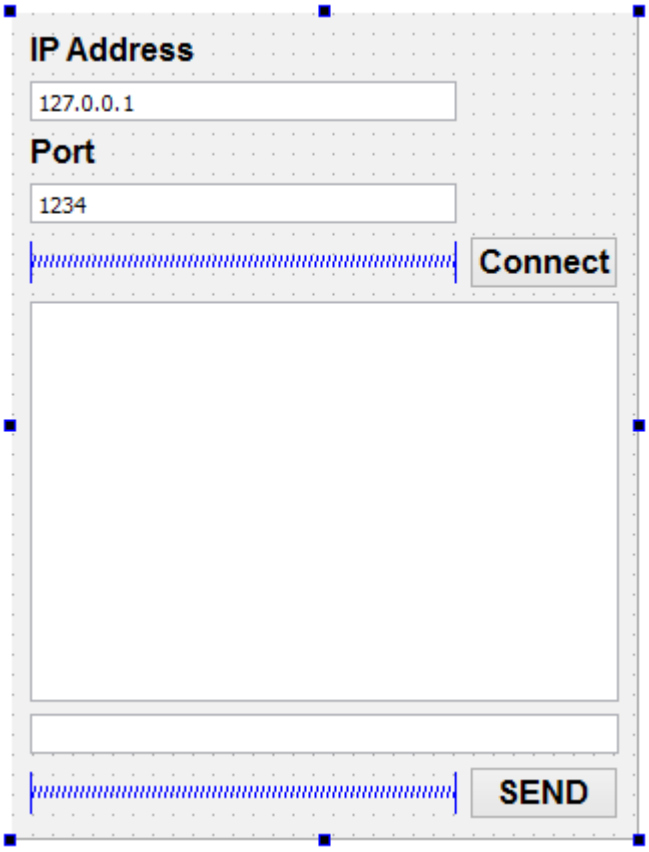
Malwala Arachchige Jude Hashane
10516159
SOFT336SL
CHAT APPLICATION
JUDE HASHANE

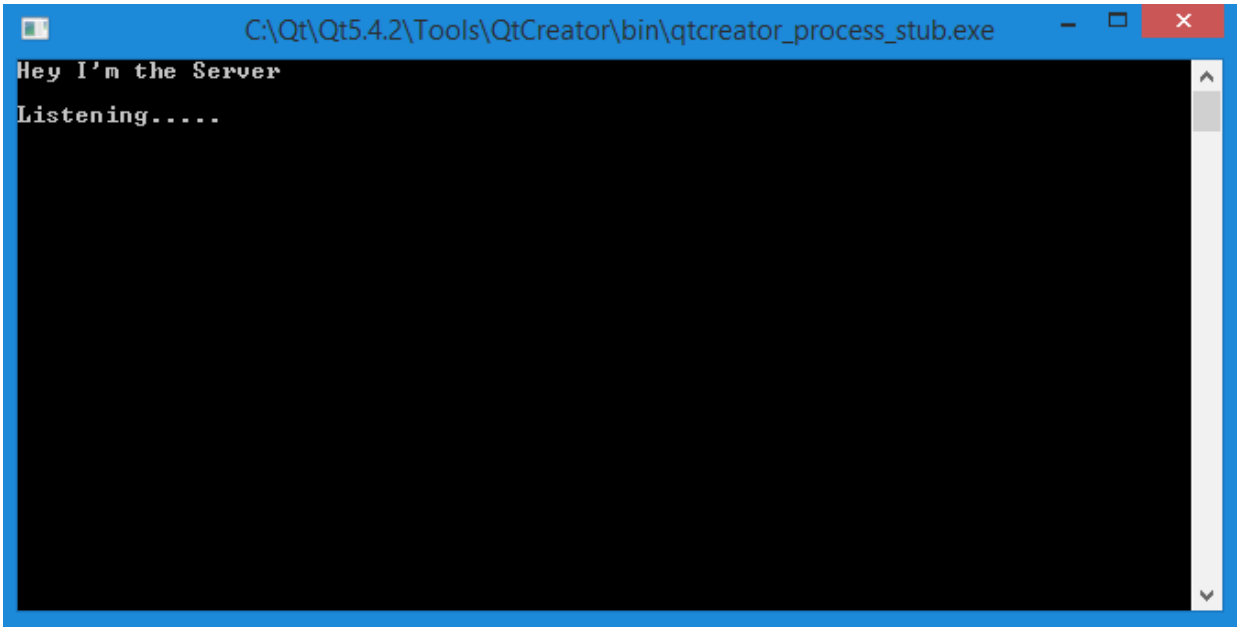
Cross Platform Application Development in C++ [SOFT336SL]

