Group 7 - Andrew Lam , Jia Li, Wendy Zheng

CSE 310 Tic Tac Toc – The Modified Tic Tac Toe

Game Description:

In Tic Tac Toc, the first player to force their opponent to complete three in a row wins. This is a 2-player game and each game will have either a winner or a draw.

Brief Overview:

We have finished 100% of the project, excluding all the extra credit. We implemented single game and multi game. The client and server python files **MUST be run on linux/unix systems that support ePoll (Windows does not support ePoll).** This game is programmed in Python 2.7.

Single game functionality:

- Commands: Help, Login, Place, Exit
- One of the major functionalities implemented is AutoPlay. AutoPlay automatically pairs 2 players on the server and starts a game. The player who logs in first will get to place his/her move first. When a game ends, the server will automatically start a new game between the 2 players. If only one player is on the server, he/she will be forced to wait for another player to log on.
- The single game server supports up to a maximum of 2 players.

Multi game functionality:

- Commands: Help, Login, Place, Exit, Games, Who, Play
- One of the major functionalities implemented is Play. Players will have the ability to play another user on the server. Players will not be able to play themselves. AutoPlay is removed in multi game server.
- The multi game server supports an unlimited number of players until all resources on the machine is utilized. At most 6 requests can be sent to the server at a given time.
  E.g.: Only 6 players will be able to issue a command at the same moment.
  If there is a delay of at least 100 milliseconds between requests, then there is no limit on how much the server can handle.

Single Game User Documentation:

The single game server only supports a maximum of 2 players and a maximum of 1 ongoing game at any time. To start, make sure you are on a **linux/unix system that supports ePoll (Windows does not support ePoll)** and that Python 2.7 is installed.

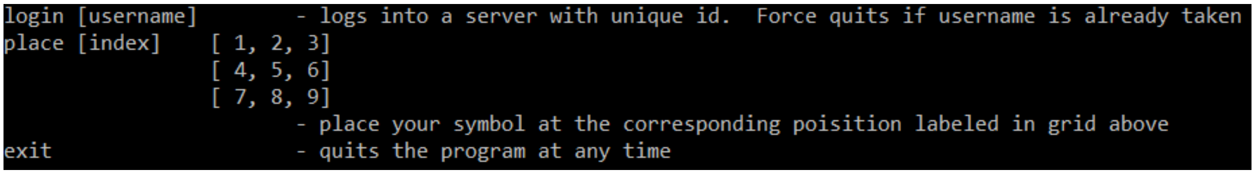
Instructions for setting up game:

1. Run “python singleServer.py” on a linux/unix environment. The default port is 9347.
2. Run “python singlePlayer.py serverIP 9347” on a linux/unix environment. This will set up the first client for the first player. You will need to repeat this step to set up for the second player.

Supported commands:

help

- Displays a help menu:



```
login [username]        - logs into a server with unique id.  Force quits if username is already taken
place [index]    [ 1, 2, 3]
                 [ 4, 5, 6]
                 [ 7, 8, 9]
                        - place your symbol at the corresponding poisition labeled in grid above
exit                    - quits the program at any time
```

login username

- Log into the server with *username* as user id
- Possible successful messages:
  - “Please wait … searching for opponents”
    You are the only player on the server right now.
  - “Your opponent is: otherPlayerName”
    The server has paired you with another player where otherPlayerName is your opponent’s name.
    A 3 by 3 board is also printed and all empty fields are shown with dots. The player who logged in first will get to place their move first.
    Picture of board: . . .
    . . .
    . . .
- Possible error messages:
  - “Username has been taken, please enter another name:”
    You are asked to use another username because the current username is in use.
  - “Username must not contain any spaces!”
    Your username currently contains a space and spaces are not allowed. Only alphanumeric characters are allowed.

place number

- Place a move on the game board at position *number*
- You are only allowed to use numbers 1 to 9. The 3 by 3 game board is represented in the following numerical representation:

  ```
  1 2 3
  4 5 6
  7 8 9
  ```

- Possible successful messages:

  -
    ```
    X .  .
    .  .  .
    .  .  .
    ```
    If you are the first player, the game board will be printed with X at the position specified with the place command. In this case, it is "place 1".

  -
    ```
    .    .    .
    O    .    .
    .    .    .
    ```
    If you are the second player, the game board will be printed with O at the position specified with the place command. In this case, it is "place 4".

  - Both players will see a board on their screen after a move is done.

  - If there is a winner, winner of the game will be displayed and a new game will be automatically started.

  - If it is a draw, both players will be notified that the game is a draw and a new game will be automatically started.

- The error message "invalid move: number" is caused by one of the following:
  - It is currently not your turn to place a move
  - You tried to place a move on a spot that is already placed.
  - You used a number less than 0 or greater than 9 or your number is not an integer. The game board in numerical representation will be displayed on your screen to help you.

exit

- You may type this command at any time to exit the client program.
- If you are not in a game, the client program automatically exits.
- If you are in a game, the following will happen:
  - The other player will be notified that you left the game and will be the winner of the game.
  - Your client program automatically exits.

Multiple Game User Documentation:

The multiple game server supports as many players and ongoing games as permitted by hardware resources. To start, make sure you are on a **linux/unix system that supports ePoll (Windows does not support ePoll)** and that Python 2.7 is installed.

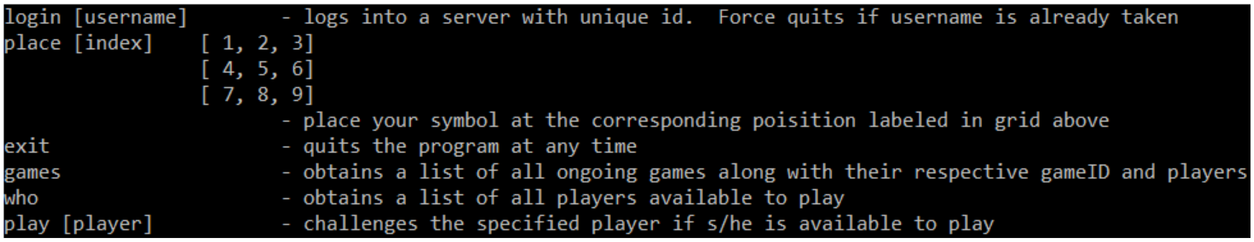
Instructions for setting up game:

3. Run “python multipleServer.py” on a linux/unix environment. The default port is 9347.
4. Run “python player.py serverIP 9347” on a linux/unix environment. This will set up the first client for the first player. You will need to repeat this step to set up for other players.

Supported commands:

help

- Displays a help menu:



```
login [username]        - logs into a server with unique id.  Force quits if username is already taken
place [index]     [ 1, 2, 3]
                  [ 4, 5, 6]
                  [ 7, 8, 9]
                        - place your symbol at the corresponding poisition labeled in grid above
exit                    - quits the program at any time
games                   - obtains a list of all ongoing games along with their respective gameID and players
who                     - obtains a list of all players available to play
play [player]           - challenges the specified player if s/he is available to play
```

login username

- Log into the server with *username* as user id

- Possible successful messages:
  - “Logged in successfully at time: [Month Day Year CurrentTime]”
    The message displays the date and current time of your successful login.

- Possible error messages:
  - “Username has been taken, please enter another name:”
    You are asked to use another username because the current username is in use.

  - “Username must not contain any spaces!”
    Your username currently contains a space and spaces are not allowed. Only alphanumeric characters are allowed.

place number

- Place a move on the game board at position *number*

- You are only allowed to use numbers 1 to 9. The 3 by 3 game board is represented in the following numerical representation: 1 2 3
  4 5 6
  7 8 9

- Possible successful messages:

  -
    ```
    X .  .
    .  .  .
    .  .  .
    ```
    If you are the first player, the game board will be printed with X at the position specified with the place command. In this case, it is “place 1”.

  -
    ```
    .    .    .
    O    .    .
    .    .    .
    ```
    If you are the second player, the game board will be printed with O at the position specified with the place command. In this case, it is “place 4”.

  - Both players will see a board on their screen after a move is done.

  - If there is a winner, winner of the game will be displayed and a new game will be automatically started.

  - If it is a draw, both players will be notified that the game is a draw and a new game will be automatically started.

- The error message “invalid move: number” is caused by one of the following:
  - It is currently not your turn to place a move
  - You tried to place a move on a spot that is already placed.
  - You used a number less than 0 or greater than 9 or your number is not an integer. The game board in numerical representation will be displayed on your screen to help you.

exit

- You may type this command at any time to exit the client program.
- If you are not in a game, the client program automatically exits.
- If you are in a game, the following will happen:
  - The other player will be notified that you left the game and will be the winner of the game.
  - Your client program automatically exits.

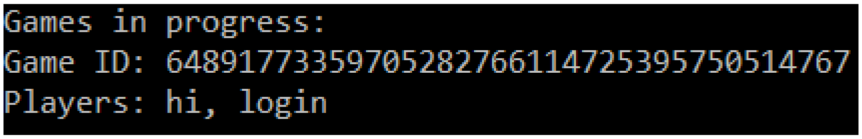
who

- You may type this command at any time when you are logged in to see user id of all available players, excluding yourself.
- Possible output:
  - “No available users online!”
  - “Available users online: [list of all available player’s username]”

games

- You may type this command at any time when you are logged in to see game id and players for each ongoing game.
- Possible output:
  - "No games in progress!"
  - "Games in progress: [list of all ongoing games]"

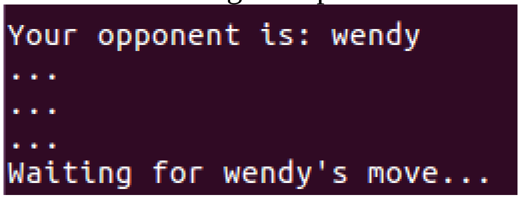
    Sample output:



```
Games in progress:
Game ID: 64891773359705282766114725395750514767
Players: hi, login
```

play username

- Challenges a player with a user id of username. The player being challenged must be available (not currently in a game).
- Successful message sample:



```
Your opponent is: wendy
...
...
...
Waiting for wendy's move...
```

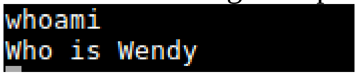
- Possible error messages:
  - "Opponent [username] does not exist!"

