Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-451-M2024/it114-module-5-project-milestone-1/grade/sa2796

IT114-451-M2024 - [IT114] Module 5 Project Milestone 1

Submissions:

Submission Selection

1 Submission [active] 6/17/2024 1:25:49 PM

Instructions

COLLAPSE

Overview Video: https://youtu.be/A2yDMS9TS1o

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. Copy in the code from Sockets Part 5 into the Project folder (just the files)
    2. https://github.com/MattToegel/IT114/tree/M24-Sockets-Part5
4. Fix the package references at the top of each file (these are the only edits you should do at this point)
5. Git add/commit the baseline and push it to github
6. Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
7. Ensure the sample is working and fill in the below deliverables 1. Note: Don't forget the client commands are /name and /connect
8. Generate the output file once done and add it to your local repository
9. Git add/commit/push all changes
10. Complete the pull request merge from the step in the beginning
11. Locally checkout main
12. git pull origin main

Branch name: Milestone1

Tasks: 8 Points: 10.00

Start Up (3 pts.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Start Up

Details:

Important: Code screenshots should be fairly concise (try to show only the sections of code relevant to the question)

Capturing all possible code (i.e., including a lot of irrelevant code) can lead to a reduced grade. The goal is to show you understand what segments are related to the prompts.

#1) Show the Server starting via

Caption (required) ✓

Describe/highlight what's being shown

Displaying Server starting via command line and listening for connections

#2) Show the Server Code that

Caption (required) ✓

Describe/highlight what's being shown (ucid/date must be present)

Displaying Server Code that listens for connections

Explanation (required) ✓

Briefly explain the code related to starting up and waiting for connections

PREVIEW RESPONSE

The code shows that the "start" is taking on the parameter port in which specifies the port number where the server will listen to. It

#3) Show the Client starting via

Caption (required) ✓

Describe/highlight what's being shown

Displaying the Client starting via command line

#4) Show the Client Code that

Caption (required) ✓

Describe/highlight what's being shown (ucid/date must be present)

Displaying the Client Code that prepares the client and waits for user input

Explanation (required) ✓

Briefly explain the code/logic/flow leading up to and including waiting for user input

PREVIEW RESPONSE

The method listentoinput() handles user input from the keyboard and scanner is then initialized to read

also prints a mesage showing that the server's listening on the specific port.

from System.in which makes it so where the program can get input. The program prints "waiting for input". The while loop makes the method run forever as long as the flag is true.

COLLAPSE

Task #2 - Points: 1

Text: Connecting

Details:
Important: Code screenshots should be fairly concise (try to show only the sections of code relevant to the question)

Capturing all possible code (i.e., including a lot of irrelevant code) can lead to a reduced grade. The goal is to show you understand what segments are related to the prompts.

#1) Show 3 Clients connecting

Caption (required) ✓
Describe/highlight what's being shown
Displaying 3 Clients connecting to the Server

#2) Show the code related to

Caption (required) ✓
Describe/highlight what's being shown (ucid/date must be present)
Displaying the code related to Clients connecting to the Server (including the two needed commands)

Explanation (required)

✓

Briefly explain the code/logic/flow

PREVIEW RESPONSE

The two needed commands were "/connect" and "/disconnect". The whole gist of it is that a user inputs "/connext" which parses the command, which leads to the extraction of the port and host. This ultimately establishes the connection. The disconnect part has the user input "/disconnect" which sends the payload to the sercer and closes all the connections and terminates it.

Communication (3 pts.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Communication

Details:

Important: Code screenshots should be fairly concise (try to show only the sections of code relevant to the question)

Capturing all possible code (i.e., including a lot of irrelevant code) can lead to a reduced grade. The goal is to show you understand what segments are related to the prompts.

#1) Show each Client sending

#2) Show the code related to

#3) Show the code related to

#4) Show the code related to

Caption (required) ✓
Describe/highlight what's being shown
Displaying Client sending and receiving messages

Caption (required) ✓
Describe/highlight what's being shown (ucid/date must be present)
Displaying the code related to the Client-side of getting a user message and sending it over the socket

Explanation (required)
✓
Briefly explain the code/logic/flow involved
PREVIEW RESPONSE
The method reads user input by utlizing "scanner" and it sees if it is a command by "processClientCommand(lir and if it's not a command and client is connected it sends message using "sendMessage(line)".

The next part has the method make a "Payload" and sets it to "MESSAGE" assigning the user's message and calls "send(p)" to simply send it over to the payload.

Finally, the method send (Payload p) sends "Payload" to the socket by "ObjectOutputStream".

Caption (required) ✓
Describe/highlight what's being shown (ucid/date must be present)
Displaying the code related to the Server-side receiving the message and relaying it to each connected Client

Explanation (required)
✓
Briefly explain the code/logic/flow involved
PREVIEW RESPONSE
First on "ServerThread.java", to recieve the message, "processPayload" handles the payloads that are coming and when "MESSAGE" is brought , it then calls " currentRoom.sendMessage payload.getMessage())".

When relaying message, on "Room.java", the sendMessage method iterates to the connected clients and sends message by "client.sendMessage(sende message)". In the event the client does not recieve message, it will disconnect

Caption (required) ✓
Describe/highlight what's being shown (ucid/date must be present)
Displaying the code related to the Client receiving messages from the Server-side and presenting them

Explanation (required)
✓
Briefly explain the code/logic/flow involved
PREVIEW RESPONSE
The listentoserver() method listens for payloads from the server and once a payload is recieved it passes it to " processPayload(Payload payload)". That method then sees the type of recieved payload. If it's PayloadType.MESSAGE it calls processMessage(payload.getClie payload.getMessage()) to present message. This method allows it to format message with the user's name and prints it.

