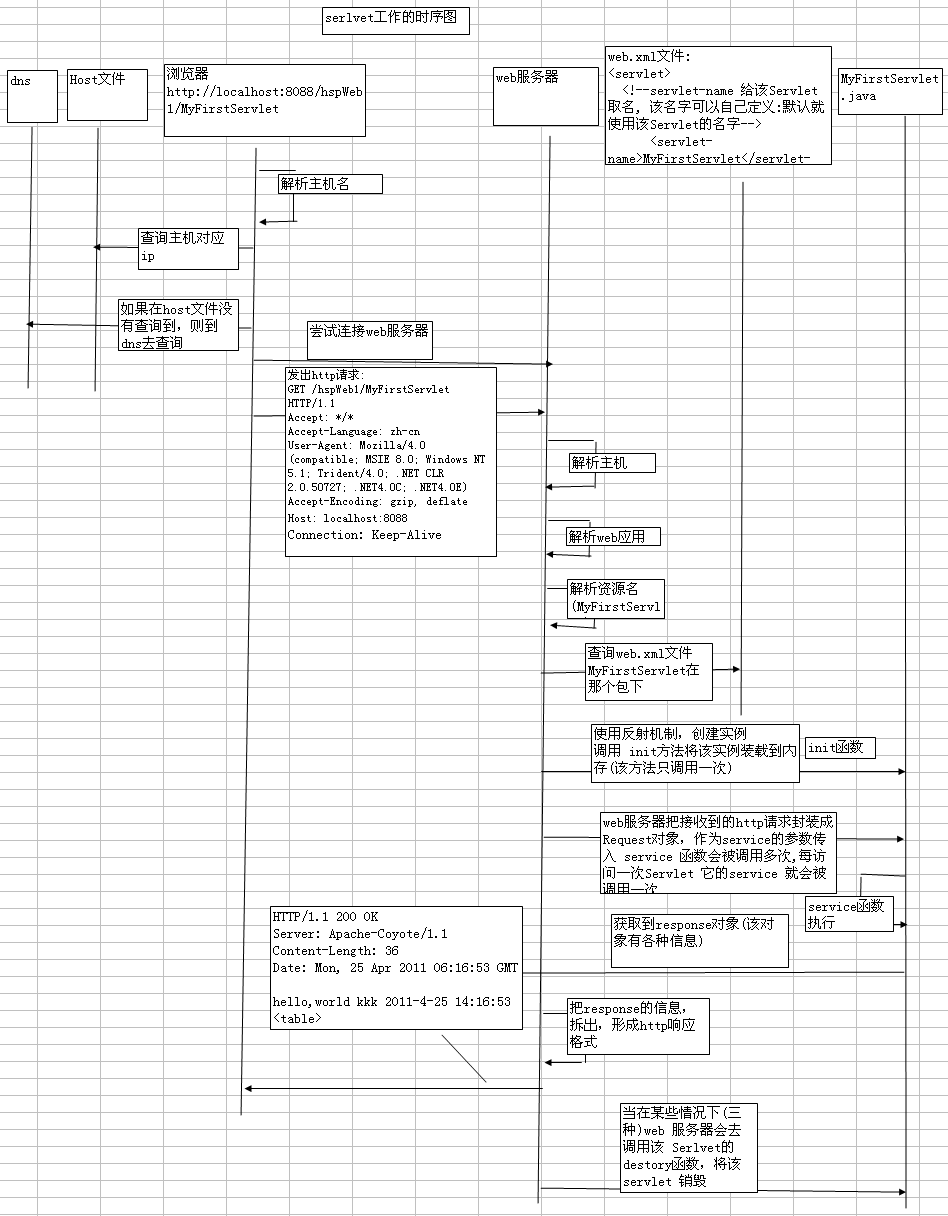




* **问题原因**

**Wireshark抓包是根据网卡的，没有抓到包，说明请求消息没有到网卡。**

1. **由上面的问题，引出http是怎么访问servlet的？？**



1. 在上面一题中，我们看到了host文件的作用，现在我们来玩个小游戏

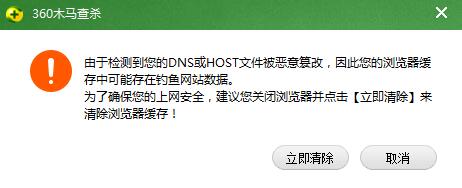
* 修改host文件，如下图所示：



* 访问自己写的域名：



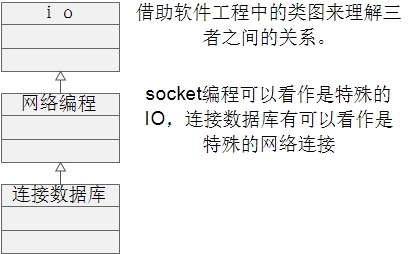




结果，如上，访问百度的IP是不成功的，很明显百度此行为有应对措施；访问地大的IP是成功的，说明地大没有针对此采取措施。并且修改host文件是杀毒软件被当做病区入侵处理的。

1. IO，socket编程，连接数据库“mysql/user”三者之间有关系吗？？

个人认为io和网络编程和连接数据库之间的关系如下

****

1. 公网IP和内网IP；

什么是内网IP：

一些小型企业或者学校，通常都是申请一个固定的IP地址，然后通过IP共享（IP Sharing），使用整个公司或学校的机器都能够访问互联网。而这些企业或学校的机器使用的IP地址就是内网IP，内网IP是在规划IPv4协议时，考虑到IP地址资源可能不足，就专门为内部网设计私有IP地址（或称之为保留地址），一般常用内网IP地址都是这种形式的：10.X.X.X、172.16.X.X-172.31.X.X、192.168.X.X等。需要注意的是，内网的计算机可向Internet上的其他计算机发送连接请求，但Internet上其他的计算机无法向内网的计算机发送连接请求。我们平时可能在内网机器上搭建过网站或者FTP服务器，而在外网是不能访问该网站和FTP服务器的，原因就在于此。