    You tried to play a non-existing player.
  - "Opponent [username] is busy!"

    You tried to play a player who is currently in a game.

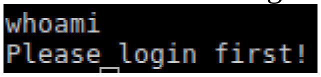
whoami

- Prints your username/user id. This is an extra functionality not required in specifications.
- Successful message sample:



```
whoami
Who is Wendy
```

- Possible error messages:



```
whoami
Please login first!
```

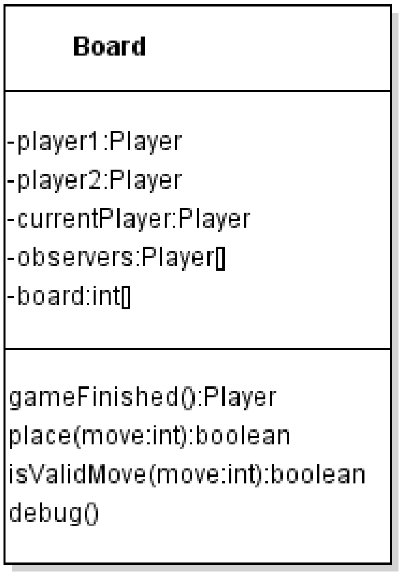
Group 7 - Andrew Lam , Jia Li, Wendy Zheng

System Documentation:

Source Code Files

board.py

Code for the functionality of the Tic Tac Toc game



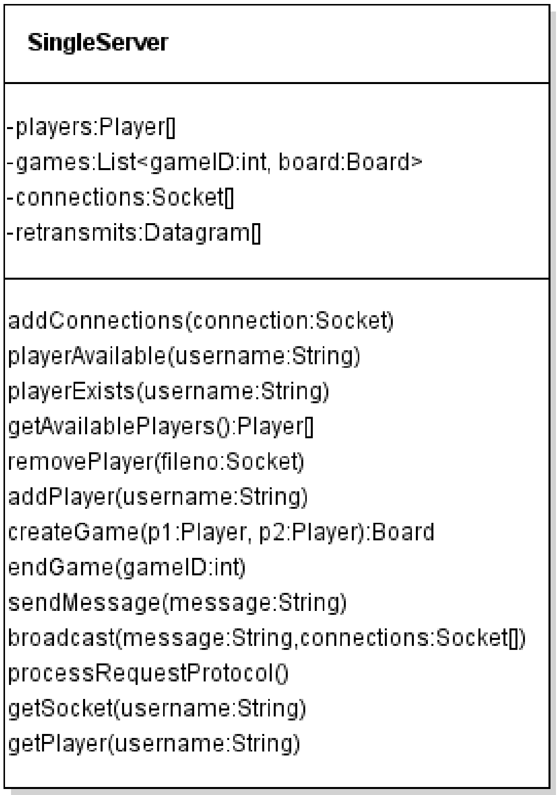
Board

-player1:Player
-player2:Player
-currentPlayer:Player
-observers:Player[]
-board:int[]

gameFinished():Player
place(move:int):boolean
isValidMove(move:int):boolean
debug()

singleServer.py

Server that handles only 1 ongoing game and maximum of 2 players.



SingleServer

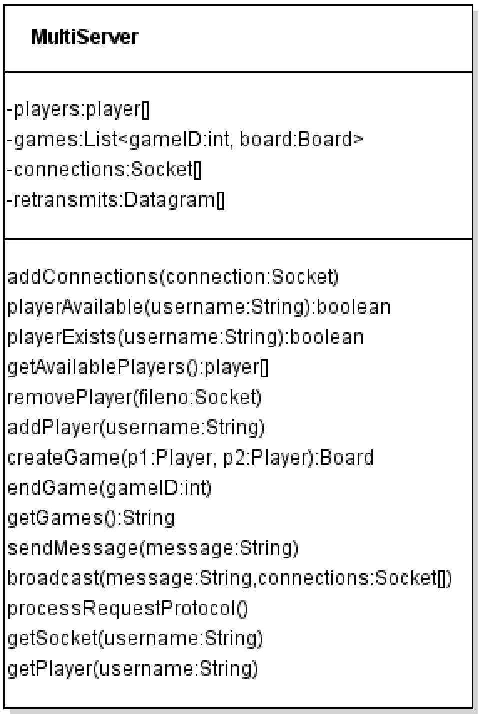
-players:Player[]
-games:List<gameID:int, board:Board>
-connections:Socket[]
-retransmits:Datagram[]

addConnections(connection:Socket)
playerAvailable(username:String)
playerExists(username:String)
getAvailablePlayers():Player[]
removePlayer(fileno:Socket)
addPlayer(username:String)
createGame(p1:Player, p2:Player):Board
endGame(gameID:int)
sendMessage(message:String)
broadcast(message:String,connections:Socket[])
processRequestProtocol()
getSocket(username:String)
getPlayer(username:String)

Group 7 - Andrew Lam , Jia Li, Wendy Zheng

multiServer.py

Server that handles multiple ongoing games and unlimited players.



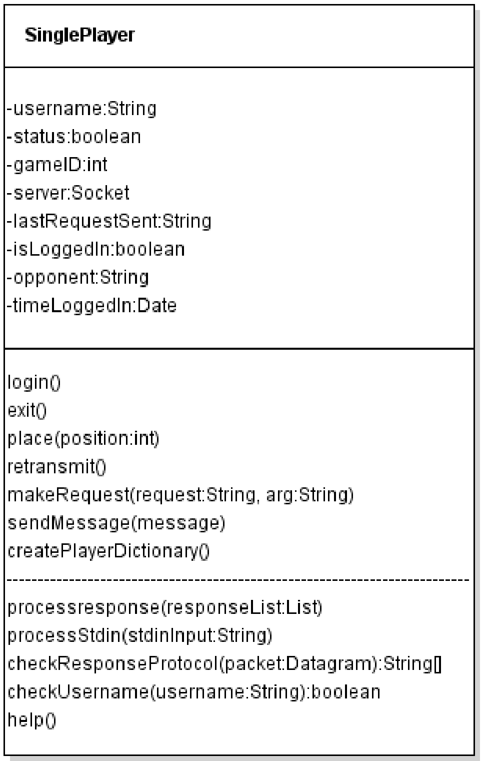
MultiServer

```
-players:player[]
-games:List<gameID:int, board:Board>
-connections:Socket[]
-retransmits:Datagram[]
```

```
addConnections(connection:Socket)
playerAvailable(username:String):boolean
playerExists(username:String):boolean
getAvailablePlayers():player[]
removePlayer(fileno:Socket)
addPlayer(username:String)
createGame(p1:Player, p2:Player):Board
endGame(gameID:int)
getGames():String
sendMessage(message:String)
broadcast(message:String,connections:Socket[])
processRequestProtocol()
getSocket(username:String)
getPlayer(username:String)
```

singlePlayer.py

Client for singleServer.py



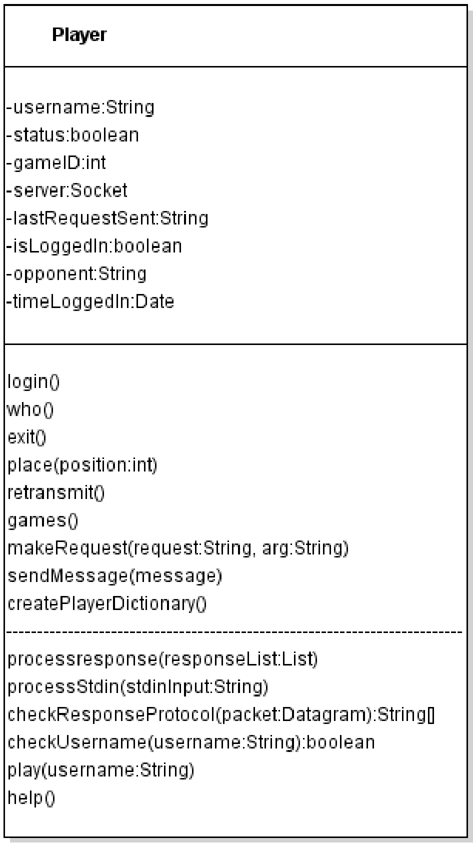
SinglePlayer

```
-username:String
-status:boolean
-gameID:int
-server:Socket
-lastRequestSent:String
-isLoggedIn:boolean
-opponent:String
-timeLoggedIn:Date
```

```
login()
exit()
place(position:int)
retransmit()
makeRequest(request:String, arg:String)
sendMessage(message)
createPlayerDictionary()
--------------------------------------------------------------------------
processresponse(responseList:List)
processStdin(stdinInput:String)
checkResponseProtocol(packet:Datagram):String[]
checkUsername(username:String):boolean
help()
```

player.py

Client for multiServer.py



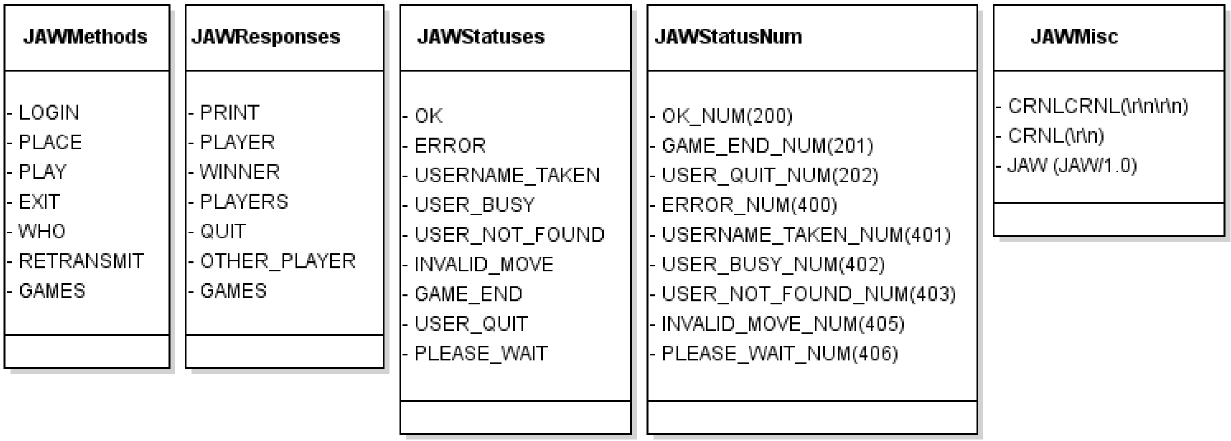
Player

-username:String
-status:boolean
-gameID:int
-server:Socket
-lastRequestSent:String
-isLoggedIn:boolean
-opponent:String
-timeLoggedIn:Date

login()
who()
exit()
place(position:int)
retransmit()
games()
makeRequest(request:String, arg:String)
sendMessage(message)
createPlayerDictionary()
----
processresponse(responseList:List)
processStdin(stdinInput:String)
checkResponseProtocol(packet:Datagram):String[]
checkUsername(username:String):boolean
play(username:String)
help()

jaw_enums.py

Contains variables and all information about the JAW protocol



JAWMethods

- LOGIN
- PLACE
- PLAY
- EXIT
- WHO
- RETRANSMIT
- GAMES

JAWResponses

- PRINT
- PLAYER
- WINNER
- PLAYERS
- QUIT
- OTHER_PLAYER
- GAMES

JAWStatuses

- OK
- ERROR
- USERNAME_TAKEN
- USER_BUSY
- USER_NOT_FOUND
- INVALID_MOVE
- GAME_END
- USER_QUIT
- PLEASE_WAIT

JAWStatusNum

- OK_NUM(200)
- GAME_END_NUM(201)
- USER_QUIT_NUM(202)
- ERROR_NUM(400)
- USERNAME_TAKEN_NUM(401)
- USER_BUSY_NUM(402)
- USER_NOT_FOUND_NUM(403)
- INVALID_MOVE_NUM(405)
- PLEASE_WAIT_NUM(406)

JAWMisc

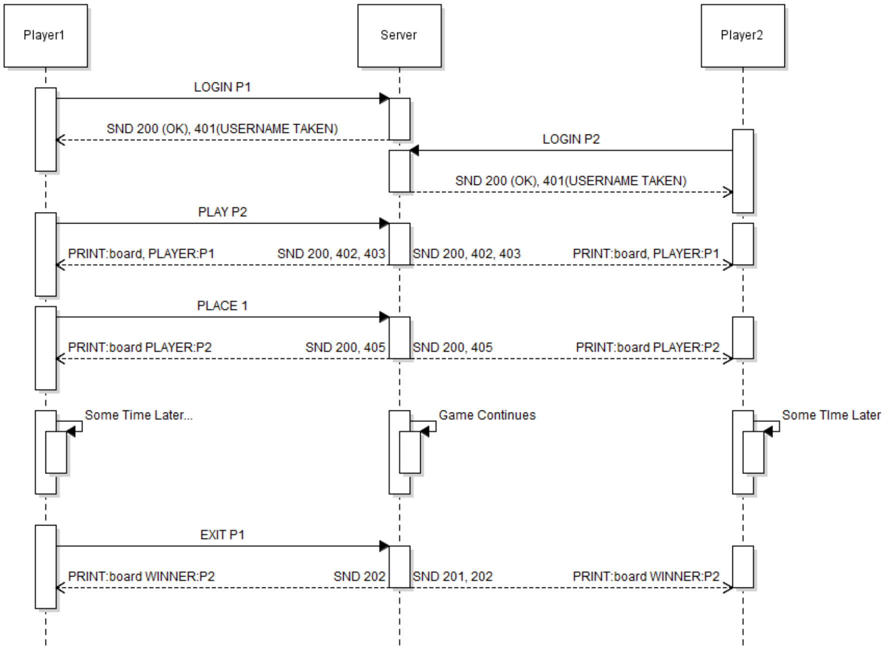
- CRNLCRNL(\r\n\r\n)
- CRNL(\r\n)
- JAW (JAW/1.0)

Changing Server Port

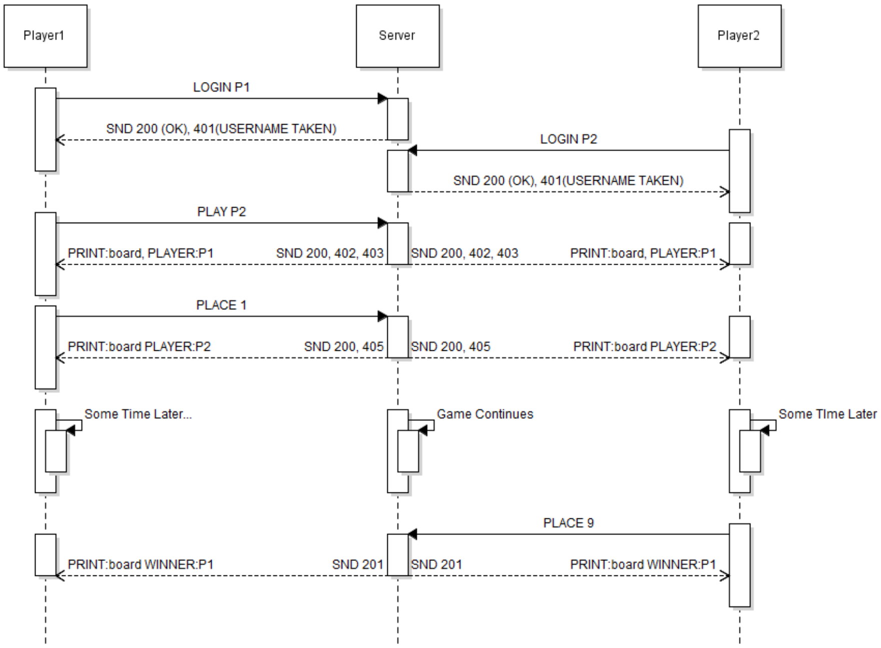
Default Single/Multiple Game Server Port: 9347
To change the server port for single/multiple game server, edit the *serverPort* variable in the source code.

Client Server Communication Protocol Diagram

Protocol diagram for player quitting:

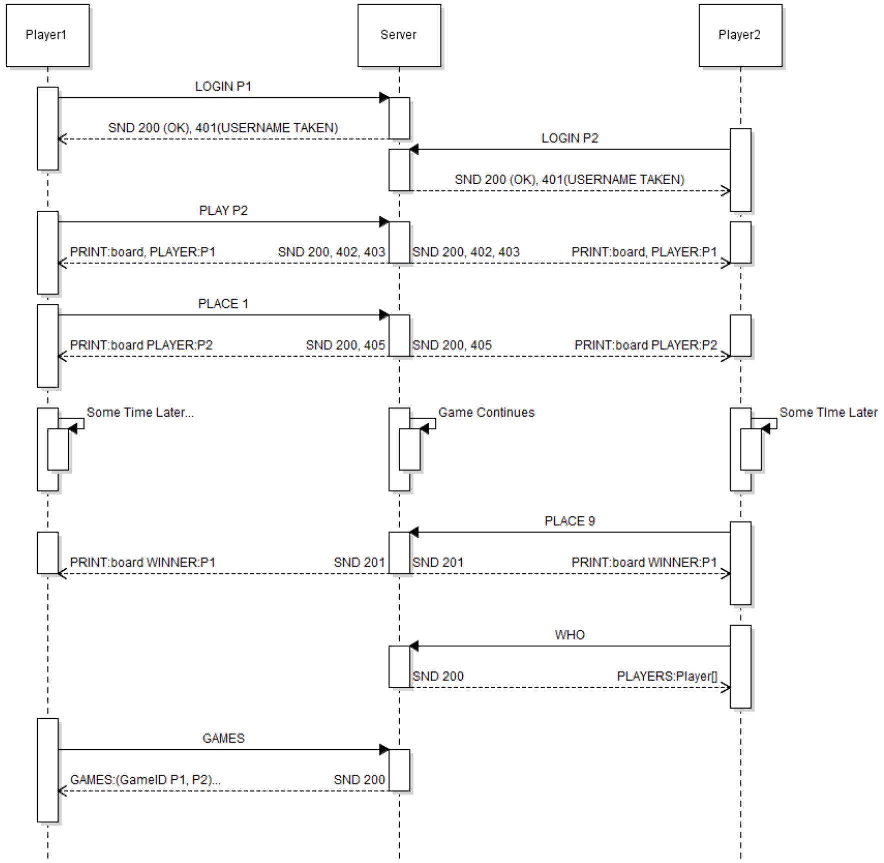


Protocol diagram for single game server:



Group 7 - Andrew Lam , Jia Li, Wendy Zheng

Protocol diagram for multiple game server:



Concurrency

We did not use multi-threading in this project. We used ePoll which is Linux/Unix specific and is part of select. ePoll is a very efficient technology for multiplexing I/O.

Major Data Structures

The 2 major data structures used in this project are list and dictionary in Python 2. By utilizing key value pairs in a dictionary, we are able to have fast operations improving the speed of the server. The game board is stored in a list which allows ease of use when a player places a move.

