

**Solution approach:**

Both client and server need to establish connection by socket. In server side, server keep listening the targeted port and use ServerSocket to catch incoming socket from client. Then, use thread to deal with this client (socket) and exchange data with the client. For example, if 10 clients connect to server successfully, then server would generate 10 threads accordingly. Server is able to detect the number of clients that is trying to connect to the server by <Server is authenticating to client #>.

After client create socket and set up port number etc. Console of client requires user to input username and access code information and send them to the server. Server has the correct access code so it can authenticate client. Pay attention here that the access code cannot be at the client side. The reason is that attacker can: 1. Get correct access code by checking codes in local. 2. Interfere server behavior. So, after server compare correct access code (in server side) with access code from client, server will send two messages to indicate connection status: "incorrect access code" and "you are connected". Server also update Boolean variable isConnected to record the success connection. If isConnected is TRUE, then within that thread, server will process message from client and I will talk about that in detail later on. In case of incorrect access code, then server will wait again for the client to try and send access code.

Note that in server side we could set limitation (number of attempts) to avoid from brute-force attack. I could implement counter or timer to limit the guesses. But for simplicity, client can try unlimited times by replying <enter access code> promoted by client side. In my implementation, if client knows that it is verified, then it is able to send message to the server following by <write your message >>. However, client is still able to send message to the server if attacker modify the code and change variable isConnected to TRUE, but server will ignore the message.

From the client side, in a nutshell, there are totally 3 main components: interaction with user through console, create a thread for listening message from server, send message to the server. Here I used two threads so that client can write message while listening any update from server (or messages from other client).

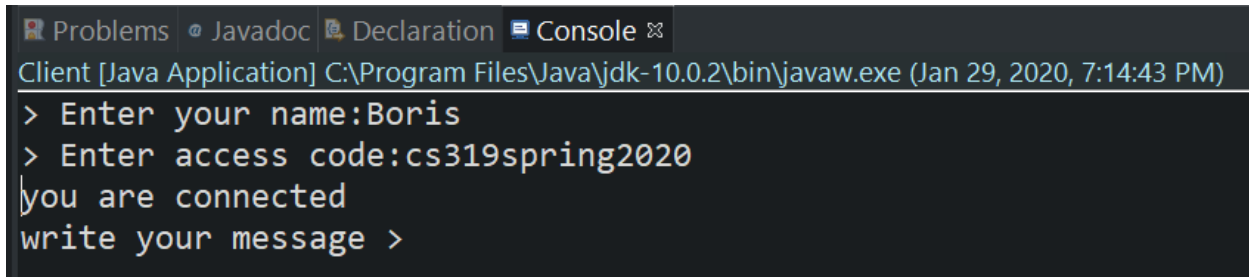
Right now, we set up a channel between client and server by thread. Next challenge is client can talk to each other. In other words, sender send message to server and server broadcast that message to anyone within connection. To accomplish this, I used ArrayList<clientHandler> to keep track of all connected clients so that I can access each clientHandler's PrintWriter. So, if condition matched, then loop through all printwriter and send message to all clients. The condition is: 1. Client who passed the authentication 2. Not the sender itself.

In my implementation, there could be some format issues where client 1 is prompted to write message and then received a message, then client 1 can start his message from new line rather than same line after <write your message> closely.

## Screenshot:

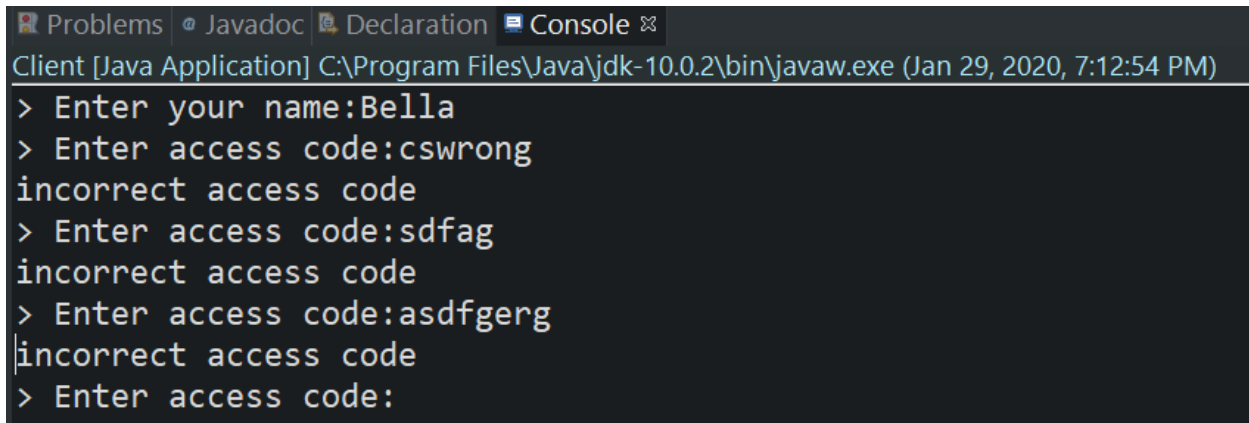
### 1.1

get notified that "you are connected"



```
Problems | Javadoc | Declaration | Console
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 7:14:43 PM)
> Enter your name:Boris
> Enter access code:cs319spring2020
you are connected
write your message >
```

get notified that "incorrect access code".



