



# Java Battleship

Brad Berezowski, Ryan Huber | COMP 258 | March 1<sup>st</sup>, 2019

## Game Description

The Client/Server Chat application has been modified to include a UI for the game called Battleship. The objective of the game is to be the first person to eliminate all your opponent's ships. When the game begins, two preset Battleship boards are assigned to each player (player one and player two). Each player takes turns guessing the coordinates of where they think their opponent has a ship placed. The first player that guesses all the correct coordinates, wins the game.

## User Interface Design

This game is set up in the Client/Server Chat UI that was built in the Framework Labs. It has been modified and extended to include:

- An additional JTextArea (on the right side of the UI).
- Three additional buttons (Quit Game, Battle Ship and Attack).
- Two labels and textboxes (Row and Column).

Figure 1. Initial log in screen and after logging in

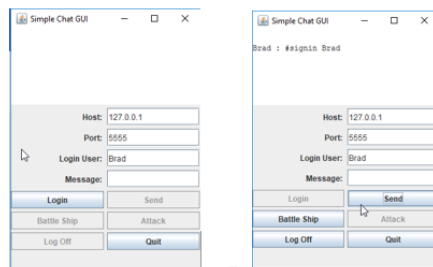
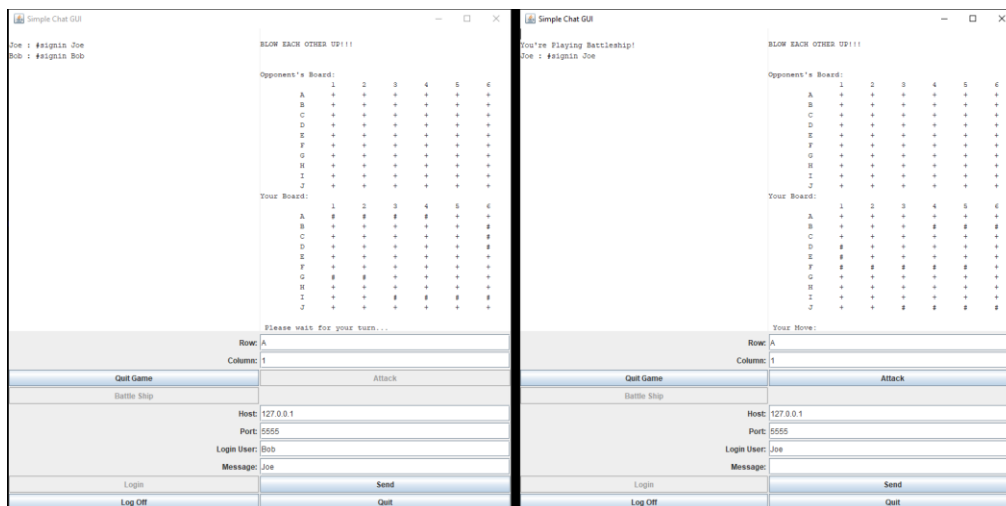


Figure 2. When Battle Ship Button is clicked with a valid user



On application load, the Battleship game components are disabled and when the user logs in the Battle Ship button is enabled, as illustrated in Figure 1. Once the user selects the “Battle Ship” button with a legitimate opponent’s name (opponent must be logged in) in the “Message” textbox, the game components will be enabled, as shown in Figure 2. More on this is covered in the Game Instructions section

## Game Instructions

- Start the client application (while EchoServer is running).
- Login the user.
- Type the user’s name (who you would like to challenge) in the “Message” textbox and select “Battleship”.
- The JTextArea will print your opponent’s board (without showing his/her ships) and your board (with all your ships preset).
- When it’s your turn, the attack button enable and you can then enter the row and column that you would like to attack in the correct textboxes.
- The opponent’s board will display an “X” if you hit a ship or an “O” if you missed.
- Your board will display your ships “#” and where your opponent is attacking (“X” is a hit, “O” is a miss).
- If you would like to quit the game early, select the “Quit Game” button.
- Once a player eliminates all the ships first, it will display who won/lost and it will exit out of the game UI and reset it back to the default UI for the chat application.