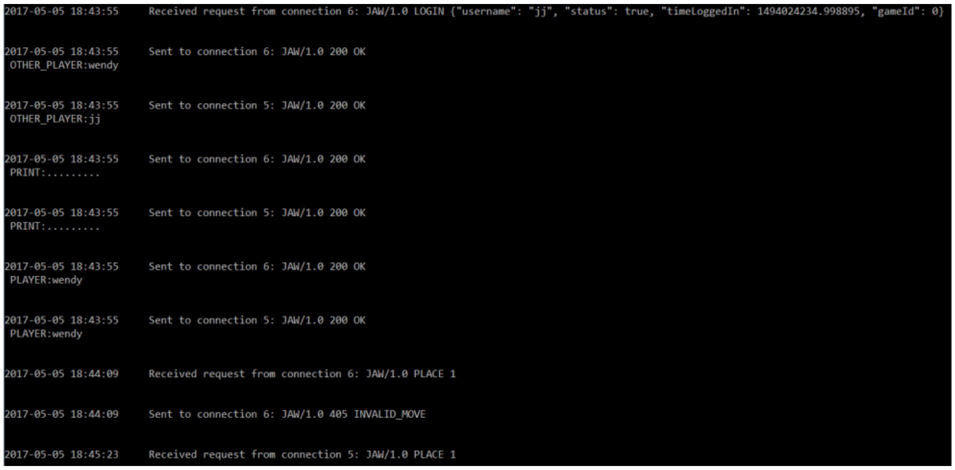
Important Algorithms

Because TCP packets can be lost, we implemented retransmit in server and client files. It is possible for clients to receive an empty packet because we are using ePoll to multiplex I/O. When a client receives an empty packet, it asks the server for a retransmission to receive the data again. The server stores the most recent message of each client's action. If the client's action requires several messages to be sent consecutively from the server, those consecutive messages are stored and counted as one retransmission.

Debugging/Log Output

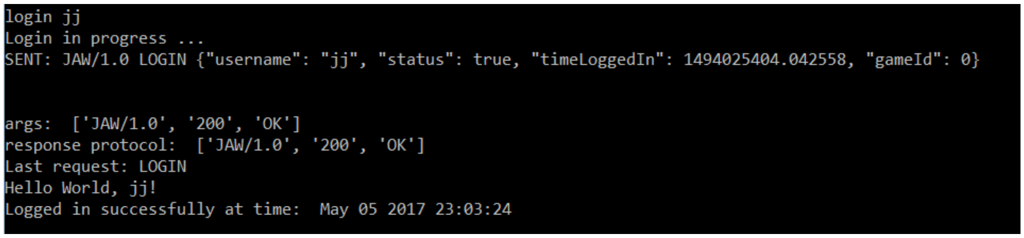
All client and server python files come with logging/debugging functionality. The server logs all received requests and sent responses with a timestamp. The client logs all sent requests, received responses, and parsed responses. All logging/debugging are turned off by default.

Sample Server Log



```
2017-05-05 18:43:55     Received request from connection 6: JAW/1.0 LOGIN {"username": "jj", "status": true, "timeLoggedIn": 1494024234.998895, "gameId": 0}

2017-05-05 18:43:55     Sent to connection 6: JAW/1.0 200 OK
 OTHER_PLAYER:wendy

2017-05-05 18:43:55     Sent to connection 5: JAW/1.0 200 OK
 OTHER_PLAYER:jj

2017-05-05 18:43:55     Sent to connection 6: JAW/1.0 200 OK
 PRINT:.........

2017-05-05 18:43:55     Sent to connection 5: JAW/1.0 200 OK
 PRINT:.........

2017-05-05 18:43:55     Sent to connection 6: JAW/1.0 200 OK
 PLAYER:wendy

2017-05-05 18:43:55     Sent to connection 5: JAW/1.0 200 OK
 PLAYER:wendy

2017-05-05 18:44:09     Received request from connection 6: JAW/1.0 PLACE 1

2017-05-05 18:44:09     Sent to connection 6: JAW/1.0 405 INVALID_MOVE

2017-05-05 18:45:23     Received request from connection 5: JAW/1.0 PLACE 1
```

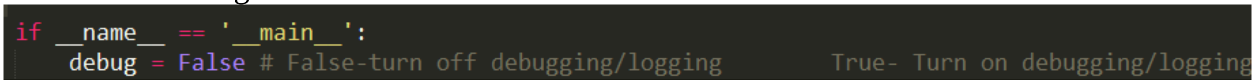
Sample Client Log (with regular output)



```
login jj
Login in progress ...
SENT: JAW/1.0 LOGIN {"username": "jj", "status": true, "timeLoggedIn": 1494025404.042558, "gameId": 0}


args:  ['JAW/1.0', '200', 'OK']
response protocol:  ['JAW/1.0', '200', 'OK']
Last request: LOGIN
Hello World, jj!
Logged in successfully at time:  May 05 2017 23:03:24
```

To turn on/off logging/debugging:

1. Look for the debug variable in the main method.



```
if __name__ == '__main__':
    debug = False # False-turn off debugging/logging        True- Turn on debugging/logging
```

2. Set debug to True to turn on debugging. Otherwise, set debug to False.

Adding/Editing/Removing server functionalities

To add a new server functionality such as more commands:

1. Open the correct python server file in an editor of your choice. singleServer.py for single game and multServer.py for multiple game.
2. Define new functions as necessary in the Server class
3. Add a new *elif* clause in the function *processRequestProtocol()* located in the server class
4. To add new instance/class variables, add the variables in the _ _init_ _ function in the Server class.

To edit an existing server functionality such as a command:

1. Go to the correct python server file. singleServer.py for single game and multServer.py for multiple game.
2. Go to the function *processRequestProtocol()* located in the Server class.
3. Look for the corresponding *elseif* or *if* clause in *processRequestProtocol()* and edit the code inside. If there are function(s) called inside the clause, edit the called function(s) to achieve desired functionality without impacting other functionalities.
4. To edit existing instance/class variables, edit the variables in the _ _init_ _ function in the Server class.

To remove an existing server functionality such as a command:

1. Open the correct python server file in an editor of your choice. singleServer.py for single game and multServer.py for multiple game.
2. Go to the function *processRequestProtocol()* located in the Server class.
3. Look for the corresponding *elseif* or *if* clause in *processRequestProtocol()* and remove the clause. If there are function(s) called inside the clause, you may remove the called function(s) as long as they do not impact other functionalities.
4. To remove existing instance/class variables, remove the variables in the _ _init_ _ function in the Server class.

Changing Tic Tac Toc gameplay

1. Open board.py in an editor of your choice.
2. To add/edit/remove instance/class variables, look in the _ _init_ _ function in the Board class.
3. To change the winning conditions of the game, edit the *gameFinished()* function in the Board class.
4. To change how placing a move works, edit *place()* and *isValidMove()* functions in the Board class.
5. DO NOT change the _ _str_ _ function in the Board class.

Adding/Editing/Removing client functionalities:

1. Open the correct python client file in an editor of your choice. singlePlayer.py for single game and player.py for multiple game.
2. To add new/edit/remove instance/class variables, look in the _ _init_ _ function in the Player class.
3. To add client command, edit *processStdin()* and *processResponse()*. Add new functions in Player class as needed.
4. To edit client command, edit *processStdin()* and *processResponse()*. Edit the corresponding functions to the command you want to edit in Player class as needed.
5. To remove client command, edit *processStdin()* and *processResponse()*. Remove the corresponding functions to the command you want to remove in Player class without altering other client functionalities.
6. DO NOT change the _ _str_ _ function in the Player class.

Error Conditions

- USERNAME_TAKEN with status code 401
  This happens when a player tries to log onto the server with an existing user id/username.
  Player will have to log in with another username.

- USER_BUSY with status code 402
  This happens when a player tries to play another player who is in a game
  Player will have to play an available player found from WHO command.

- USER_NOT_FOUND with status code 403
  This happens when a player tries to play a non-existing player.
  Player will have to play an available player found from WHO command.

- INVALID_MOVE with status code 405
  This happens when a player tries to play when it is not his/her turn, tries to place a move at a position less than 0 or greater than 9, or tries to place a move on a position that already contains a move.
  Player will have to place move at his/her turn with a valid position.

- ERROR with status code 400
  This happens when a player performs an unsupported operation that is not listed above.
  These errors are not important enough to have their own error status code and message.
  For single server, this includes when a player tries to log onto a server that has 2 players.
  Player will be notified something has gone wrong and will need to retry their last action.

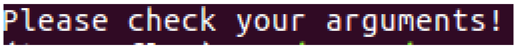
Generating/Compiling Executables

By running “python fileName.py”, Python will automatically compile and run the python code. A pyc file will be created in the same directory as the py files for all imported py files. The pyc file contains byte code which is the Python compiled version of the source (py) code.

