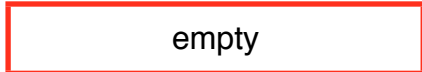


Frame 1:

Sign-on frame (Client <-> Server)

Client -> Server

*Empty payload, just indicate that they want to play ex: * 1*

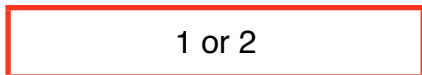


Server -> Client

Sent after sign-on, informs which color player is

1: black 2: white (black always moves first)

Note: first player assigned is black which will wait for white to sign-on



Once white signs-on a board (frame 4) is sent out indicating start of game

Frame 2:

Move Frame (Client -> Server)

Client -> Server

Informs server where they would like to move on the x y grid

X Y range from 0 to 7

X

	0	1	2	3	4	5	6	7
0								
1								
2								
...								



Frame 3:

Invalid Move frame (Server -> Client)

Server -> Client

Empty payload, informs client that move is invalid can be interpreted as try again

empty

Frame 4:

Board Frame (Server -> Client)

Server -> Client

Updates board after move

Series of 65 integers (0: null 1: black 2: white)

The first 64 integers represent the board layout

The 65th character represents whose turn it is (1: black 2: white)

Order of the integers can be seen in the below table

ex: 0111011120111011...

	0	1	2	3	4	5	6	7
0	1	2	3	4	5	6	7	8
1	9	10	11	12	13	14	15	...

Series of 65 integers (0,1, or 2)

Frame 5:

End of game (Server <-> Client)

Server -> Client

Sent to declare winner, can be sent at anypoint to end game if player is disqualified (i.e. timeouts on turn, to many invalid moves....)

Sends number indicating winner

1: black 2: white

1 or 2

Client -> Server

Empty frame sent to 'quit' game

empty

Frame 6:

Error (Server -> Client)

Server -> Client

Sent whenever an error occurs or unexpected frame is received

(i.e. trying to make move when not turn)

Contains a string can either be "" or contain information on the error

String

[Optional] **Frame 7:**

Message (Client <-> Server)

Client <-> Server

Message to be sent to other player

String

Additional Info:

