目录

[连接 4](#_Toc40047858)

[自动输入端口号 4](#_Toc40047859)

[自动输入IP地址 4](#_Toc40047860)

[注册 4](#_Toc40047861)

[可以保存注册 6](#_Toc40047862)

[可以检测是否重复注册 6](#_Toc40047863)

[实时导出注册表 6](#_Toc40047864)

[服务器端可以读取注册表 7](#_Toc40047865)

[登录 7](#_Toc40047866)

[登录检测是否注册 7](#_Toc40047867)

[未登录不能使用功能 7](#_Toc40047868)

[可以判断密码是否正确 7](#_Toc40047869)

[用户名设置 8](#_Toc40047870)

[日志界面 8](#_Toc40047871)

[可以记录一些操作 9](#_Toc40047872)

[支持导出日志内容并且可以保存中文内容 9](#_Toc40047873)

[密码设置 9](#_Toc40047874)

[密码设置\*号代替 9](#_Toc40047875)

[20位密码 9](#_Toc40047876)

[在线用户获取 9](#_Toc40047877)

[用户上线提醒 11](#_Toc40047878)

[用户下线提醒 11](#_Toc40047879)

[个人资料 11](#_Toc40047880)

[设置三种个人资料，并上传服务器 12](#_Toc40047881)

[查看他人资料 13](#_Toc40047882)

[选中用户可以查看资料 14](#_Toc40047883)

[上传文件 14](#_Toc40047884)

[支持上传文件（暂时为txt和png，通过代码修改可以添加其他文件类型）并且有错误判断 16](#_Toc40047885)

[上传至服务器自动保存至程序根目录下 ”文件“文件夹 17](#_Toc40047886)

[服务器端自动获取上传文件名，并保存为同名文件，并且以错误判断 17](#_Toc40047887)

[下载文件 17](#_Toc40047888)

[可以查看服务器端文件列表 21](#_Toc40047889)

[选中文件可以下载，并选择下载位置 21](#_Toc40047890)

[下载文件自动获取下载文件名，并默认保存同名文件 21](#_Toc40047891)

[拉入黑名单 21](#_Toc40047892)

[选中用户拉入黑名单 23](#_Toc40047893)

[禁用单独聊天功能 23](#_Toc40047894)

[拉黑用户接收不到被拉黑用户消息 23](#_Toc40047895)

[恢复黑名单 23](#_Toc40047896)

[选中用户恢复聊天 24](#_Toc40047897)

[黑名单列表 24](#_Toc40047898)

[支持List Box控件双击选中 24](#_Toc40047899)

[公共聊天室 24](#_Toc40047900)

[群发消息，在线用户均可收到 26](#_Toc40047901)

[单独聊天室 26](#_Toc40047902)

[设置单独和某位用户聊天 26](#_Toc40047903)

[显示背景图片（娱乐功能） 26](#_Toc40047904)

[服务器端显示在线人数 27](#_Toc40047905)

[当前状态 28](#_Toc40047906)

[设置最大载荷 28](#_Toc40047907)

[启动 29](#_Toc40047908)

[关闭 29](#_Toc40047909)

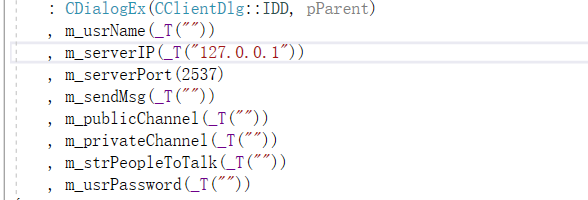
[自动获取服务器计算机名称 30](#_Toc40047910)

**综合局域网聊天室**

# 连接

## 自动输入端口号

## 自动输入IP地址



# 注册

void CClientDlg::OnRegister()

{

// TODO: 在此添加控件通知处理程序代码

if (m\_clientSocket.m\_hSocket == INVALID\_SOCKET)

{

MessageBox(\_T("未连接到服务器！"), \_T("错误"), MB\_ICONERROR);

return;

}

else

{

UpdateData();

if (m\_usrName.IsEmpty()||m\_usrPassword.IsEmpty())

{

MessageBox(\_T("请输入用户名/密码！"), \_T("提示"), MB\_ICONWARNING);

GetDlgItem(IDC\_USERNAME)->SetFocus();

return;

}

if (m\_usrName.GetLength()!=8)

{

MessageBox(\_T("请输入八位用户名"), \_T("提示"), MB\_ICONWARNING);

GetDlgItem(IDC\_USERNAME)->SetFocus();

return;

}

else

{

unsigned char buf[MAX\_BUFSIZE];

buf[0] = REGISTER;

m\_usrNP = m\_usrName + m\_usrPassword;

memcpy(buf + 1, m\_usrNP.GetBuffer(0), m\_usrNP.GetLength() \* 2);

m\_clientSocket.Send(buf, m\_usrNP.GetLength() \* 2 + 1);

GetDlgItem(IDC\_USERNAME)->EnableWindow(FALSE); //登陆后后不可更改用户名

}

}

}

case REGISTER://注册

memcpy(wbuf, buf + 1, 16); //获取用户名

usrName = wbuf;

memcpy(wbuf, buf + 1, len - 1); //获取用户名和密码

usrNP = wbuf;

clientSocket->m\_usrNP = usrNP; //设置用户名和密码

clientSocket->m\_userName = usrName; //设置用户名

for (int i = 0; i < m\_ctrlUsers.GetCount(); i++)

{

CString str;

m\_ctrlUsers.GetText(i, str);

str = str.Left(8);

if (usrName == str)

{

undefine = false;

buf[0] = HADDEFINED;

clientSocket->Send(buf, 1); //用户已注册过

clientSocket->Close();

delete clientSocket;

clientList.pop\_back();

m\_clientCnt--;

return;