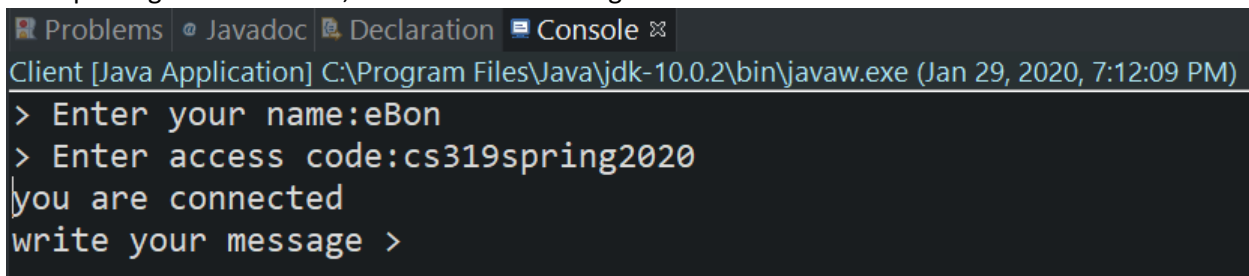
```
Problems | Javadoc | Declaration | Console
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 7:12:54 PM)
> Enter your name:Bella
> Enter access code:cswrong
incorrect access code
> Enter access code:sdfag
incorrect access code
> Enter access code:asdfgerg
incorrect access code
> Enter access code:
```

### 1.2

Client side:

Ask username and access code for user authentication.

After passing authentication, user can write message.



```
Problems | Javadoc | Declaration | Console
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 7:12:09 PM)
> Enter your name:eBon
> Enter access code:cs319spring2020
you are connected
write your message >
```

Server side: when get a new connection, display that the number of clients is authenticating

After getting correct access code, then prompt <username> connection status.

```
Problems Javadoc Declaration Console
Server [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:44:51 PM)
Server is authenticating to client1
eBon is connected successfully!
```

Client 2 side: if user enter wrong access code, then prompt again to allow user to try again

```
Problems Javadoc Declaration Console
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 7:12:54 PM)
> Enter your name:Bella
> Enter access code:cswrong
incorrect access code
> Enter access code:sdfag
incorrect access code
> Enter access code:asdfgerg
incorrect access code
> Enter access code:
```

```
Problems Javadoc Declaration Console
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:51:36 PM)
> Enter your name:Bella
> Enter access code:cs309spring2020
> Enter access code:cs329spring2020
> Enter access code:cs319spring2019
> Enter access code:
```

After connecting successfully Server side:

```
Problems Javadoc Declaration Console
Server [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:13 PM)
Server is authenticating to client1
eBon is connected successfully!
Server is authenticating to client2
Bella is connected successfully!
```

If another client wants to join, then server will show:

```
Problems Javadoc Declaration Console x
Server [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:13 PM)
Server is authenticating to client1
eBon is connected successfully!
Server is authenticating to client2
Bella is connected successfully!
Server is authenticating to client3
```

After three clients talk to each other, server will show:

```
Problems Javadoc Declaration Console x
Server [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:13 PM)
Server is authenticating to client1
eBon is connected successfully!
Server is authenticating to client2
Bella is connected successfully!
Server is authenticating to client3|
Eric is connected successfully!
Eric:Hi, this is Eric. Nice to meet you!
eBon:I am eBon, I am a good student.
Bella:I am Bella, eBon is naughty
```

And this is what showed in eBon, Bella, Eric's console respectively:

```
Problems Javadoc Declaration Console x
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:15 PM)
> Enter your name:eBon
> Enter access code:cs319spring2020
write your message >
Eric:Hi, this is Eric. Nice to meet you!
I am eBon, I am a good student.
write your message >|
Bella:I am Bella, eBon is naughty
```

```
Problems Javadoc Declaration Console x
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:35 PM)
> Enter your name:Bella
> Enter access code:cs319spring2020
write your message >
Eric:Hi, this is Eric. Nice to meet you!

eBon:I am eBon, I am a good student.
I am Bella, eBon is naughty
write your message >
```

```
Problems Javadoc Declaration Console x
Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:56:15 PM)
> Enter your name:Eric
> Enter access code:cs319spring2020
write your message >Hi, this is Eric. Nice to meet you!
write your message >
eBon:I am eBon, I am a good student.

Bella:I am Bella, eBon is naughty
```

This is an example of client terminate console:

Failed client:

```
Problems Javadoc Declaration Console x
<terminated> Client [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 7:07:01 PM)
> Enter your name:haha
> Enter access code:wrong
> Enter access code:
```

Server side:

```
Problems | Javadoc | Declaration | Console x
Server [Java Application] C:\Program Files\Java\jdk-10.0.2\bin\javaw.exe (Jan 29, 2020, 6:54:13 PM)
Server is authenticating to client1
eBon is connected successfully!
Server is authenticating to client2
Bella is connected successfully!
Server is authenticating to client3
Eric is connected successfully!
Eric:Hi, this is Eric. Nice to meet you!
eBon:I am eBon, I am a good student.
Bella:I am Bella, eBon is naughty
Server is authenticating to client4
haha connection failed!
```

## In conclusion:

- Client is able to send username and chat message to the server.
- The server is able to broadcast client message to every other client.
- In server console, it is able to record authentication status, who send what message (username + message)
- In client console, it is able to display other clients message with their name.