

Submission Worksheet

CLICK TO GRADE

<https://learn.ethereallab.app/assignment/IT114-002-S2024/it114-project-milestone-1/grade/am3485>

IT114-002-S2024 - [IT114] Project Milestone 1

Submissions:

Submission Selection

1 Submission [active] 4/17/2024 9:25:18 AM

Instructions

^ COLLAPSE ^

1. Create a new branch called Milestone1
2. At the root of your repository create a folder called Project if one doesn't exist yet
 1. You will be updating this folder with new code as you do milestones
 2. You won't be creating separate folders for milestones; milestones are just branches
3. Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
4. Copy in the latest Socket sample code from the most recent Socket Part example of the lessons
 1. Recommended Part 5 (clients should be having names at this point and not ids)
 2. <https://github.com/MattToegel/IT114/tree/Module5/Module5>
5. Fix the package references at the top of each file (these are the only edits you should do at this point)
6. Git add/commit the baseline and push it to github
7. Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
8. Ensure the sample is working and fill in the below deliverables
 1. Note: The client commands likely are different in part 5 with the /name and /connect options instead of just "connect"
9. Generate the worksheet output file once done and add it to your local repository
10. Git add/commit/push all changes
11. Complete the pull request merge from step 7
12. Locally checkout main
13. git pull origin main

Branch name: Milestone1

Tasks: 9 Points: 10.00



Start Up (3 pts.)

^ COLLAPSE ^

Task #1 - Points: 1

Text: Server and Client Initialization

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Server should properly be listening to its port from the command line (note the related message)	
<input type="checkbox"/> #2	1	Clients should be successfully waiting for input	
<input type="checkbox"/> #3	1	Clients should have a name and successfully connected to the server (note related messages)	

Task Screenshots:

Gallery Style: Large View

SmallMediumLarge

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

EXTENS

bob: hi
Debug Info: Type[MESSAGE], Number[0], Message[join room 3]
bob: join room 3
Debug Info: Type[DISCONNECT], Number[0], Message[disconnect]
bob disconnected
/joinroom room3
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
alan connected
yay
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[yay]
alan: yay

* History restored

merca@pear MINGW64 ~/an3485-it114-002 (module5)
\$ java MS/Server
Starting Server
Server is listening on port 3000
waiting for next client
waiting for next client
Client connected
Thread[29]: Thread created
Thread[29]: Thread starting
Thread-0 leaving room lobby
Thread-0 joining room lobby
Thread[29]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[31]: Thread created
Thread-2 leaving room lobby
Thread[31]: Thread starting
Thread-2 joining room lobby
Thread[31]: Received from client: Type[CONNECT], Number[0], Message[null]
[]

Waiting for input
/connect localhost:3000
java.net.ConnectException: Connection refused: connect
at java.base/sun.nio.ch.Net.connect0(Native Method)
at java.base/sun.nio.ch.Net.connect(Net.java:589)
at java.base/sun.nio.ch.Net.connect(Net.java:578)
at java.base/sun.nio.ch.NioSocketImpl.connect(NioSocketImpl.java:583)
at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:327)
at java.base/java.net.Socket.connect(Socket.java:751)
at java.base/java.net.Socket.connect(Socket.java:686)
at java.base/java.net.Socket.<init>(Socket.java:555)
at java.base/java.net.Socket.<init>(Socket.java:324)
at MS.Client.connect(Client.java:47)
at MS.Client.processCommand(Client.java:120)
at MS.Client\$1.run(Client.java:160)
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
Bob connected
Debug Info: Type[DISCONNECT], Number[0], Message[disconnect]
null disconnected
Debug Info: Type[CONNECT], Number[0], Message[connected]
Alan connected
[]

24 mins

Java Ready

You, 18 hours ago

Ln 1, Col 12

Tab Size: 4

UTF-8

CR/LF

() Java

the screen shot

Checklist Items (0)

Task #2 - Points: 1

Text: Explain the connection process

Details:

Note the various steps from the beginning to when the client is fully connected and able to communicate in the room.

Emphasize the code flow and the sockets usage.

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention how the server-side of the connection works
<input type="checkbox"/> #2	1	Mention how the client-side of the connection works
<input type="checkbox"/> #3	1	Describe the socket steps until the server is waiting for messages from the client

Response:

In order to connect the client and the server you must first make sure each part is running then from the server type `connect localhost:3000`, `localhost` is how it sounds and means you are hosting it locally and the `3000` part is a port number that is usually available. next name the two clients and then connect both to the same port as the server.