}

}

if (undefine)

{

/\*CString log;

CTime tm = CTime::GetCurrentTime();

log.Format(\_T("用户："), clientSocket->m\_userName);

log = tm.Format("%Y-%m-%d %X : ") + log;\*/

this->m\_ctrlUsers.AddString(usrNP);

CString strPath, strLog;

strPath = \_T("users.txt");

CFile file(strPath, CFile::modeCreate | CFile::modeWrite);

for (int i = 0; i < m\_ctrlUsers.GetCount(); i++)

{

CString str;

m\_ctrlUsers.GetText(i, str);

strLog += str;

strLog += \_T("\r\n");

}

WORD unicode = 0xFEFF; //这句重要，注意这里是F E FF，中间有个E

file.Write(&unicode, 2); //这句重要

file.Write(strLog, strLog.GetLength() \* 2);

file.Close();

tm = CTime::GetCurrentTime();

log.Format(\_T("【%s : %u】与服务器断开。"), clientSocket->m\_userIP, clientSocket->m\_userPort);

log = tm.Format("%Y-%m-%d %X : ") + log;

m\_ctrlChatRoomInfo.AddString(log);

buf[0] = REGISTER;

clientSocket->Send(buf, 1); //注册用户

clientSocket->Close();

delete clientSocket;

clientList.pop\_back();

m\_clientCnt--;

UpdateData(false);

return;

}

break;

## 可以保存注册

## 可以检测是否重复注册

## 实时导出注册表

## 服务器端可以读取注册表

# 登录

void CClientDlg::OnLogoIn()

{

// TODO: 在此添加控件通知处理程序代码

if (m\_clientSocket.m\_hSocket == INVALID\_SOCKET)

{

MessageBox(\_T("未连接到服务器！"), \_T("错误"), MB\_ICONERROR);

return;

}

else

{

UpdateData();

if (m\_usrName.IsEmpty()||m\_usrPassword.IsEmpty())

{

MessageBox(\_T("请输入用户名/密码！"), \_T("提示"), MB\_ICONWARNING);

GetDlgItem(IDC\_USERNAME)->SetFocus();

return;

}

unsigned char buf[MAX\_BUFSIZE];

buf[0] = NEWCLIENT;

m\_usrNP = m\_usrName + m\_usrPassword;

//m\_usrNP +=m\_usrPassword;

memcpy(buf + 1, m\_usrNP.GetBuffer(0), m\_usrNP.GetLength() \* 2);

m\_clientSocket.Send(buf, m\_usrNP.GetLength() \* 2 + 1);

GetDlgItem(IDC\_USERNAME)->EnableWindow(FALSE); //登陆后后不可更改用户名

GetDlgItem(IDC\_BUTTON1)->EnableWindow(FALSE); //登陆后后不可再点击登录

GetDlgItem(IDC\_BUTTON2)->EnableWindow(false); //登陆后不可以再点击注册

GetDlgItem(IDC\_DOWNLOADFILE)->EnableWindow(true); //连接后可上传文件

}

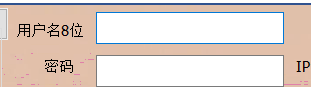
}

## 登录检测是否注册

## 未登录不能使用功能

## 可以判断密码是否正确

# 用户名设置



# 日志界面

void CServerDlg::OnBnClickedBtnExport()

{

// TODO: 在此添加控件通知处理程序代码

CFileDialog dlg(FALSE,NULL,NULL,OFN\_HIDEREADONLY | OFN\_OVERWRITEPROMPT,

\_T("Text Files(\*.txt)||"),AfxGetMainWnd());

CString strPath,strLog;

if (dlg.DoModal() == IDOK)

{

strPath=dlg.GetPathName();

if (strPath.Right(4)!=".txt")

strPath+=".txt";

}

CFile file(strPath,CFile::modeCreate|CFile::modeWrite);

for (int i = 0; i < m\_ctrlChatRoomInfo.GetCount(); i++)

{

CString str;

m\_ctrlChatRoomInfo.GetText(i,str);

strLog+=str;

strLog+=\_T("\r\n");

}

WORD unicode = 0xFEFF; //这句重要，注意这里是F E FF，中间有个E

file.Write(&unicode, 2); //这句重要

//const unsigned char LeadBytes[] = { 0xEF, 0xBB, 0xBF };//UTF-8带bom

//const unsigned char LeadBytes[] = { 0xFE, 0xFF };

//file.Write(LeadBytes, sizeof(LeadBytes));

//int i = strLog.GetLength()\*2;

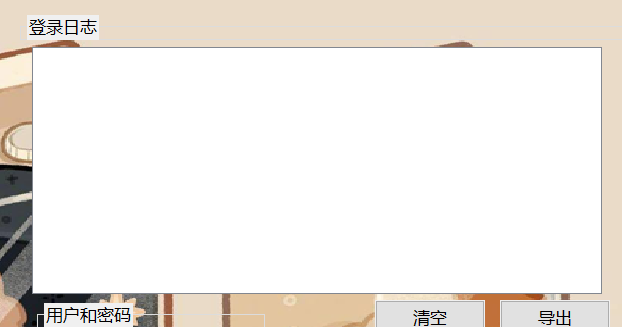
//wchar\_t str[i] = { 0 };

file.Write(strLog,strLog.GetLength()\*2);

file.Close();

MessageBox(\_T("导出成功！"),\_T("提示"),MB\_ICONINFORMATION);