Table of Content

1.Technical Documentation

1.1. GUI Design And Description





QDialog - Is the container of all Qt components/ this provides main application dialog.

QPushButtons – This widget provides a command button which commands computer to perform some action.

QSpacerItem – Horizontal spacer provides a horizontal blank space in the layout.

QLabel – This widget provides a text display in this software to make it user-friendly.

QLineEdit – This widget is a one-line text editor to hold small,single values.

QTextEdit – This widget is used to edit and display both plain and rich text.

Grid Layout - The Grid Layout is the container that holds and arranges the placement of the components inside it. This is very useful when cross platform usability, because this component done the arrangements of its child components according to running environment.

1.2. Code Documentation

The whole program documentation is also generated by using doxygen. See attached doxygen.zip file (extract it and find the "Doxygen/html/index.html") documentation for further information of the software. The index.html page contains all the relevant documentation which is generated by doxygen.

2.1 Introduction

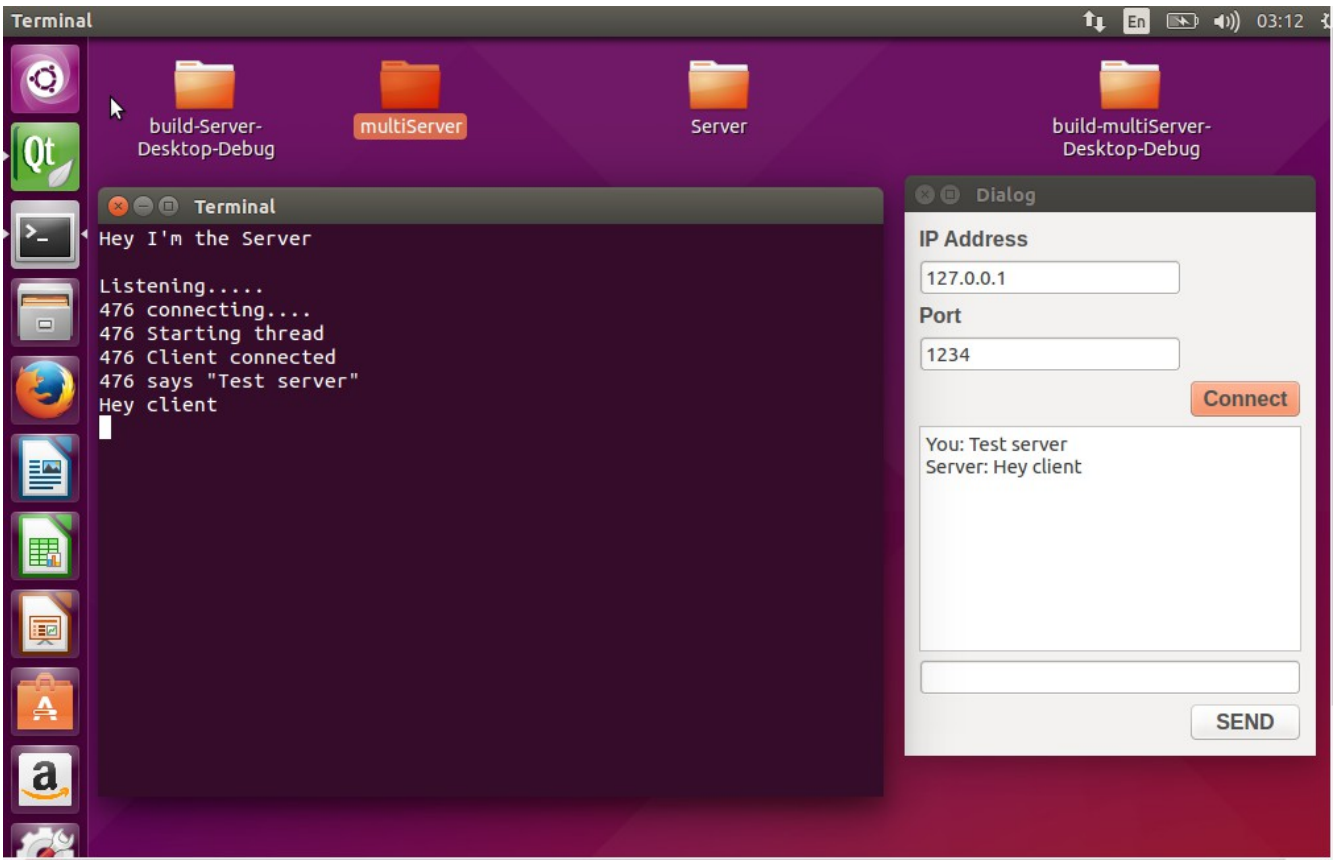
This is a multithreaded Client-Server chat application. So this is in two projects one for the Client which is a GUI and the the most critical part ,the Server which works in console application.

