1、概念

Socket 套接字，是由系统提供用于网络通信的技术，是基于 TCP/IP 协议的网络通信的基本操作单元。基于 Socket套接字的网络程序开发就是网络编程。

2、分类

Socket套接字主要针对传输层协议划分为如下三类：

1. 流套接字

使用传输层TCP协议

TCP，即Transmission Control Protocol（传输控制协议），传输层协议。

TCP协议的特点：

1.有连接（打电话）

2.可靠传输（发送方能够知道对方是否收到了）

3.面向字节流（）

4.全双工（有接受缓冲区，也有发送缓冲区）(A和B可以同时向对方发送接收数据)

5.大小不限

对于字节流来说，可以简单的理解为，传输数据是基于IO流，流式数据的特征就是在IO流没有关闭的情况下，是无边界的数据，可以多次发送，也可以分开多次接收。

2. 数据报套接字

使用传输层UDP协议

UDP，即User Datagram Protocol（用户数据报协议），传输层协议。

UDP协议的特点:

1.无连接(发微信)

2.不可靠传输(发送方不知道对方是否收到了)

3.面向数据报

4.半双工(有接收缓冲区,无发送缓冲区)(单向通信,要么A给B发,要么B给A发,不能同时发)

5.大小受限:一次最多传输64K

对于数据报来说，可以简单的理解为，传输数据是一块一块的，发送一块数据假如100个字节，必须一次发送，接收也必须一次接收100个字节，而不能分100次，每次接收1个字节。

