Network Protocol

JEJN

May 2023

Our network protocol is split up into a Server Protocol and a Client Protocol.
The packages sent are built up the same way on both ends:
COMMAND + SEPARATOR + ARGUMENT_1 + SEPARATOR + ARGUMENT... + SEPARATOR + ARGUMENT_N

1 Server Protocol

1.1 SEPARATOR

returns <&!>
Used to separate arguments in the packages sent from client to server or vice versa
Example: LOBBY_EXITED<&!>lobbyName
Command sent form server to client. Here the separator is used to separate the argument LOBBY_EXITED from the argument recipient (the lobby name)

1.2 SUBSEPARATOR

returns <&.>
The separator of the subarguments
Example: UPDATE_LOBBY_LIST<&!>client1 color<&.>client2 color<&.>client3 color
Subseparator is used when sending lists of something. Elements of these lists are assembled as shown above

1.3 SUBSUBSEPARATOR

returns <&..>
Used to separate lobby information when a list of clients is requested
Example: SEND_CRITICAL_BLOCKS<&!><&.>x<&..>y<&..>color<&.>x<&..>y
<&..>color<&.>x<&..>y<&..>color
Subsubseparator is used when sending nested lists of something. Elements of these lists are assembled as shown above

1.4 USERNAME_SET_TO

Inform client that their username has been changed.
Example: USERNAME_SET_TO<&!>username
Command sent form server to client to tell the client that their username is now successfully set to this.username

1.5 NO_USER_FOUND

No user with that username was found
Example: NO_USER_FOUND<&!>username
Command sent form server to client. This string is going to be sent to server to tell client that user was not found

1.6 SEND_PRIVATE_MESSAGE

Used to communicate from client to client
Example: SEND_PRIVATE_MESSAGE<&!>recipient<&!>message
Command sent form server to client. This string is sent from server client to send message to specific client

1.7 SEND_PUBLIC_MESSAGE

A message is being sent to the whole server
Example: SEND_PUBLIC_MESSAGE<&!>sender<&!>message
Command sent form server all clients, to relay the message of one client to all clients connected to the server

1.8 SEND_LOBBY_MESSAGE

A message is being sent to all clients in the lobby
Example: SEND_LOBBY_MESSAGE<&!>username<&!>message
Command sent form server all clients in the lobby, to send message to all clients connected to the lobby

1.9 LOBBY_JOINED

A client has successfully joined a lobby
Example: LOBBY_JOINED<&!>lobbyName
Command sent form server to client. This string is sent to client to inform them that they have successfully joined a lobby.

1.10 LOBBY_EXITED

Sent to a client to inform them that they have successfully exited a lobby. Enables them to print a message to the console.
Example: LOBBY_EXITED<&!>lobbyName
Command sent form server to client. This string is sent to client to inform them that they have successfully exited a lobby.

1.11 UPDATE_FULL_LIST

Sends a list of all lobbies and their clients to the client.
Example: UPDATE_FULL_LIST<&!>lobby1 username1 username2 username3<&.> lobby2 username4 username5
Command sent form server to client. This string is sent to client for them to have access to a full list of all lobbies and their clients.

1.12 UPDATE_CLIENT_LIST

Sends a list of all clients in the server to the client.
Example: UPDATE_CLIENT_LIST<&!>username1<&.>username2<&.>username3
Command sent form server to client. This string is sent to client for them to have access to a list of all clients connected to the server.

1.13 UPDATE_LOBBY_LIST

Sends a list of all clients in the lobby to the client.
Example: UPDATE_LOBBY_LIST<&!>username1 true #f57dc6<&.>username2 false #ffffff<&.>username3 true #b35h6e
Command sent form server to client. This string is sent to client for them to have access to a list of all clients connected to the lobby.

1.14 UPDATE_GAME_LIST

Sends a list of all games that have been played or are currently being played to the client. Contains whether the games have been ended or not.
Example: UPDATE_GAME_LIST<&!>game1 true<&.>game2 false<&.>game3 false
Command sent form server to client. This string is sent to client for them to have access to a list of all games that have been played or are currently being played.

1.15 TOGGLE_READY_STATUS

Informs the client that their ready status has successfully been changed.
Example: TOGGLE_READY_STATUS<&!>true
Command sent form server to client. This string is sent to client to inform them that their ready status has successfully been changed.

1.16 START_GAME

Informs the client that the game has started.
Example: START_GAME
Command sent form server to clients in lobby. This string is sent to client to inform them that the game has started.

1.17 SERVER_PING

Signal regularly sent from server to client to confirm connection
Example: SERVER_PING
Command sent form client to server. Server checks if client is still connected by sending this command to client.

1.18 SERVER_PONG

Signal sent to client upon receiving a PING from the client
Example: SERVER_PONG;
Command sent form server to client. Server has received a PING from client and sends this command to client to confirm connection.

1.19 GAME_ENDED

Signal sent to clients upon receiving an end game request.
Example: GAME_ENDED
Command sent form server to client. Server has received an end game request from client and sends this command to all clients to end the game and return to their lobbys.

1.20 SEND_CRITICAL_BLOCKS

Sends the critical blocks and their colour to the client.
Example: SEND_CRITICAL_BLOCKS<&!><&.>55<&..>39<&..>0xf57dc6ff<&.>55<&..>41<&..>0xf57dc6ff<&.>6
Command sent form server to client. Server sends this command to all clients to update the critical blocks.

1.21 POSITION_UPDATE

Updates the position of the cube for the client.
Example: POSITION_UPDATE<&!>x_pos<&!>y_pos
Command sent form server to client. Server sends this command to all clients to update the position of the cube.

