Subject: Re: Socket - send multiple lines
Posted by nlneilson on Thu, 18 Nov 2010 09:15:12 GMT
View Forum Message <> Reply to Message

Thanks for the quick reply, I left after my last post and just back.

It took a bunch of hours but got it working.

"for(;*q;q++)" is a bit new for me. for(;;) until break I understand. If I debugged some code it would become apparent.

What I did was manually added several lines into char buf including the \n and that worked.

Ln = in.GetLine(); putting that into chbuf for each char dropped the '\n'

cc = Ln.GetCount();
if(j==cc || ch=='\n') ch = '\n';
chBuf[kB] = ch;

'\n' is just int 10, it wasn't necessary to tinker with escape codes.

Also at the end of a set of lines:
chBuf[kB] = 0;
Otherwise if a previous set was longer there is extra char in the buf.

In TheIde in debug the chbuf just shows the first part, select all or copy didn't work for me.

Data2<<= chBuf; into an EditField the select all->copy worked.
The chbuf held up to ~2100 char.
Then I could paste that into Notepad++ and what was received by the Java app it interacts with to compare.
The C++ to Java is why a socket is necessary. Trying shared memory was a real pain that made my head hurt.

Also what someone may find useful:
```
//	LeftPosZ(13, 248).TopPosZ(64, 54); // for deploy
	LeftPosZ(13, 248).TopPosZ(66, 100); // for debug
```
That way there is room for 5 extra data fields for debugging.

The highest I have tested this is 50 lines per chbuf and sent once every second for one of 60 sets, or when replaying one second represents one minute. The chbuf is made 6o times per second with 50 lines each.
CPU usage <2% running by itself.

edit: I did notice it may take longer than a second but that is not important now. At most in real time only one chbuf will be made and sent through the socket per second.
For replay of data these are also useful:
pos = in.GetPos();

```
in.SeekCur(fwd);
```

No changes to the socket server or client was necessary.