Testing the Player (Client) Program

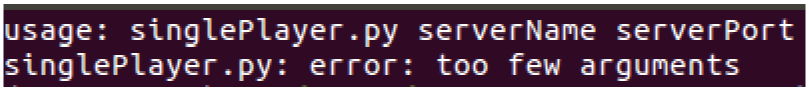
Starting the Player (Client) Application

- Running with wrong arguments



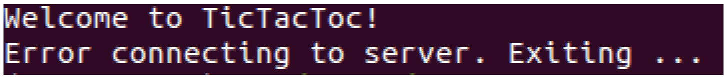
```
Please check your arguments!
```

- Running with missing arguments



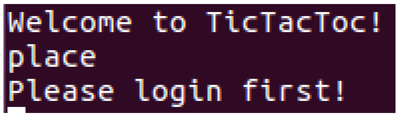
```
usage: singlePlayer.py serverName serverPort
singlePlayer.py: error: too few arguments
```

- Running without arguments, missing arguments, wrong arguments



```
Welcome to TicTacToc!
Error connecting to server. Exiting ...
```

- Running with correct arguments, place without logging in

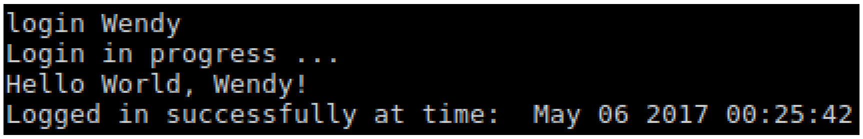


```
Welcome to TicTacToc!
place
Please login first!
```

❑ Single and MultiPlayer/Server

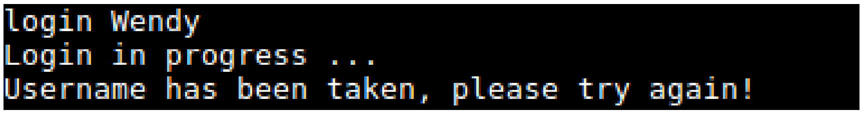
- Username Taken

First user:



```
login Wendy
Login in progress ...
Hello World, Wendy!
Logged in successfully at time:  May 06 2017 00:25:42
```

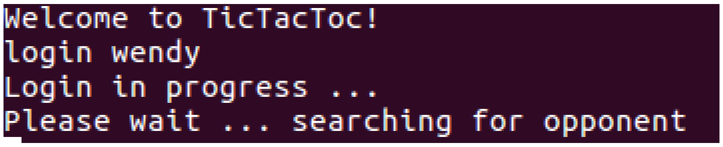
Other user:



```
login Wendy
Login in progress ...
Username has been taken, please try again!
```

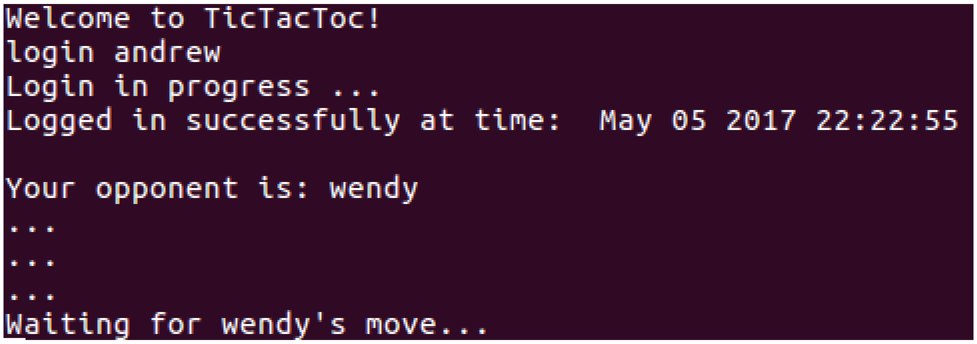
❑ Single Player/Server

- Running with correct arguments and login successful (no users online)



```
Welcome to TicTacToc!
login wendy
Login in progress ...
Please wait ... searching for opponent
```

- Running with correct arguments and login successful (autoplay)

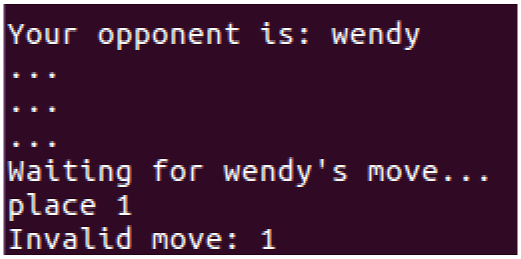


```
Welcome to TicTacToc!
login andrew
Login in progress ...
Logged in successfully at time:  May 05 2017 22:22:55

Your opponent is: wendy
...
...
...
Waiting for wendy's move...
```

- ❑ Multiplayer/Server
- ● Running with correct arguments and login

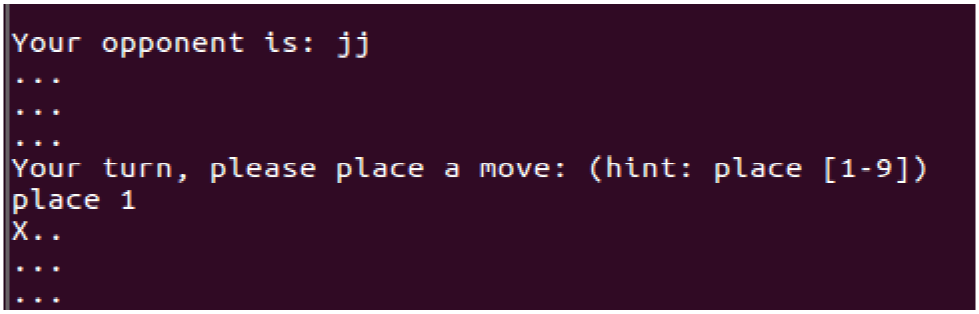
Playing a Game

- ❑ Single and MultiPlayer/Server
- ● Placing a move when not player turn



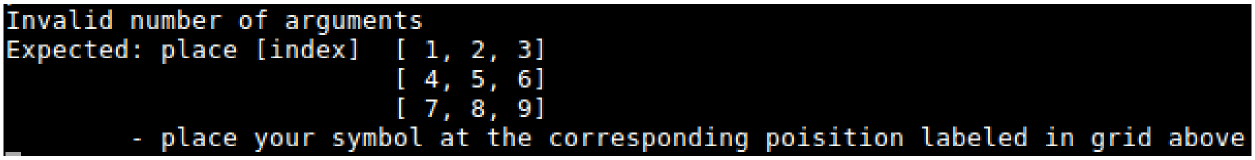
```
Your opponent is: wendy
...
...
...
Waiting for wendy's move...
place 1
Invalid move: 1
```

- ● Placing a move when player turn (valid)



```
Your opponent is: jj
...
...
...
Your turn, please place a move: (hint: place [1-9])
place 1
X..
...
...
```

- ● Placing a move when player turn (invalid)

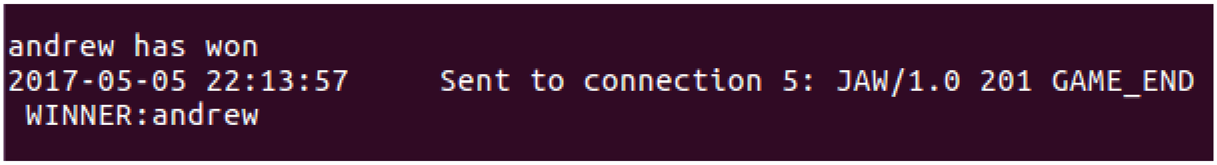


```
Invalid number of arguments
Expected: place [index]  [ 1, 2, 3]
                         [ 4, 5, 6]
                         [ 7, 8, 9]
        - place your symbol at the corresponding poisition labeled in grid above
```

Winning, Losing and Drawing Games

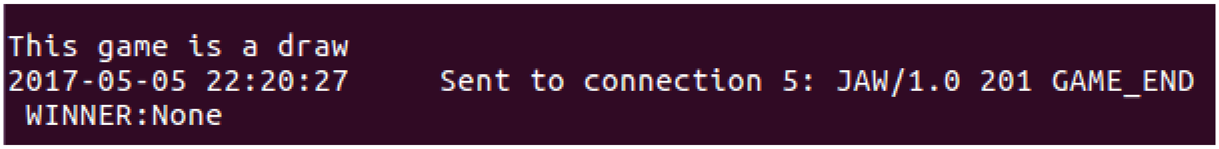
❏ Single and MultiPlayer/Server

- Winning/Losing



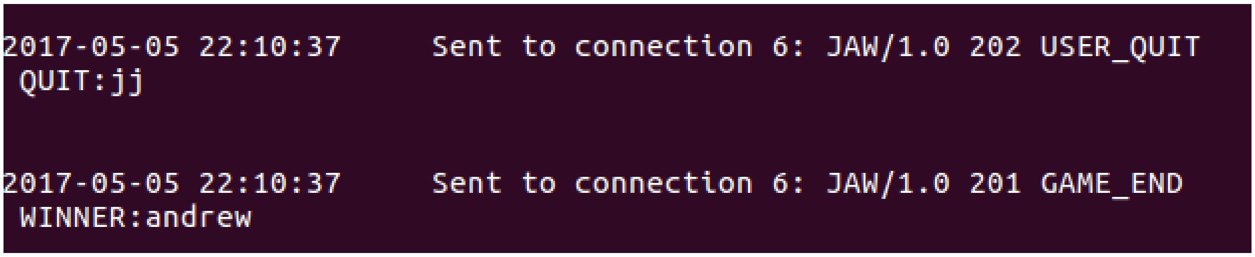
```
andrew has won
2017-05-05 22:13:57     Sent to connection 5: JAW/1.0 201 GAME_END
 WINNER:andrew
```

- Draw



```
This game is a draw
2017-05-05 22:20:27     Sent to connection 5: JAW/1.0 201 GAME_END
 WINNER:None
```

- User Quit

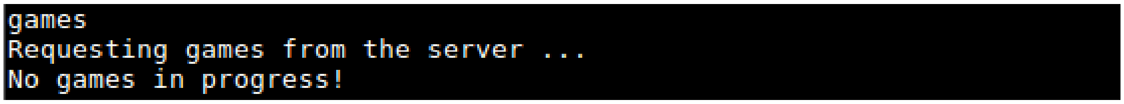


```
2017-05-05 22:10:37     Sent to connection 6: JAW/1.0 202 USER_QUIT
 QUIT:jj


2017-05-05 22:10:37     Sent to connection 6: JAW/1.0 201 GAME_END
 WINNER:andrew
```