3. 原始套接字

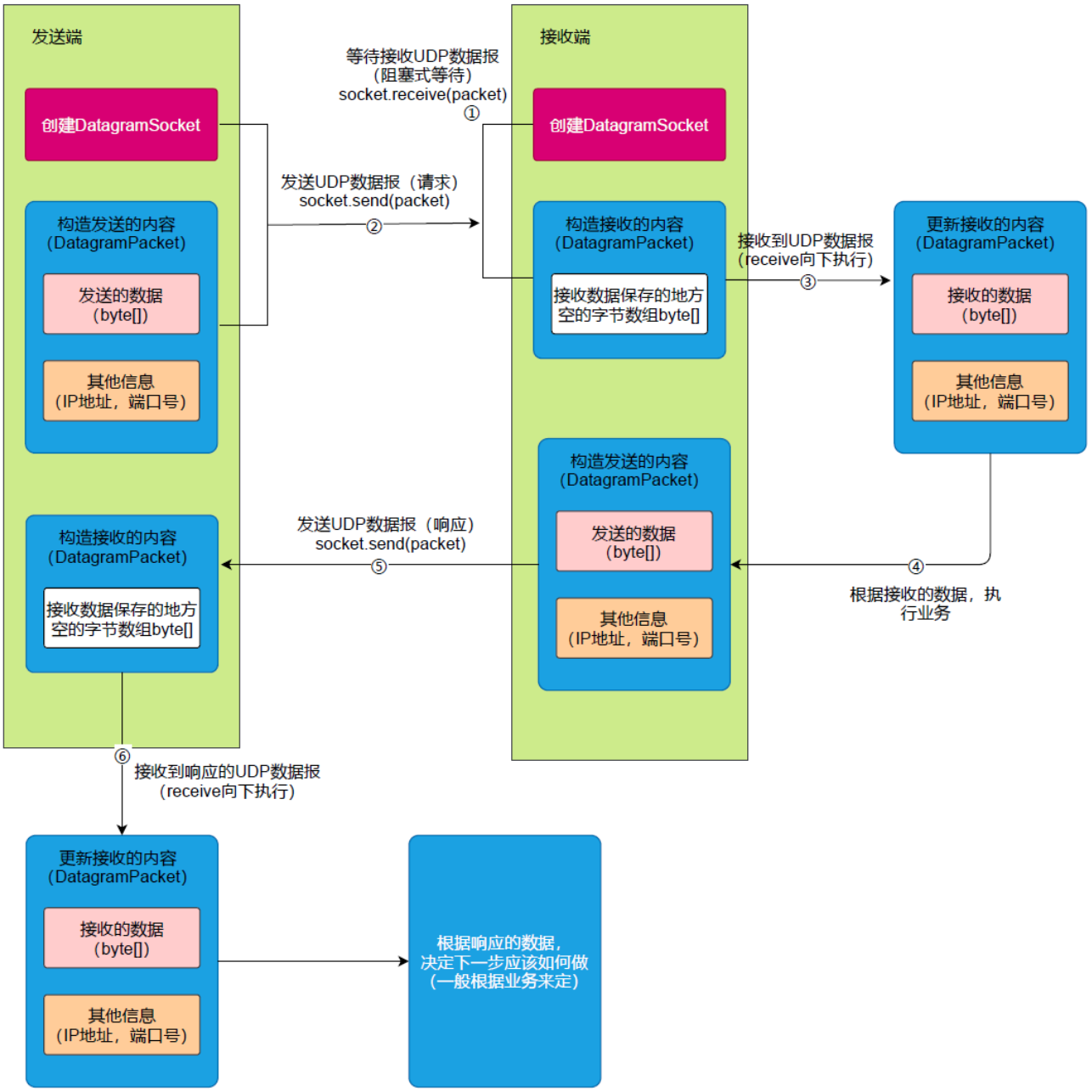
原始套接字用于自定义传输层协议，用于读写内核没有处理的IP协议数据。

3、Java流套接字通信模型

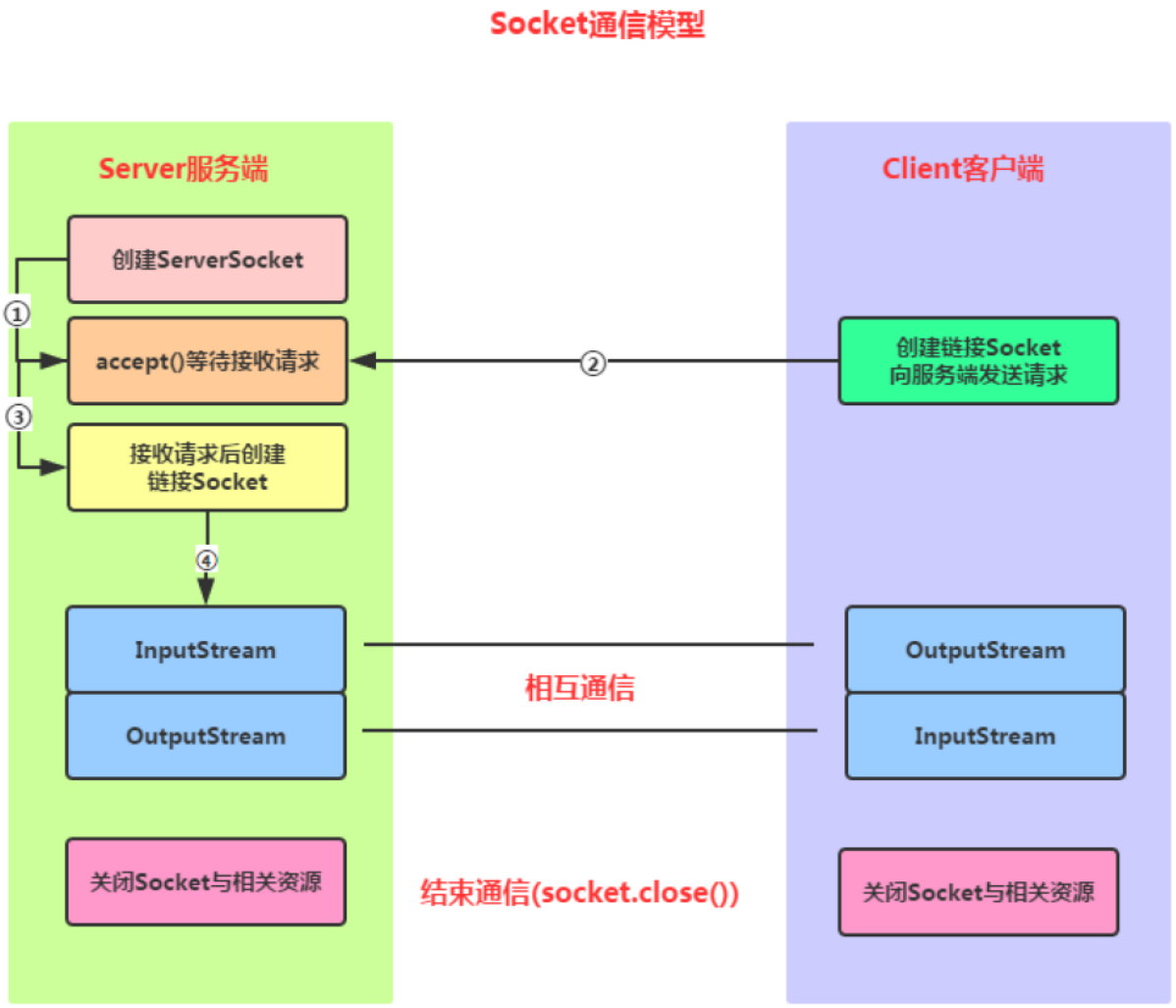
1. Java数据报套接字通信模型

