说明文档

高钰洋 2120190419 网络第四次作业 UDP 聊天室

文档说明

编程环境：介绍编程所用语言以及所用库
关键问题：设计思路即实现方法
用法及测试截图：介绍本程序使用方法，以及使用的截图
文件说明：说明作业文件夹架构

文件说明

- 作业中源代码见 src 文件夹，可执行文件在可执行文件文件夹下
- 源代码文件夹下，client.py 为客户端代码，server.py 为服务器代码，mylog.py 为日志代码，createuserfile.py 为穿件 LogedUser.txt 文件的（没有该文件启动服务器会报错，需要先运行改脚本），setup.py 是 cx_freeze 生成可执行文件的脚本
- 可执行文件中 client.exe 为客户端，server.exe 为服务器端，LogedUser.txt 为用户信息的记录，用来维护注册过的用户信息（此文件没有启动服务器报错，需要先生成）

编程环境

本程序编写与 Windows10 系统，python3.7.0 环境下，图形界面使用 python3 自带的 tkinter 库，UDP 通信采用 socket 库，多线程采用 threading 库，，用户列表的保存和传输用到了 pickle 和 json 库，上次由于 py2exe 生成的 exe 文件在 win64 下面没法打包成一个文件，会有很多附带的库文件，看起来很乱，所以这次采用 cx_freeze 生成 exe 文件

关键问题

程序功能

- **服务器的开启和关闭**：输入监听端口，服务器会自动获取本机 ip，之后开启服务器，运行中会保存日志记录以及用户信息。
- **用户注册**：用户若未拥有账户需要先注册，输入服务器的地址端口，以及用户名和密码可以注册
- **用户登录**，用户可以登录已经注册的账号，输入服务器的地址端口，以及用户名和密码

可以登录

- **一键换肤**：可以点击换肤按钮更换软件皮肤
- **发送消息**：可以选择用户列表中的一位发送消息，或者点击 **Ctrl 键**多选，一对多的发送消息
- **发送文件**：用户可以发送 txt 文件到选择的用户
- **登出**：用户可以点击登出按钮登出

注册用户信息保存

服务器端对每个注册的用户都会保存其用户信息，具体包括用户名，密码，登录 ip 和端口，用户信息保存在 **Logeduser.txt** 中，服务器每次启动时会读取改文件的内容，每次更新后会将更新写回文件，该功能使用 pickle 库实现
使用者可以通过查看该文件来查看服务器保存的用户信息
“可以改进的点”：这里存密码才用的是明文，其实可以再客户端发送的时候传递密文，这里的实现没有更多的考虑安全性，可以在以后的工作中改进

交互命令

为了方便服务器与客户端，以及客户端之间的交互，定义了一系列命令，发送时采用 ***开始符号+分隔符+信息+分隔符+命令+分隔符+结束符*** 的方式来定义命令，这里的信息在同命令下是不同的，注册时和登录时为用户名和密码，发送消息时为信息内容和用户名，服务器端的登录密码错误命令信息为空等。
这里为了区分服务器和客户端发出的命令，服务器端的开始符号为"$"，客户端的开始符号为“>”

例如，客户端的注册命令为
>!USERNAME!PASSWORD!REG!$

全部的命令如下表所示

命令（发送方）	*功能*	*信息内容*
reg(客户端)	注册	用户名+密码
login(客户端)	登录	用户名+密码
msg(客户端)	发送聊天消息	用户名+消息
logout(客户端)	登出	用户名
keepalive(客户端)	获取用户列表	用户名
regok(服务器)	注册成功	用户名

haveuser(服务器)	注册时用户名已存在	无
Loginok(服务器)	登录成功	用户名
Loginnopass(服务器)	登录密码错误	无
loginnouser(服务器)	登录用户名不存在	无
userlist(服务器)	向客户端更新用户列表	用户列表
quit(服务器)	服务器关闭	无