List of Ongoing Games

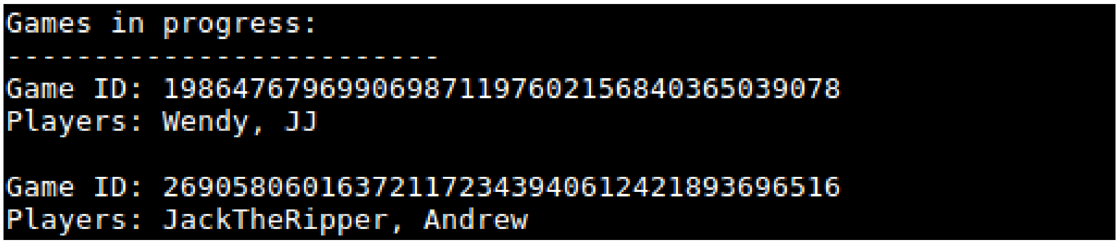
❏ MultiPlayer/Server

- No games in progress



```
games
Requesting games from the server ...
No games in progress!
```

- Games in progress



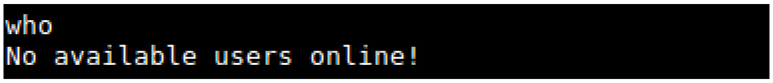
```
Games in progress:
-------------------------
Game ID: 198647679699069871197602156840365039078
Players: Wendy, JJ

Game ID: 269058060163721172343940612421893696516
Players: JackTheRipper, Andrew
```

Group 7 - Andrew Lam , Jia Li, Wendy Zheng

List of Available Players Online

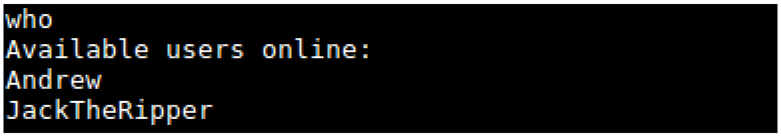
❑ MultiPlayer/Server

- No available players online



```
who
No available users online!
```

- Players online

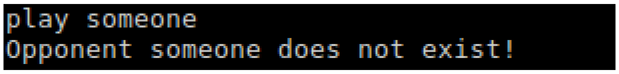


```
who
Available users online:
Andrew
JackTheRipper
```

Playing Specific Player

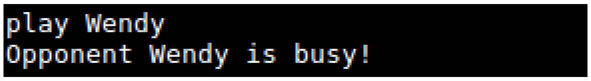
❑ MultiPlayer/Server

- User does not exist



```
play someone
Opponent someone does not exist!
```

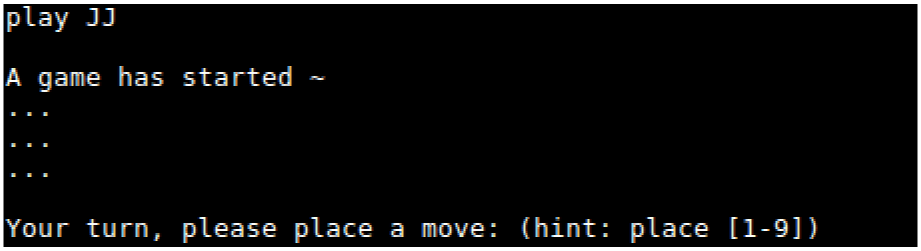
- User is busy



```
play Wendy
Opponent Wendy is busy!
```

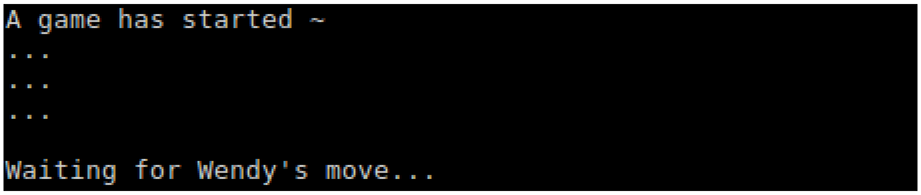
- User is available

First user:



```
play JJ

A game has started ~
...
...
...

Your turn, please place a move: (hint: place [1-9])
```

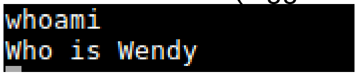
Opponent:



```
A game has started ~
...
...
...

Waiting for Wendy's move...
```

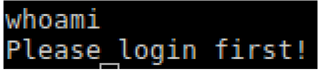
Who Am I?

- MultiPlayer/Server
- Who am I (logged in)



```
whoami
Who is Wendy
```

- Who am I (not logged in)

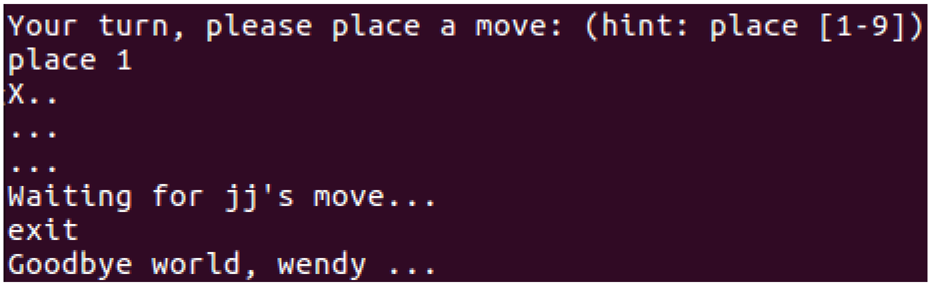


```
whoami
Please login first!
```

Exiting the Player (Client)

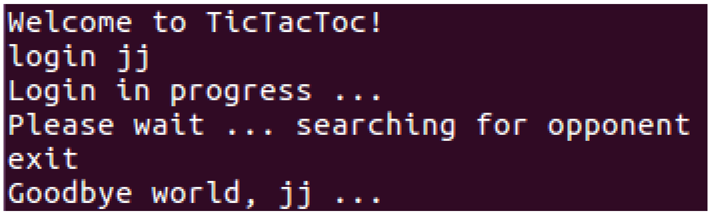
□ Single Player/Server

- Exiting when in middle of game



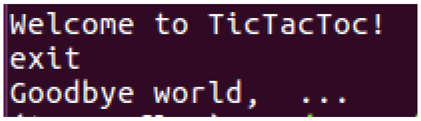
```
Your turn, please place a move: (hint: place [1-9])
place 1
X..
...
...
Waiting for jj's move...
exit
Goodbye world, wendy ...
```

- Exiting when searching for opponents



```
Welcome to TicTacToc!
login jj
Login in progress ...
Please wait ... searching for opponent
exit
Goodbye world, jj ...
```

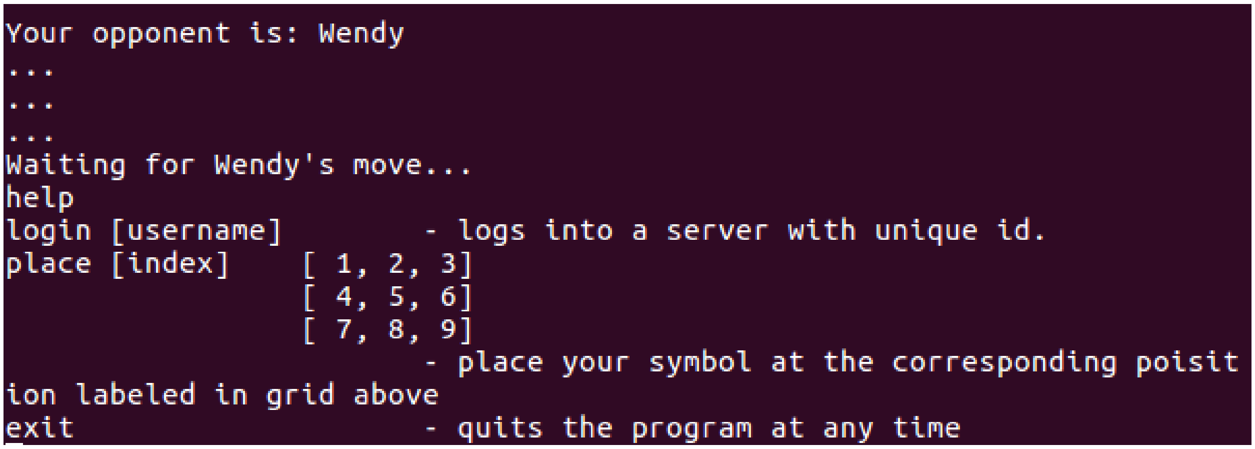
- Exiting prior to login



```
Welcome to TicTacToc!
exit
Goodbye world,  ...
```

