Document of P2P

这是一个基于UDP的P2P Demo， 可以在两个内网主机中进行点对点通信， 但需要有辅助服务器支持才能够实现NAT的穿透，原理如下图所示。但由于是基于UDP，丢包是不可避免的，在代码中会进行一定程度的尝试连接，但仍可能会出现在恶劣网络环境下出现无法连接的情况。（为什么需要进行NAT穿透，可以看我的Blog中有阐释）

Server:

根据UID储存

指令队列

客户端：

1. 需定义辅助服务器的地址，端口
2. 提供通信双方的UID，需要传输的信息Msg

通信格式：

效验段\_MsgID\_指令段\_MyUID\_TargetUID\_TargetIP\_TargetPort\_Text

内部处理格式 ：

源IP\_源端口\_效验段\_MsgID\_指令段\_MyUID\_TargetUID\_ TargetIP\_TargetPort\_Text

客户端输入格式：

指令段\_MyUID\_TargetUID\_TargetIP\_TargetPort\_Text

（不需要的字段可用0填充）

与服务器通信：

接收：

1. 登记MyUID：发送格式：LOGIN\_UID\_MyUID
2. 连接TargetUID： 发送格式：ASK\_CONNECT\_UID\_TargetUID
3. 注销MyUID： 发送格式：LOGOUT\_UID\_MyUID

发送：

1. 回复收到： 发送格式：Server\_RECEIVE
2. 连接TargetUID： 发送格式：CONFIRM\_CONNECT\_TargetIP\_ Port

客户端之间通信：

1. 发送：格式：SEND\_MSG\_MyUID\_text
2. 收到：格式：RECEIVE\_MSG\_MyUID\_text
3. 发给服务器：格式RECEIVE\_ CONFIRM\_CONNECT

客户端：

1. 主线程：ClientMain 用于启动监听、消息处理线程，并定义了消息队列、Msg是否收到的标记容器、其他客户端地址端口的记录容器、已完成消息的限定长度队列。
2. 监听端口线程：ClientReceiveMsg 用于监听UDP端口，接受信息，去重后加入消息队列
3. 消息处理线程：ClientExeMsg 用于处理消息队列中收到的消息，根据消息种类的不同，进行相应的操作。（处理完后放入已完成队列，避免重复操作）
4. 发送线程：ClientSendMsg 由消息处理线程、用户添加指令函数调用。用于发送UDP包，有两种发送数量可选，可更改发包数量。并可以在对方确认收到后停止发送。
5. 其他重要的类：UserAddInstr 用户可以调用这个类中的函数，实现SendMsg、Connect、 Login、Logout操作。

服务器：

1. 主线程：与客户端差不多
2. 监听端口线程：差不多
3. 消息处理线程：与客户端处理的消息种类不同而已
4. 发送线程：相同

工具类UDPtools中封装了收UDP包和发送UDP包的两个方法：

boolean SendMsg(DatagramSocket client, String IP, int Port, String Msg)

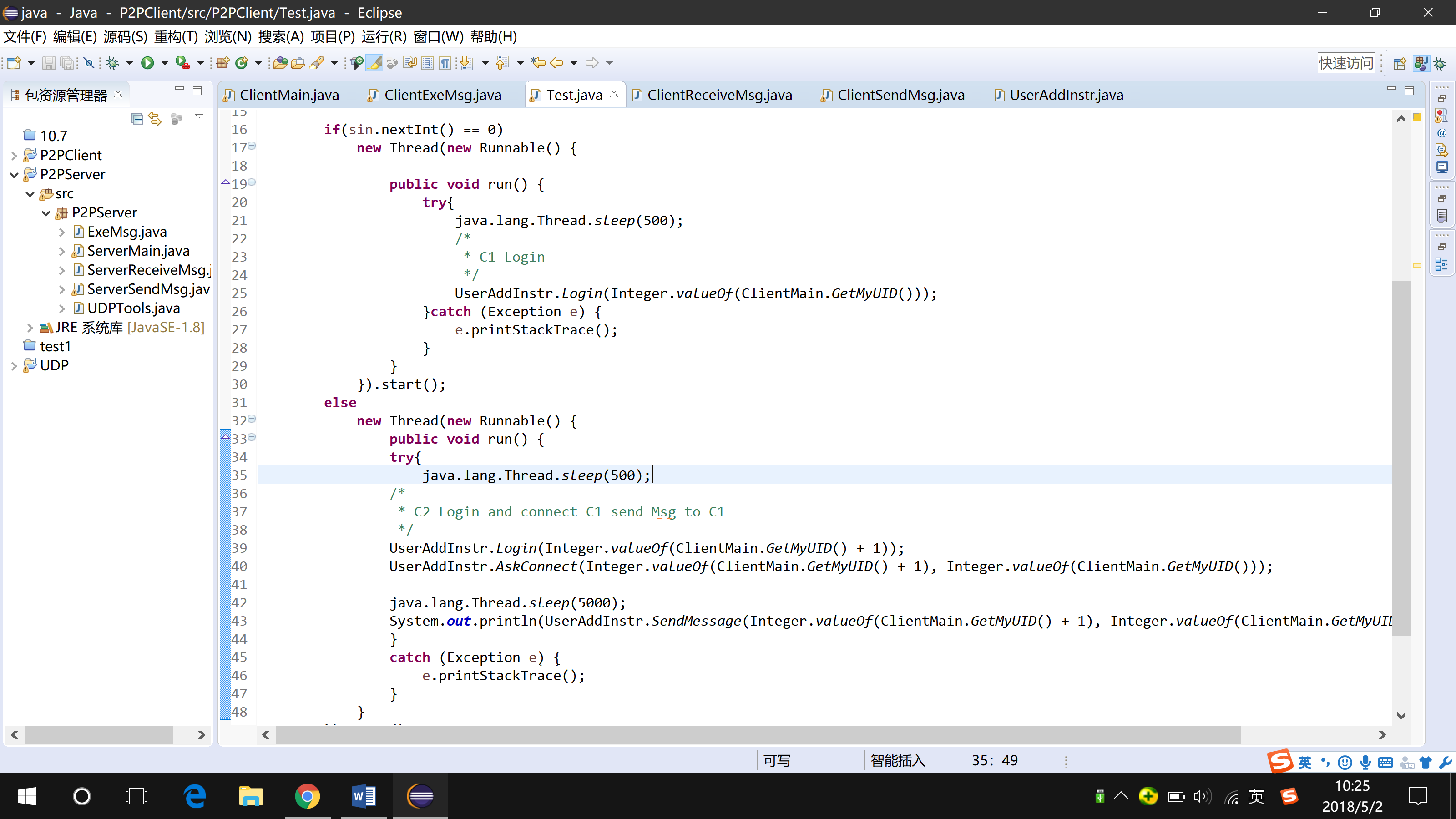
boolean SendMsg(DatagramSocket client, SocketAddress target, String Msg)

String ReceiveMsg(DatagramSocket client)

使用方法：

Client：

1. 启动ClientMain线程，并传入参数（服务器的IP、 服务器监听的UDP端口、 本设备的ID值、 （可选： 本设备的UDP端口））
2. 在等待一会（让ClientMain初始化后），可以添加指令，如下图所示：



含义为：输入为0时，等待500ms后，执行Login（向服务器登记自己的ip、端口）

输入为1时，等待500ms后，执行Login，再执行Askconnect（连接之前登记过的某个Client）实现打洞过程。

Server：

直接启动即可，默认UDP监听端口为30000，可在源代码中自行更改