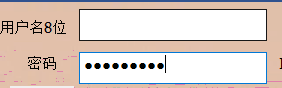
}



## 可以记录一些操作

## 支持导出日志内容并且可以保存中文内容

# 密码设置



## 密码设置\*号代替

## 20位密码

# 在线用户获取

if (NEWCLIENT == buf[0]) //新用户上线消息

{

memcpy(usrName, buf + 1, len-1); //用户名

if (LB\_ERR == m\_ctrlUserOnline.FindString(-1,usrName)) //在线用户增加

{

m\_ctrlUserOnline.AddString(usrName);

}

if (LB\_ERR == m\_ctrlPeopleToTalk.FindString(-1,usrName)) //可以私聊的人增加

{

m\_ctrlPeopleToTalk.AddString(usrName);

}

str.Empty();

str+=\_T("\t\t【");

str+=usrName;

str+=\_T("】");

str+=enterStyle[rand()%MAX\_ENTERSTYLE];

str+=\_T("聊天室，大家欢迎！\r\n");

m\_publicChannel+=str;

}

case NEWCLIENT: //收到的是新用户上线消息，转发给所有人，并给新用户发送全部在线用户信息

memcpy(wbuf, buf + 1, 16); //获取用户名

usrName = wbuf;

memcpy(wbuf, buf + 1, len - 1); //获取用户名和密码

usrNP = wbuf;

clientSocket->m\_userName = usrName; //设置用户名

for (int i = 0; i < m\_ctrlUsers.GetCount(); i++)

{

CString str;

m\_ctrlUsers.GetText(i, str);

if (usrNP == str)

{

undefine = false;

break;

}

}

if (undefine)

{

tm = CTime::GetCurrentTime();

log.Format(\_T("【%s : %u】与服务器断开。"), clientSocket->m\_userIP, clientSocket->m\_userPort);

log = tm.Format("%Y-%m-%d %X : ") + log;

m\_ctrlChatRoomInfo.AddString(log);

buf[0] = ERROR;

clientSocket->Send(buf, 1); //用户未注册或密码不正确

clientSocket->Close();

delete clientSocket;

clientList.pop\_back();

m\_clientCnt--;

return;

}

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if (\*iter != NULL)

{

(\*iter)->Send(buf, 17); //转发到每个客户端

//前十七个字节时是 一个字节标志和8个宽字符

}

}

Sleep(100);

for (iter = clientList.begin(); iter < clientList.end() - 1; iter++)

{

if (\*iter != NULL)

{

buf[0] = ONLINECLIENT;

memcpy(wbuf, (\*iter)->m\_userName.GetBuffer(0), (\*iter)->m\_userName.GetLength() \* 2);

memcpy(buf + 1, wbuf, (\*iter)->m\_userName.GetLength() \* 2);

clientSocket->Send(buf, (\*iter)->m\_userName.GetLength() \* 2 + 1); //发送在线用户的信息给新上线的用户

Sleep(100);

}

}

break;

## 用户上线提醒

## 用户下线提醒

# 个人资料

void CClientDlg::OnMydata()

{

// TODO: 在此添加控件通知处理程序代码

DataDlg datadlg = new DataDlg();

ot\_dataName = m\_usrName;

m\_judgeSelf = true;

if (IDOK == datadlg.DoModal())

{

m\_dataAge = datadlg.Data\_dataAge;

m\_dataLove = datadlg.Data\_dataLove;

m\_dataSex = datadlg.Data\_dataSex;

}

unsigned char data\_buf[MAX\_BUFSIZE] = { 0 };

data\_buf[0] = UpDATA; //发送资料

memcpy(data\_buf + 1, m\_usrName.GetBuffer(0), m\_usrName.GetLength() \* 2); //16字节上传资料的用户名

memcpy(data\_buf + 17, m\_dataAge.GetBuffer(0), m\_dataAge.GetLength() \* 2); //6字年龄

memcpy(data\_buf + 23, m\_dataSex.GetBuffer(0), m\_dataSex.GetLength() \* 2); //2字性别

memcpy(data\_buf + 25, m\_dataLove.GetBuffer(0), m\_dataLove.GetLength() \* 2); //40字节爱好

m\_clientSocket.Send(data\_buf, m\_dataLove.GetLength() \* 2 + 25); //发送消息

}

case UpDATA://上传资料

memcpy(wbuf, buf + 1, 16);

usrName = wbuf;

memcpy(wbuf + 16, buf + 17, 6);

usrAge = wbuf + 16;

memcpy(wbuf + 22, buf + 23, 2);

usrSex = wbuf + 22;

memcpy(wbuf + 24, buf + 25, len - 25);

usrLove = wbuf + 24;

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if ((\*iter)->m\_userName == usrName)

{

//保存用户上传的资料

(\*iter)->m\_usrAge = usrAge;

(\*iter)->m\_usrSex = usrSex;

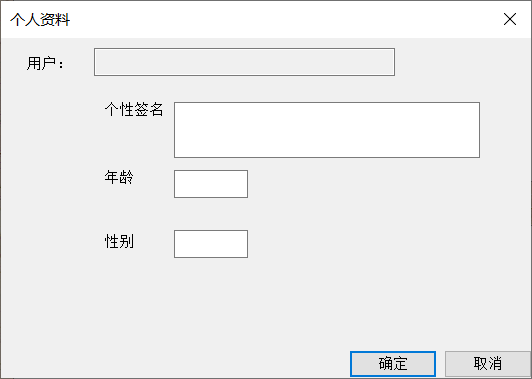
(\*iter)->m\_usrLove = usrLove;

break;

}

}

break;



## 设置三种个人资料，并上传服务器

# 查看他人资料

//查看他人资料

void CClientDlg::OnBnClickedBtnOtherdata()

{

// TODO: 在此添加控件通知处理程序代码

if (LbnDblclkUseronline)

