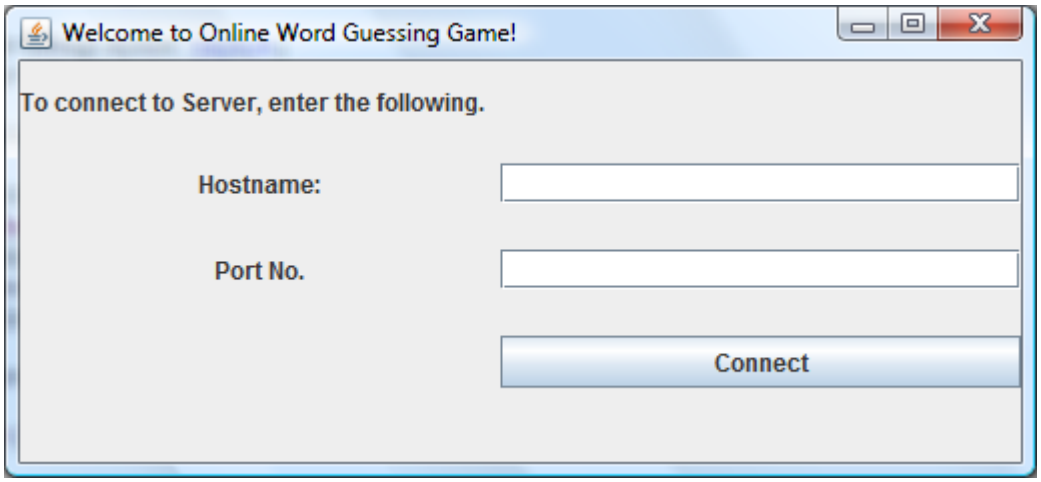
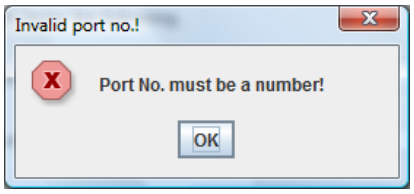
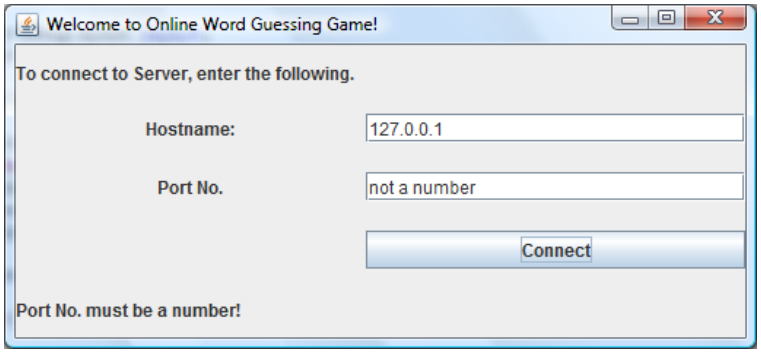
Assignment 3 Report