什么是公网IP（外网IP）：

公网IP就是除了保留IP地址以外的IP地址，可以与Internet上的其他计算机随意互相访问。我们通常所说的IP地址，其实就是指的公网IP。互联网上的每台计算机都有一个独立的IP地址，该IP地址唯一确定互联网上的一台计算机。这里的IP地址就是指的公网IP地址。

在java套接字编程的时候，一方写公网IP，另一方写内网IP是连接不上的。

1. **java 没有无符号整形 java char是两个字节**
2. **byte** a = (**byte**)234;
3. System.out.println(a);

上面的代码，结果是-22，因为java中byte是有符号的，byte范围是-128~127。

如果想输出234，该怎么做呢，首先想到的是将a 赋给大一点的类型，如下：

1. **byte** a = (**byte**)234;
2. **int** i = a;
3. System.out.println(a);

执行后，还是-22，因为int也是有符号的，所以a赋给i时，a的符号位在i中成为了i的符号位。

正确方法应该是：

1. **byte** a = (**byte**)234;
2. **int** i = a;
3. i = a&0xff;
4. System.out.println(i);

原因是：

0xff是int，占4个字节，a是byte，占1个字节，进行&操作的细节如下：

    00000000 00000000 00000000 11101010    (a)

&

   00000000 00000000 00000000 11111111    (i)

