# CS3524 Assessment 2 MUD

Alexandar Dimitrov
Student ID: 51769662
Email: a.dimitrov.17@abdn.ac.uk

## 1. Running Instructions:

- Ensure you have Java version 11.0.5 installed (this is what my code was run on)
- Download and unzip the zip file
- Open the bash and navigate to the directory that contains the make file
- Type “make mud” (to compile the java files). The output should look like this:

```
javac cs3524/solutions/mud/Edge.java; \
javac cs3524/solutions/mud/Vertex.java; \
javac cs3524/solutions/mud/MUD.java; \
javac cs3524/solutions/mud/MUDServer.java; \
javac cs3524/solutions/mud/MUDServerInterface.java; \
javac cs3524/solutions/mud/MUDServerMainline.java; \
javac cs3524/solutions/mud/MUDClient.java; \
javac cs3524/solutions/mud/MUDClientImpl.java; \
javac cs3524/solutions/mud/MUDClientInterface.java;
```

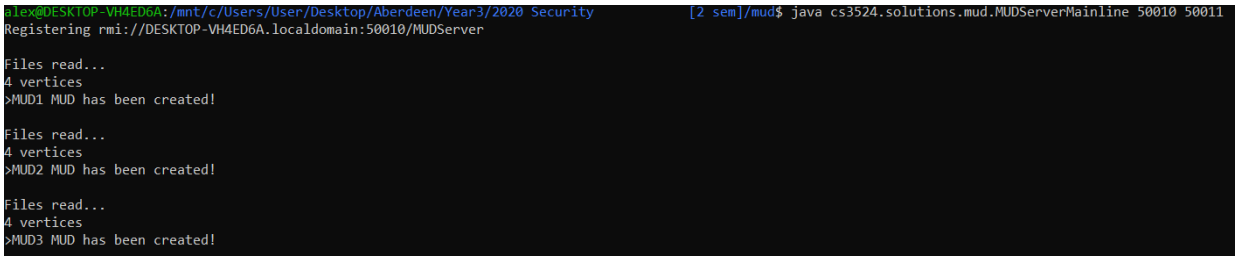
- Open terminals and navigate them to the same directory where the make file is
  - Terminal 1

    Firstly, you must run the rmi registry on an appropriate port (within the range of 50010 – 50019) using the following command:

    rmiregistry <port1>  e.g. rmiregistry 50010
  - Terminal 2

    Once, the rmi registry has been run, open a new terminal window and run the server mainline on the **same** rmi port + one more port (different from the first one) to specify where the objects will be exported to, using the following command:

    java cs3524.solutions.mud.MUDServerMainline <port1> <port2>




  - Terminal 3

    Finally, you must connect to the server as a client. To connect to the server and start playing the game, specify the machine name the server resides on + use the **same** rmi port the rmiregistry is run on + one more call-back port (different from the previous ones), using the following command:

    java cs3524.solutions.mud.MUDClient < hostname> <port1> <port3>
  - Terminal 4,5…

    You may also run multiple clients using the same command as above, just make sure the call-back ports are different from the already used ports if you are running all clients on the same host.

Example:

Terminal 1 - rmiregistry 50010

Terminal 2 (Server) - java cs3524.solutions.mud.MUDServerMainline 50010 50011

Terminal 3 (Client) - java cs3524.solutions.mud. MUDClient <hostname> 50010 50012

T4 (C) - java cs3524.solutions.mud.MUDClient < hostname > 50010 50013

T5 (C) - java cs3524.solutions.mud. MUDClient < hostname > 50010 50014

## 2. Client Instructions:

The game has 2 main loops – one for choosing a MUD game that is presented to the client upon connecting and one game loop for operating throughout the game. Each loop iterates until a specific command is entered. For the choosing of a MUD game, the loop continues until a game is joined – initially the user is presented with a list of 3 MUD games to choose from. Once a game is joined, the client is asked to provide a username before they can begin playing and enter the second (game) loop.

```
Looking up rmi://DESKTOP-VH4ED6A:50010/MUDServer

>Available MUD games:
        -MUD1
        -MUD2
        -MUD3
>Join a MUD
        1) Type 'join <MUD name>' to connect to an existing game
        2) Type 'create <MUD name>' to create and connect to your own
        3) Type 'view' to see the list of available MUDS
>Command: join MUD1
>You successfully joined MUD1!

>Provide a username: Aya
```

