Communication protocol explaination

Client side

Instances

- **Main:** launches the client, instantiates the client view and the client controller, initiates the connection
- **ClientController** : creates the clientModel passing the view, instantiates the socket for the connection, creates the reading and writing streams from the socket, sends data to the server and handles the server response
- **ClientModel** : Holds relevant client informations such as the playerID given from the server and the gameID
- **ClientView** : Gets the input from the user, builds up the packet according to the input given, notifies the controller, prints data from the server

Client first connection

1. The app class instantiates the ClientView
2. The app class instantiates the ClientController passing the view, the host and the port.
3. The controller creates a ClientModel in the constructor and adds itself to the list of view observers in its constructor
4. The app class calls the `connect` method inside the clientController
5. The client controller creates a socket then creates the inputStream (`in`) and the outputStream (`out`)
6. The client controller creates a new thread passing its own instance and waits for data from the server

Sending messages to the server

1. The user inputs a command into the command line
2. The view updates the controller passing a Packet
3. The clientController sends the packet to the server

Receiving a response

1. The clientController receives an object from the stream and casts it into a packet object
2. The clientController calls the `getAction()` method from the Packet and updates the model
3. The clientController passes the Packet to the view
4. The view calls the `getAction()` method and prints the informations in the payload

Server side

Instances

- **Main** : instantiates a gameController and a serverController, starts the server
- **ServerController** : accepts client connections, creates the clientHandler for each connection, adds the clientHandlers to the clientHandlerController
- **ClientHandlerController** : Holds a list of all the client connections, handles the requests from the clients
- **ClientHandler** : instantiates the socket for the connection, creates the reading and writing streams from the socket, sends data to the server and handles the server response

Handling a request

Requests arrive to a single ClientHandler that forwards them to the ClientHandlerController.

The clientHandlerController check for the player id and game id in the packet and calls the action in the gameController.

When an action is performed usually the view is updated. When a view is updated instead of printing in the consolle they send the data back to the client.