## Major Requirements

| Requirement                                  | Implemented | Feedback  |
|--|-------------|---|
| Extend GUI for the game                      | YES         | Created a series of buttons to execute the appropriate game commands, an additional JTextArea to display the game, additional labels and textboxes for the user to enter specific commands (row, column). |
| Transmission of an object across the network | YES         | A game object is created when a command (#battleship [player]) is sent to the server. This object and the opponent is sent through an envelope to begin the game.   |
| Project Proposal                             | YES         | Sent to the instructor and approved.  |
| UML Diagram                                  | YES         | Compiled and attached to this report  |
| Documentation                                | YES         | Enclosed in this report.  |

## Minor Requirements

| Requirement   | Implemented | Feedback   |
|---|-------------|--|
| Player cannot attack the opponent's board when it is not their turn                                     | YES         | When it is player one's turn, player two's attack button is disabled so he/she can't attack while it's not their turn. When it is their turn, the attack button is enabled.  |
| Player's boards are set up by the players   | NO          | Due to time constraints, we were unable to implement this. Instead, we set the ships on the board manually.  |
| Notify Player when it's their turn  | YES         | A message will display in the game's JTextArea notifying him/her if it's their turn or not. Also, the attack button will be enabled when it's their turn.  |
| Display the opponent's (without showing their ships) board and the player's board (showing their ships) | YES         | The opponent's board will be printed in the JTextArea. This board will only show where the player is attacking (whether they hit or miss). The player's board will show where the player's ships are and where the opponent is attacking (where they hit or miss).   |
| Validation for user input on rows and columns   | YES         | Added validation so the user can only enter a correct row or column. If an incorrect value is entered, the field is turned red and a message is displayed.   |
| Allow the players to be able to still communicate in the chat application                               | YES         | Both players can send messages to each other, the same as before. While the game is running, messages will display in the left JTextArea, while the right JTextArea will only display the game. Note: You can only talk to each other if you are in the same room or via PM if you are in a different room, as you can play each other across different rooms. |
| Player can challenge a person regardless of which room they are in                                      | YES         | As mentioned above you can challenge but can only talk to each other via PM when in a different room.  |
| Check to see if a player has attacked the coordinates (row, column) before                              | NO          | Due to time constraints and lack of knowledge, we were unable to implement this.   |
| Game end scenario   | YES         | Once a player has eliminated all their opponent's ships first, the game will exit, and a message will display on both their UIs to notify them whether they won or lost.   |

## User Validation when Battle Ship is Clicked

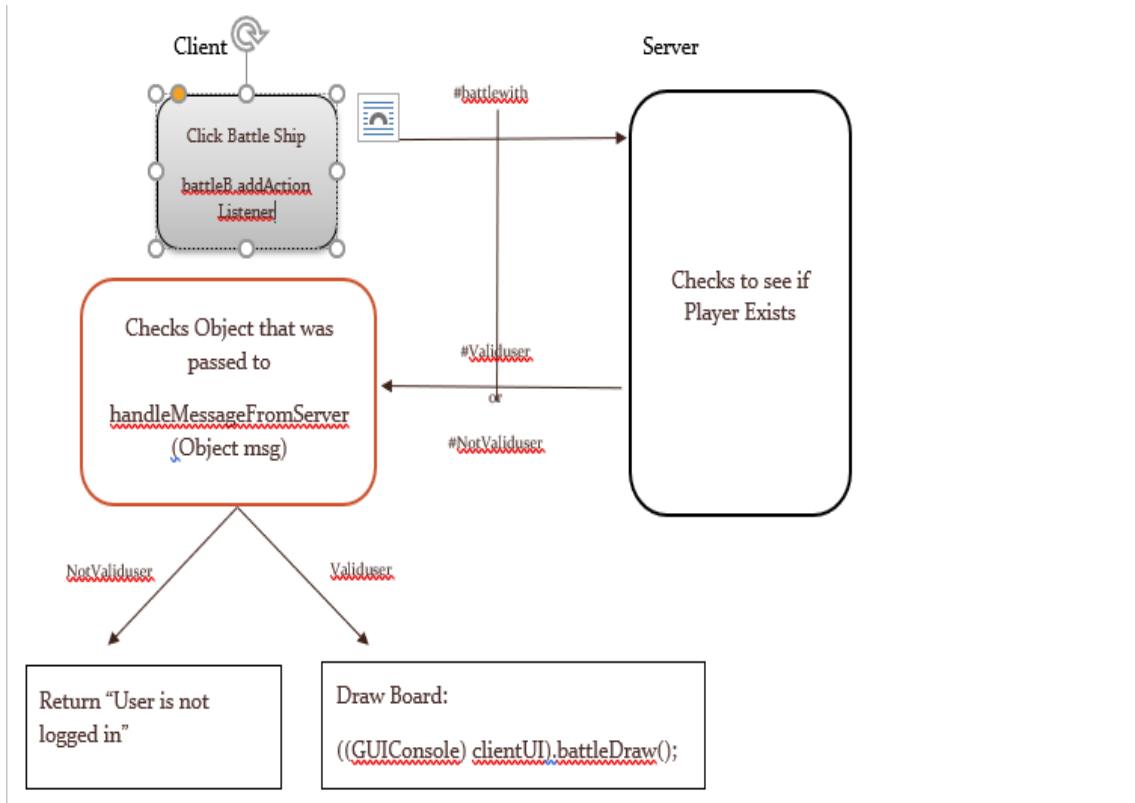


Figure 3. User doesn't exist message

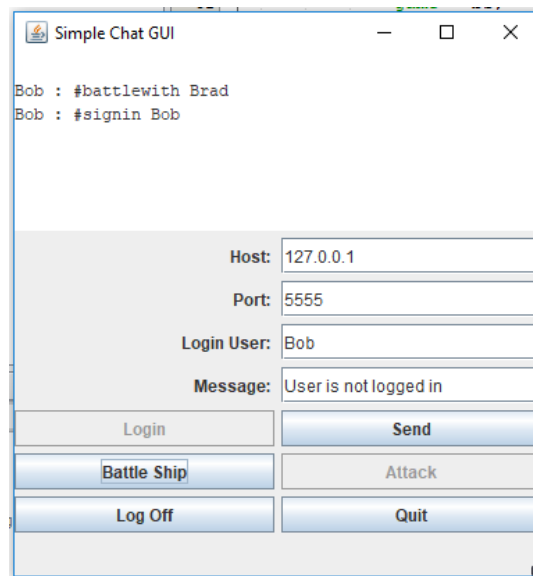


Figure 4 and 5 illustrate what happens when a user inputs an incorrect Row or Column and then hit the Attack Button. The Row validation verifies that the Row Value is between and including A and J either upper or lower. The Column Validation verifies that the Column Value is a numeric value between and including 1 and 6

Figure 4. Row Validation

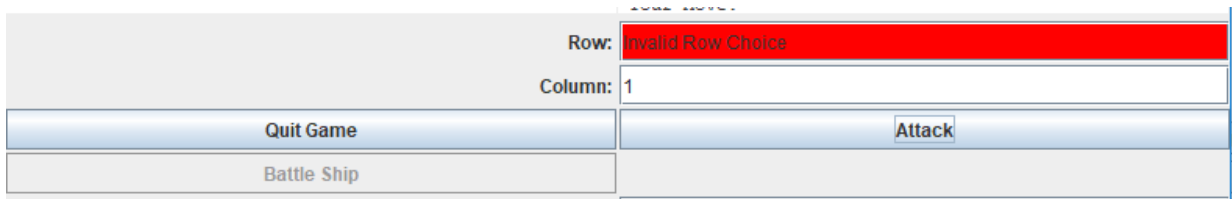


Figure 5. Column Validation

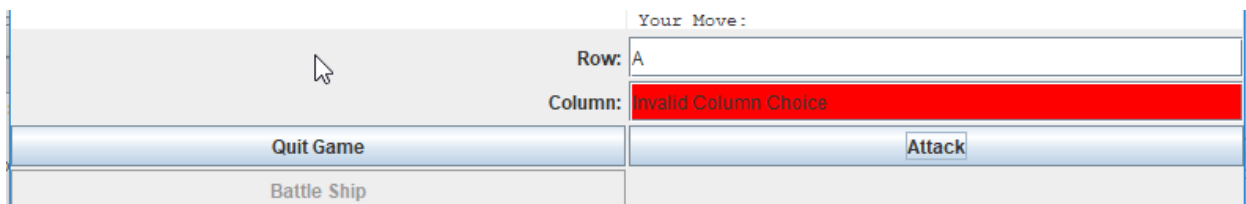


Figure 6 displays the Win and Lose scenario once all shups have been destroyed on one side. The Board is closed and the GUI is reverted back to the original chat window

Figure 6. Win/Lose Screen

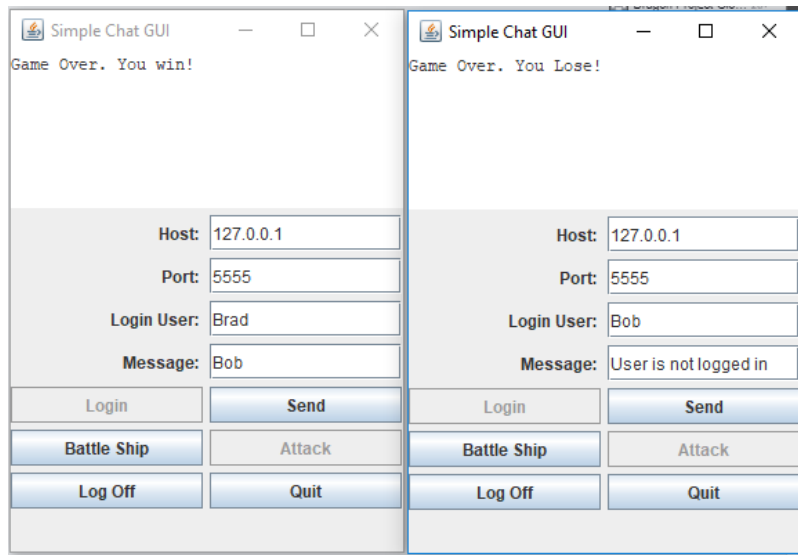
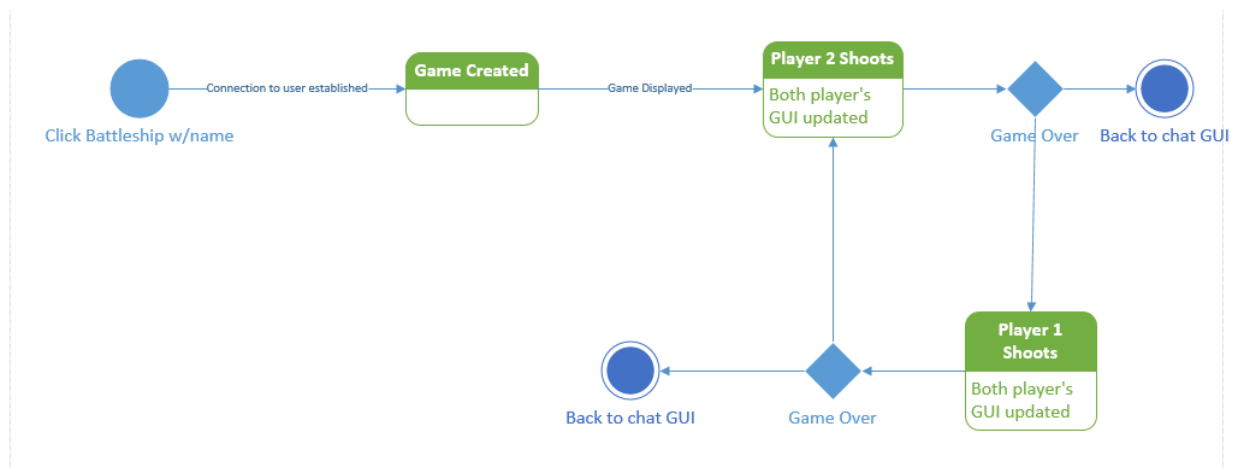
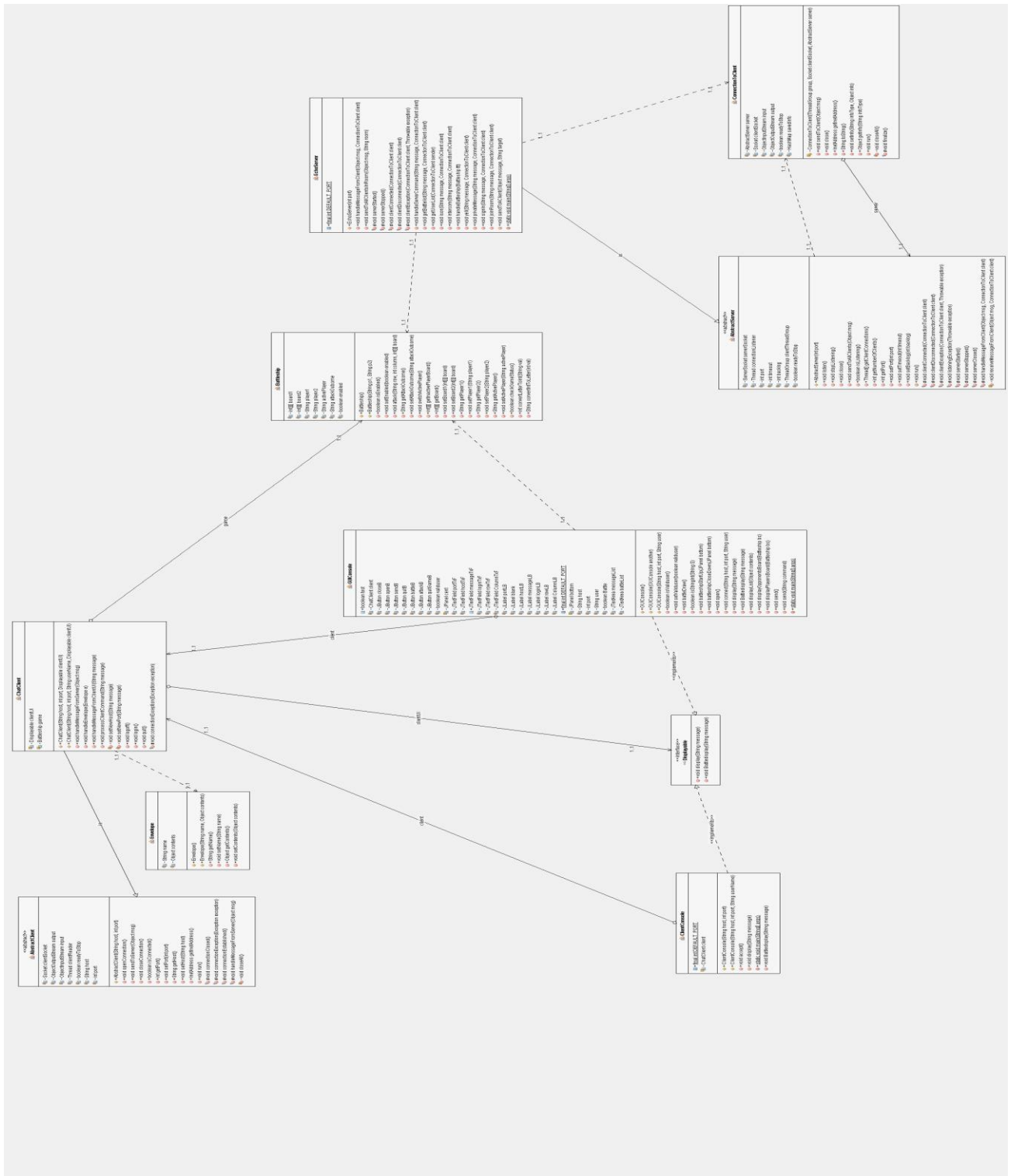


