Protokollet

1. Innehålla en utförlig beskrivning av datan som skickas mellan server och klient.

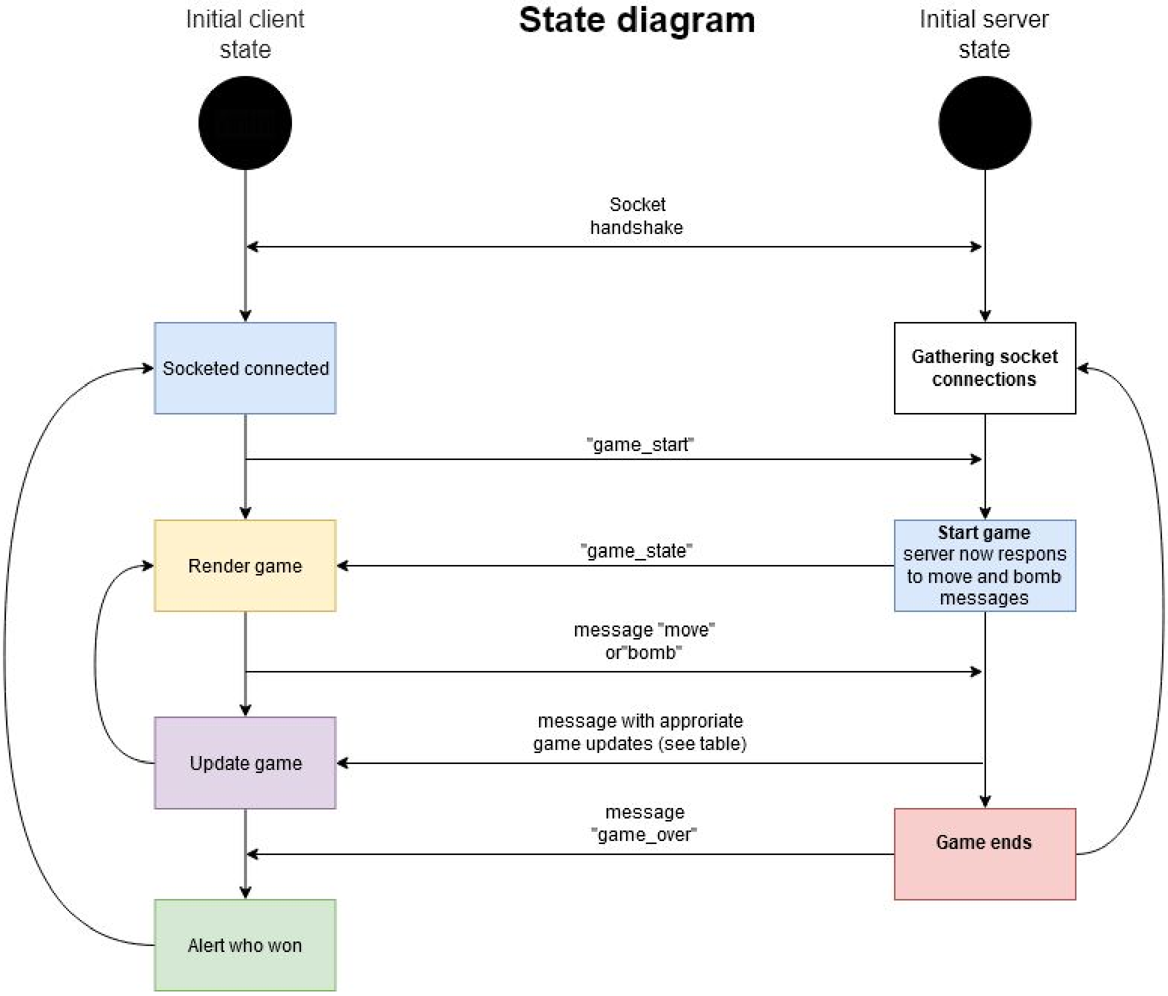
The strings sent are UTF-8 JSON encoded. “arg1” and “arg2” are the two parameters sent via the socket connection and aren’t actually called “arg1” and “arg2” anywhere, but replaced with the things we have specified in the tables below. We use Socket.io which uses WebSockets and we structured our socket communication the same way specified in this link (https://socket.io/docs/#Using-it-just-as-a-cross-browser-WebSocket) which makes it compatible with regular WebSockets.

INPUT (server to client)		
arg1	**arg2**	**Purpose**
“game_state”	JSON string with keys “players” and “tiles”. The “players” value is a JSON string with keys representing each players’ game ID and values with a JSON representation of the player. The player is represented as a JSON string with keys “id” (integer), “x” (integer), “y” (integer), “isAlive” (boolean), “color” (string). The “tiles” value is a string representation of an array. Each element represents a game tile which is a JSON string with keys “x” (integer), “y” (integer), “item” (string), “deadly” (boolean), “explosionId” (integer).	Only needs to be called to get the initial state of the game. The changes to the game state will be sent by the server and are described in the rest of the protocol.
“player_move”	JSON object with keys “id” (integer), “x” (integer), “y” (integer).	For moving the player with the corresponding id to the new x and y coordinates graphically.
“remove_player”	JSON string of a simplified player object with keys “id” (integer), “x” (integer), “y” (integer), “color” (string).	Getting info about which player was removed and where they were placed for easy “drawing over” their pos and removing them from the playerlist.
“new_player”	JSON string with same info as in “remove_player”	To know where to draw the new player

		that joined the game.
“explosion”	JSON string representation of an array with tile objects. Each tile has attributes “x” (integer) and “y” (integer).	To mark tile as “deadly” and draw it with the “deadly” colour defined by the client’s draw function.
“made_not_deadly”	Same as the one right above	To mark tile as not “deadly” and draw it with the appropriate color (the “empty” color, since the explosion destroyed any barrel that may have been there.
“game_over”	A binary representation of the id of the player who won the game.	For displaying who won the game.
“update_tile”	JSON string representation of an object with keys “item” (string), “x” (integer), “y” (integer)	For updating the tile as position x, y with the correct tile item.

OUTPUT (client to server)		
arg1	**arg2**	**Purpose**
“start_game”	null	Signalling to the server that someone has clicked the start game button which places the players, emits the gamestate and starts the game.
“move”	“up”	Signalling to the server that the client wants to move its player up.
“move”	“down”	Same as the one above, but moving down instead.
“move”	“left”	Same as above but left.
“move”	“right”	Same as above but right.
“bomb”	null	Signalling to the server that the client wishes to place a bomb at its player’s current location.

2. Innehålla ett tillståndsdiagram (state diagram) som beskriver vilka tillstånd klienten och servern kan ha samt vad som gör att de övergår från ett tillstånd till ett annat.


State diagram
Initial client state
Initial server state
Socket handshake
Socketed connected
Gathering socket connections
"game_start"
Render game
"game_state"
Start game server now respons to move and bomb messages
message "move" or"bomb"
Update game
message with approriate game updates (see table)
message "game_over"
Game ends
Alert who won