ConnectionGUI

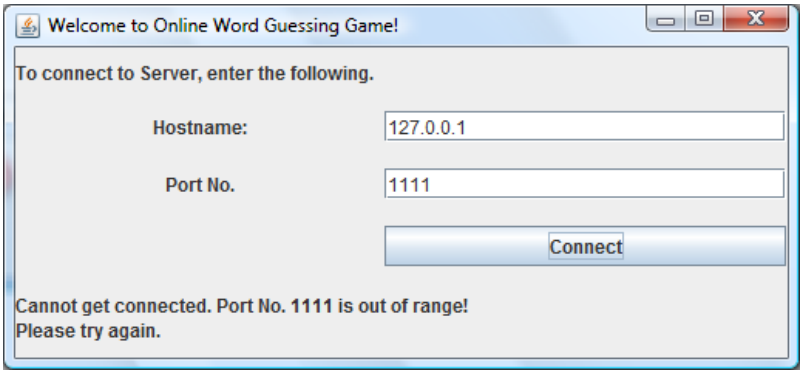


If port no. is not a number

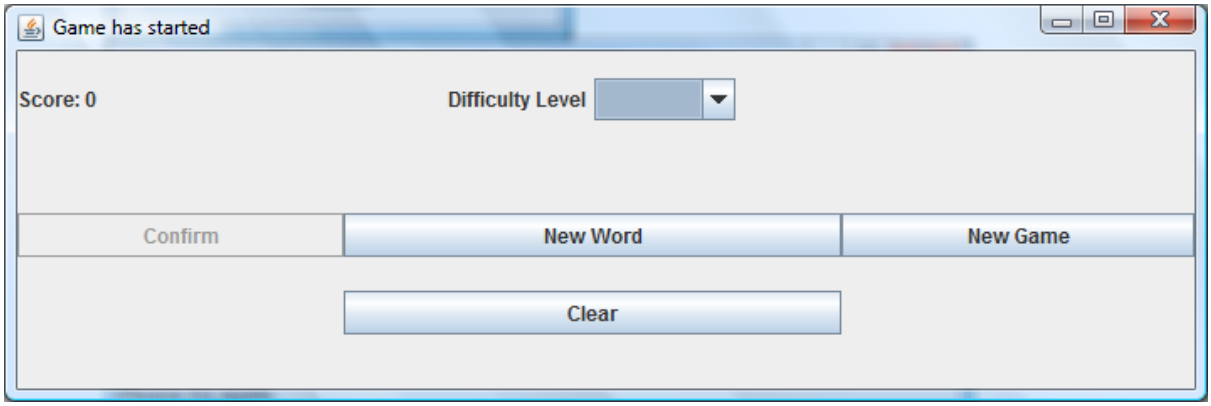




If cannot get connected

