CS408 - PROJECT STEP 1 – 2018/19 FALL TERM

Our system creates a game where two players play a quiz game. Our aim was to create a working game between two for now. We created safe-threads through the use of delegate and invoke methods.

Server starts a safe-thread of “Accept” and from Accept Thread, we allow clients to connect to server. Accept always runs as long as server is running. After accepting a client, client’s socket” is added to “clientSocket” list and name of the client to “nameList” list (all characters are lowercased to compare two strings easier). Furthermore, a safe “Receive” thread starts running. In receive thread, if there are at least 2 clients connected to the server, the game starts. If there are not enough players, lone player waits until someone joins to the server. After player number is enough the game start.

Our game is mainly designed according to two player game. However, we prepared our ideas for how to reshape it to multiple players (we simply couldn’t implement it yet). At the very beginning of the game, we decide which player is going to play first. Since it should be the one which is front in alphabetical order, we compare both strings and set the preceding one as `first`, latter one as `second`. These terms help us to get the socket from list of client sockets. However, after finishing round, order should exchange so we check the round number to decide who starts first. After order is done, assuming first player is first, second player is second, server tells first player to ask a question. First player sends a question and answer together. Every time a question is send from any player, that question MUST have ‘?’ because, after server receiving the question, it splits the text according to ‘?’. Takes everything before ‘?’ and creates the question from it. The part after ‘?’ is considered as answer. After implementing first’s player question and answer, server asks for second player’s question and answer. Server does the same thing as it did on first player. Split, take first part as question, second part as answer. When both questions are ready, server sends second player’s question to first player and waits for a response. After receiving an answer from first player for second player’s question, it reveals if the answer was correct or not. Both players are able to see if the first player did correct or wrong. Next, second player’s answer to first player’s question is expected. When received, it compares the answer of second player with the real answer. Reveals if second player did right or wrong. Also, before checking answers, all received answer information, including the real answers are lowercased to precisely compare strings. Round number is increased, order is changed and play again!

These were the things we accomplished. However, there are several things missing in our project’s first part. We cannot handle the situation which when a server exits abruptly, GUI freezes. In addition, since we’re gathering the questions and answers in the format of “question?answer”, if user enters a question without question mark, the program is unable to detect and separate the question and the answer properly, so we expect users to put a “?” after question is ready, and answer afterwards.

Atilla Alpay Nalçacı

Deniz Ademoğlu