日志记录

客户端和服务器端都会在运行时记录日志信息，日志信息的记录方法函数定义在 mylog.py 中。
服务器端日志统一记录在 SERVER.log 中，客户端日志记载在客户端本地，不同的用户有不同的日志，分别名为 CLIENT_USERNAME.log,其中 USERNAME 为用户名
记录格式为 ***时间 [事件] 信息***
具体信息可以见下一部分的截图

用法即测试截图

注册

（进行测试此时已有注册用户 gyy，没有用户登录）
（发送消息时请先选择右侧用户，点击空白处等操作有时候会让右侧列表取消选择，请发送前确认选择了用户）

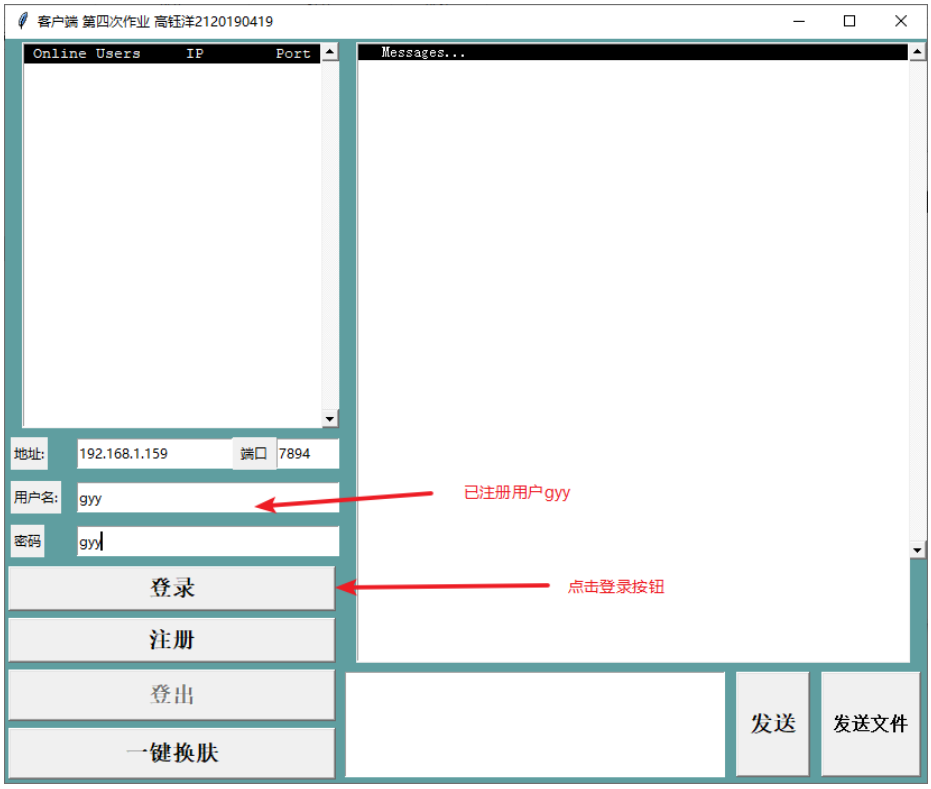
开启服务器

点击 Start 按钮

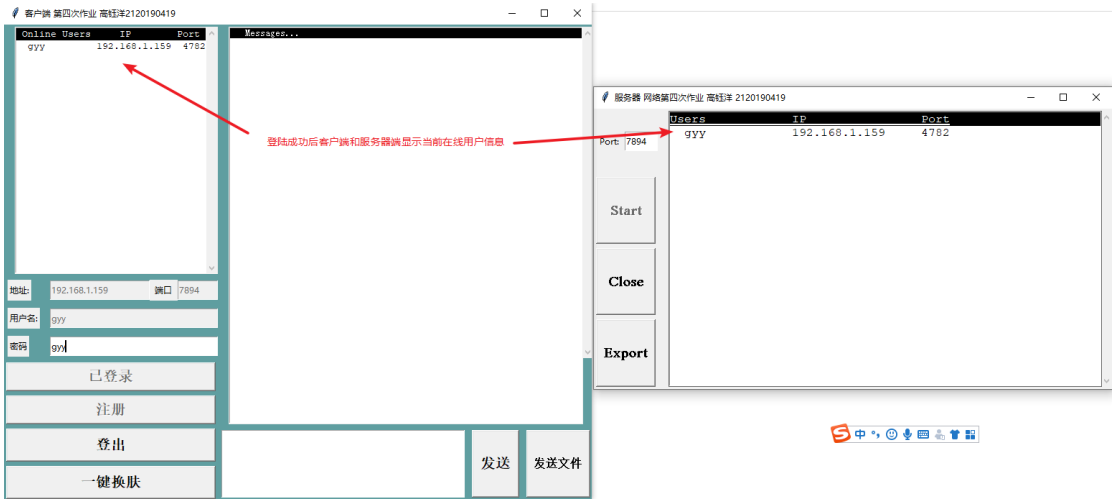
开启成功弹出提示框，显示服务器的 ip 地址和端口

登录

输入已经注册的用户的用户名和密码（未注册请先注册，步骤可见下方注册的截图）

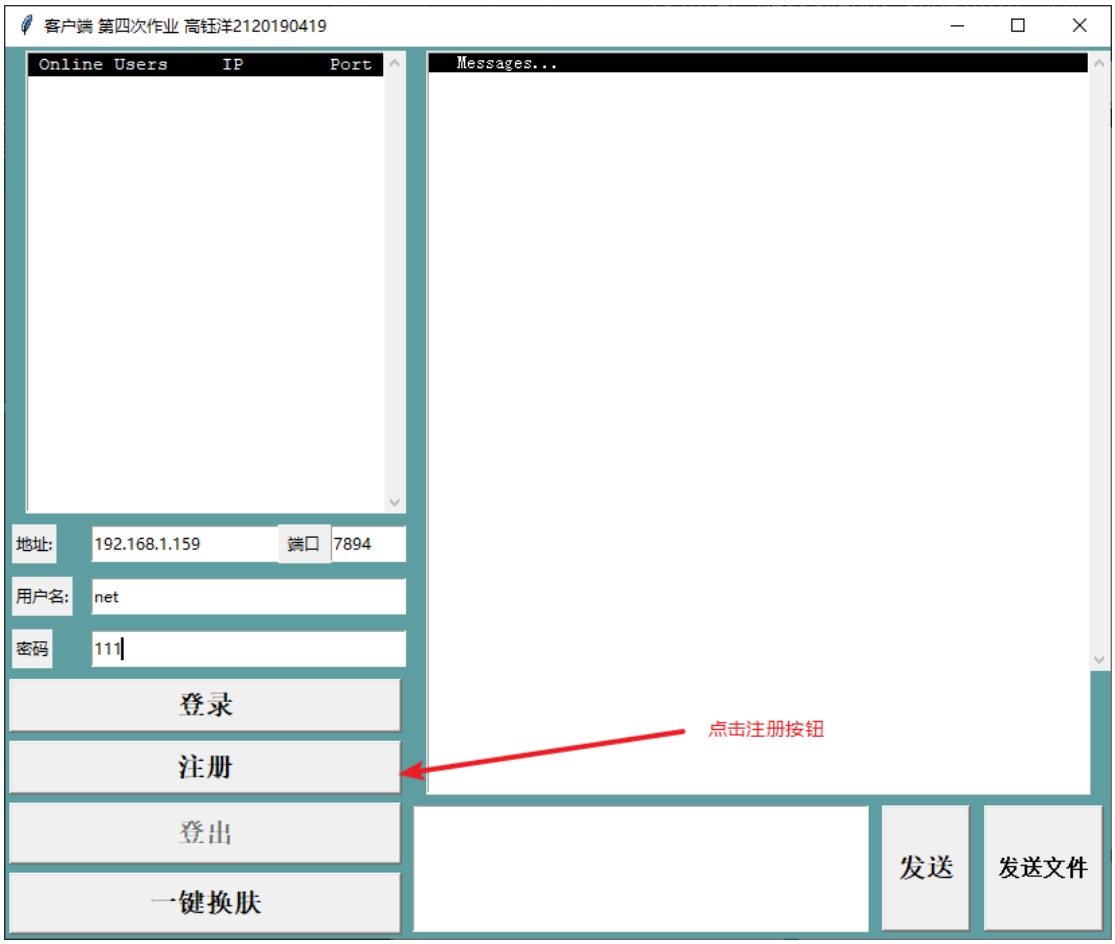