{

unsigned char buf[MAX\_BUFSIZE];

buf[0] = WDATA;

memcpy(buf + 1, ot\_dataName.GetBuffer(0), ot\_dataName.GetLength() \* 2);//16字节所看资料用户名

memcpy(buf + 17, m\_usrName.GetBuffer(0), m\_usrName.GetLength() \* 2);//16字节请求用户名

m\_clientSocket.Send(buf,33);

}

else

MessageBox(\_T("未选择用户，请双击选择！"), \_T("提示"), MB\_ICONWARNING);

}

//双击listbox某一行的响应函数

void CClientDlg::OnLbnDblclkUseronline()

{

// TODO: 在此添加控件通知处理程序代码

m\_ctrlUserOnline.GetText(m\_ctrlUserOnline.GetCurSel(), ot\_dataName);//得到指定行内容，给text

LbnDblclkUseronline = true;

}

case WDATA://发送他人资料

memcpy(dstName, buf + 1, 16);//被看的

memcpy(srcName, buf + 17, 16);//要看的

data\_c\_srcName = srcName;

data\_buf[0] = WDATA;

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if ((\*iter)->m\_userName == dstName) //被看的用户

{

//发送资料

usrAge = (\*iter)->m\_usrAge;

usrSex = (\*iter)->m\_usrSex;

usrLove = (\*iter)->m\_usrLove;

break;

}

}

memcpy(data\_buf + 1, data\_c\_srcName.GetBuffer(0), data\_c\_srcName.GetLength() \* 2);//16字节要看资料的用户名

memcpy(data\_buf + 17, usrAge.GetBuffer(0), usrAge.GetLength() \* 2);//6字节年龄

memcpy(data\_buf + 23, usrSex.GetBuffer(0), usrSex.GetLength() \* 2);//2字节性别

memcpy(data\_buf + 25, usrLove.GetBuffer(0), usrLove.GetLength() \* 2);//爱好

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if ((\*iter)->m\_userName == srcName) //被看的用户

{

//发送资料

(\*iter)->Send(data\_buf, usrLove.GetLength() \* 2 + 25); //发送消息

break;

}

}

break;

## 选中用户可以查看资料

# 上传文件

void CClientDlg::OnUPFILE()

{

// TODO: 在此添加控件通知处理程序代码

unsigned char file\_buf[MAX\_BUFSIZE] = { 0 };

file\_buf[0] = UPFILE; //发送文件

CFileDialog dlg(TRUE, NULL, NULL, OFN\_HIDEREADONLY | OFN\_OVERWRITEPROMPT,

\_T("文本文档 \*.txt|\*.txt|图片 \*.png|\*.png|"), AfxGetMainWnd());

CString strPath, strLog,strName;

unsigned int strName\_length = 0;

if (dlg.DoModal() == IDOK)

{

strPath = dlg.GetPathName();

strName = dlg.GetFileName();

strName\_length = strName.GetLength()\*2;

}

char chName\_length = strName\_length;

file\_buf[1] = chName\_length;

memcpy(file\_buf + 2, strName, strName\_length);

HANDLE hFile = CreateFile(strPath, FILE\_GENERIC\_READ, // 打开文件，获得文件读句柄

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE | FILE\_SHARE\_DELETE, // 共享方式打开，避免其他地方需要读写此文件

NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

DWORD dwSize = GetFileSize(hFile, NULL);

if (INVALID\_HANDLE\_VALUE == hFile) // 文件打开失败，返回错误值

MessageBox(\_T("打开文件失败"), \_T("提示"), MB\_ICONWARNING);

ReadFile(hFile, file\_buf+2+strName\_length, dwSize, &dwSize, NULL);

if (dwSize == GetFileSize(hFile, NULL))

{

CloseHandle(hFile);

AfxMessageBox(\_T("发送完成"));

}

m\_clientSocket.Send(file\_buf, dwSize+2+strName\_length); //发送消息

}

case UPFILE:

CH\_nameLength = NULL;

IN\_namelength= buf[1]-CH\_nameLength;

strPath.Empty();

strPath += \_T("文件\\");

memcpy(wbuf, buf + 2, IN\_namelength);

strPath += wbuf;

temp\_File\_Name.Empty();

temp\_File\_Name += buf[1];//每个文件名的长度

temp\_File\_Name += wbuf;

hFile = CreateFile(strPath, GENERIC\_WRITE, 0, 0, OPEN\_ALWAYS, 0, 0);

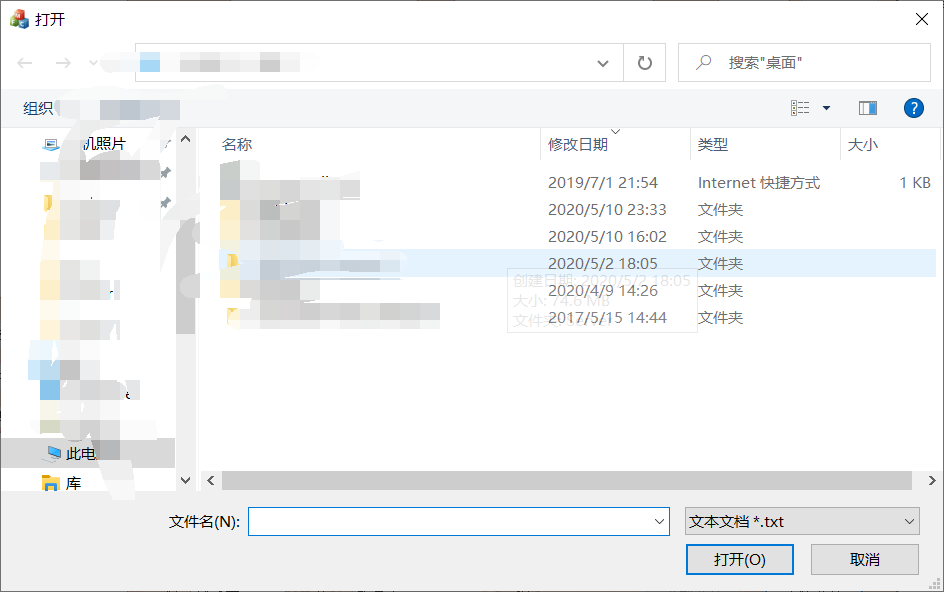
DWORD nBytes;