To be honest I must that I didn't know anything regarding Qtcp connections and things early before I start doing this project. I had to go through many articles to get the idea behind those concepts and all. The initial idea was to create a simple chat room using QT but due to some technical errors occurred using Qlist to loop through all the socket descriptors and send message from server to all clients connected I had to change it to a client server program which is capable of connecting as many as clients we want and chat with the main server directly. As the first step I created the server which was the hardest to me as I've used threads in a console format. I tried to make it a gui but with the time I was given and to make the server in GUI format using same threads in there was hard and couldn't complete it. Below you can see a screenshot of the incompleted gui of the Server. But I must say that the console which I've used to the main server in this project and attached works perfectly with the client's gui.

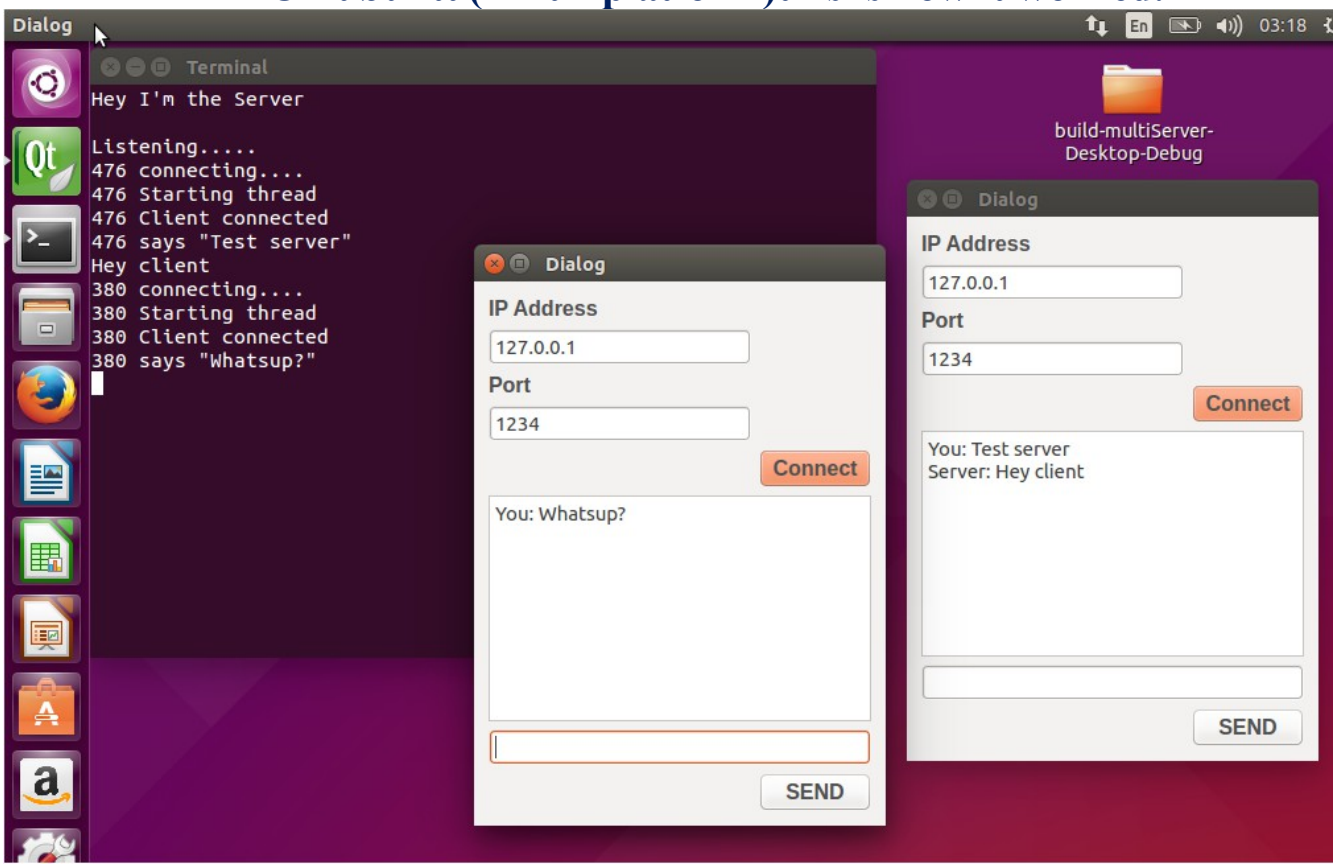