点击登录按钮，若密码错误或用户名不存在，会根据服务器返回的相应的返回值进行提示，提示登陆成功用户列表中显示当前在线用户

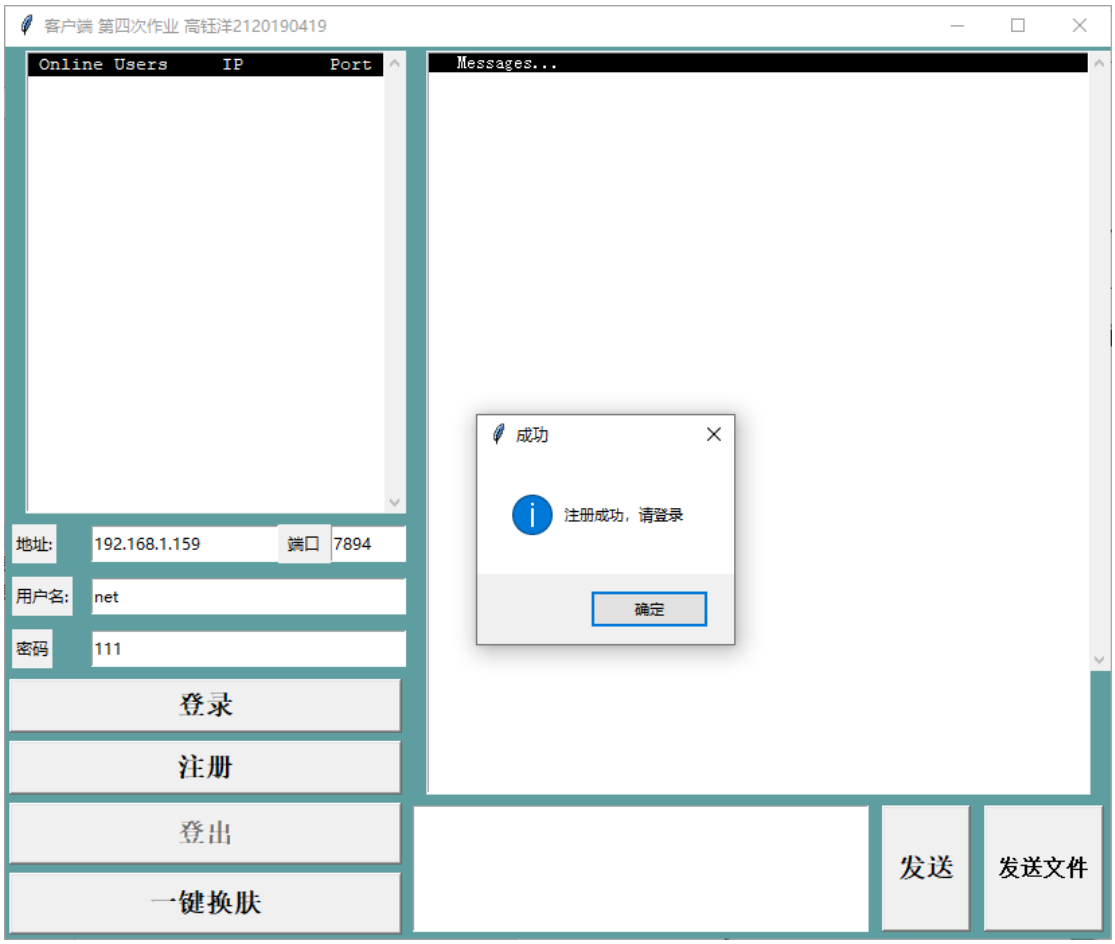


注册

在开启一个客户端，输入服务器的地址和端口，输入想注册的用户名和密码，如我们想注册用户 net,密码为 111，如下图所示



若用户名已存在会返回响应的返回值进行提示，若注册成功则弹出提示框



点击确定，之后便可以用刚才的注册的用户名进行登录