1.22 JUMP_UPDATE

The cube has just jumped. Informs the client of the coordinates of the rotation point
Example: JUMP_UPDATE<&!>x_pos<&!>y_pos<&!>
Command sent form server to client. Server sends this command to all clients to update the position of the cube.

1.23 GAME_STATUS_UPDATE

Updates the status (lives left and levels completed) of the game for the client.
Example: GAME_STATUS_UPDATE<&!>lives_left<&!>levels_completed
Command sent form server to client. Server sends this command to all clients to update the status of the game.

1.24 LOAD_LEVEL

Loads a level for the clients.
Example: LOAD_LEVEL<&!>level
Command sent form server to client. Server sends this command to all clients to load a level.

2 ClientProtocol

2.1 COMMAND_SYMBOL

Symbol inputted by the client to indicate that the following input is a command
returns "!"
Example: if (command.startsWith(commandSymbol)) send command to server
Here the client detects if a user input is meant as a command and if yes sends it to the server

2.2 SET_USERNAME

Set client username
Example: SET_USERNAME<&!>username
Sends command to server which asks server to update the username. The server then has the right to modify that request if the username is already taken.

2.3 SEND_LOBBY_MESSAGE

Sends message to the whole lobby
Example: SEND_LOBBY_MESSAGE<&!>message
Command sent from client to server. Server then sends message to all clients in lobby.

2.4 SEND_PRIVATE_MESSAGE

Sends message to specific client
Example: SEND_PRIVATE_MESSAGE<&!>recipient<&!>message
Command sent from client to server. Server then sends message only to specific client.

2.5 SEND_PUBLIC_MESSAGE

Send a chat message to the whole server
Example: SEND_PUBLIC_MESSAGE<&!>message
Command sent from client to server. Server then sends message to all clients connected

2.6 EXIT

Used when client is exiting the program
Example: EXIT
Command sent from client to server. Server get information, that client is disconnecting –¿ Logout protocol on serverside

2.7 JOIN_LOBBY

Client wants to join a lobby
Example: JOIN_LOBBY<&!>lobbyName<&!>lobbyPassword
Command sent from client to server. Client requests to join specific lobby.

2.8 CREATE_LOBBY

Client wants to create a lobby
Example: CREATE_LOBBY<&!>lobbyName<&!>lobbyPassword
Command sent from client to server. Client requests to create lobby on server.

2.9 EXIT_LOBBY

Client wants to exit a lobby. If they are not in a lobby, the server will ignore their request. Example: EXIT_LOBBY

2.10 GET_FULL_SERVER_LIST

Signal sent to server to tell server that client wants to get a list of all clients and lobbies. Used when the client has loaded their menu screen.
Example: GET_FULL_SERVER_LIST
Command sent form client to server. Client sends this command to server to tell server that client wants to get a list of all clients and lobbies.

2.11 GET_FULL_MENU_LISTS

Signal sent to server to tell server that client wants to get a list of all clients, lobbies and games. Used when the client has loaded their menu screen.
Example: GET_FULL_MENU_LISTS
Command sent form client to server. Client sends this command to server to tell server that client wants to get a list of all clients, lobbies and games,

2.12 GET_FULL_LOBBY_LIST

Signal sent to server to tell server that client wants to get a list of all clients in their lobby. Used when the client has loaded their lobby screen.
Example: GET_FULL_LOBBY_LIST
Command sent form client to server. Client sends this command to server to tell server that client wants to get a list of all clients in their lobby.

2.13 TOGGLE_READY_STATUS

Client wants to toggle their ready status
Example: TOGGLE_READY_STATUS<&!>clientIsReady
Command sent from client to server. Client wants to toggle their ready status

2.14 CLIENT_PING

Signal regularly sent from client to server to confirm connection
Example: PING
Command sent form client to server. Client sends this command regularly to server to detect connection issues.

2.15 CLIENT_PONG

Signal sent to server upon receiving a PING from the server
Example: PONG
Command sent form client to server. Client sends this command as a respond to server PING

2.16 SPACE_BAR_PRESSED

Signal sent to server to tell server that client wants to jump
Example: SPACE_BAR_PRESSED
Command sent form client to server. Client sends this command to server to tell server that client has pressed space bar server then checks whether the client is allowed to jump and either makes all clients jump or does nothing.

2.17 READY_UP

Signal sent to server to tell server that the client has the game opened and is ready to start
Example: READY_UP
Command sent form client to server. Client sends this command to server to tell server that client is ready to start,

2.18 REQUEST_CRITICAL_BLOCKS

Signal sent to server when client has loaded the level successfully. Client requests the critical blocks of the level.
Example: REQUEST_CRITICAL_BLOCKS
Command sent form client to server. Client sends this command to server to tell server that the client has loaded the level successfully and requests the critical blocks of the level.

2.19 REQUEST_END_GAME

Signal sent to server to tell server that the client has pressed the quit game button and requests that the game ends
Example: REQUEST_END_GAME
Command sent form client to server. Client sends this command to server to tell server that the client has pressed the quit game button and requests that the game ends,

2.20 SKIP_LEVEL

Signal sent to server to tell server that client wants to skip the current level
Example: SKIP_LEVEL
Command sent form client to server. Client sends this command to server to tell server that client wants to skip the current level,

2.21 SET_IMMORTAL

Signal sent to server to tell server that client wants to be immortal
Example: SET_IMMORTAL
Command sent form client to server. Client sends this command to server to tell server that client wants to be immortal,

2.22 SET_MORTAL

Signal sent to server to tell server that client wants to be mortal
Example: SET_MORTAL
Command sent form client to server. Client sends this command to server to tell server that client wants to be mortal,