Documento di rete gruppo AM50

Communication protocol

Communication between client and server is based on the exchange of JSON strings serialized and deserialized via GSON library.

Strings are sent via a PrintWriter through the socket output stream and are received via a BufferedReader on the socket input stream constantly listening on a thread.

There are two main objects that are serialized to JSON respectively for messages from the client to the server and from the server to the client.

* Client to Server

Messages from the client to the server holds commands to be executed on the server so the JSON sent is structured as follows:

1. {
3. “command”: String,
4. “value1”: String,
5. “value2”: String
7. }

The command field holds a string mapped on the server with the relative method, value1 and value2 are optional fields used to pass parameters.

* Server to client

Messages from the server to the client are mostly responses or results of a command and they are structured as follows:

1. {
2. “type”:String,
3. “context”:String,
4. “message”:String
5. }

The tipe field indicates the message type (error, confirmation, notify, islands dashboard ecc…), the context field is optional and used when managing errors or confirmations and contains the specific error code to be handled by the client, the message field can be used to send a simple message or store a nested JSON with more information, for example islands or dashboard state.

## Game setup

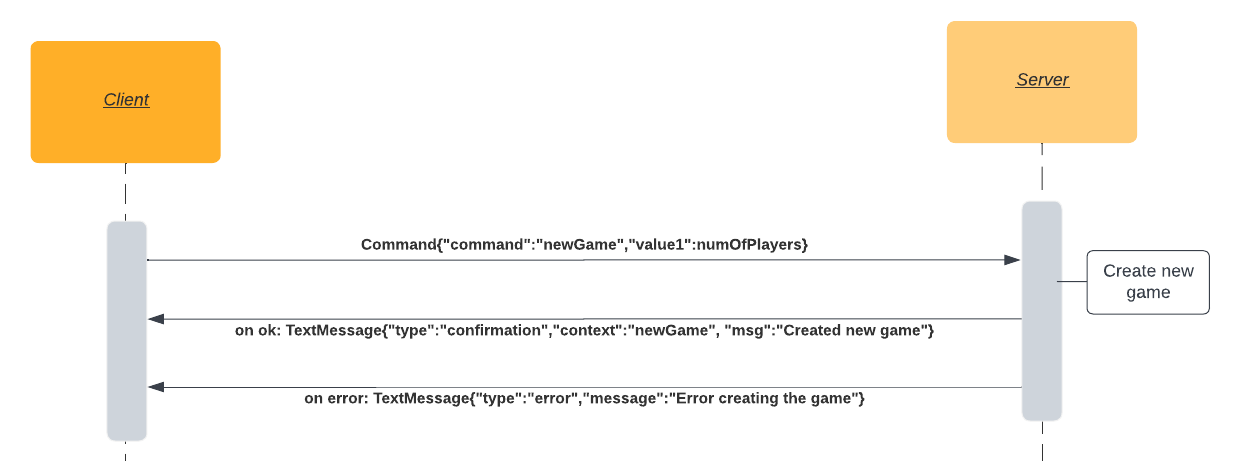
Communication between client and server begin when a new client connects.

The server logs the connection and starts listening for messages from the client.

The first message sent by the client depends on user choise between starting a new game or joining an existing one.