2.2 **Aim-** The aim of this project is to make a advanced client-server chat application using multithreading.

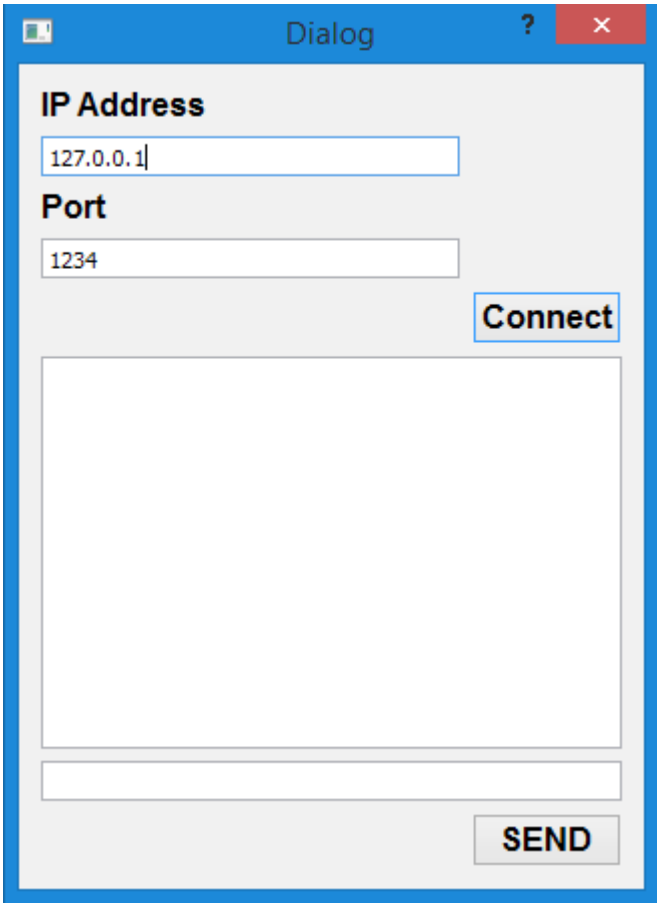
2.3 Scope – This application helps many clients connect to the server and communicate with the server simultaneously. First the Server which is the console program starts listening on a specific port which I've set as 1234 by default. Any client application will be able connect to server through that specific port in real time. The connection will be aborted and rejected if they try to connect through another Tcp port. Server works as the QtcpServer and both write messages to the network stream in other words to the Qtcpsocket. As the client gets connected server notifies it and triggers the IncomingConnection Signal to start the thread and start messaging. When the client get the reply from server they're able to type a message again and press Enter or to press Send button on GUI. When a client gets disconnected it's shows on the console that the specific client from the specific socketDescriptor got disconnected.

