

Poker network protocol

May 11, 2016

1 Introduction

A protocol for playing poker remotely via a text interface, usually (but not necessarily) over a network. The protocol is based on passing text commands between the server and a client, containing all the necessary information to play the game. The server will send the settings, information about the other players, and their decisions as they are made. The client will wait for their turn, and send ack decisions after the server requests it.

Each command is terminated by a newline, and commands are separated into tokens as specified in the “tokens” section. The first token is always the command name. If the client or server receives a command with an unknown command name, the whole command should be ignored, and communication should continue as normal afterwards. If a known command is illegally formatted, however, the client should show an error.

2 Tokens

Each command consists of one or more tokens. Normally, each token is a single “word” (sequence of characters delimited by one or more whitespaces), but a token can also be several words inside ascii quotation marks. For example:

- **stackSizes 0 1000 1 1000** tokenizes to [“stacksizes”, “0”, “1000”, “1”, “1000”]
- **name “Svein Jonny”** tokenizes to [“name”, “Svein Jonny”]

3 Decision and card format

Cards

Each card is stored as a single text token (no whitespace characters), represented as the suit followed by the rank. The ranks are given by the integers 2-14, the suits are given by their names (diamonds, spades, hearts, clubs). Examples: diamonds14, spades7, clubs10.

Decisions

A decision is stored as a single token (no whitespace characters), consisting of the type of decision and optionally the “size” of the decision in case of raises or bets. Possible decisions are:

- fold
- check
- call
- smallBlind
- bigBlind
- bet⟨amount⟩
- raise⟨amount⟩ (Raises *by* the amount given, not *to* the amount)
- allIn

4 Server to client

upiok ⟨version⟩

Sent after it receives the initial upi command from client, to complete the initialization handshake. After receiving this, the client should wait for further input.

getName

Requests the client’s name, to which it expects a name command in return.

newGame

Tells the client that a whole new game is being played, with new players and new stack sizes.

newHand

Tells the client that a new hand is starting.

clientId ⟨ID⟩

Sends the client’s ID in the game. Each player on the table has a unique non-negative ID, which is used when sending names, positions, etc. This should be sent immediately after a newgame command, before amountOfPlayers and other information about the players.

amountOfPlayers $\langle n \rangle$

Sends the number of players on the table, including the client itself. This should be sent after `clientId`, but before any information about the other players (stack sizes, blinds, names, etc) is sent.

playerNames $\langle id1\ name1 \rangle \langle id2\ name2 \rangle \dots$

Sends the names of all the players. This should be sent at the start of each game. These are not used by the protocol, but are for the client to display.

playerPositions $\langle id1\ position1 \rangle \langle id2\ position2 \rangle \dots$

Sends the position for all players' IDs. This is sent at the start of each hand, i.e. when the positions change. The position is sent as a non-negative integer, where position 0 is small blind, position 1 is big blind, position 2 is UTG, etc.

stackSizes $\langle id1\ stackSize1 \rangle \langle id2\ stackSize2 \rangle \dots$

Sends the stackSizes for all players' IDs. This is sent at the start of each hand, and also at the end of the final hand.

smallBlind $\langle n \rangle$

Sends the small blind.

bigBlind $\langle n \rangle$

Sends the big blind.

levelDuration $\langle n \rangle$

Sends the duration of each blind level, in minutes.

setHand $\langle ID \rangle \langle card1 \rangle \langle card2 \rangle$

Sends the hole cards of the player's ID. The client's own hole cards are sent at the start of each hand; other players' hole cards are sent during showdown, or when reviewing games.

setFlop $\langle card1 \rangle \langle card2 \rangle \langle card3 \rangle$

Sends the flop cards to the client. See section "decision and card formats" for the format for cards.

setTurn <card>

Sends the turn card to the client. See section “decision and card formats” for the format for cards.

setRiver <card>

Sends the river card to the client. See section “decision and card formats” for the format for cards.

playerBust <ID> <rank>

Sent when a player is busted (is out of the game). Also sends the position the player lost in.

showdown <line1> <line2> ...

Sent after a showdown, sending n lines of text describing who won the pot and any side pots. These are meant to be displayed in the middle of the GUI.

getDecision <timeToThink>

It's the client's turn, and the server is asking for a decision. The server expects a decision command in return. The timeToThink field is the time the client has to think, in milliseconds. If the client does not return a decision in time, the server may give the client a “default” decision, or otherwise override the client's wishes.

playerMadeDecision <ID> <decision>

Sent to every client whenever any player makes a decision in the game. Clients are also sent their own decisions this way. See section “decision and card formats” for the format for decisions.

5 Client to server

upi <version>

Used to initialize communication over the protocol, and tell the client server that it is using universal poker interface. The version parameter is a sequence-based identifier (like 1.0.0). The client should wait for a “upiok” command to complete the handshake.

chat textToken

Used to send chat messages to other players. The chat message is sent as a single token, so it usually needs to be wrapped in quotation marks to be received properly.

name <name>

Sends the client's name to the server. Should only be sent after the server requests it with a getName command. The name field may be empty, in which case the server may assign a name to the client. The name cannot contain whitespace characters.

decision <decision>

Sends the client's decision to the server. See section "decision and card formats" for the format for decisions.

6 Example playthrough

Client upi

Server upiok

Server getName

Client name Doyle

Server newGame

Server setId 2

Server amountOfPlayers 3

Server playerNames 0 Morten 1 Kristian 2 Doyle

Server playerPosition 0 0 1 1 2 2

Server stackSizes 0 5000 1 5000 2 5000

Server setHand 2 diamonds14 ace14

Server playerMadeDecision 0 smallBlind

Server playerMadeDecision 1 bigBlind

Server getDecision 5000

Client decision call

Client playerMadeDecision 2 call
Server playerMadeDecision 0 fold
Server playerMadeDecision 1 check
Server set