Figure 7. Workflow Diagram of the Game



## UML Diagram

Figure 8. UML Diagram for BattleShip



### Changes from the Original Client/Server Application



- Updated the GUIConsole to include and expanded UI for the game. This includes 3 new buttons (Attack, Quit Game and Battleship), one additional JTextArea, 2 labels and 2 textboxes (for row and column).
- Created a Battleship class that sets the boards (populates them with ships) and the players. This class also includes:
  - i. Some helper methods (for getting a number value from a char and getting a char from a number value).
  - ii. A method that handles each player's attack.
  - iii. A method that checks the status of the game (makes sure a player hasn't eliminated all their opponent's ships yet).
  - iv. Validation methods (for user input, active user, etc.).
  - v. A method for switching the active player (the player who's turn it is).
  - vi. A method for locating the active player (the player who's turn it is).
  - vii. A method for locating the inactive player (the player who's turn it isn't).
- The GUIConsole class includes a few additional methods for:
  - i. Displaying the player's board in the JTextArea.
  - ii. Displaying the opponent's board in the JTextArea.
  - iii. Setting up the UI when the game is initialized.
  - iv. Closing the UI for when the game is exited.
  - v. Click events for the new buttons.
- The EchoServer class includes:
  - i. Additional if statements within the "handleMessageFromClient" that handle the Envelope containing the battleship game.
  - ii. A "handleBattleship" method that sends the boards to the players.
- The ChatClient class includes:
  - i. Additional if statement within the "processClientCommand" method that determines if the message being sent contains "#battleship". If it contains "#battleship", it sets up the boards (by creating an instance of the Battleship class), packages the boards into an Envelope and sends it to the server.
  - ii. Additional if statement within the "processClientCommand" method that determines if the message being sent contains "#attack". If it contains "#attack", it trims the message to assign the coordinates to variables and initiates the attack method within the Battleship class (by creating an instance of the Battleship class).