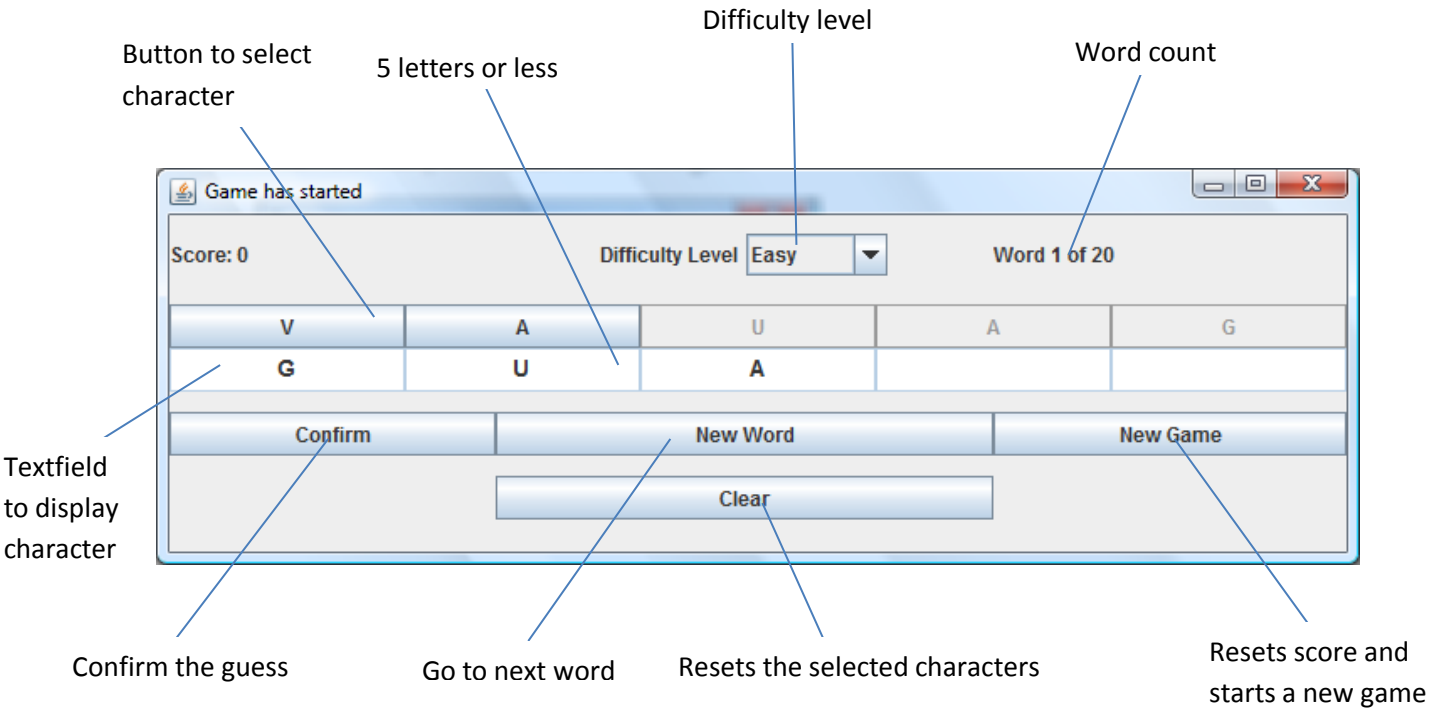


GuessAWordGUI

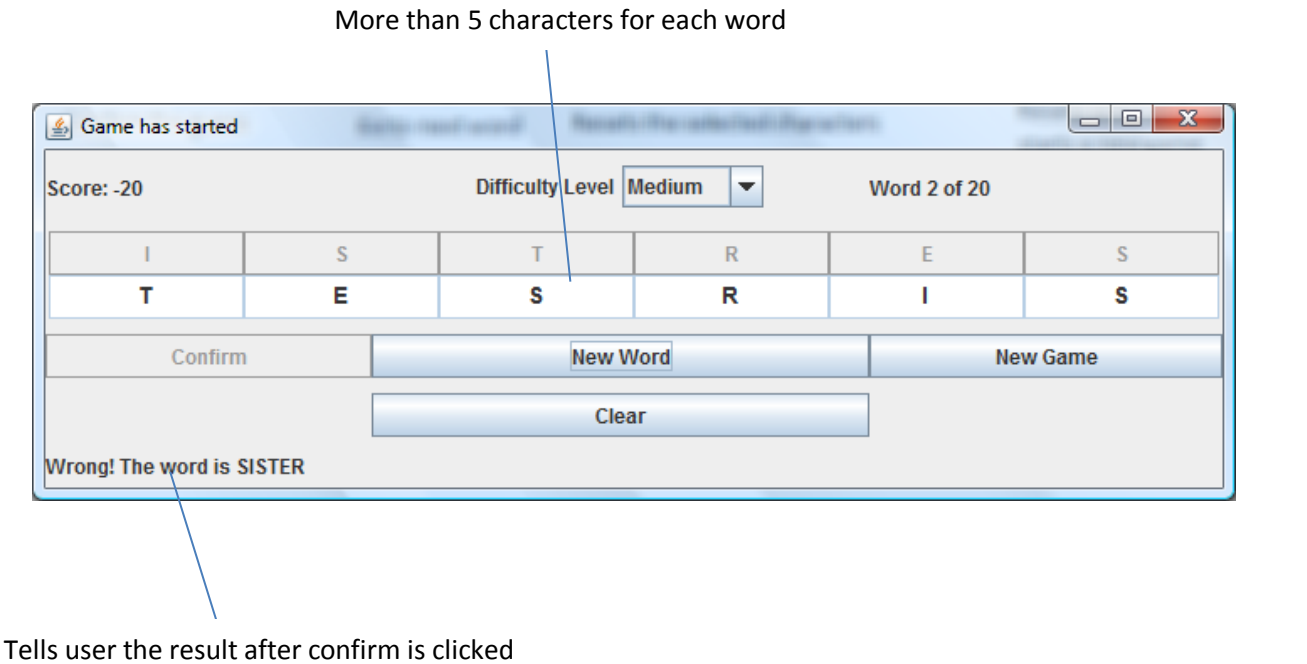


If start a difficulty level not set before new game