对于UDP协议来说，具有无连接，面向数据报的特征，即每次都是没有建立连接，并且一次发送全部数据报，一次接收全部的数据报。

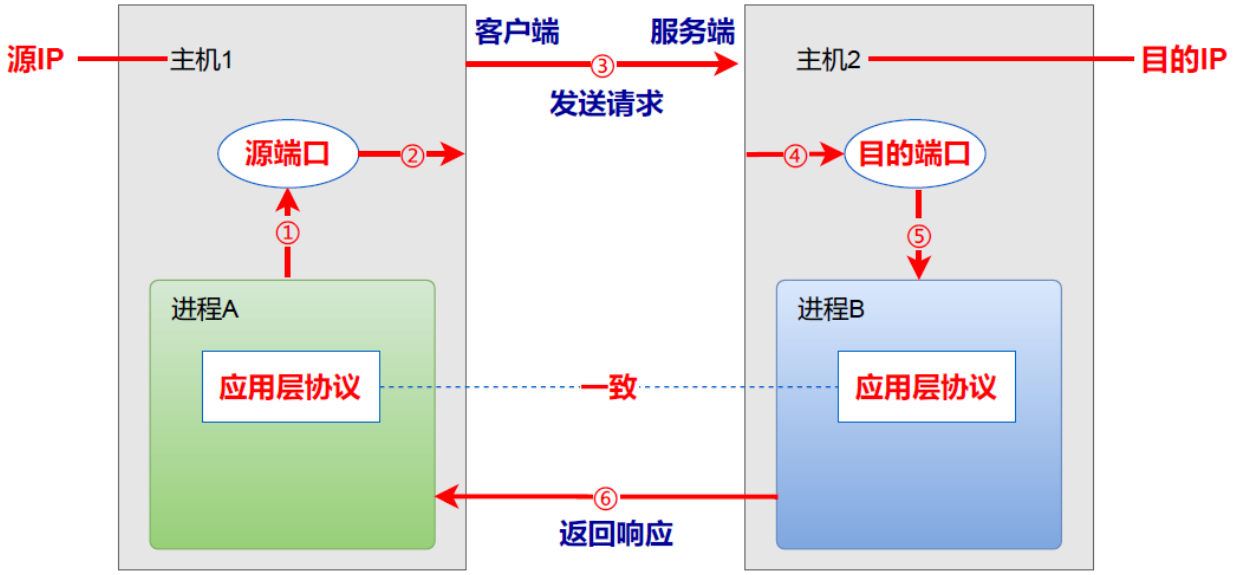
java中使用UDP协议通信，主要基于 DatagramSocket 类来创建数据报套接字，并使用DatagramPacket 作为发送或接收的UDP数据报。多个客户端的请求处理及响应，流程如下:



2. Java流套接字通信模型



4、Socket编程注意事项



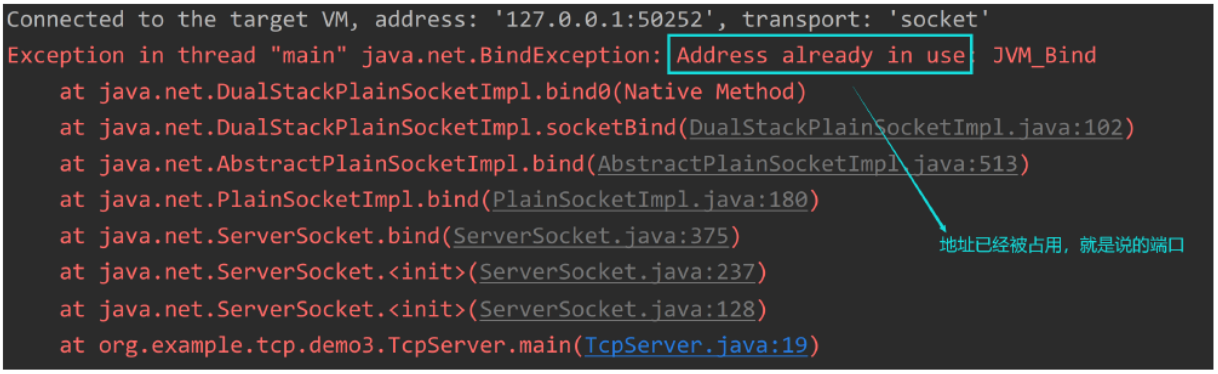
1.客户端和服务端:开发时,经常是基于一个主机开启两个进程作为客户端和服务端,但真实的场景时,一般都是在不同主机.