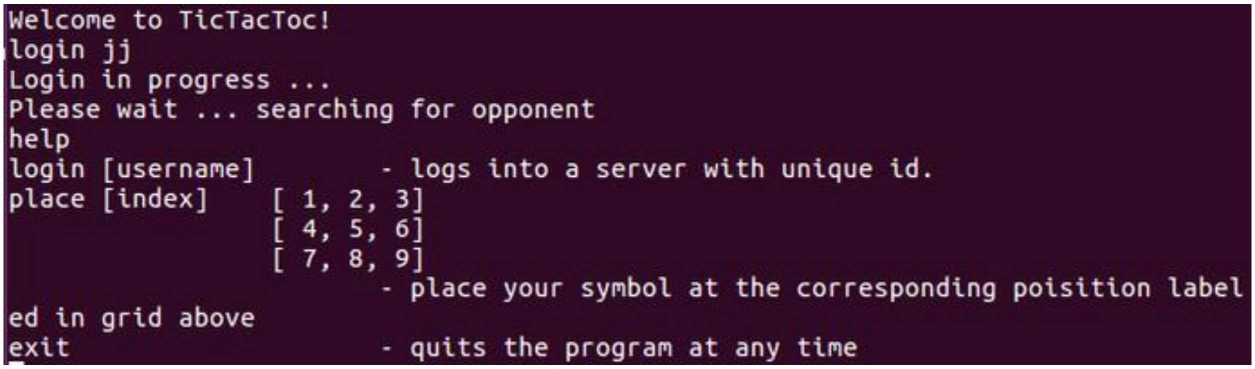
Help

- ❏ Single and MultiPlayer/Server
- Help when in middle of game



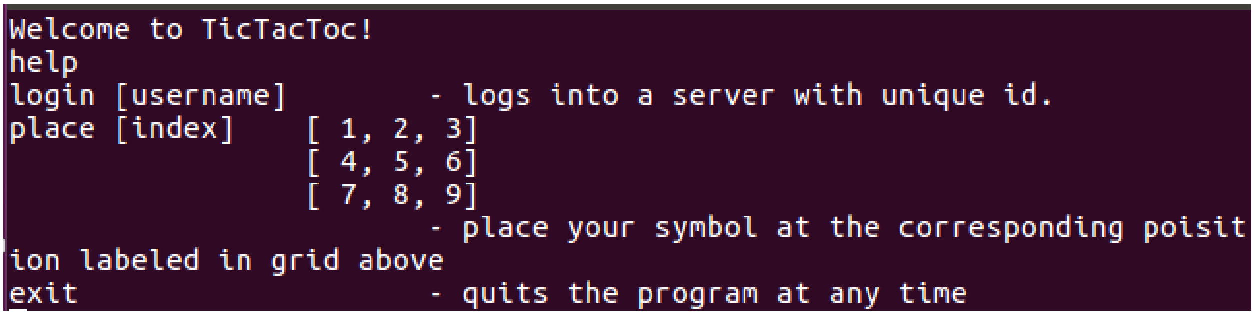
```
Your opponent is: Wendy
...
...
...
Waiting for Wendy's move...
help
login [username]        - logs into a server with unique id.
place [index]    [ 1, 2, 3]
                 [ 4, 5, 6]
                 [ 7, 8, 9]
                        - place your symbol at the corresponding poisit
ion labeled in grid above
exit                    - quits the program at any time
```

- Help when searching for opponents



```
Welcome to TicTacToc!
login jj
Login in progress ...
Please wait ... searching for opponent
help
login [username]        - logs into a server with unique id.
place [index]    [ 1, 2, 3]
                 [ 4, 5, 6]
                 [ 7, 8, 9]
                        - place your symbol at the corresponding poisition label
ed in grid above
exit                    - quits the program at any time
```

- Help prior to login



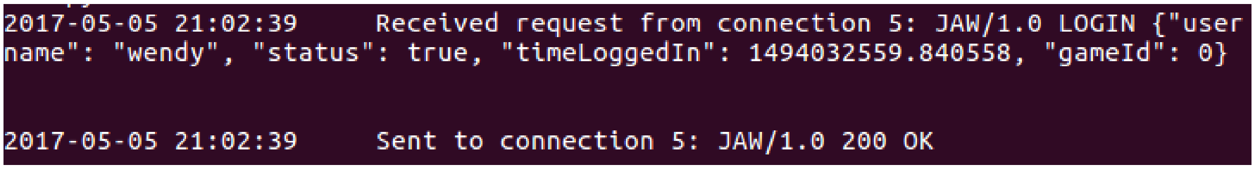
```
Welcome to TicTacToc!
help
login [username]        - logs into a server with unique id.
place [index]    [ 1, 2, 3]
                 [ 4, 5, 6]
                 [ 7, 8, 9]
                        - place your symbol at the corresponding poisit
ion labeled in grid above
exit                    - quits the program at any time
```

Testing the Server Program

Starting the Server Application

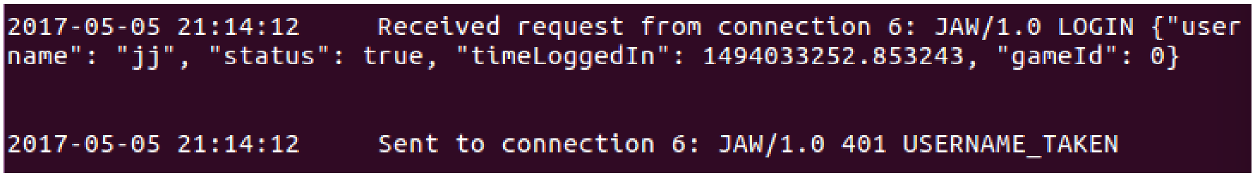
❏ Single and MultiPlayer/Server

- Login successful



```
2017-05-05 21:02:39     Received request from connection 5: JAW/1.0 LOGIN {"user
name": "wendy", "status": true, "timeLoggedIn": 1494032559.840558, "gameId": 0}


2017-05-05 21:02:39     Sent to connection 5: JAW/1.0 200 OK
```

- Login username taken

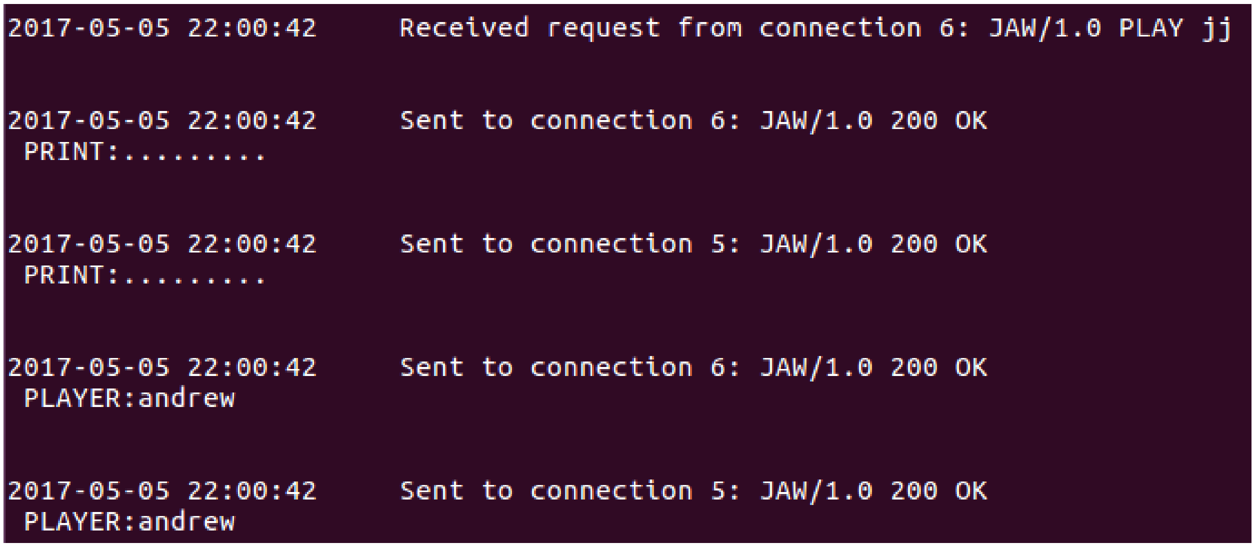


```
2017-05-05 21:14:12     Received request from connection 6: JAW/1.0 LOGIN {"user
name": "jj", "status": true, "timeLoggedIn": 1494033252.853243, "gameId": 0}


2017-05-05 21:14:12     Sent to connection 6: JAW/1.0 401 USERNAME_TAKEN
```

Playing a Game

❏ Single and MultiPlayer/Server

- Broadcast Board and Play

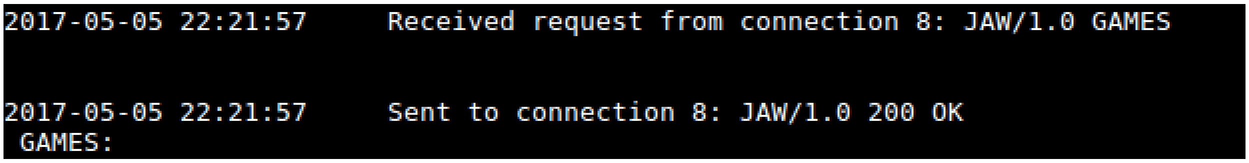


```
2017-05-05 22:00:42     Received request from connection 6: JAW/1.0 PLAY jj


2017-05-05 22:00:42     Sent to connection 6: JAW/1.0 200 OK
 PRINT:.........


2017-05-05 22:00:42     Sent to connection 5: JAW/1.0 200 OK
 PRINT:.........


2017-05-05 22:00:42     Sent to connection 6: JAW/1.0 200 OK
 PLAYER:andrew


2017-05-05 22:00:42     Sent to connection 5: JAW/1.0 200 OK
 PLAYER:andrew
```

List of Ongoing Games

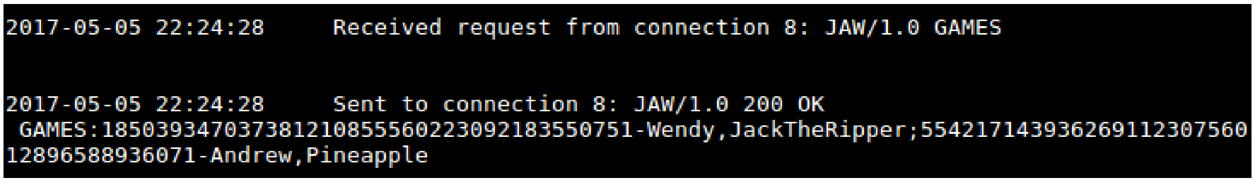
❏ MultiPlayer/Server

- No games in progress



```
2017-05-05 22:21:57     Received request from connection 8: JAW/1.0 GAMES


2017-05-05 22:21:57     Sent to connection 8: JAW/1.0 200 OK
 GAMES:
```