Communication (3 pts.)

^COLLAPSE ^



Task #1 - Points: 1

Text: Add screenshot(s) showing evidence related to the checklist

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	At least two clients connected to the server
<input type="checkbox"/> #2	1	Client can send messages to the server
<input type="checkbox"/> #3	1	Server sends the message to all clients in the same room
<input type="checkbox"/> #4	1	Messages clearly show who the message is from (i.e., client name is clearly with the message)
<input type="checkbox"/> #5	2	Demonstrate clients in two different rooms can't send/receive messages to each other (clearly show the clients are in different rooms via the commands demonstrated in the lessons)
<input type="checkbox"/> #6	1	Clearly caption each image regarding what is being shown

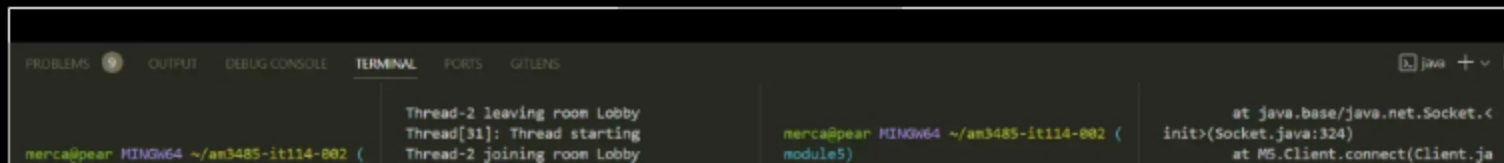
Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



```
module5)
$ java MS/Client

Listening for input
Waiting for input
/connect localhost:3000
You must set your name before you can c

Users\merca\am3485-it114-002\MS\Client.class
Waiting for input
/name Alan
Name set to Alan
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Alan connected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Bill connected*
/createroom big'
Waiting for input
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Alan connected*
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Bill connected*
Debug Info: Type[MESSAGE], Number[0], M
essage[hey all whats up]
Bill: hey all whats up
[]

Thread[31]: Received from client: Type[
CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[33]: Thread created
Thread-4 leaving room Lobby
Thread[33]: Thread starting
Thread-4 joining room Lobby
Thread[33]: Received from client: Type[
CONNECT], Number[0], Message[null]
Thread[31]: Received from client: Type[
MESSAGE], Number[0], Message[/createroo
n big']
Room[Lobby]: Sending message to 3 clien
ts
Created new room: big'
Thread-2 leaving room Lobby
Thread-2 joining room big'
Thread[33]: Received from client: Type[
MESSAGE], Number[0], Message[/joinroom
big']
Room[Lobby]: Sending message to 2 clien
ts
Thread-4 leaving room Lobby
Thread-4 joining room big'
Thread[33]: Received from client: Type[
MESSAGE], Number[0], Message[hey all wh
ats up]
Room[big']: Sending message to 2 clien
ts
Thread[29]: Received from client: Type[
MESSAGE], Number[0], Message[where'd ev
eryone go]
Room[Lobby]: Sending message to 1 clien
ts
[]

$ java MS/Client
Listening for input
Waiting for input
/name Bill
Name set to Bill
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Bill connected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*Alan disconnected*
/joinroom big'
Waiting for input
Debug Info: Type[CONNECT], Number[0], M
essage[connected]
*Bill connected*
hey all whats up
Debug Info: Type[MESSAGE], Number[0], M
essage[hey all whats up]
Bill: hey all whats up
[]

va:47)
at MS.Client.processCommand(Cl
ient.java:128)
at MS.Client$1.run(Client.java
:160)
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0],
Message[connected]
*Bob connected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0],
Message[connected]
*Alan connected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0],
Message[connected]
*Bill connected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*Alan disconnected*
Debug Info: Type[DISCONNECT], Number[0]
, Message[disconnected]
*Bill disconnected*
where'd everyone go
Waiting for input
Debug Info: Type[MESSAGE], Number[0],
Message[where'd everyone go]
Bob: where'd everyone go
[]
```

red = message shown blue = proves connection green = shows messages being sent from inside the same room yellow = shows different rooms can't see each others messages

Checklist Items (0)

^COLLAPSE ^

Task #2 - Points: 1

Text: Explain the communication process

Details:

How are messages entered from the client side and how do they propagate to other clients?