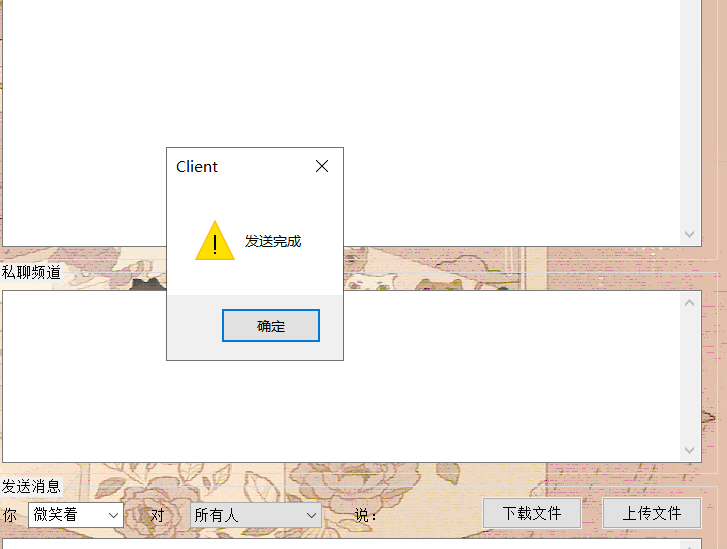
WriteFile(hFile, buf+2+IN\_namelength, len-2-IN\_namelength, &nBytes, 0);

CloseHandle(hFile);

filelist.push\_back(temp\_File\_Name);

break;





## 支持上传文件（暂时为txt和png，通过代码修改可以添加其他文件类型）并且有错误判断

## 上传至服务器自动保存至程序根目录下 ”文件“文件夹

## 服务器端自动获取上传文件名，并保存为同名文件，并且以错误判断

# 下载文件

void CClientDlg::OnDOWNLOADFILE()

{

// TODO: 在此添加控件通知处理程序代码

unsigned char down\_file\_buf[MAX\_BUFSIZE] = { 0 };

down\_file\_buf[0] = WATCH\_FILELIST; //发送资料

memcpy(down\_file\_buf + 1, m\_usrName.GetBuffer(0), m\_usrName.GetLength() \* 2); //16字节上传资料的用户名

m\_clientSocket.Send(down\_file\_buf, m\_usrName.GetLength() \* 2 + 1); //发送消息

}

void CClientDlg::DownLoad()

{

unsigned char down\_file\_buf[MAX\_BUFSIZE] = { 0 };

down\_file\_buf[0] = DOWNLOAD\_FILE; //发送资料

down\_file\_buf[1] = num;

memcpy(down\_file\_buf + 2, m\_usrName.GetBuffer(0), m\_usrName.GetLength() \* 2); //16字节上传资料的用户名

m\_clientSocket.Send(down\_file\_buf, m\_usrName.GetLength() \* 2 + 2); //发送消息

}

case WATCH\_FILELIST:

memcpy(wbuf, buf + 1, 16);

usrName = wbuf;

temp\_File\_List.Empty();

for (file\_iter = filelist.begin(); file\_iter != filelist.end();file\_iter++)

{

temp\_File\_List +=\*file\_iter;

}

data\_buf[0] = WATCH\_FILELIST;

memcpy(data\_buf + 1, temp\_File\_List.GetBuffer(0), temp\_File\_List.GetLength() \* 2+1);

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if ((\*iter)->m\_userName == usrName)

{

(\*iter)->Send(data\_buf, temp\_File\_List.GetLength() \* 2 + 1);

break;

}

}

break;

case DOWNLOAD\_FILE:

CH\_nameLength = NULL;

File\_num = buf[1] - CH\_nameLength;

memcpy(wbuf, buf + 2, 16);

usrName = wbuf;

strPath.Empty();

strPath += \_T("文件\\");

strPath += filelist[File\_num].Mid(1);

data\_buf[0] = DOWNLOAD\_FILE;

//\_splitpath((char\*)(LPCTSTR)strPath, NULL, NULL, NULL, charFileLog);

//File\_num = strlen(charFileLog);

//CH\_nameLength = File\_num;//传输后缀名长度；

//data\_buf[1] = CH\_nameLength;

memcpy(data\_buf + 1, filelist[File\_num], filelist[File\_num].GetLength()\*2);

hFile = CreateFile(strPath, FILE\_GENERIC\_READ, // 打开文件，获得文件读句柄

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE | FILE\_SHARE\_DELETE, // 共享方式打开，避免其他地方需要读写此文件

NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL);

dwSize = GetFileSize(hFile, NULL);

if (INVALID\_HANDLE\_VALUE == hFile) // 文件打开失败，返回错误值

MessageBox(\_T("打开文件失败"), \_T("提示"), MB\_ICONWARNING);

ReadFile(hFile, data\_buf + 1+ filelist[File\_num].GetLength() \* 2, dwSize, &dwSize, NULL);

CloseHandle(hFile);

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if ((\*iter)->m\_userName == usrName)

{

(\*iter)->Send(data\_buf, dwSize+1+ filelist[File\_num].GetLength() \* 2);//信号+后缀长度+后缀名+文件长度

break;