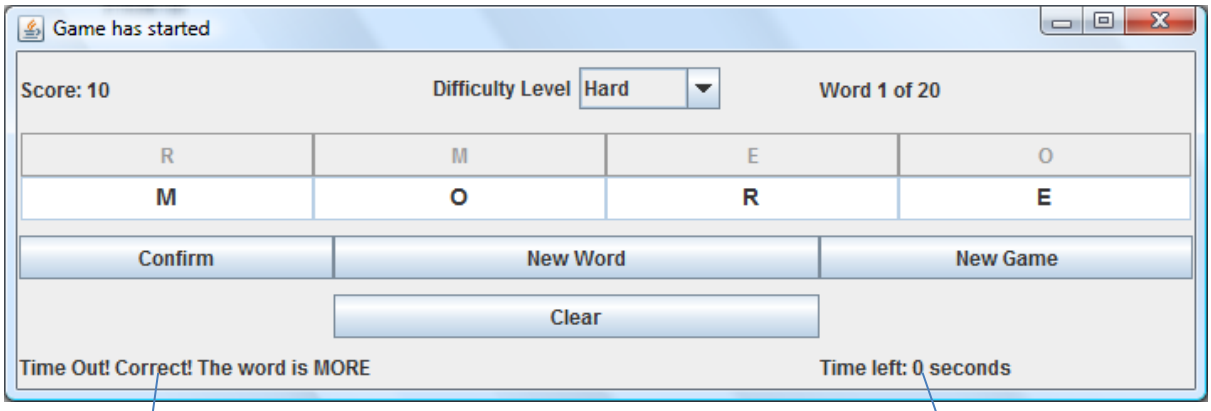
Easy mode



Medium



Hard

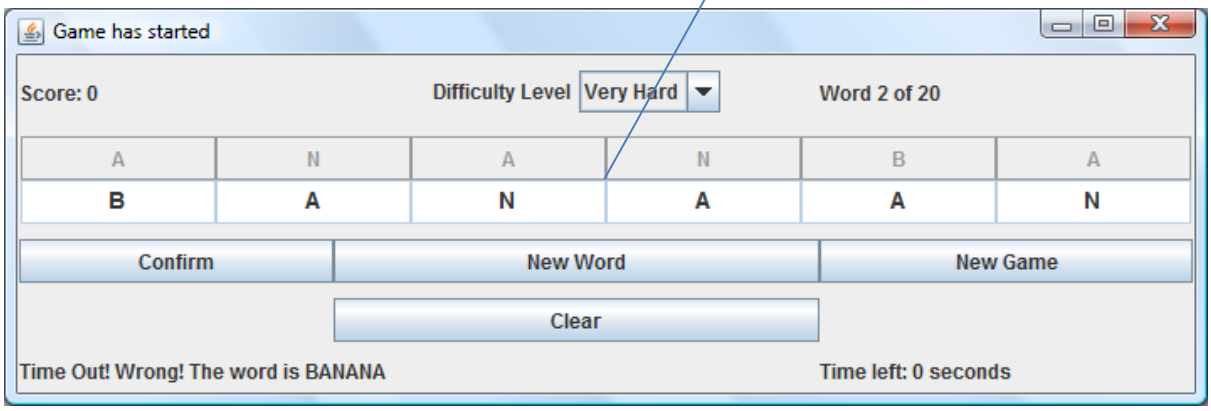


Displays result. (In this case it's a time out)

