Report on the Implementation of Chat Rooms in C#

**December 2017**

**EFREI M1 IFM**

**ZHENG Yujie, LIAO Xingwei**



CONTENT

1 Introduction 2

2 Implementation 2

3 Tests 2

3.1 Server 2

3.2 Login 2

3.3 Register 4

3.4 Topic Selection 4

3.5 Chatting 5

4 Conclusion 8

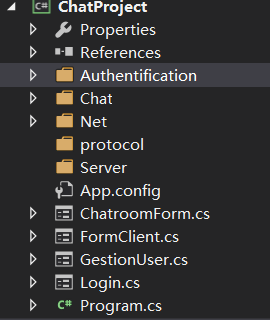
# Introduction

In this project, we want to create several chat rooms for the users. A user can login with his or her username and password, and if he or she does not have an account, he or she can register for a new account. Once logged in , the user can create a new topic or just directly choose an existed chat room. The users can communicate in the chat room and send messages. We successfully implemented all of the functions, but there are still something to be improved in the program.

# Implémentation

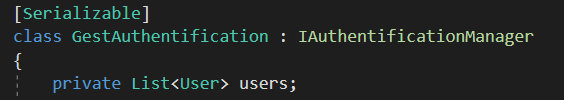
2.1 structure

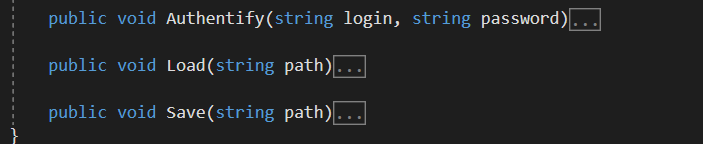
We have 2 grand parts the first part have 5 packages and 4 forms.



2.2 Storage of username and password

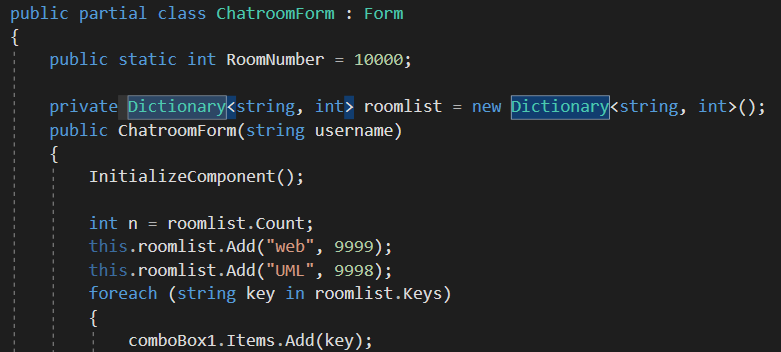
We use serialization to save the username in a .txt document and we load this .txt document to get the username and his password.





2.3 How to realize the different topic and different chatroom?

We give each chatroom a different port. So, we can connect to the different chatroom at the same time. And we use a Dictionary to save the name of chatroom and its port number. Dictionary is faster than a list and easy to find its port number，



2.4 How to show the text from the different treads

We have two ways to avoid the cross-thread exceptions

First way is set “TextBox.CheckForIllegalCrossThreadCalls = false;” And second way more safety is use delegate and InvokeRequired

# Tests

## 3.1 Server

First, start the server.