}

}

break;

else if (WATCH\_FILELIST == buf[0])

{

for (int i = 1; buf[i] != 0; i+=IN\_namelength+2)

{

CH\_nameLength = NULL;

IN\_namelength = buf[i] - CH\_nameLength;

wchar\_t wbuf[100] = { 0 };

memcpy(wbuf, buf + 1+i+1,IN\_namelength);

temp\_File\_Name = wbuf;

filelist.push\_back(temp\_File\_Name);

}

CSelect\_Files select\_file = new CSelect\_Files;

if (IDOK == select\_file.DoModal())

{

num = select\_file.CSelect\_num;

Select\_File\_OK = select\_file.CSelect\_Select\_File\_OK;

filelist.clear();

}

else

filelist.clear();

if (Select\_File\_OK)

{

DownLoad();

}

}

else if (DOWNLOAD\_FILE == buf[0])

{

CString strPath, strName;

CH\_nameLength = NULL;

IN\_namelength = buf[1] - CH\_nameLength;//文件名长度

wchar\_t wbuf[100] = { 0 };

memcpy(wbuf, buf + 3, IN\_namelength);

strName = wbuf;

CFileDialog dlg(FALSE, NULL, strName, OFN\_HIDEREADONLY | OFN\_OVERWRITEPROMPT,

\_T("所有文件(\*.\*)||"), AfxGetMainWnd());

if (dlg.DoModal() == IDOK)

{

strPath = dlg.GetPathName();

}

//strPath += strName;

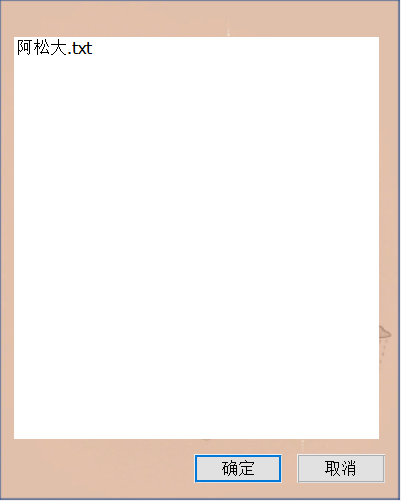
HANDLE hFile = CreateFile(strPath, GENERIC\_WRITE, 0, 0, OPEN\_ALWAYS, 0, 0);

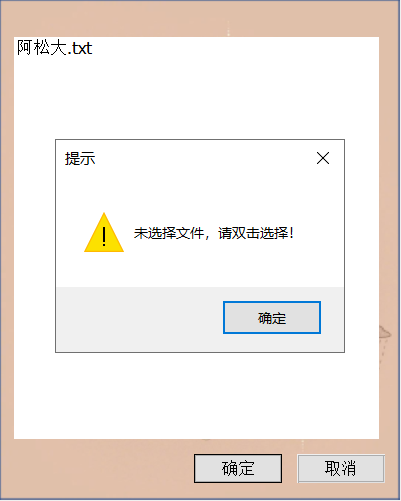
DWORD nBytes;

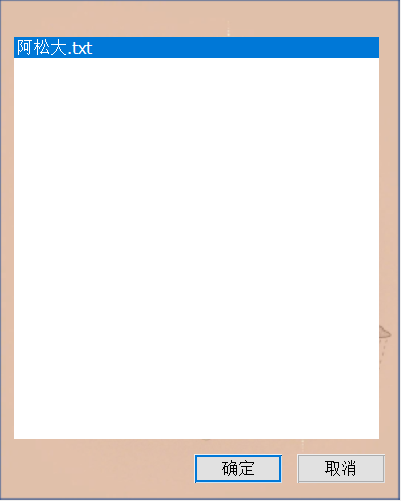
WriteFile(hFile, buf + 3+ IN\_namelength, len -1, &nBytes, 0);

CloseHandle(hFile);

}







## 可以查看服务器端文件列表

## 选中文件可以下载，并选择下载位置

## 下载文件自动获取下载文件名，并默认保存同名文件

# 拉入黑名单

//拉黑

void CClientDlg::OnBnClickedToBlacklist()

{

// TODO: 在此添加控件通知处理程序代码

m\_BlackName = ot\_dataName;

if (LbnDblclkUseronline&&m\_BlackName!=m\_usrName)

{

/\*unsigned char buf[MAX\_BUFSIZE];

buf[0] = BLACK;

memcpy(buf + 1, m\_BlackName.GetBuffer(0), m\_BlackName.GetLength() \* 2);//16字节所看资料用户名

m\_clientSocket.Send(buf, 17);\*/

m\_ctrlBlackName.AddString(m\_BlackName);

int n = m\_ctrlPeopleToTalk.FindString(-1, m\_BlackName);

if (LB\_ERR != n)

{

m\_ctrlPeopleToTalk.DeleteString((UINT)n);

}

}

else if(!LbnDblclkUseronline)

MessageBox(\_T("未选择用户，请双击选择！"), \_T("提示"), MB\_ICONWARNING);

else if (m\_BlackName==m\_usrName)

MessageBox(\_T("不能拉黑自己！"), \_T("提示"), MB\_ICONWARNING);

/\*if (LbnDblclkUseronline)

{

unsigned char buf[MAX\_BUFSIZE];

buf[0] = BLACK;

memcpy(buf + 1, m\_BalckName.GetBuffer(0), m\_BalckName.GetLength() \* 2);//16字节所看资料用户名

m\_clientSocket.Send(buf, 17);

m\_ctrlBlackName.AddString(m\_BalckName);

}

else

MessageBox(\_T("未选择用户，请双击选择！"), \_T("提示"), MB\_ICONWARNING);\*/

LbnDblclkUseronline = false;

