# 服务器基本框架

采用单进程多线程的模式。

监听线程，一个线程专门用来监听，一有用户连接就为他专门建立一个线程来为他服务，即为用户线程。

用户线程，用户发来信息，用户线程根据信息做出相应的动作，返回信息给用户。

用户发出的信息协议格式

(消息号，消息参数)

消息参数不止一个，消息参数之间用符号隔开，消息号也要与消息参数隔开，隔开符可以一样的。

用户信息处理的一个简单例子

C#的一个方法，

说白了就是在if语句中加功能，涉及复杂问题以后再处理。

public void ClientService()

{

string[] acceptStr = null;//保存消息字符

byte[] buff = new byte[1024];//缓冲区

bool keepConnected = true;

//用循环不断地与客户端进行交互,直到其发出EXIT或者QUIT命令

//将keepConnected设置为false,退出循环，关闭当前连接,中止当前线程

while (keepConnected && server.ServerFlag)

{

acceptStr = null;

try

{

if (currentSocket == null || currentSocket.Available < 1)

{

Thread.Sleep(300);

continue;

}

//接收信息存入buff数组中

int length = currentSocket.Receive(buff);

//将字符数组转换为字符串

string Command = Encoding.Default.GetString(buff, 0, length);

//对字符串按分隔符拆分

acceptStr = Command.Split(new char[] { '|' });

if (acceptStr == null)

{

Thread.Sleep(200);

continue;

}

}

catch (Exception ex)

{

server.UpdateMsg("程序发生异常,异常信息:" + ex.Message);

}

if (acceptStr[0] == "CONNECT")

{

//如果为CONECT，则它的命令格式为“CONNECT|发送者的用户名：附件信息”

//则acceptStr[1]保存的就是用户名

this.name = acceptStr[1];

if (server.clients.Contains(this.name))

{

SendToClient(this, "ERORR|User" + this.name + "已经存在！");

}

else

{

Hashtable synClients = Hashtable.Synchronized(server.clients);

synClients.Add(this.name, this);

//更新界面

server.AddUser(this.name);

//对每一个在线用户发送JOIN和LIST消息命令更新用户列表

IEnumerator myIEnumerator = server.clients.Values.GetEnumerator();

while (myIEnumerator.MoveNext())

{

ChatClient cClient = (ChatClient)myIEnumerator.Current;

SendToClient(cClient, "JOIN|" + this.name + "|");

Thread.Sleep(100);

}

//更新连接状态

connState = "connected";

SendToClient(this, "OK");

//向客户端发送LIST命令更新用户列表

string msgUsers = "LIST|" + server.GetUserList();

SendToClient(this, msgUsers);

}

}

else if (acceptStr[0] == "LIST")

{

if (connState == "CONNECTED")

{

//向客户端发送LIST命令，以此更新客户端的当前在线用户列表

string msgUsers = "LIST|" + server.GetUserList();

SendToClient(this, msgUsers);

}

else

{

SendToClient(this, "ERROR|服务器未连接,请重新登录");

}

}

else if (acceptStr[0] == "CHAT")

{

if (connState == "connected")

{

//此时接收到的命令的格式为：

//命令标志符（CHAT）|发送者的用户名：发送内容|

//向所有当前在线的用户转发此信息

IEnumerator myEnumerator = server.clients.Values.GetEnumerator();

while (myEnumerator.MoveNext())

{

ChatClient client = (ChatClient)myEnumerator.Current;

//将“发送者的用户名：发送内容”转发给用户

SendToClient(client, acceptStr[1]);

}

server.UpdateMsg(acceptStr[1]);

}

else

{

//发送错误信息

SendToClient(this, "ERR|服务器未连接,请重新登录");

}

}

else if (acceptStr[0] == "PRIV")

{

if (connState == "connected")

{

//此时接收到的命令格式为：

//命令标志符（PRIV）|发送者用户名|接收者用户名|发送内容|

//acceptStr[1]中保存了发送者的用户名

string sender = acceptStr[1];

//acceptStr[2]中保存了接收者的用户名

string receiver = acceptStr[2];

//acceptStr[3]中保存了发送的内容

string content = acceptStr[3];

string message = sender + " ---> " + receiver + ": " + content;

//仅将信息转发给发送者和接收者

if (server.clients.Contains(sender))

{

SendToClient(

(ChatClient)server.clients[sender], message);

}

if (server.clients.Contains(receiver))

{

SendToClient(

(ChatClient)server.clients[receiver], message);

}

server.UpdateMsg(message);

}

else

{

//发送错误信息

SendToClient(this, "ERROR|服务器未连接,请重新登录");

}

}

else if (acceptStr[0] == "EXIT")

{

//此时接收到的命令的格式为：命令标志符（EXIT）|发送者的用户名

//向所有当前在线的用户发送该用户已离开的信息

if (server.clients.Contains(acceptStr[1]))

{

ChatClient client = (ChatClient)server.clients[acceptStr[1]];

//将该用户对应的Client对象从clients中删除

Hashtable syncClients = Hashtable.Synchronized(

server.clients);

syncClients.Remove(client.name);

server.RemoveUser(client.name);

//向客户端发送QUIT命令

string message = "QUIT|" + acceptStr[1];

System.Collections.IEnumerator myEnumerator =

server.clients.Values.GetEnumerator();

while (myEnumerator.MoveNext())

{

ChatClient newClient = (ChatClient)myEnumerator.Current;

SendToClient(newClient, message);

}

server.UpdateMsg("QUIT");

}

}

else if (acceptStr[0] == "LOGIN")

{

//MessageBox.Show("sdfsa");

string connString = "Data Source =BCOMPUTER\\SQLEXPRESS; Initial Catalog=server; Integrated Security = true";

using (SqlConnection conn = new SqlConnection(connString))

{

conn.Open();

string sql\_select = string.Format("select \* from user\_login where name = '{0}' and password = '{1}'", acceptStr[1], acceptStr[2]);

SqlCommand comm = new SqlCommand(sql\_select, conn);

MessageBox.Show(acceptStr[1] + " " + acceptStr[2]);

string count = comm.ExecuteScalar().ToString();

if (count.Length > 0)

{

SendToClient(this, "LOGIN\_OK");

}

else

{

SendToClient(this, "USER\_NOT\_EXIST");

}

}

}

//退出当前线程

break;

}

Thread.Sleep(200);

}