2.4 Tested platforms – This “Chat application” currently tested on three platforms which are Windows,Machintosh and Linux(Ubuntu). This cross-platform client-server chat application compiled and ran smoothly on each platforms everytime. But in some cases though it compiles and runs well it didn't get connected due to some technical reasons on Machintosh and Linux which were because some ports were occupied at the time by some other default applications running at the time. So I had to change the port listening and check. In order to troubleshoot my application on both Linux and Machintosh I downloaded another built in server application and tried to connect even using Telnet protocol. By enabling telnet protocol on Linux I tried to connect but it wasn't shown connected sometimes even on other server application which I used for troubleshooting purposes. In my case also it's the same ,the client gets connected but it wasn't shown sometimes. Anyway Everytime it compiled and ran well without any small bug.On windows this application's all functions also worked perfect always without any bugs. All the screenshots of all the tested platforms are shown below.

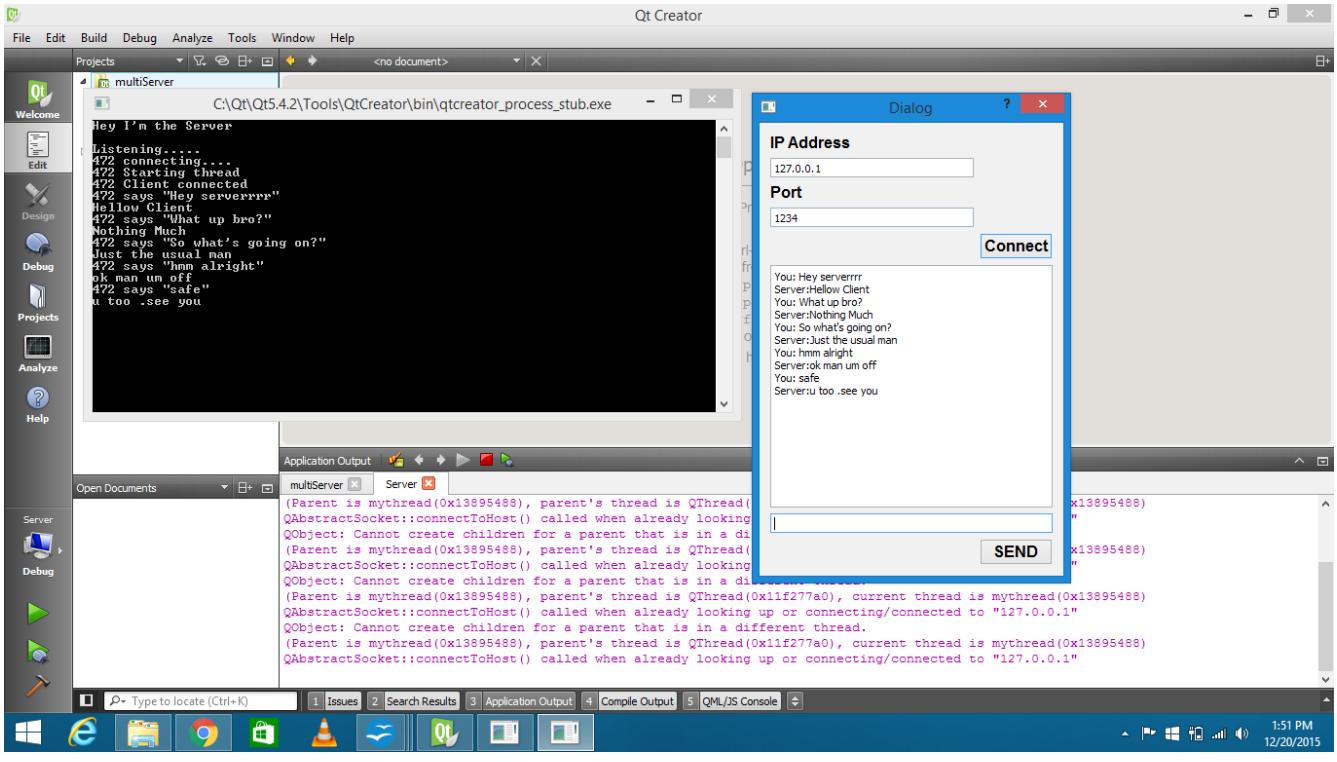


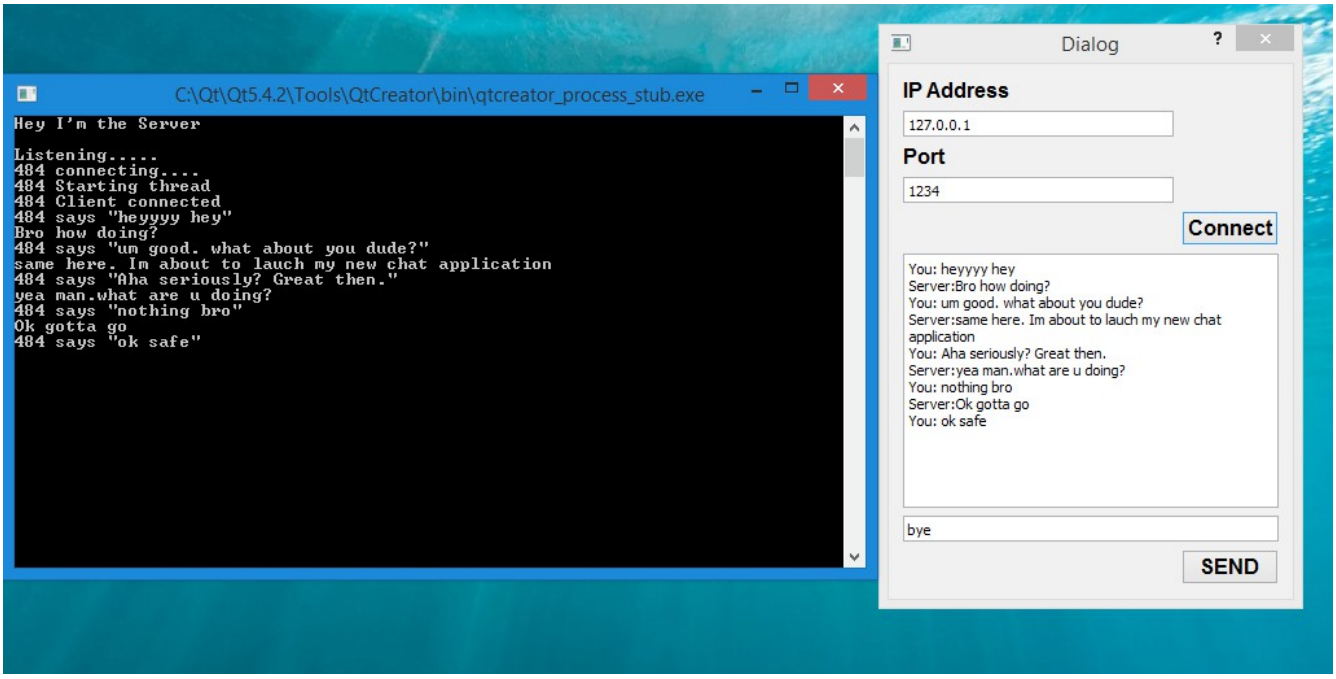
On ubuntu(Linux platform)this is how it worked.