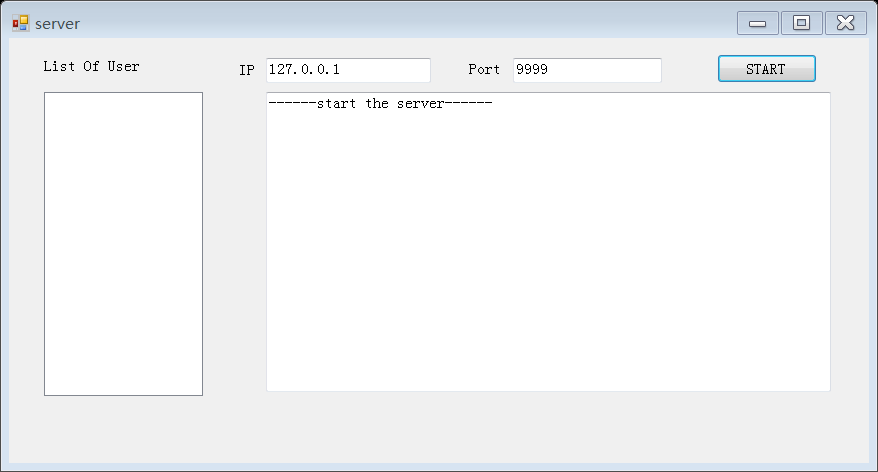


Figure 3-1

## 3.2 LOGIN

Then, the user needs to login with a username and a password. We have two users already stored in our server “bob” and “alice”.

Below we will test some login errors first.

If we input the username that does not exist, we will get a warning:

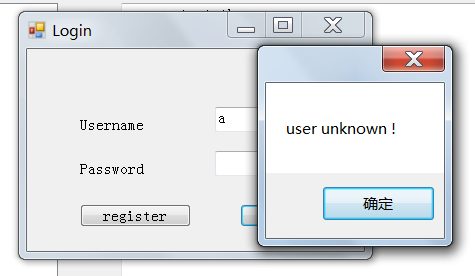


Figure 3-2

If we do not enter any usernames, we will get a warning too:

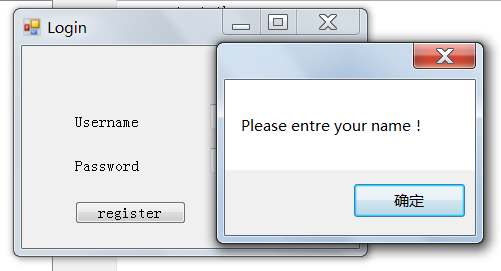


Figure 3-3

## 3.3 Register

Here we test the register function. If you do not have an account, you can click on the “register” button and create an account. For example, you can create an account “Jack” with password “1234”.

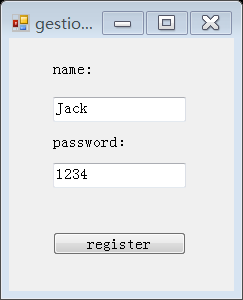


Figure 3-3

## 3.4 Topic Selection

Then, we will enter the second form. On the left side, the user can create a new topic, once the user clicks on the “new topic” button, the new topic will be added to the drop - down menu list on the right side. On the right side, the user can choose a topic to join in in the related chat room. Here we have two default topics “web” and “UML”.

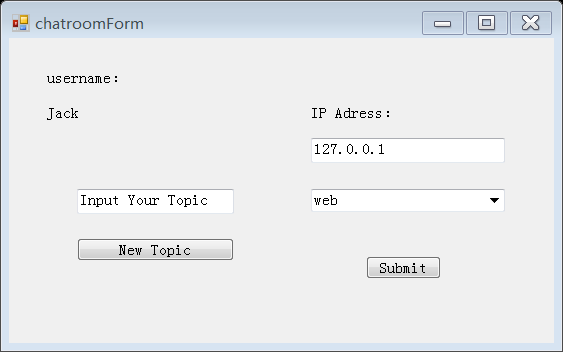


Figure 3-4

We create a “GAME” topic and it will show in the list.