List of commands for choosing a MUD game to connect to

After a unique and valid username (no spaces allowed) is provided, the user is presented a list of possible commands – typing any other command would be considered invalid.

```
>Provide a username: Aya

============================== WELCOME TO THE MUD WORLD, Aya! ==============================
>List of commands that you can use:
       1.help (to see the list of available commands)
       2.return (to return information about the location you are currently in)
       3.move <cardinal direction: {north, east, south, west}> (to move around)
       4.pick <item> (to pick an item)
       5.show items (to see the list of picked items)
       6.show users (to see the list of active users in the current MUD world)
       7.show muds (to see the list of available MUD games)
       8.change (to switch to another MUD game)
       9.create <MUD name> (to create a new MUD game and join it)
       10.message (to send messages to other users)
       11.exit (to exit the game)

>Your start location is node: A
>
```

List of commands to operate the MUD game

The game loop iterates until the client enters ‘exit’ which removes him/her from the current game and send him/her back to another loop for choosing a new MUD to be connected to. Both loops can be stopped if the client decides to abort the game (by pressing Ctrl+C, Ctrl+D, closing the terminal window) – in this case all client's logins are being deleted and all client’s items are being dropped.

There are certain limits applied to clients:

- They can play up to 5 five games simultaneously (simply because of the limit of games that can be created)
- Only 2 players at max can play in 1 game instance at a time.

```
private Integer maxPlayers = 2;    // Restrict the number of logged players in a single game
private Integer maxGames = 5;      // Restrict the number of running MUD games that can be
created
```

## 3. What has been done?

| | | | |
|---|---|---|---|
| Edge | 3/4/2020 12:41 PM | JAVA File | 1 KB |
| MUD | 4/3/2020 8:52 PM | JAVA File | 12 KB |
| MUDClient | 4/3/2020 11:12 PM | JAVA File | 17 KB |
| MUDClientImpl | 4/3/2020 8:46 PM | JAVA File | 1 KB |
| MUDClientInterface | 4/2/2020 6:05 PM | JAVA File | 1 KB |
| MUDServer | 4/3/2020 11:33 PM | JAVA File | 9 KB |
| MUDServerInterface | 4/3/2020 8:37 PM | JAVA File | 2 KB |
| MUDServerMainline | 4/3/2020 9:49 AM | JAVA File | 2 KB |
| mymud.edg | 1/31/2010 12:21 AM | EDG File | 1 KB |
| mymud | 1/31/2010 12:21 AM | Outlook Item | 1 KB |
| mymud.thg | 1/31/2010 12:21 AM | THG File | 1 KB |
| Vertex | 3/21/2020 12:20 PM | JAVA File | 2 KB |

The Multiplayer Game Server is following the server-client model, using the JAVA RMI(Remote Method Invocation). The MUD server makes the functionalities available to the client through the MUDServer Interface and the corresponding interface implementation by creating a stub which is registered with the RMI registry.

The MUD file contains most of the functionalities of the game and hence most of the function calls in the MUD service implementation (the MUDServer file) are calls to functions from that file. The functions interact with various data structures and manipulate the information extracted from them to ensure responsiveness and robustness. Because of the better lookup HashMaps are the mainly used data structures for the application needs.

CGS D:

All functional requirements satisfied.

- RMI client-server basis set - the server implements remote methods that are called by the client
- User can move around
- User gets feedback at start location and after moving

CGS C:

All functional requirements satisfied.

- RMI client-server extended to be available for multiple clients which access the remote methods simultaneously
- Users are asked to specify a unique and valid username
- Users can move in any direction (if there is a path)
- Users can see other users in the MUD world (list of online/active users)
- Users can pick up things in the MUD
- Users can see their picked items
- Users can see more information about the location they are currently in
- Users can see a list of available commands

CGS B:

All functional requirements satisfied.

- RMI client-server extended to operate with multiple MUD instances (multiple games)
- Users can choose between 3 pre-defined running MUD game instances
- Users can see the list of running/active MUD games on the server
- Users can leave a MUD game at any time

CGS A5:

All functional requirements satisfied.

- RMI client-server restricted to have a limited number of logged clients (2) in each MUD game at a time and a limited number of MUDs (5) running on the server simultaneously
- Users can create their own MUD games unless the limit is not reached
- Users can take part of/ be logged in multiple games
- Users can switch the game focus between the games they are taking part of without losing their previous locations and items

CGS A4-A1:

All functional requirements satisfied.

- RMI client-server extended to be robust and responsive; call-backs introduced
- Application tested and edge-cases handled

- User’s input restricted (no unexpected commands allowed, no blank usernames, moves, etc.)
- Users receive live updates for certain activities within their scope (upon changes, such as other users joining or leaving a mud game and moving, relevant users get notified)
- Users aborting the game (by pressing Ctrl+C, Ctrl+D, closing the terminal window) handled - logins cleared, items dropped, HashMaps updated
- Users can benefit from a messaging service, i.e. message other active users