Note all the steps involved and use specific terminology from the code.
Don't just translate the code line-by-line to plain English, keep it concise.

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Mention the client-side (sending)	
<input type="checkbox"/> #2	1	Mention the ServerThread's involvement	
<input type="checkbox"/> #3	1	Mention the Room's perspective	
<input type="checkbox"/> #4	1	Mention the client-side (receiving)	

Response:

from the client side all that you have to do is type something out and press enter, on the server thread side its forwards the message so people can see it, on the room side it makes it so that only people in the same room can see a message, and the receiving side they are displayed the message and who its from.

Task #1 - Points: 1

Text: Add screenshot(s) showing evidence related to the checklist

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show a client disconnecting from the server; Server should still be running without issue (it's ok if an exception message shows as it's part of the lesson code, the server just shouldn't terminate)
<input type="checkbox"/> #2	1	Show the server terminating; Clients should be disconnected but still running and able to reconnect when the server is back online (demonstrate this)
<input type="checkbox"/> #3	1	For each scenario, disconnected messages should be shown to the clients (should show a different person disconnected and should show the specific client disconnected)
<input type="checkbox"/> #4	1	Clearly caption each image regarding what is being shown

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

```

m big']
Room[Lobby]: Sending message to 3 clients
Thread-2 leaving room Lobby
Thread-2 joining room big'
Thread[33]: Received from client: Type[MESSAGE], Number[0], Message[/joinroom big']
Room[Lobby]: Sending message to 2 clients
Thread-4 leaving room Lobby
Thread-4 joining room big'
Thread[33]: Received from client: Type[MESSAGE], Number[0], Message[hey all whats up]
Room[big']: Sending message to 2 clients
Thread[29]: Received from client: Type[MESSAGE], Number[0], Message[where'd everyone go]
Room[Lobby]: Sending message to 1 clients
Thread[29]: Received from client: Type[MESSAGE], Number[0], Message[/disconnect]
Room[Lobby]: Sending message to 1 clients
Thread[29]: Passed in room was null, this shouldn't happen
Thread[29]: Thread being disconnected by server
Thread[29]: Thread cleanup() start
Thread[29]: Thread cleanup() complete
Thread[29]: Exited thread loop. Cleaning up connection
Thread[29]: Thread cleanup() start
Thread[29]: Thread cleanup() complete

merca@pear MINGW64 ~/an3485-it114-002 (module5)
$ java M5/Client

Listening for input
Waiting for input
/connect localhost:3000
You must set your name before you can connect via: /name your_name
Waiting for input
/name Alan
Name set to Alan
Waiting for input
/connect localhost:3000
Client connected
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Alan connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
>null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Bill connected*
/createroom big'
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Alan connected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Bill connected*
Debug Info: Type[MESSAGE], Number[0], Message[hey all whats up]
Bill: hey all whats up

null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Bill connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*Alan disconnected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*Bill disconnected*
where'd everyone go
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[where'd everyone go]
Bob: where'd everyone go
/disconnect
Waiting for input
java.io.EOFException
    at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
    at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1713)
    at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
    at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
    at M5.Client$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
  
```

red disconnected from yellow

Checklist Items (0)

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS EXTENS

Bill: yo
/server
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[/server]

Alan: /server
/disconnect
Waiting for input
java.io.EOFException
at java.base/java.io.ObjectInputStream\$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:1713)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
at MS.Client\$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input

Alan: /server
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
Alan disconnected
/disconnect
Waiting for input
java.io.EOFException
at java.base/java.io.ObjectInputStream\$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:1713)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
at MS.Client\$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
/connect localhost:3000
java.net.ConnectException: Connection refused: connect
at java.base/sun.nio.ch.Net.connect(Native Method)
at java.base/sun.nio.ch.Net.connect(Net.java:589)
at java.base/sun.nio.ch.Net.connect(Net.java:578)
at java.base/sun.nio.ch.NioSocketImpl.connect(NioSocketImpl.java:583)
at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:327)
at java.base/java.net.Socket.connect(Socket.java:751)
at java.base/java.net.Socket.connect(Socket.java:686)
at java.base/java.net.Socket.<init>(Socket.java:555)
at java.base/java.net.Socket.<init>(Socket.java:324)
at MS.Client.connect(Client.java:47)
at MS.Client.processCommand(Client.java:120)
at MS.Client\$1.run(Client.java:160)
Waiting for input

Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
Bill disconnected
where'd everyone go
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[where'd everyone go]
Bob: where'd everyone go
/disconnect
Waiting for input
java.io.EOFException
at java.base/java.io.ObjectInputStream\$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:1713)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
at MS.Client\$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input

27 mins Java Ready You, 19 hours ago Ln 145, Col 21 (5 selected) Spaces: 4 UTF-8

when server disconnects

Checklist Items (0)

COLLAPSE

Task #2 - Points: 1
Text: Explain the various Disconnect/termination scenarios

Details:

Include the various scenarios of how a disconnect can occur. There should be around 3 or so.

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention how a client gets disconnected from a Socket perspective
<input type="checkbox"/> #2	1	Mention how/why the client program doesn't crash when the server disconnects/terminates.
<input type="checkbox"/> #3	1	Mention how the server doesn't crash from the client(s) disconnecting

Response:

the scenarios of a disconnect are quitting the client, terminating the server, or pressing control C from either the client or the server

one way to terminate is to x the instance of the terminal out or close it lets say you did this from the server side it wouldn't crash the client because of the check connection function that uses a try catch loop to stop this from happening so you don't have to rerun the program anytime a disconnect might happen the client will be set back to waiting for input like when you start

another way to terminate is by pressing control c in the client or server lets say you do this on the client side the server wouldn't crash because of a try catch loop that would catch when a client disconnects and send the server back to the same point where it started as waiting for client and all the server threads and rooms will be closed if nobody is using them

Misc (1 pt.)

^COLLAPSE ^

Task #1 - Points: 1

Text: Add the pull request link for this branch

URL #1

<https://github.com/alanpear/am3485-it114-002/pull/10>

Task #2 - Points: 1

Text: Talk about any issues or learnings during this assignment

i Details:

Few related sentences about the Project/sockets topics

Response:

I had no issues with this milestone it was more of an assignment that helped me get familiar with the process and helped me recount what i learned in class.

Task #3 - Points: 1

Text: WakaTime Screenshot

i Details:

Grab a snippet showing the approximate time involved that clearly shows your repository.

The duration isn't considered for grading, but there should be some time involved.

Task Screenshots:

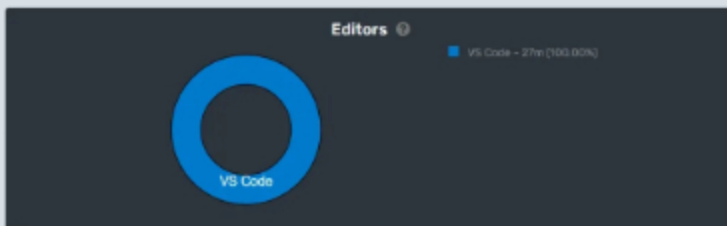
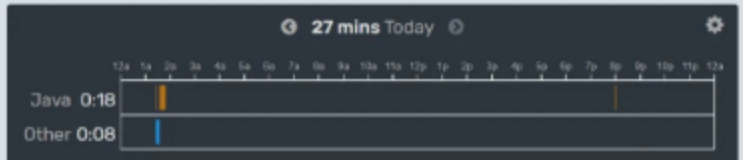
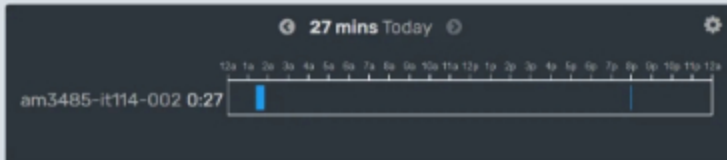
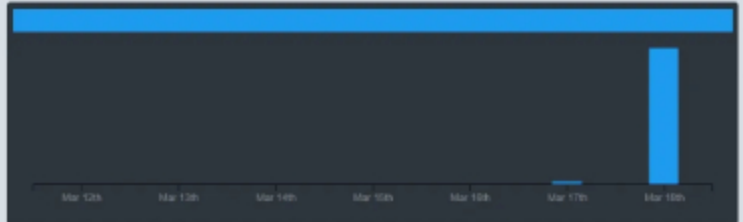
Gallery Style: Large View

Small

Medium

Large

27 mins over the Last 7 Days. 📈



Operation Systems ⓘ

Machines ⓘ

waka time ss

End of Assignment