一对一发送消息

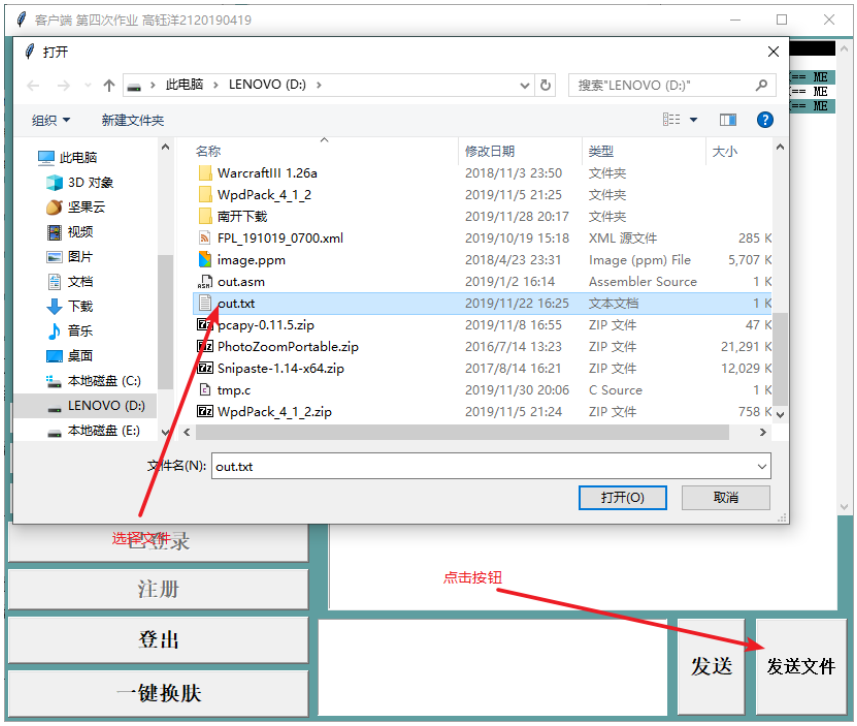
我们用刚才刚注册的用户 net 跟之前登录的 gyy 用户之间演示发送消息

我们令 net 向 gyy 发送一条信息，gyy 向 net 返回三条信息，双方显示如下

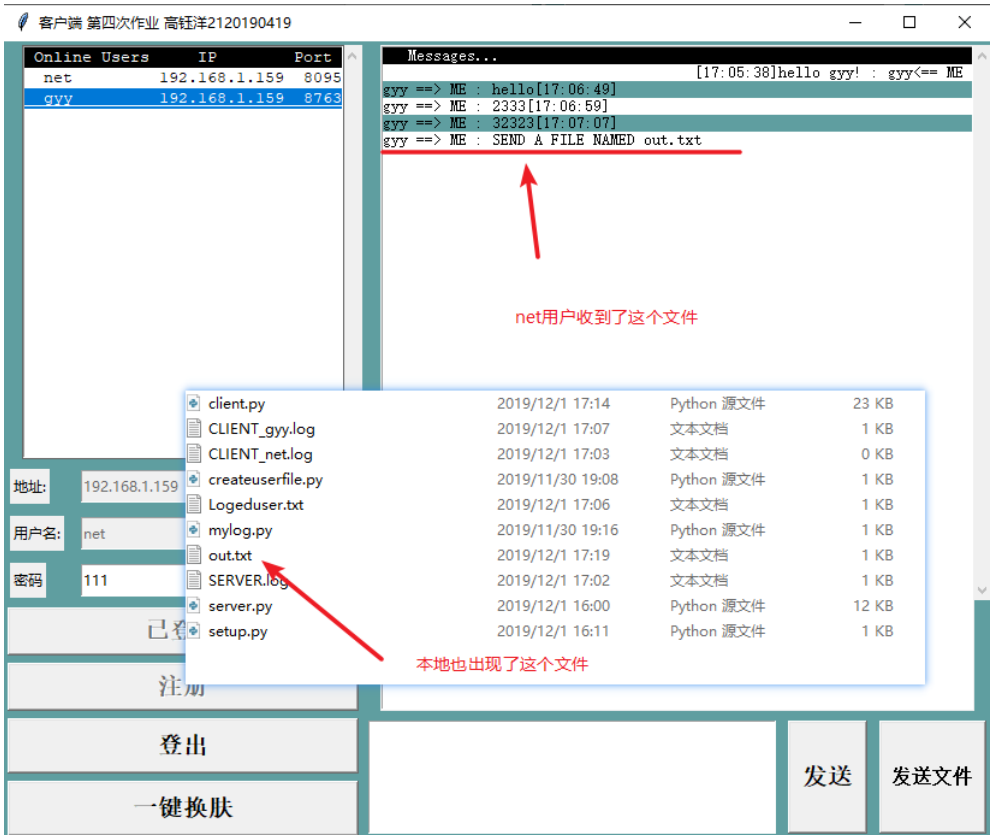
可以看到，发送方的消息显示在右侧，接收方的消息显示在右侧

发送文件

我们通过 gyy 账号像 net 账号发送文件，首先点击发送文件按钮，选择想要发送的文件