}

void CClientDlg::OnLbnDblclkBlacklist()

{

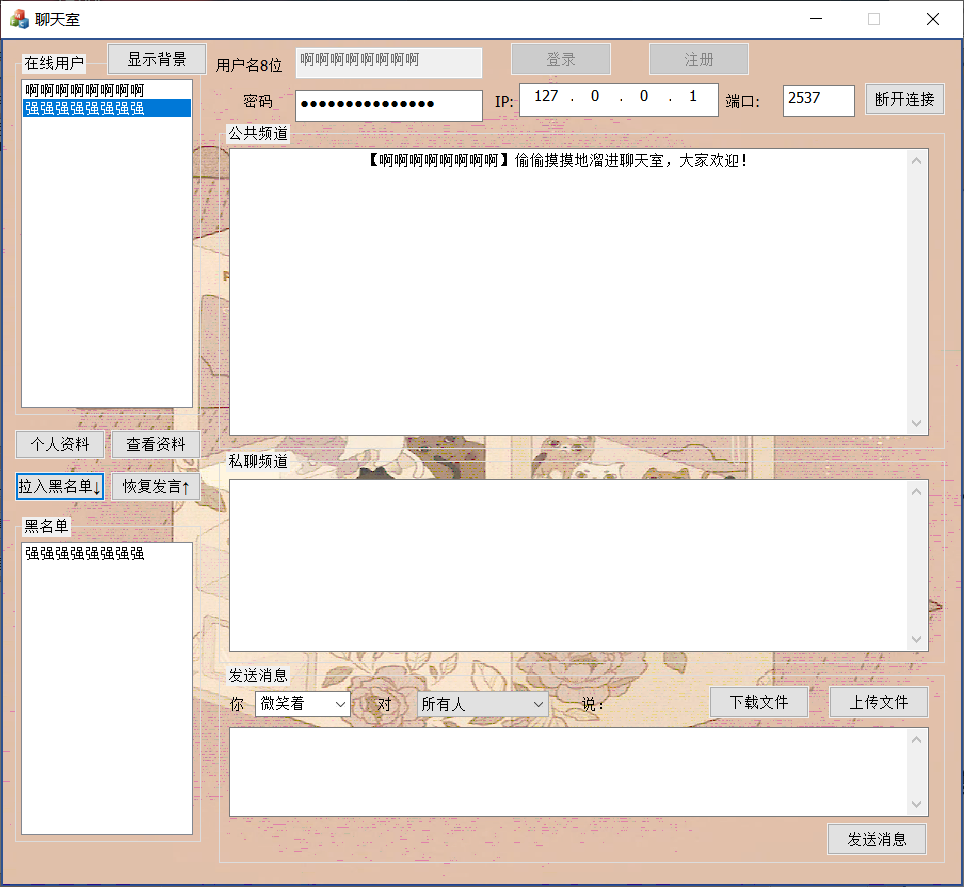
// TODO: 在此添加控件通知处理程序代码

i = m\_ctrlBlackName.GetCurSel();

m\_ctrlBlackName.GetText(i, m\_BlackName);//得到指定行内容，给text

DblcBlacklist = true;

}



## 选中用户拉入黑名单

### 禁用单独聊天功能

### 拉黑用户接收不到被拉黑用户消息

# 恢复黑名单

void CClientDlg::OnBnClickedRestroe()

{

// TODO: 在此添加控件通知处理程序代码

if (DblcBlacklist)

{

m\_ctrlBlackName.DeleteString(i);

m\_ctrlPeopleToTalk.AddString(m\_BlackName);

m\_ctrlBlackName.SetRedraw(FALSE); //更新内容

}

else

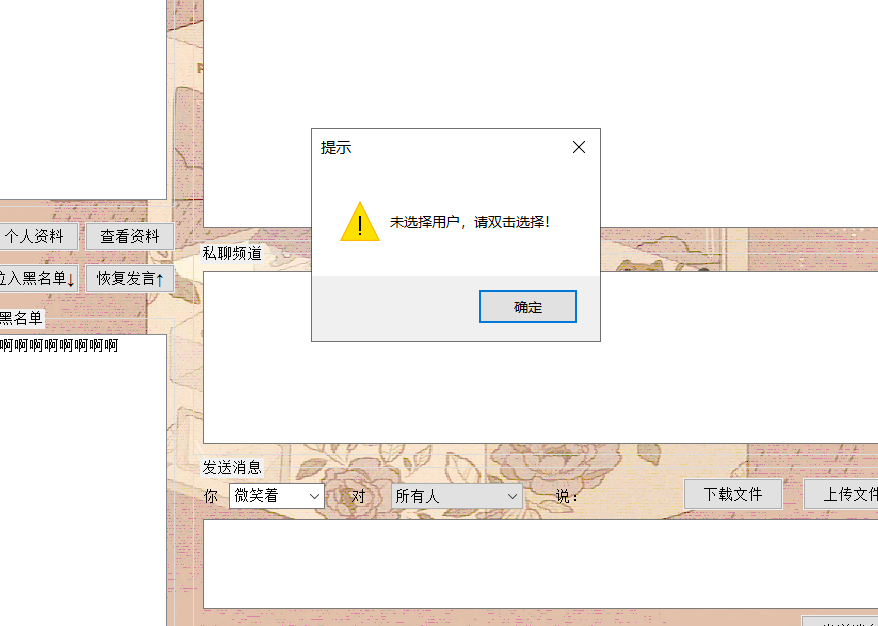
MessageBox(\_T("未选择用户，请双击选择！"), \_T("提示"), MB\_ICONWARNING);

DblcBlacklist = false;