Displays number of seconds left before time out

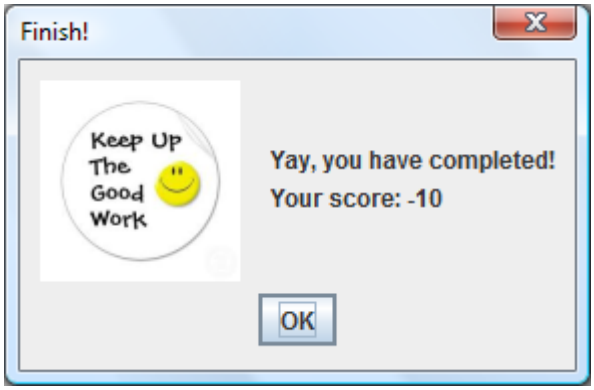
Very Hard

More than 5 characters for each word

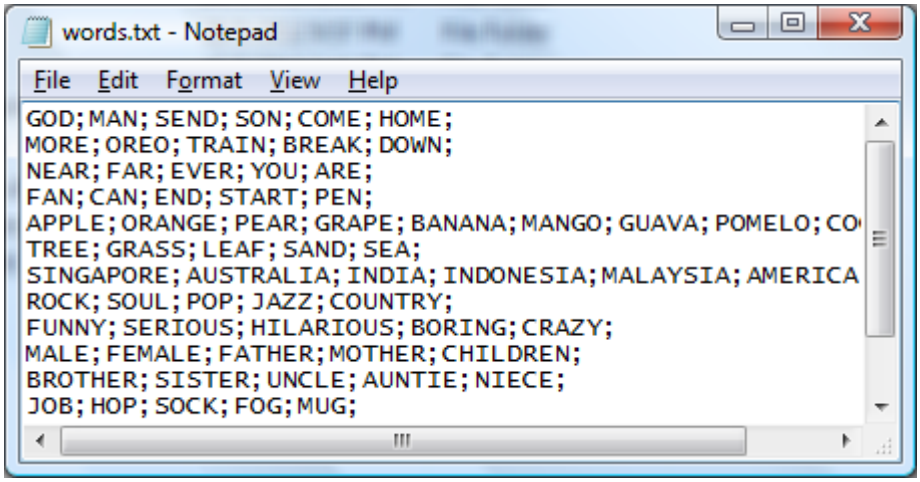


If player guessed all the words

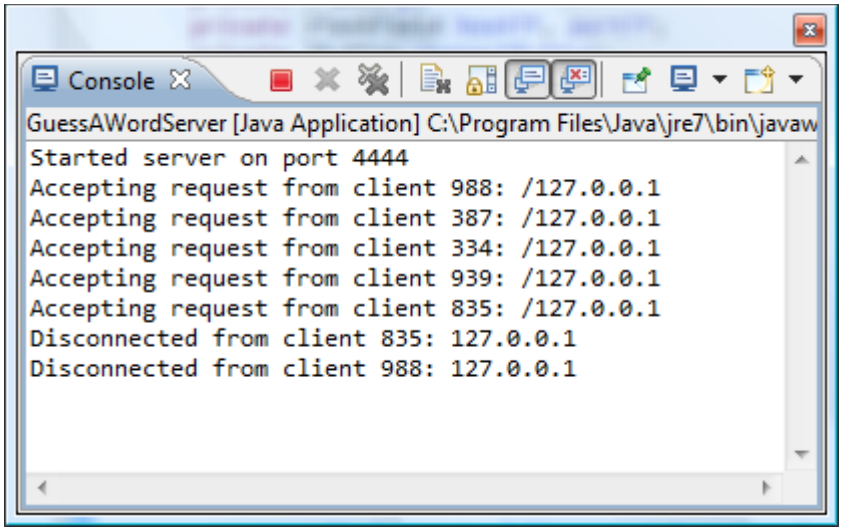
If player did not guess all correctly



words.txt



Sample of server output



Extra Features

- Displays current word number
- Combo box to set difficulty level
- Difficulty level can be set anytime and the game will be reset
- User can type instead of entering the characters by clicking the buttons
- Clear button to clear the selected characters
- Time limit is set to 20 seconds for each word in hard and very hard mode
- User can press backspace to clear characters entered in the text field

Design of GUI

ConnectionGUI

- Used a GridBagLayout
- A label to display instructions
- A label to label the text field for entering hostname
- A label to label the text field for entering port number
- A text field for entering hostname
- A text field for entering port number
- A connect button to start connection
- A label at the bottom to display error messages

GuessAWordGUI

- Used a GridBagLayout
- A label to display score
- A label to label the combo box for setting difficulty level
- A label to display the current word number (eg. Word n of 20)
- A subpanel (Word Panel) to display each word
    - A row of buttons to display each character of the scrambled word
    - A row of text fields to display the user's guess
- A confirm button to submit the guessed word to the server
- A new word button to request for a new word from the server
- A new game button to request for a new game from the server
- A clear button to reset the user's guess (if not yet submitted)
- A label at the bottom left to display the result (Correct, Wrong, Time out, answer)
- A label at the bottom right to display time left until time out (for Hard and Very Hard mode)