^COLLAPSE ^

Task #2 - Points: 1

Text: Rooms

Details:

Important: Code screenshots should be fairly concise (try to show only the sections of code relevant to the question)

Capturing all possible code (i.e., including a lot of irrelevant code) can lead to a reduced grade. The goal is to show you understand what segments are related to the prompts.

#1) Show Clients can Create

Caption (required) ✓

Describe/highlight what's being shown

Displaying Clients can Create Rooms

#2) Show Clients can Join Rooms

Caption (required) ✓

Describe/highlight what's being shown

Displaying Clients can Join Rooms (leave/join messages should be visible)

#3) Show the Client code

Caption (required) ✓

Describe/highlight what's being shown (ucid/date must be present)

Displaying the Client code related to the create/join room commands

Explanation (required) ✓

Briefly explain the code/logic/flow involved

PREVIEW RESPONSE

The "processClientCommand(St text)" method sees if the input is a command and detects the "createroom" and "joinroom" commands to then call the specific methods to handle. The "sendCreateRoom(String

#4) Show the ServerThread/Ro

Caption (required) ✓

Describe/highlight what's being shown (ucid/date must be present)

Displaying the ServerThread/Room code handling the create/join process

Explanation (required) ✓

Briefly explain the code/logic/flow involved

PREVIEW RESPONSE

In "ServerThread.java" the processPayload calls currentRoom.handleCreateRoom payload.getMessage()) when the payload "ROOM_CREATE" is given. Then in "Room.java", the "handleJoinRoom"

room)" method makes a "Payload" with the type "ROOM_CREATE" and it sends it to server.

The "sendJoinRoom(String room)" method makes a "Payload with the type "ROOM_JOIN" and sends it to server.

method adds client to the room if it is already present.

Similarly, the same process happens when handling the join room.

The server class handles the making and joining of rooms and maintains the names of rooms to the "Room" objects by giving them methods for making and joining rooms.

#5) Show the Server code for

Caption (required) ✓

Describe/highlight what's being shown (ucid/date must be present)

Displaying the Server code for handling the create/join process

Explanation (required) ✓

Briefly explain the code/logic/flow involved

PREVIEW RESPONSE

In "Room.java", the method "handleCreateRoom" utilizes (Server.INSTANCE.createRo " to make a new room and Server.INSTANCE.joinRoom sender) adds client to the newly created room

#6) Show that Client messages

Caption (required) ✓

Describe/highlight what's being shown

Displaying that Client messages are constrained to the Room (clients in different Rooms can't talk to each other)

Explanation (required) ✓

Briefly explain why/how it works this way

PREVIEW RESPONSE

The sendMessage method in the Room class gives out a mesage to the clients in "clientsInRoom" which is a collection that has only clients in same

the newly created room.

As mentioned before, the server class handles the making and joining of rooms. The "createRoom(String name)" method sees if room is present and then creates new Room object in the event it is not present. Then the " handleJoinRoom(ServerThr sender, String room)" method moves client to the said room which update's the current room and adds it to new room.

room.

The sendMessage methos iteratres over this collection which makes it so that the messages are being sent only to clients in the same room. This ensures that clients in different rooms will not be able to communicate with each other.

Disconnecting/Termination (3 pts.)

^COLLAPSE ^

^COLLAPSE ^

Task #1 - Points: 1

Text: Disconnecting

Details:

Important: Code screenshots should be fairly concise (try to show only the sections of code relevant to the question)

Capturing all possible code (i.e., including a lot of irrelevant code) can lead to a reduced grade. The goal is to show you understand what segments are related to the prompts.

#1) Show Clients gracefully

Caption (required) ✓

Describe/highlight what's being shown

#2) Show the code related to

#3) Show the Server terminating

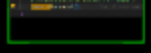
Caption (required) ✓

Describe/highlight what's being shown

#4) Show the Server code

Caption (required) ✓

Displaying Clients gracefully disconnecting (should not crash Server or other Clients)



Caption (required) ✓
Describe/highlight what's being shown (ucid/date must be present)
Displaying the code related to Clients disconnecting

Explanation (required)
✓
Briefly explain the code/logic/flow involved
PREVIEW RESPONSE
The client sends "DISCONNECT" payload to the server and it then closes connection by utilizing "close" and "closeServerConnection" methods.

The method "processClientCommand" handles "/quit" which begins the disconnection.

In "ServerThread.java" the method processPayload handles "Disconnect" payload by calling "currentRoom.disconnect(this)".

Displaying the Server terminating (Clients should be disconnected but still running)

Describe/highlight what's being shown (ucid/date must be present)
Displaying the Server code related to handling termination

Explanation (required)
✓
Briefly explain the code/logic/flow involved
PREVIEW RESPONSE
The constructor "server" sets the shutdown hook by utlizing "Runtime.getRuntime().addShutdo which does the cleanup once the JVM is shut down.

In "Server.java", the shutdown method sets "isRunning" to false to stop bringing in new clients.

It iterates all rooms and calls "disconnectAll" for each room to disconnect all the clients.

Misc (1 pt.)
COLLAPSE

COLLAPSE

Task #1 - Points: 1
Text: Add the pull request link for this branch

URL #1
https://github.com/SHUAIB2796/sa2796-IT114-451/pull/9

COLLAPSE

Task #2 - Points: 1

Text: Talk about any issues or learnings during this assignment

Details:

Few related sentences about the Project/sockets topics

Response:

I initially had some trouble with having the files running but I realized it was because I didn't have the proper package references and I did need some guidance at first but overall I had little to no problems in doing this assignment.

COLLAPSE

Task #3 - Points: 1

Text: WakaTime Screenshot

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

Wakatime Milestone1

Wakatime Milestone 1

End of Assignment