2.注意目的IP和目的端口号，标识了一次数据传输时要发送数据的终点主机和进程

3. Socket编程我们是使用流套接字和数据报套接字，基于传输层的TCP或UDP协议，但应用层协议，也需要考虑，这块我们在后续来说明如何设计应用层协议

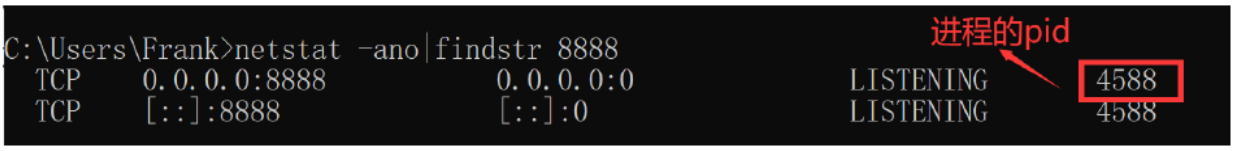
4. 关于端口被占用的问题

如果一个进程A已经绑定了一个端口，再启动一个进程B绑定该端口，就会报错，这种情况也叫端口被占用。对于java进程来说，端口被占用的常见报错信息如下：

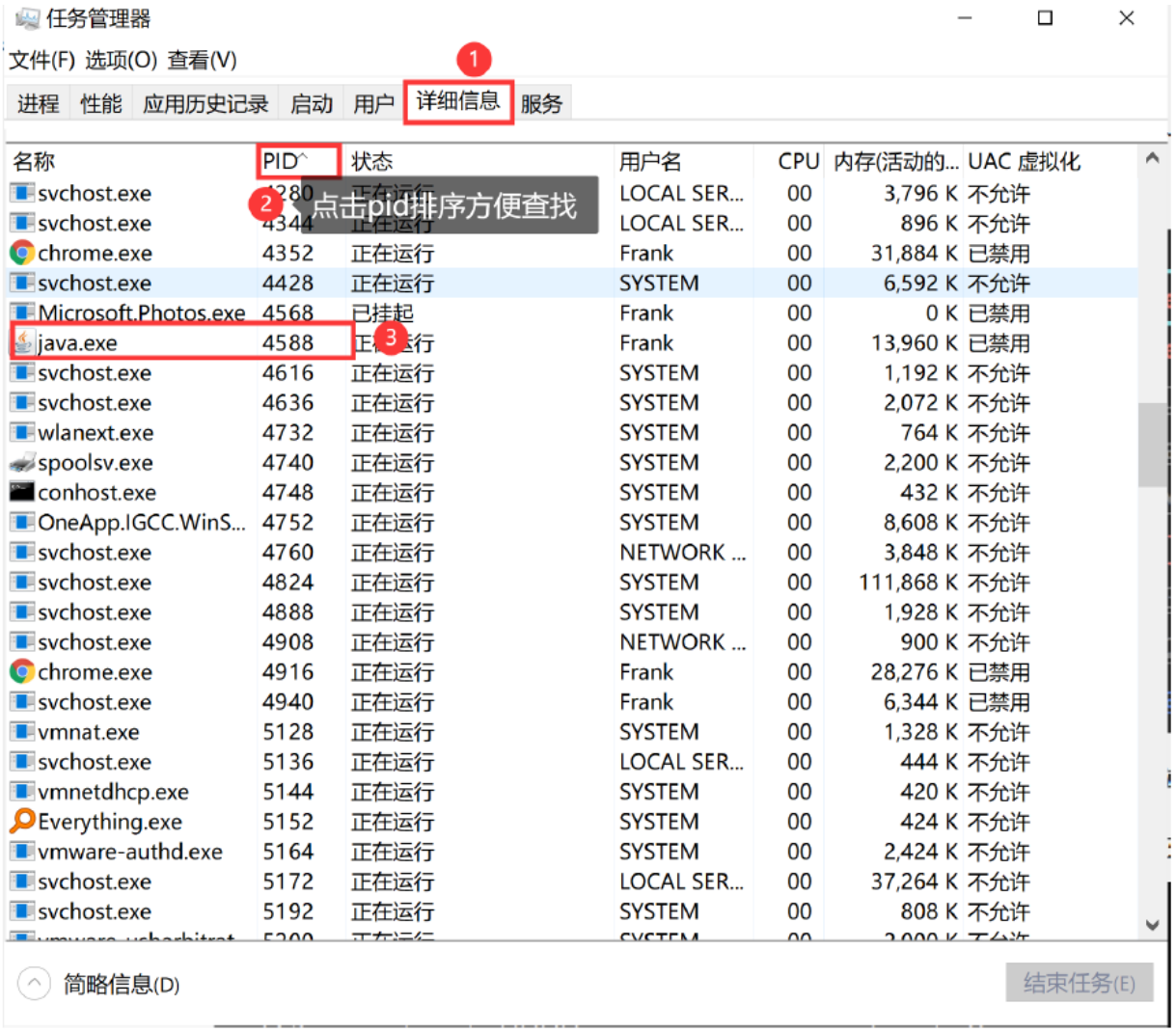


此时需要检查进程B绑定的是哪个端口，再查看该端口被哪个进程占用。以下为通过端口号查进程的方式：

- 在cmd输入 netstat -ano | findstr 端口号，则可以显示对应进程的pid。如以下命令显示了8888进程的pid



- 在任务管理器中，通过pid查找进程



解决端口被占用的问题：

- 如果占用端口的进程A不需要运行，就可以关闭A后，再启动需要绑定该端口的进程B
- 如果需要运行A进程，则可以修改进程B的绑定端口，换为其他没有使用的端口。