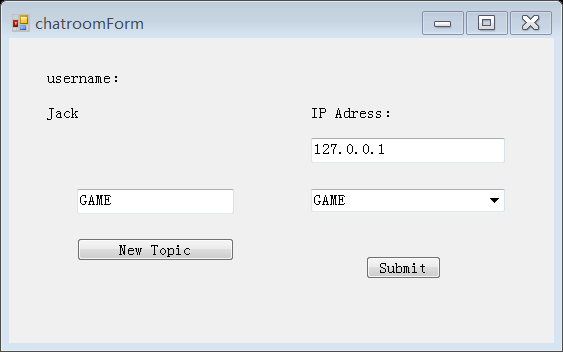


Figure 3-5

## 3.5 Chatting

Next, we can enter the chat room. For example, we can choose to enter the “web” chat room and it will show like below:

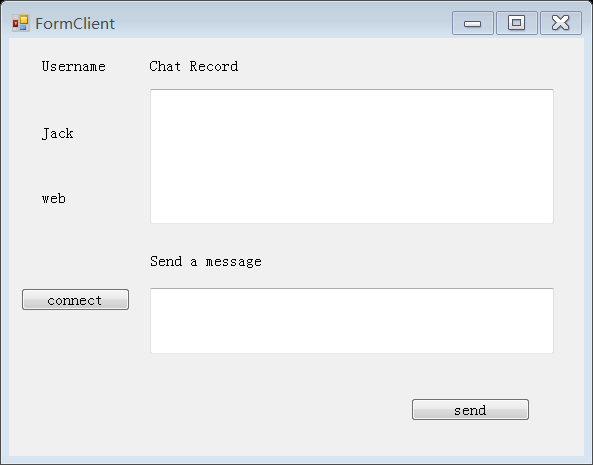


Figure 3-6

And then, we need to click on the “connect” button. Via this button, we can try more than once to connect the server if the server was not online.

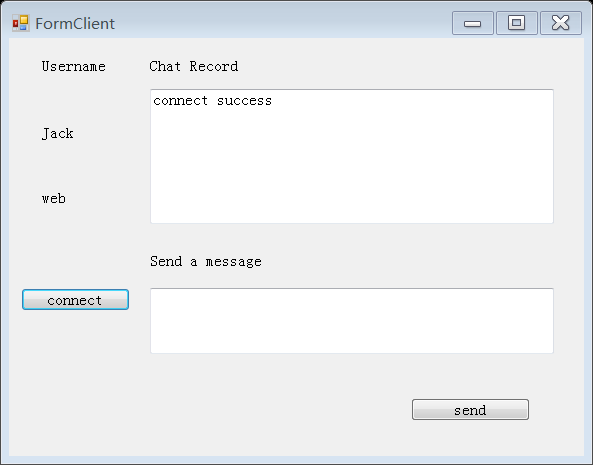


Figure 3-7

Now we can start another client “bob” and enter the “web” chat room.

Now we can send messages freely and every client in the same chat room will see it.

