CS2524: DISTRIBUTED SYSTEMS ASSESSMENT MUD GAME

Bradley Scott
Student ID: 51661169
Email: b.scott.16@aberdeen.ac.uk

Tasks completed

- CGS D
- CGS C
- CGS B
- CGS A5

Installation and running instructions

- please ensure your directory structure is as follows:

/3524/cs3524/solutions/mud

this was the structure assumed in the practicals

- also ensure you have java version 1.8 installed (this is what my code was run on)

1. Firstly you must run rmi registry on an open port (within the range of 50010 – 50019 should work) using the following command:

```
rmiregistry <port>
```

for example:

```
user > rmiregistry 50010
```

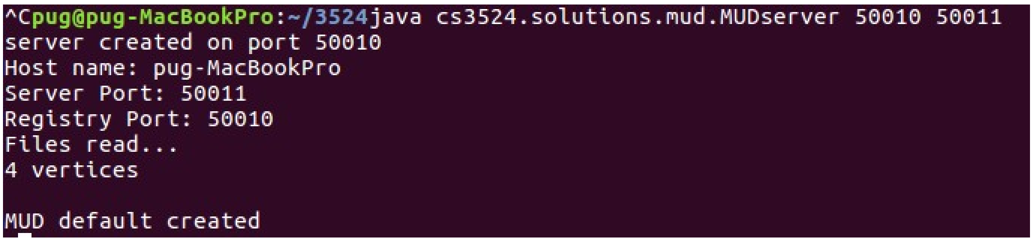
2. Once your rmi registry has been run, open a new terminal window and run the server mainline using the following command:

```
java cs3524.solutions.mud.MUDserver <registry port> <server port>
```

where <registry port> is the port configured with the rmi registry in the previous step and <server port> is a new open port for the server.

for example:

java cs3524.solutions.mud.MUDserver 50010 50011



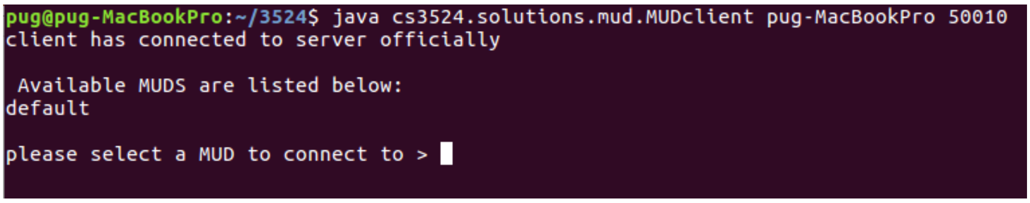
```
^Cpug@pug-MacBookPro:~/3524java cs3524.solutions.mud.MUDserver 50010 50011
server created on port 50010
Host name: pug-MacBookPro
Server Port: 50011
Registry Port: 50010
Files read...
4 vertices

MUD default created
```

3. Finally, you must connect to the server as a client. To connect to the server and begin playing the game, use the following command:

java cs3524.solutions.mud.MUDclient <hostname> <registry port>

for example:

java cs3524.solutions.mud.MUDclient pug-MacBookPro 50010



```
pug@pug-MacBookPro:~/3524$ java cs3524.solutions.mud.MUDclient pug-MacBookPro 50010
client has connected to server officially

 Available MUDS are listed below:
default

please select a MUD to connect to > 
```

Client Instructions

The client will be presented with a list of MUD worlds upon connecting, assuimg this will be the first connect the only world available will be default. The user simply types which MUD world they wish to join, then they will be asked for a username before they can begin playing.

The player can play the game using the following commands:

Movement controls:

- north – moves player north if possible
- east- moves player east if possible
- south – moves player south if possible
- west – moves player south if possible

other gameplay commands:

- look – will display all users and items in your current location
- take -will present you with a list of items you can add to your inventory
    - typing that item name will then add that item to your inventory
- online – displays a list of online users in current MUD world
- help – will display the list of available commands

MUD world commands

- make mud – will prompt the user to create a new MUD world
- change mud – will prompt the user to select a new MUD world to join on the server
- quit – will leave the game and close client connection

How it works

In order to implement the solution I extended the solution given in the second practical to create server and client functionalities. Most of the functionality to make the game work was already provided in the MUD.java and hence most of the function calls in my MUDservice implementation are calls to functions from that file. Extra functions were required to make and change MUDs at run time and these can be seen in the MUDServiceImplementation.

The MUDserver makes the functionality available to the client through the Mud interface, and Mud interface implementation by creating a stub which is registered with the RMI registry.

The client then connects to the RMI registry to make use of these functions to make the game playable for the client(s).