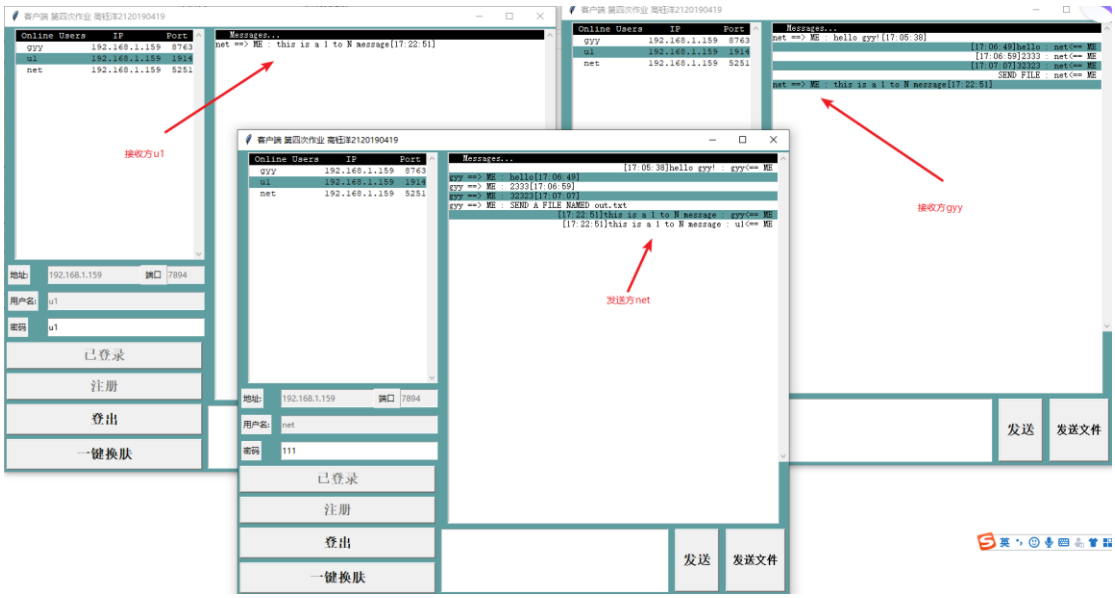


点击打开，可以看到另一端收到了文件发送的消息，另一方收到消息后会把这个文件存到本地文件夹下，可以看到本地也出现了这个文件



一对多发送消息

让我们在注册一个用户 u1,来演示一对多通信的过程，注册和登录过程省略
我们通过 net 账号像另外两个账号 gyy 和 u1 发送消息，可以看到，双方都收到了消息，（**注意多选需要按住 Ctrl**）