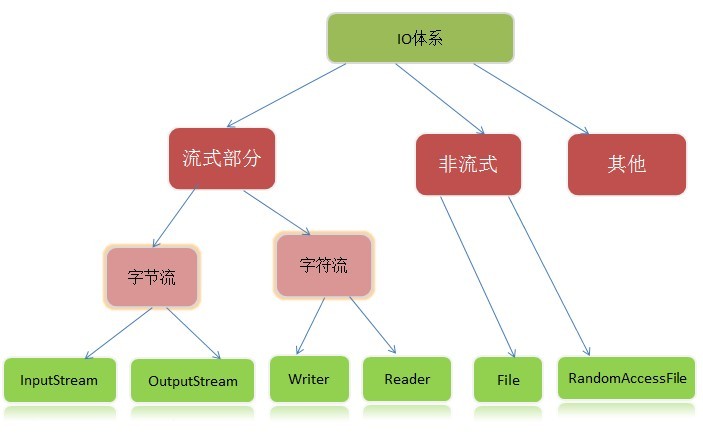
---------------------------------------------------------------------

= 00000000 00000000 00000000 11101010

结果是int，但是符号位是0，说明是正数，最后就是正整数234.

其实这个方法在C语言中也可以获取有符号char的无符号值，但是C语言中可以直接使用unsigned来转换就可以，比这个方便。

1. **Java中流的关闭顺序**



一般情况下是：先打开的后关闭，后打开的先关闭

另一种情况：看依赖关系，如果流a依赖流b，应该先关闭流a，再关闭流b

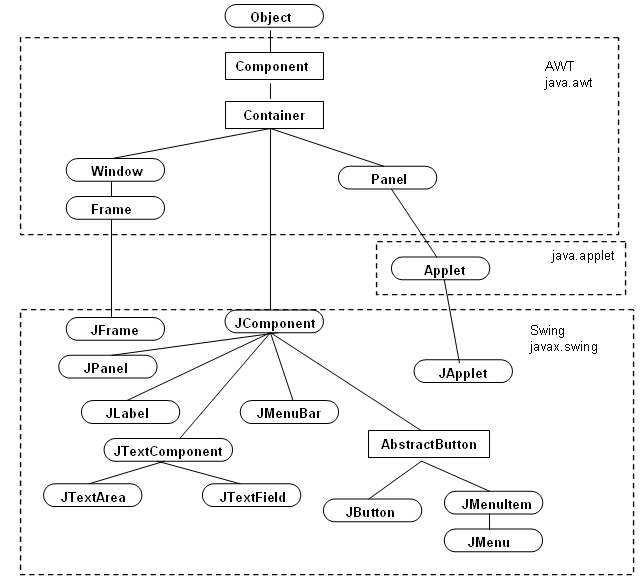
例如处理流a依赖节点流b，应该先关闭处理流a，再关闭节点流b

当然完全可以只关闭处理流，不用关闭节点流。处理流关闭的时候，会调用其处理的节点流的关闭方法

如果将节点流关闭以后再关闭处理流，会抛出IO异常

1. java 容器中的关系

* **Swing基本框架**



Java中组件容器包含顶层容器和中间容器。

在java中顶层容器有三种，分别是JFrame（框架窗口，即通常的窗口）、JDialog（对话框）、JApplet（用于设计嵌入在网页中的java小程序），顶层容器是容纳其它组件的基础，即设计图形化程序必须要有顶层容器。

Java中间容器是可以包含其它相应组件的容器，但是中间容器和组件一样，不能单独存在，必须依附于顶层容器。

常见的中间容器有：

• JPanel：最灵活、最常用的中间容器。

• JScrollPane：与 JPanel 类似，但还可在大的组件或可扩展组件周围提供滚动条。

• JTabbedPane：包含多个组件，但一次只显示一个组件。用户可在组件之间方便地切换。

• JToolBar：按行或列排列一组组件（通常是按钮）。

1. Cs有缺陷？？只能客户端先发送消息。

**每一方解释客户端有事服务器端，变成p2p。**

12.传输文件时不知道Client发送的是什么类性的文件（doc？bmp？avi？？）传输文件的时候是TCp好一些还是UDP好一些？？

线传输文件的基本信息，好让接收方做相应的处理。

Tcp好一些，因为TCP保证可靠的传输，Udp不行，需要自己进行控制（确认、重传）。