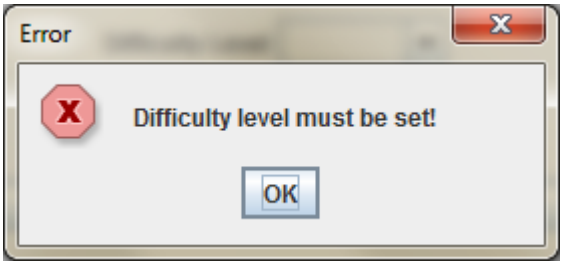
Error Handling

ConnectionGUI

- Port number entered is not a number
- Unable to establish connection to server

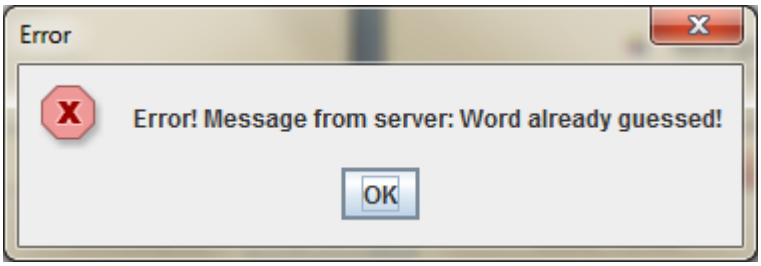
GuessAWordGUI

- New word requested before game start



- New game requested before difficulty level set

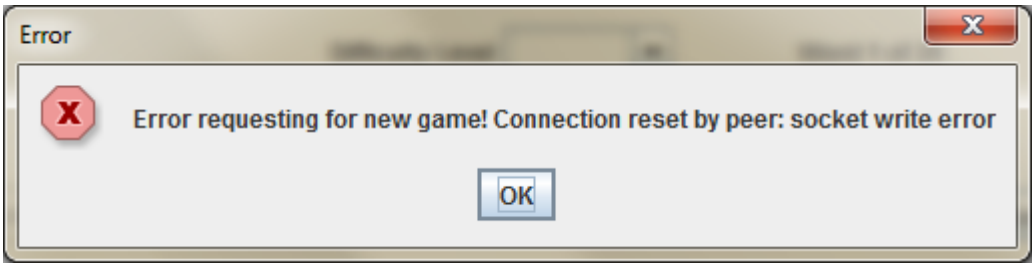
- Same word submitted as the scrambled word



- If confirm button is not disabled upon click, server also can handle a word submitted more once for the same question

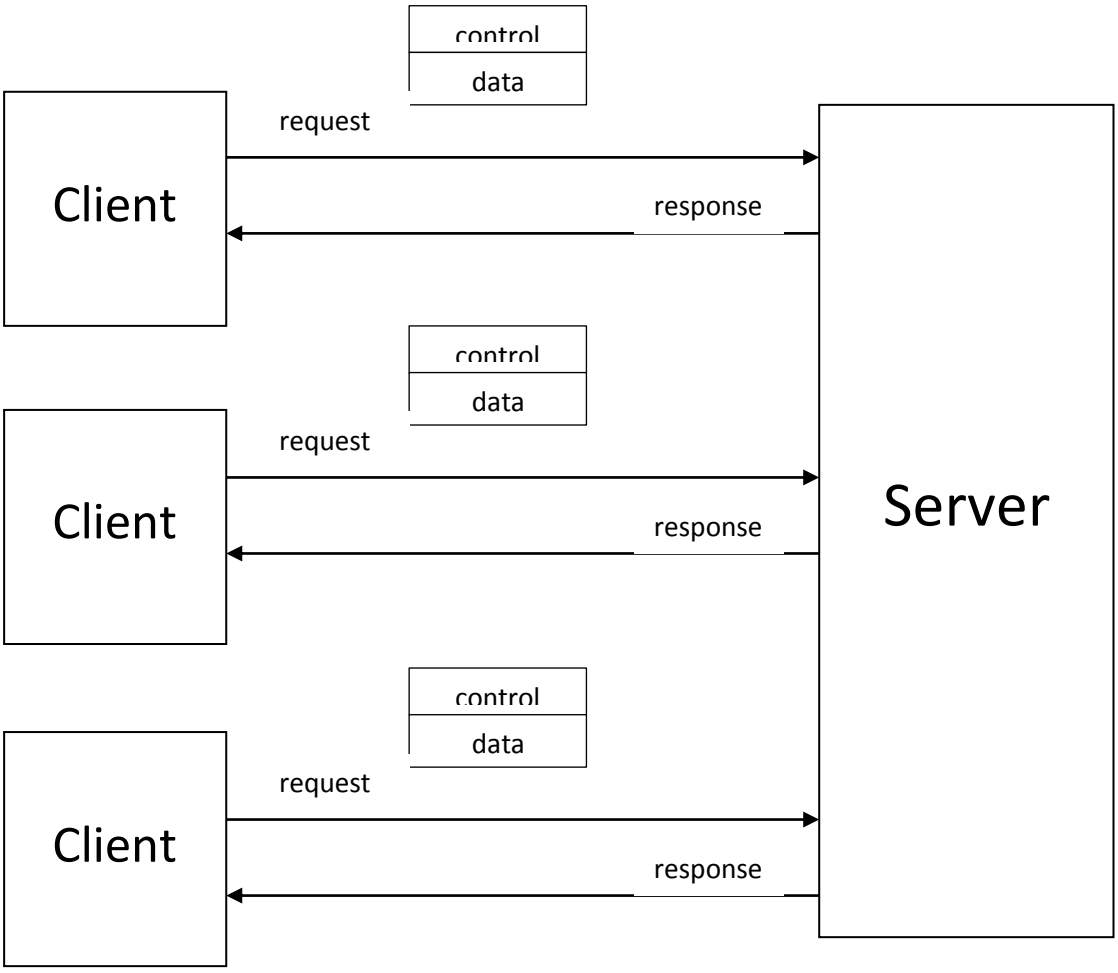
- Server down

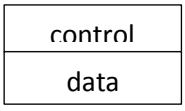

Error
Error requesting for new word! Connection reset by peer: socket write error
OK