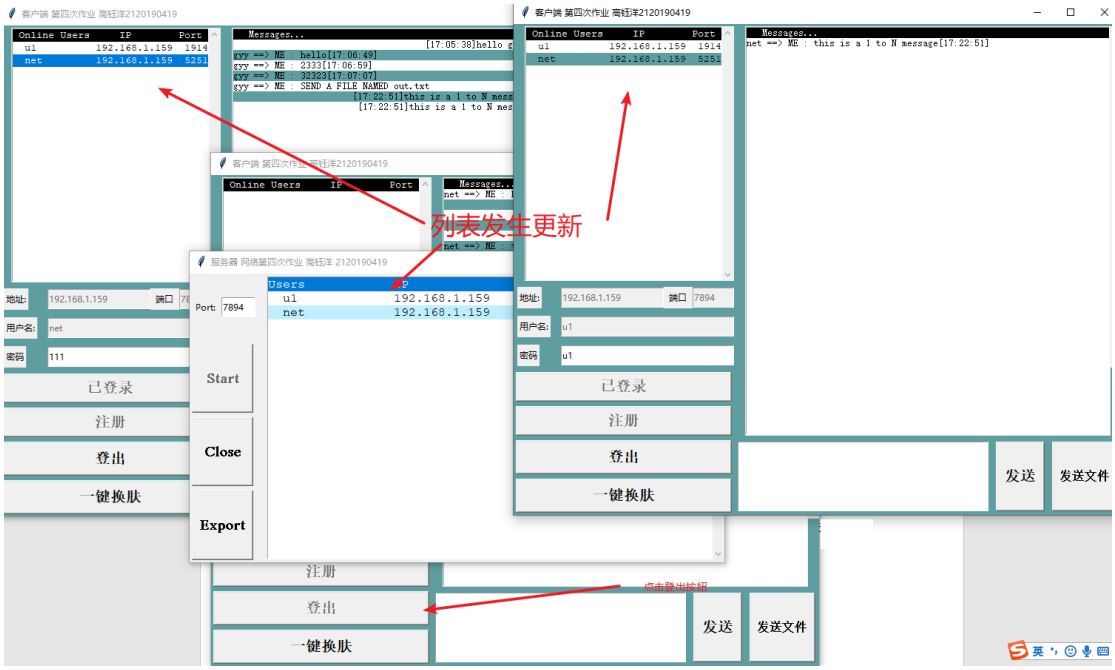


一键换肤

点击客户端一键换肤按钮，系统会自动更换颜色，共有十几种颜色可以更换，用户可以自行体验，下图为几种皮肤实例

登出

点击登出按钮，我们点击 gyy 用户登出按钮，可以看到其他用户的和服务器的列表会进行更新

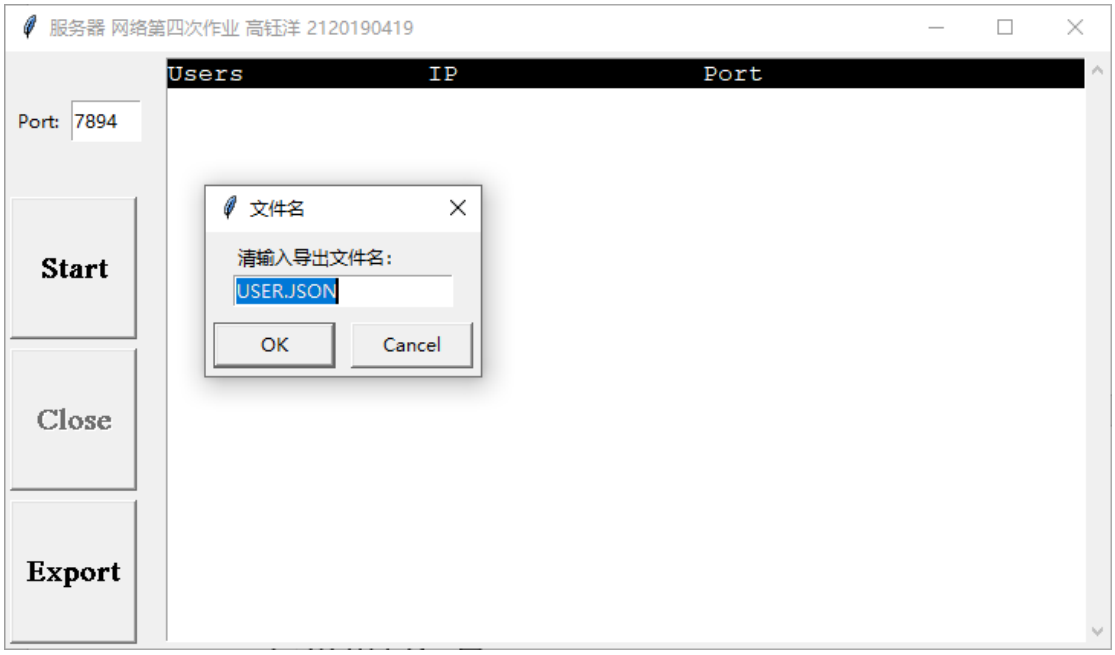


关闭服务器

点击 Close 按钮，关闭服务器，其他客户端被动登出，提示登出成功

导出用户信息

服务器可以在任何时候点击导出用户信息，弹出提示框询问导出文件名，导出文件为 json 格式，默认为 USER.JSON



点击确认，本地出现 USER.JSON 文件，打开可以看到用户的信息


```
{
    "gyy": {
        "name": "gyy",
        "pass": "gyy",
        "ip": "192.168.1.159",
        "port": 8763
    },
    "net": {
        "name": "net",
        "pass": "111",
        "ip": "192.168.1.159",
        "port": 5251
    },
    "u1": {
        "name": "u1",
        "pass": "u1",
        "ip": "192.168.1.159",
        "port": 1914
    }
}
```

日志信息截图

客户端和服务器端的日志如图所示

服务器端

```
2019-12-01 17:02:10 [GET LOGED USER] size = 0
2019-12-01 17:02:15 [Server started] IP: 192.168.1.159 Port: 7894
2019-12-01 17:02:53 [  User login  ] gyy (192.168.1.159:1954)
2019-12-01 17:03:33 [  User login  ] net (192.168.1.159:8095)
2019-12-01 17:06:23 [  User logout ] gyy (192.168.1.159:1954)
2019-12-01 17:06:28 [  User login  ] gyy (192.168.1.159:8763)
2019-12-01 17:21:43 [  User login  ] u1 (192.168.1.159:1914)
2019-12-01 17:21:58 [  User logout ] net (192.168.1.159:8095)
2019-12-01 17:22:02 [  User login  ] net (192.168.1.159:5251)
2019-12-01 17:28:43 [  User logout ] gyy (192.168.1.159:8763)