- Games in progress

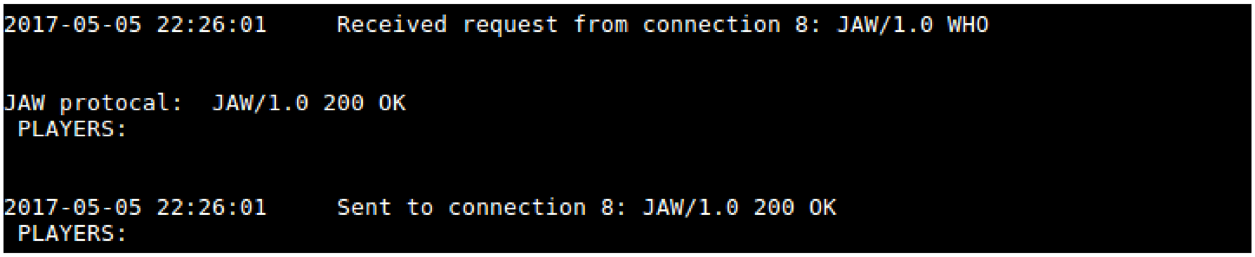


```
2017-05-05 22:24:28     Received request from connection 8: JAW/1.0 GAMES


2017-05-05 22:24:28     Sent to connection 8: JAW/1.0 200 OK
 GAMES:185039347037381210855560223092183550751-Wendy,JackTheRipper;554217143936269112307560
12896588936071-Andrew,Pineapple
```

List of Available Players Online

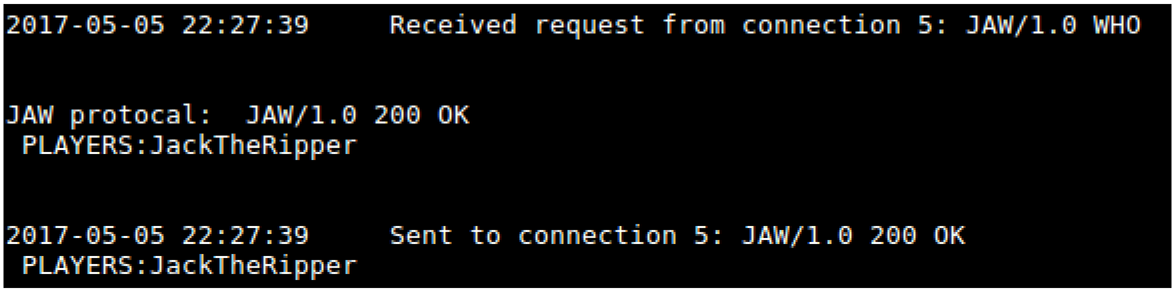
❏ MultiPlayer/Server

- No available players online



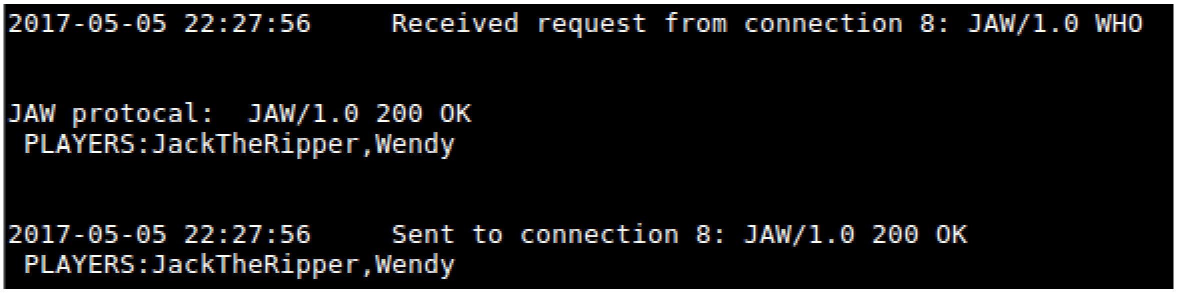
```
2017-05-05 22:26:01     Received request from connection 8: JAW/1.0 WHO


JAW protocal:  JAW/1.0 200 OK
 PLAYERS:


2017-05-05 22:26:01     Sent to connection 8: JAW/1.0 200 OK
 PLAYERS:
```

- Players online (from a free player)



```
2017-05-05 22:27:39     Received request from connection 5: JAW/1.0 WHO


JAW protocal:  JAW/1.0 200 OK
 PLAYERS:JackTheRipper


2017-05-05 22:27:39     Sent to connection 5: JAW/1.0 200 OK
 PLAYERS:JackTheRipper
```

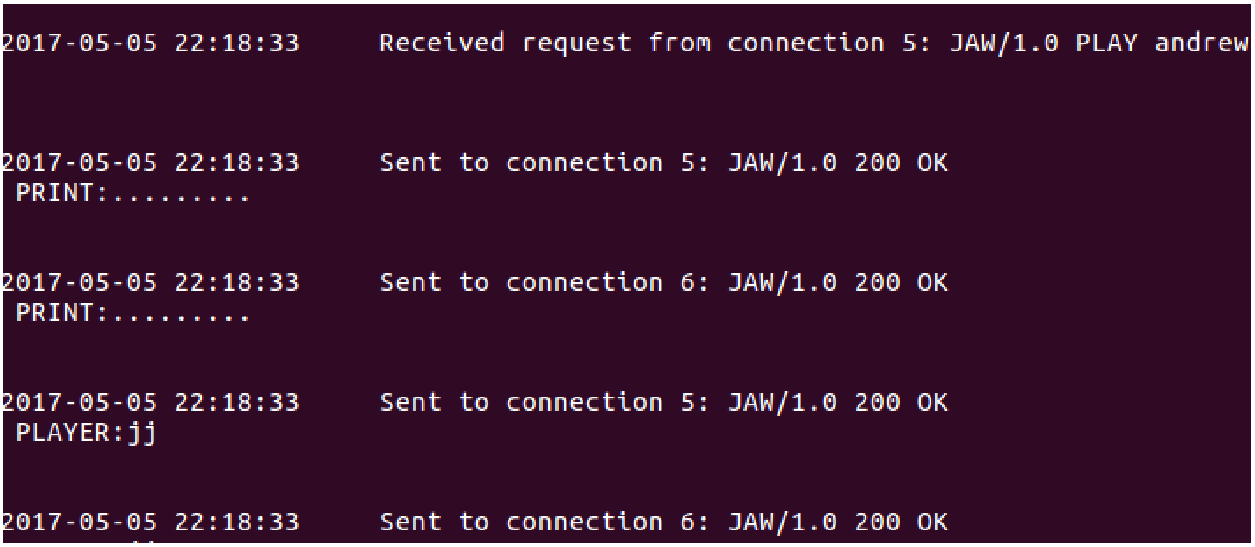
- Players online (from a in-game player)



```
2017-05-05 22:27:56     Received request from connection 8: JAW/1.0 WHO


JAW protocal:  JAW/1.0 200 OK
 PLAYERS:JackTheRipper,Wendy


2017-05-05 22:27:56     Sent to connection 8: JAW/1.0 200 OK
 PLAYERS:JackTheRipper,Wendy
```

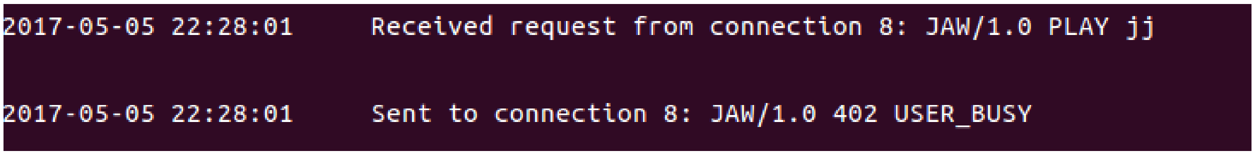
Playing Specific Player

- ❑ MultiPlayer/Server
- User is available



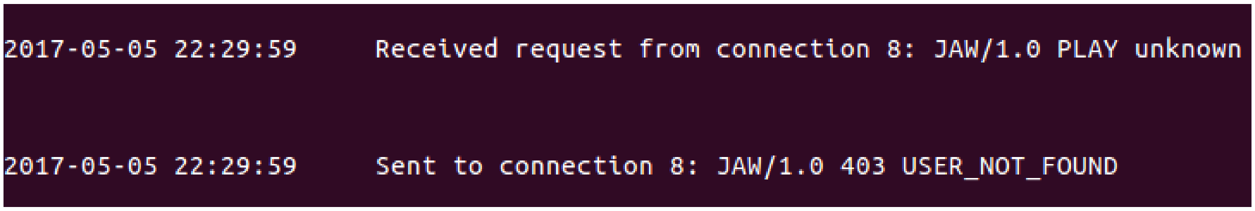
```
2017-05-05 22:18:33     Received request from connection 5: JAW/1.0 PLAY andrew


2017-05-05 22:18:33     Sent to connection 5: JAW/1.0 200 OK
 PRINT:.........


2017-05-05 22:18:33     Sent to connection 6: JAW/1.0 200 OK
 PRINT:.........


2017-05-05 22:18:33     Sent to connection 5: JAW/1.0 200 OK
 PLAYER:jj


2017-05-05 22:18:33     Sent to connection 6: JAW/1.0 200 OK
```

- User is busy



```
2017-05-05 22:28:01     Received request from connection 8: JAW/1.0 PLAY jj


2017-05-05 22:28:01     Sent to connection 8: JAW/1.0 402 USER_BUSY
```

- User is does not exist



```
2017-05-05 22:29:59     Received request from connection 8: JAW/1.0 PLAY unknown


2017-05-05 22:29:59     Sent to connection 8: JAW/1.0 403 USER_NOT_FOUND
```