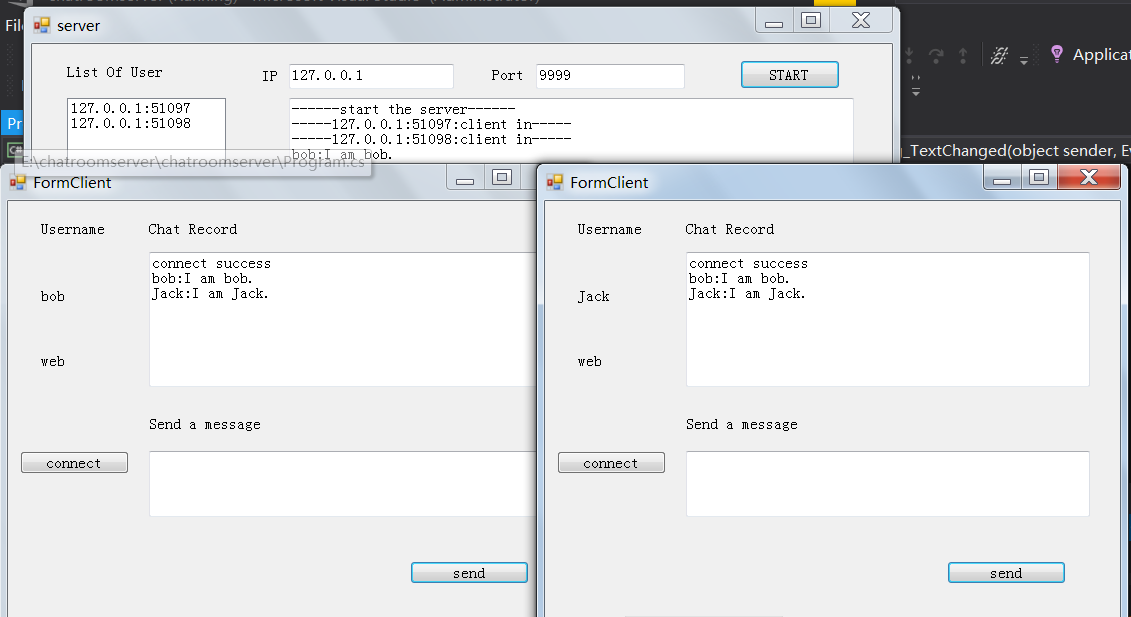


Figure 3-8

We can also enter a different chat room at the same time, for example, bob and Jack can enter the “UML” chat room and the “web” chat room at the same time.

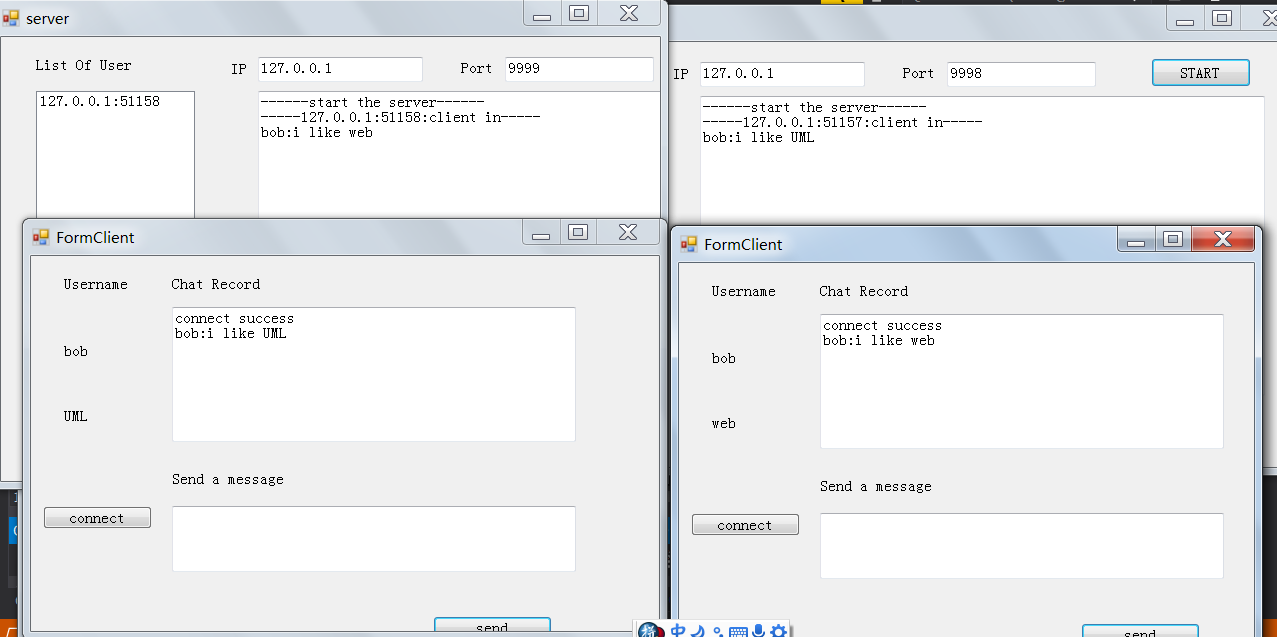


Figure 3-9

# Conclusion

Through this project, we have used a lot of technologies such as inherit, thread, resource control, event, form, TCP server and client and so on. Still, there are a lot of technologies we still do not know and it will be very interesting for us to continue exploring in the ocean of knowledge in C#.