2019-12-01 17:31:44 [Server closed ] IP: 192.168.1.159 Port: 7894
```

gyy 客户端

```
2019-12-01 17:02:55 [Login Succeed ] User: gyy  Server: 192.168.1.159:7894
2019-12-01 17:02:55 [Logging in ...] User: gyy  Server: 192.168.1.159:7894
2019-12-01 17:05:38        |from net(192.168.1.159:8095): hello gyy![17:05:38]
2019-12-01 17:06:24 [Logged out ...] User: gyy  Server: 192.168.1.159:7894
2019-12-01 17:06:29 [Login Succeed ] User: gyy  Server: 192.168.1.159:7894
2019-12-01 17:06:29 [Logging in ...] User: gyy  Server: 192.168.1.159:7894
2019-12-01 17:06:49        |to net(192.168.1.159:8095): hello[17:06:49]
2019-12-01 17:06:59        |to net(192.168.1.159:8095): 2333[17:06:59]
2019-12-01 17:07:07        |to net(192.168.1.159:8095): 32323[17:07:07]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): Start Sending File!out.txt[17:19:59]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): this is a test file[17:19:59]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): my name is gyy[17:19:59]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): 2120190419[17:19:59]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): [17:19:59]
2019-12-01 17:19:59        |to net(192.168.1.159:8095): Sending Completed![17:19:59]
2019-12-01 17:22:51        |from net(192.168.1.159:5251): this is a 1 to N message[17:22:51]
2019-12-01 17:29:38 [Logged out ...] User: gyy  Server: 192.168.1.159:7894
```