Submission Worksheet

Submission Data

Course: IT114-450-M2025
Assignment: IT114 Module 4 Sockets Part3 Challenge
Student: Matt T. (mt85)
Status: Submitted | **Worksheet Progress:** 4%
Potential Grade: 1.20/10.00 (12.00%)
Received Grade: 0.00/10.00 (0.00%)
Started: 6/16/2025 1:20:24 AM
Updated: 6/16/2025 1:20:24 AM
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-4-sockets-part3-challenge/grading/mt85
View Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-4-sockets-part3-challenge/view/mt85

Instructions

1. Ensure you read all instructions and objectives before starting.
2. Create a new branch from `main` called `M4-Homework`
    1. `git checkout main` (ensure proper starting branch)
    2. `git pull origin main` (ensure history is up to date)
    3. `git checkout -b M4-Homework` (create and switch to branch)
3. Copy the template code from here: GitHub Repository - M4 Homework
    - It includes Sockets `Part1`, `Part2`, and `Part3`. Put all into an `M4` folder or similar if you don't have them yet (adjust `package` reference at the top if you chose a different folder name).
    - Make a copy of `Part3` and call it `Part3HW`
        - Fix the `package` and `import` references at the top of each file in this new folder
    - Immediately record to history
        - `git add .`
        - `git commit -m "adding M4 HW baseline files"`
        - `git push origin M4-Homework`
        - Create a Pull Request from `M4-Homework` to `main` and keep it open
4. Fill out the below worksheet
    - Each Problem requires the following as you work
        - Ensure there's a comment with your UCID, date, and brief summary of how the problem was solved
        - Code solution (add/commit periodically as needed)
        - Hint: Note how `/reverse` is handled
5. Once finished, click "Submit and Export"
6. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github
    1. `git add .`
    2. `git commit -m "adding PDF"`
    3. `git push origin M4-Homework`
    4. On Github merge the pull request from `M4-Homework` to `main`
7. Upload the same PDF to Canvas

8. Sync Local
    1. `git checkout main`
    2. `git pull origin main`

Section #1: (3 pts.) Challenge 1 - Coin Flip

Progress: 33%

Task #1 (3 pts.) - Implement a Coin Flip Command

Progress: 33%

Details:

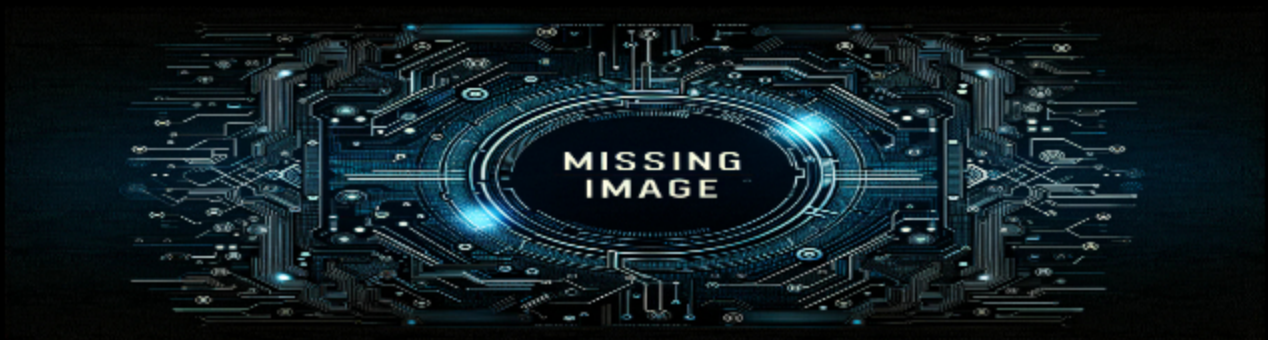
- `Client` must capture the user entry and generate a valid command per the lesson details
    - Command format must be `/flip`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic and send the result to everyone
    - The message must be in the format of `<who> flipped a coin and got <result>` and be from the Server
- Add code to solve the problem (add/commit as needed)

Part 1:

Progress: 0%

Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
    - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
    - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
    - Should only need to create a new method and pass the result message to `relay()`
4. Show 5 examples of the command being seen across all terminals (2+ Clients and 1 Server)
    1. This can be captured in one screenshot if you split the terminals side by side



Missing Caption

Saved: 6/16/2025 1:20:24 AM

Part 2:

Progress: 0%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
Missing URL

URL

Saved: 6/16/2025 1:20:24 AM

Part 3:

Progress: 100%

Details:
Briefly explain `how` the code solves the challenge (note: this isn't the same as `what` the code does)

Your Response:

Good luck

Saved: 6/16/2025 1:20:24 AM

Section #2: (3 pts.) Challenge 2 - Private Message

Progress: 0%

Task #1 (3 pts.) - Implement a Private Message Command

Progress: 0%

Details:

- `Client` must capture the user entry and generate a valid command per the lesson details
  - Command format must be `/pm <target id> <message>`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic
  - The message must be in the format of `PM from <who>: <message>` and be from the Server
  - The result must only be sent to the original sender and to the receiver/target
- Add code to solve the problem (add/commit as needed)

Part 1:

Progress: 0%

Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
    - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
    - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
    - Should only need to create a new method and pass the result message to `relay()`
4. Show 3 examples of the command being seen across all terminals (3+ Clients and 1 Server)
    1. This can be captured in one screenshot if you split the terminals side by side
    2. Note: Only the sender and the receiver should see the private message (show variations across different users)

Missing Caption

Saved: 6/16/2025 1:10:13 AM

Part 2:

Progress: 0%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
Missing URL

URL

Saved: 6/16/2025 1:10:13 AM

Part 3:

Progress: 0%

Details:
Briefly explain `how` the code solves the challenges (note: this isn't the same as `what` the code does)

Your Response:

Missing Response

Saved: 6/16/2025 1:10:13 AM

Section #3: (3 pts.) Challenge 3 - Shuffle Message

Progress: 0%

Task #1 (3 pts.) - Implement a Shuffle Message Command

Progress: 0%

Details:

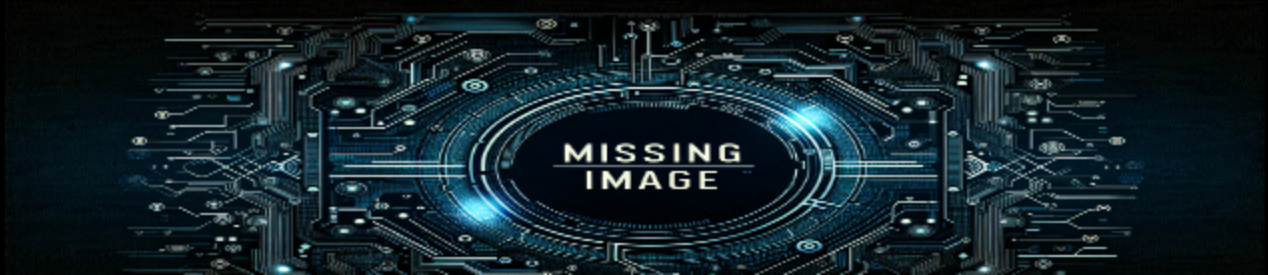
- `Client` must capture the user entry and generate a valid command per the lesson details
  - Command format must be `/shuffle <message>`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic and send the result to everyone
  - The message must be in the format of `Shuffled from <who>: <shuffled_message>` and be from the Server
- Add code to solve the problem (add/commit as needed)

Part 1:

Progress: 0%

Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
   - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
   - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
   - Should only need to create a new method and do similar logic to `relay()`
4. Show 3 examples of the command being seen across all terminals (2+ Clients and 1 Server)
   1. This can be captured in one screenshot if you split the terminals side by side



Missing Caption

Saved: 6/16/2025 1:10:13 AM

Part 2:

Progress: 0%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
Missing URL

URL

Saved: 6/16/2025 1:10:13 AM

Part 3:

Progress: 0%

Details:
Briefly explain `how` the code solves the challenges (note: this isn't the same as `what` the code does)

Your Response:

Missing Response

Saved: 6/16/2025 1:10:13 AM

Section #4: (1 pt.) Misc

Progress: 0%

Task #1 (0.33 pts.) - Github Details

Progress: 0%

Part 1:

Progress: 0%

Details:
From the Commits tab of the Pull Request screenshot the commit history Following minimum should be present

Missing Caption

Saved: 6/16/2025 1:10:13 AM

Part 2:

Progress: 0%

Details:
Include the link to the Pull Request (should end in `/pull/#`)

URL #1
Missing URL

URL

Saved: 6/16/2025 1:10:13 AM

Task #2 (0.33 pts.) - WakaTime - Activity

Progress: 0%

Details:

- Visit the WakaTime.com Dashboard
- Click `Projects` and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary

Missing Caption

Not saved yet

Task #3 (0.33 pts.) - Reflection

Progress: 0%

Task #1 (0.33 pts.) - What did you learn?

Progress: 0%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

Missing Response

Not saved yet

Task #2 (0.33 pts.) - What was the easiest part of the assignment?

Progress: 0%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

Missing Response

Not saved yet

Task #3 (0.33 pts.) - What was the hardest part of the assignment?

Progress: 0%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

Missing Response

Not saved yet