Error
Error requesting for new game! Connection reset by peer: socket write error
OK


Error
Unable to send exit message Connection reset by peer: socket write error
OK

Diagram / Illustrations of program design





Packet Object sent between client and server

Summary of implementation of each module in my program

Client

ConnectionGUI

- ConnectionListener class
  - Added to the connect button
  - Checks if port number entered is an integer
    - If port number is not a number, display error message and return
  - Creates a socket otherwise
    - If socket created unsuccessfully, display error message and return
  - Hides the current frame and calls the GuessAWordGUI frame

GuessAWordGUI (what is to be done in the correct order)

- Creates input and output streams to server

- User chooses a difficulty level from the combo box, triggers DifficultyLevelListener
  - DifficultyLevelListener class
    - Attached to the combo box
    - Contains Strings “Easy”, “Medium”, “Hard” and “Very Hard”
    - When user selects, client sends the difficulty level over to the server
    - If level is “Easy” or “Medium”, hides the countdown label (displaying the time left to answer the question)
    - If level is “Hard” or “Very Hard”, displays the countdown label
    - Resets/Clears all the labels, the timer and the displayed word

- User presses New Game button, triggers NewGameListener
  - NewGameListener class
    - Sets score to 0 and display score label
    - Sets question number to 1 and display question number label
    - Sets message label to empty string
    - Sends request for new game to the server
    - If server response with an invalid message, display error message and return
    - Sets the given scrambled word and displays it on the word panel
    - Word panel: Displays the scrambled words as a row of buttons and the user guess as a row of text fields (not editable)
    - If difficulty level is “Hard” or “Very Hard”
      - Initializes the number of seconds left (in this case 20)
      - Starts the timer
    - Enables the confirm button

- User tries to guess the word but clicking on the buttons of the scrambled word, triggers CharListener
    - CharListener class
        - Displays the character of the selected button in leftmost empty text field (done using an index to keep track of the next empty text field location)
        - Disable the button clicked

- Alternatively, user can just enter and remove letters using the keyboard, triggering CharKeyListener
    - CharKeyListener class
        - If backspace is entered and there is at least 1 character in the row of text fields
            - Removes the right most character from the text fields
            - Enables a button that contains the character
        - Else
            - If uppercase of the character key entered is equal to one of the enabled buttons
                - Disables one of the enabled button containing the uppercase of the character key entered
                - Fills the leftmost empty space with the character

- If users thinks he/she entered wrongly and hasn't pressed confirm and wants to reset the entered guess, he/she presses the Clear button, triggering the ClearListener
    - ClearListener class
        - Enables all the scrambled word buttons again
        - Clears all the text fields (user guess)
        - Resets current index to 0

- User submits guess by pressing the Confirm button, triggering the ConfirmListener
    - ConfirmListener class
        - Disables the confirm button upon click
        - If difficulty level is "Hard" or "Very Hard", stops the timer
        - Sends guess to server
        - Receives response from server
        - If server detects the guess word is similar to the scrambled word
            - Displays an error message
            - If difficulty level is "Hard" or "Very Hard", starts the timer again
            - Resets the word panel
            - Enables the confirm button
            - Return

- If server detects valid guess
    - Tells client whether guess is correct or wrong
    - Client displays message telling user if guess is correct or wrong and the correct answer
    - Client request for score update
    - Server response with updated score
    - Client updates score and displays score on the score label
    - Client ask server if game is finished (20th question)
    - Server replies client whether game has ended
    - If game ended checks the user score
        - If user has full score (answered all correctly), displays a congratulation message
        - Else displays a "keep it up" message

- User is done with the current word and want to move on to the next word, he/she clicks the New Word button, triggering the NewWordListener
    - NewWordListener class
        - Message label displaying the result (correct or wrong guess) is cleared
        - Request for a new word from server
        - Receives new word from server
        - Sets the word and updates the word panel
        - If difficulty level is "Hard" or "Very Hard", resets time left to 20 seconds and start the countdown timer
        - Increments the current question number and updates its label
        - Enables confirm button

- User selects Hard or Very Hard mode while guessing a word, a timer is counting down, triggering CountDownListener every second
    - CountDownListener
        - Decrements the number of seconds left and updates the countdown label
        - If number of seconds left equals zero (time out)
            - Submits whatever word on the text fields to server
            - Server replies whether correct or wrong
            - Displays time out message on the message label
            - Request for updated score
            - Receives updated score
            - Client ask server if game is finished (20th question)
            - Server replies client whether game has ended
            - If game ended checks the user score
                - If user has full score (answered all correctly), displays a congratulation message
                - Else displays a "keep it up" message
            - Sets the score and displays it in the score label