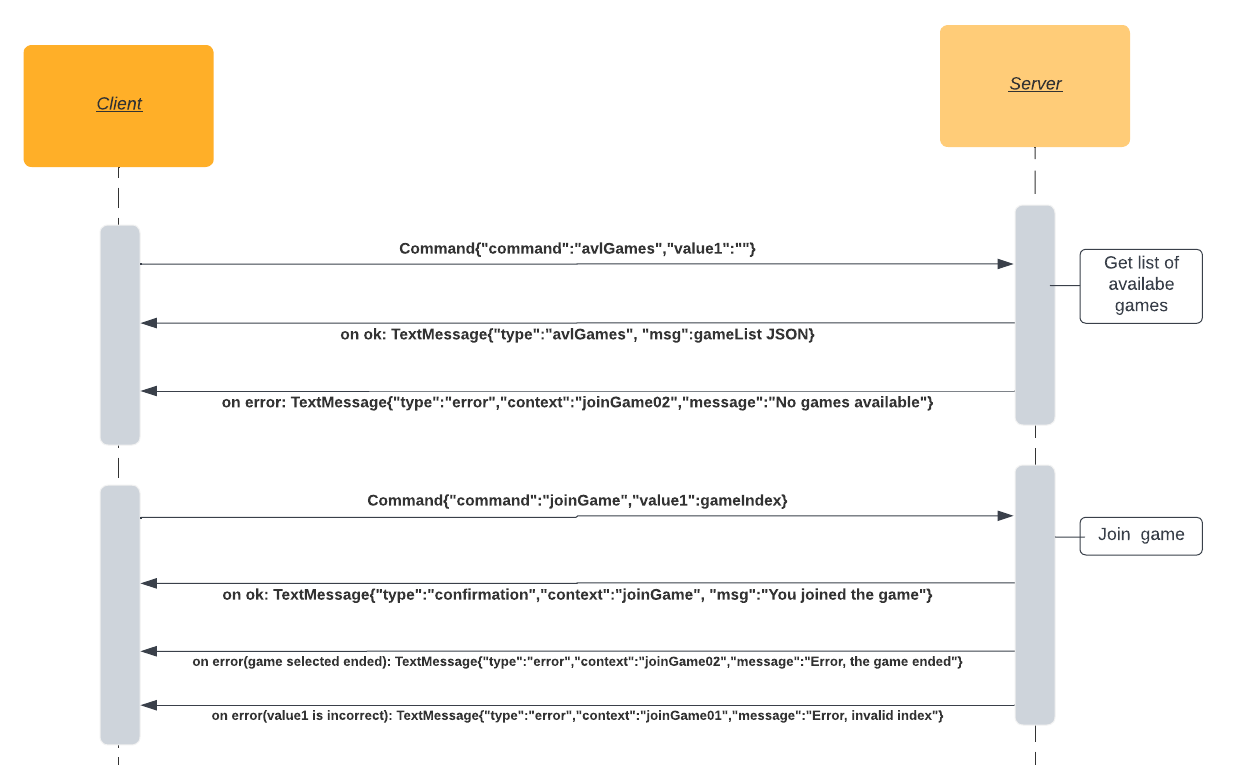
* New Game

The user inputs the number of player desired, 2 or 3, a message is sent to the server.



* Join Game

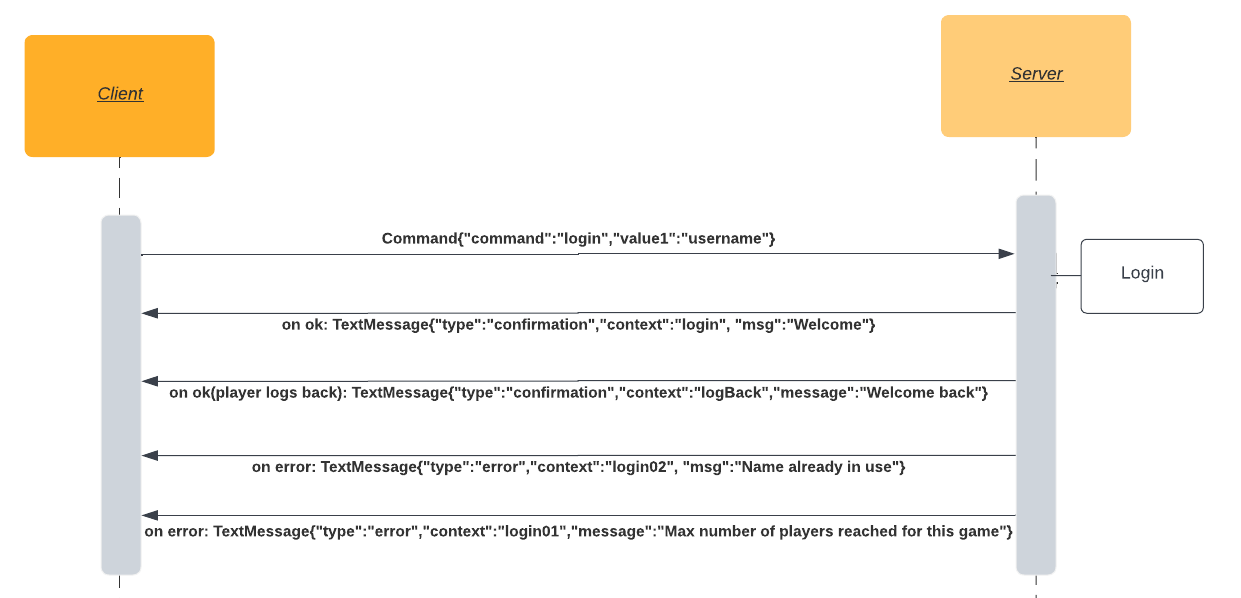
If the user chose to join a game the client asks the server for the list of available games and shows it to the user. Then sends a message to the server with the game index chosen by the user.



* Login

After starting a new game or to join a game the user needs to log in with a unique username.

The server sends back a message depending on the result, the different context defines the client behavior.



## Game commands

After game set up is finished every command to actively play the game from the client is followed by a response by the server that follows the diagram below:

Table

Description automatically generated

Server side a method, mapped to the string in the command field, is called.

The optional fields, value1 and value2, contain paramether used to execute actions, for example the card number to play or the position of a student to move.

Error and confirmation messages are strings unique for every method and follows the same stucture, except for student movement commads that in case of success send a JSON string in the message field with the updated state of dashboard or islands.

Commands mapped on the server:

* Move a student from the hall to classroom
* Move a student from the hall to an island
* Play assistant card
* Play character card
* Choose cloud card
* Move Mother nature on the islands
* Quit the game

## Update commands

To get an update on game objects like players’ dashboard, islands, cards available to play, current player ecc… the client sends to the server commands with the requested object in the command field and empty optional values.

Text

Description automatically generated with medium confidence

The server responds with a TextMessage, the type corresponds to the object requested and a JSON string is serilized in the message field.The client will deserialize twice the message received, once to get the type and a second one depending on the type to access the JSON in the message .

## Notifications

The server notifies every player when certain events happen without waiting any response from the clients.

Events:

* A player joins the game
* A player logs out from the game or disconnects
* It is a player turn
* Switch from planification phase to action phase
* A student has moved, the server sends an update of the islands or dashboard
* A player plays a character card, every other player is notified with the card’s effect
* The game ends