Network Game Skeleton

Documentation

How to compile (The project is already compiled):

…\SKJ_Project_3\src>javac GameClient.java

…\SKJ_Project_3\src>javac GameServer.java

How to run:

1. Lauch GameServer:
   a. **…\SKJ_Project_3\src>java GameServer 5000**
2. Lauch several GameClients:
   a. **…\SKJ_Project_3\src>java GameClient localhost 5000**
3. Use PLAY command on each of the clients.

How it works:

For each new client connection, server creates a separate thread and sends back to a client his new ID. PlayerManager class is responsible for assigning IDs and storing references to all players. With each new connection, server checks if there are at least two players ready to play. If so, it creates an instance of a Match (Duel) for those players is a separate thread. Each Match contains only two players. Any time while not playing client can send those commands: **"PLAY", "LIST", "LOGOUT".** After sending a logout message the client will be disconnected from the server.

Game process (It is possible to play the game through CLI):

After some client sends a **"PLAY"** (as a **User Input** message) it waits for the server response till it finds an opponent. From this point each message is being built using MessageBuilder class and has the following format: "playerID:command:data". If the command is YOUR_TURN then the client has an opportunity to make his move and input data in format "x y" (x, y – integers from 0 to 2) separated with space (**User Input**). In case if this place is already taken the client will be asked to input the data once again.

Each cycle (in a Match (Duel) thread):

1. Receive position from the player that currently makes a move and apply it to the game field.
2. Update the current state of the game field on both clients.
3. Check if there is a winner using the Field class and sends the answer to both players if the winner is determined.
4. The same operations but with the opponent.
5. And so on.

Notes:

UDP "Audience" is not implemented.