ChatAPP

Simple TCP Socket client chat program.

Releases:
Executable for windows ChatAPP_RC1.0.zip (2018-03-08)
Source code in ChatAPP_Source_RC1.0.zip (2018-03-08)
Documentation:

DOCS_README.pdf	(this document explanatory regarding program use and coding)
DOCS_Pictures_UI.pdf	(pictures showing how UI looks and act)

Prerequisites:
Windows OS for EXE, cross platform Qt code.
Neuman chatAPP server running on python 3.x

This program is a client program for text messaging. The program is written i c++ Qt framework and uses QTCP socket library for communications with the server.
The client provides the following features:

- Connect to server with status indicators and error handling
- Sending to, and receiving messages from server while typing
- Simple “enter click” to send.
- UTF-8 char support.
- Resizable windows.
- Scrollable message window automatically scrolls to last message.
- Slick and minimalistic UI with anti tired eyes color scheme.
- Nickname support through command line arguments or text field input
- Disconnect from server and connect to other.
- Quit handled with real disconnect, asks “are you sure” before quitting.
- Funny icon.

How i built this program?

I choose to do this client GUI in the Qt because of prior knowledge to the framework which i find to have really useful libraries which speeds up the coding from the otherwise slow c++ class building coding. I found the library QTCPsocket extremely easy to use for the cause. QTCP socket library includes an socket class which my main class ChatMeUp holds. This object is newed in the startup of the application and deleted with the destructor of the main program. All-through the program this socket (called mySocketConnection) handles the connection to the server. To let the program know when data is being sent from server (the socket states) these are connected to different slots via signals emitted from the QTCP socket class.

How does it work?

Connection mode:

User starts application and are greeted by an UI that contains:

- a drop down menu with the options to disconnect or quit the application. The "disconnect" option is not clickable until the program is connected.
- The Logo-like connection sign is an indicator of connection state. Gray: inactive, Red: connection lost, Green: connected.
- Nickname input for screen name in chat
- Ip address field where user can tell address of host
- Port field where user can tell port the service is running on (default is 1337)
- Big fat connect button.

The user puts the adress (IP) to the hosting server and the portnumber the service is running on. When user presses "connect" an connection attempt is being made to this address through TCP socket request. This is a try method and the program waits a maximum of 2 seconds before throwing the user an error box telling the user that the remote address is not reachable. The indicator turn to red.
If attempt to connect is successful the application switches to "message mode".

Messaging mode:

User is presented with two input text boxes, one with the server welcome message in and another for writing your message. Next to the message writing a large button with the label "send" is located but user can also press "enter" key to send msg to server. Application takes for granted that the user is familiar with the modern way chat programs work, therefore application gives no guidance here. From the top drop down menu user can disconnect from server or quit the application. If user chooses to disconnect user is taken to the connection mode again showing a red cat5 cable as indicator of being offline.

Error handling:

The following cases are handled:
client loses connection with server (throws error box, takes user to connection mode)
server shuts down (throws error box, takes user to connection mode)