- User wants to exit game, he/she closes the window frame, triggering the ExitListener
    - ExitListener class
        - Sends request to disconnect from server
        - Socket is closed, server and client are disconnected

Server

- Creates a server socket using port number 4444
- Loads words from text file into 2 Array List, one for easy words (<= 5 characters), one for hard words (> 5 characters)
- If not enough words for each (< 20 words), prints and error message and terminates
- Goes into an infinite loop waiting to accept a client
- When accepted a client
    - Generates a unique client ID
    - Creates and start a new thread to handle the client
    - Prints message to show client accepted and its ID and IP address
    - Adds the client ID to the current clients list

HandleClient thread

- Creates input/output streams to client
- Waits to receives a request from client in an infinite loop
- If request for New Game
    - If difficulty level not yet set
        - Sends back an error message
        - Skips the rest of the loop and wait for another request
    - Sets question number and score to 0
    - Gets the indexes of the 20 different words to play
    - Scrambles the first word and sends to client
    - Sets Boolean game started to true and word guessed to false
- If request for New Word
    - If difficulty level not chosen or game not yet started or word is the last word
        - Sends back an error message
        - Skips the rest of the loop and wait for another request
    - Increments question number
    - Scrambles the next word and sends it to client
    - Sets Boolean word guessed to false
- If receives a guess from client
    - If difficulty level not chosen or game not yet started or word already guessed
        - Sends back an error message
        - Skips the rest of the loop and wait for another request
    - If received same word as the scrambled word

        - Sends back a message telling client to resubmit
        - Skips the rest of the loop and wait for another request
    - Checks the guess and reply the client whether guess is correct or wrong
    - Updates score
    - Receives request for score
    - Sends back the updated score
    - Receives request from client asking if it is the last question
    - If question number is less than 19, replies “No”
    - Else replies “Yes” and set Boolean game started to false (game ended)
- If request to set difficulty level
    - Resets score and question number to 0
    - Sets game started to false
    - If Easy or Hard mode selected, use the easy words
    - If Medium or Very Hard mode selected, use the hard words
- If client sends Time Out message to server
    - Checks guess and reply the client whether the guess is correct
    - Updates score
    - Set word guessed to true
    - Receives request for updated score
    - Sends updated score
    - Receives request from client asking if it is the last question
    - If question number is less than 19, replies “No”
    - Else replies “Yes” and set Boolean game started to false (game ended)
- If received request to exit
    - Breaks the infinite loop
    - Closes the socket and disconnects from client
    - Remove client ID from the current client list

Reflections on program development

Difficulties faced

Being unfamiliar with the concept of server and client and socket connection in java, I had to try spend some time reading up on the note as well as on the web. However grasping this concept in java seems relatively easy as java objects are quite simple to understand. The time constraint we face was a challenge as it was a considerably large project for a single person given the time to complete. Initially I had a problem of the system hanging when I first attempted to connect to server. However this was solved with the help of google as I realized that the ObjectOutputStream must be declared before the ObjectInputStream.

http://www.coderanch.com/t/274824/Streams/java/Hanging-ObjectInputStream

Another challenge faced was that of the scrambled word display – the row of buttons and the row of text fields below it. This is a dynamic component that can change the number of components it contains. However after learning about the removeAll() method of the JPanel class, it is solved rather easily. I also wanted to server to handle error messages instead of just the client disabling the buttons, hence more coding was required and Boolean variables like gameStarted and wordGuessed were created. Identifying relevant components to be visible or hide upon each button click was also a little hassle. Matching the client request code to the server side response code was important in development of the program.

What could have been done better / possible enhancements in future

I had wanted to store a high score table in the server but could not do so due to time constraint. One thing I wanted to try also was the automatically attempt to reconnect to the server from the client side should they be accidentally disconnected in the middle of a game. The state of the game would have to be saved. However I do not know if this is possible. Also the GUI could be decorated with more colors and pictures instead of the default design.

What I have learnt

I have now a much better understanding of client-server concept in Java. I have also learnt the use of input and output streams, in particular the ObjectInputStream and ObjectOutputStream class. After this, I am more familiar with the kind of exceptions certain operations can throw, such as IOException, ClassNotFoundException and NumberFormatException. Although the time given was short, I have learn quite a few things in the process of doing this assignment.