}

## 选中用户恢复聊天

# 黑名单列表



## 支持List Box控件双击选中

# 公共聊天室

void CClientDlg::OnBnClickedSendmsg()

{

// TODO: 在此添加控件通知处理程序代码

UpdateData();

if (m\_clientSocket.m\_hSocket == INVALID\_SOCKET)

{

MessageBox(\_T("未连接到服务器！"),\_T("错误"),MB\_ICONERROR);

return;

}

unsigned char buf[MAX\_BUFSIZE]={0};

CString strStyle;

GetDlgItem(IDC\_MSGSTYTLE)->GetWindowTextW(strStyle);

if (m\_ctrlPeopleToTalk.GetCurSel() == 0) //选中了第0项，也就是发送给所有人

{

buf[0] = MESSAGE\_ALL; //公共消息

memcpy(buf+1,m\_usrName.GetBuffer(0),m\_usrName.GetLength()\*2); //16字节用户名

memcpy(buf+17,strStyle.GetBuffer(0),strStyle.GetLength()\*2); //10字节消息方式

memcpy(buf+27,m\_sendMsg.GetBuffer(0),m\_sendMsg.GetLength()\*2); //消息主体

m\_clientSocket.Send(buf,m\_sendMsg.GetLength()\*2+27); //发送消息

}

else

{

UpdateData();

buf[0] = MESSAGE\_ONE; //私聊消息

memcpy(buf+1,m\_strPeopleToTalk.GetBuffer(0),m\_strPeopleToTalk.GetLength()\*2); //16字节目的用户名

memcpy(buf+17,m\_usrName.GetBuffer(0),m\_usrName.GetLength()\*2); //16字节源用户名

memcpy(buf+33,strStyle.GetBuffer(0),strStyle.GetLength()\*2); //10字节消息方式

memcpy(buf+43,m\_sendMsg.GetBuffer(0),m\_sendMsg.GetLength()\*2); //消息主体

m\_clientSocket.Send(buf,m\_sendMsg.GetLength()\*2+43); //发送消息

m\_privateChannel+=\_T("【你】");

m\_privateChannel+=strStyle;

m\_privateChannel+=\_T("\* 对 【");

m\_privateChannel+=m\_strPeopleToTalk;

m\_privateChannel+=\_T("】 说：");

m\_privateChannel+=m\_sendMsg;

m\_privateChannel+=\_T("\r\n");

UpdateData(FALSE);

}

}

case MESSAGE\_ALL: //收到的是发送给所有人的聊天消息

for (iter = clientList.begin(); iter < clientList.end(); iter++)

{

if (\*iter != NULL)

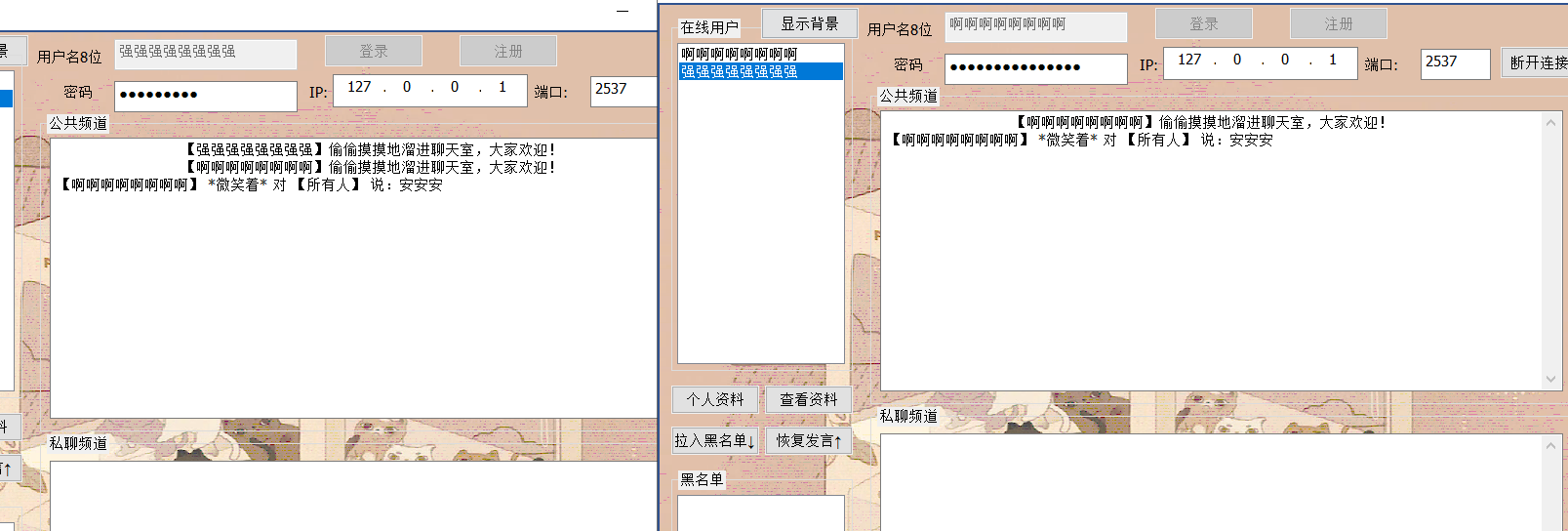
{

(\*iter)->Send(buf, len); //转发到每个客户端

}

}

break;



## 群发消息，在线用户均可收到

# 单独聊天室

## 设置单独和某位用户聊天

# 显示背景图片（娱乐功能）

void CClientDlg::OnShowBackGround()

{

CBackGround back = new CBackGround;

back.DoModal();

}



# 服务器端显示在线人数



# 当前状态



# 设置最大载荷



# 启动



# 关闭



# 自动获取服务器计算机名称