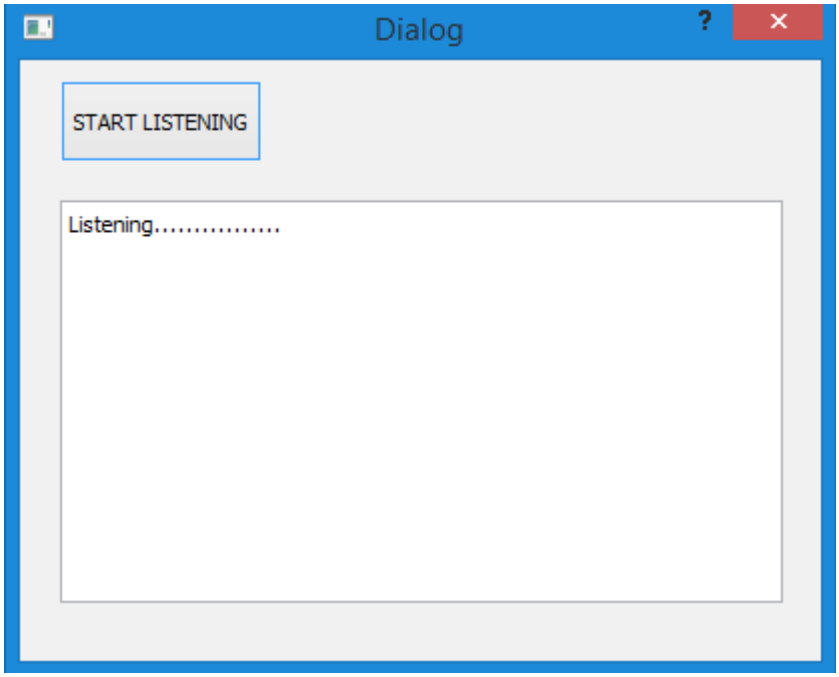


Windows platform compiled and run without any bugs.





Chat application on windows platform



This is the Server gui which is still incomplete and mentioned above.

2.5 Known Bugs and Issues – There are some small bugs as my application stands. They are,

1. First when the client gets connected Client must first send a test message to the server then only the server is able to reply.

2. Once only one message can be transferred by the each party to the other because I couldnt complete threading part on client side gui. But server side threading works fine and many clients as much could connect at once without any bug.

3.On linux platform as I mentioned above doens't get connected sometimes if the port is occupied by some other programs or sometimes it doenst show up on server's console saying Client from a specific socketdescriptor got connected due to some platform issues which I couoldn't realize.

On machintosh didn't start listening only once other than that it ran and compiled perfectly in all the other times.

1.3.Installation

This whole chat application system is in two projects.

So it's required to extracting both the zip files before installation.

The project named “MULTISERVER” is the server project which is a console program.

And the project named “SERVER” is the client's application which is a GUI.

In order to run both applications/projects should be running at the same.

First of all make sure the device which is going to run this application is already installed the Qt and compiling without any errors. The application is not required to install external dependencies. It relies on Qt core APIs, Qt GUI API and other related API's. Therefore the application can be run simply by extracting the (.zip) file and loading the project using (.pro) project file to the Qt creator. Then build and run the application.

First Start the SERVER then the CLIENT and press connect button. Then message on console pop up saying client got connected.

Then type a message on client application and hit enter or press send button. The message you sent also pop up on the console.

Then Type a message on the console and hit ENTER then it'll send the reply back to the client applciation.'

Then you'll be able to chat so on in that order which each side.

Note: The version used to develop the application is "Qt creator 3.4.1 (open source) based on Qt 5.4.2, and the compiler is MinGW 4.9.1". Similar version or higher version can be recommended as minimum requirements to run the application.

1.4. Desired Improvements

In addition to fixing about mentioned bugs, some advanced text editor functions will be added to later versions of this application. Such as,

- Making the Server GUI
- Make the Client application with multiple threads.
- Colour schemes
- Add themes and stickers.
- Simultaneous chatting.
- More user-friendliness.