The Othello server API, version 0.99.2

Jacek Malec

January 28, 2025

The interaction between the Othello server and a client must adhere to the following rules. There are two versions of it: one for playing against a simple automatic opponent, to verify your code (so called "server mode"), and one for playing against code of another human (so called "competition mode").

Please note that availability of the servers will be announced separately, via announcements on canvas. Right now this document is provided for the purpose of defining the protocol.

1 Server mode

The Othello server is running on machine `vm33.cs.lth.se` on port `9035` (testing version, with the possibility of choosing colour) or port `9045` (evaluation version, used for checking your code). You are expected to open a socket connection to this port and interact with the system through it.

The *idstring* (see below, 128 characters) will be provided to you separately. You must use your own *idstring* to get the game recorded and scored. The behaviour of the server for an invalid *idstring* is unpredictable.

All interaction is purely text based, messages end with a newline character (`\n`).

server: `Hi! I am your othello server.`
server: `What is your name?`
client: *idstring*
server: `Hello` *idstring*`! Your current win count is #`
server: `Your time limit is` # `secs`

NOTE: if you exceed the time limit, the behaviour of the server is unpredictable.

server: `Move format: column (letter) + row (number).`
server: `choose colour, 'd' for dark, 'w' for white.`
client: *d*`|`*w*

The two rows above are available for your testing of both possibilities. In evaluation mode, your program will not have any possibility of choosing the colour.

server: `you are dark|white`

or, in testing mode, when wrong colour (i.e., not *d* or *w*) was provided:

server: `error\n You provided wrong colour. Bye!`

if you are white:

server: `opponent's move`

server: `XY`

(where `X` is a letter `a-h` and `Y` is a digit `1-8`).

`repeat`

server: `your move`

client: *XY*

server: `opponent's move`

server: `XY`

`until` one of the conditions hold:

1. if the client proposed an illegal move: the server closes with the message:

   server: `error\n This move is illegal. Bye!`

2. if the game has ended: the server informs about the disk count and the possible winner, and closes the connection

   server: `The game is finished`

   server: `White:  #`

   server: `Dark:  #`

   server: `White won | Dark won | Draw`

Please note: if one of the sides has no legal moves then it should use the reserved keyword `PASS` to signal it, instead of providing the location in the format `XY`.

2 Competition mode

The two competing programs should connect to ports `9051` and `9052`. Let us call the players' identifiers *id1* and *id2*, respectively.

All interaction is purely text based, messages end with a newline character (`\n`).

server: `Hi! I am your othello server.`

server: `What is your name?`

client: *id1*

if on port `9051`, or *id2* if on port `9052`.

server: `Hello` *id1* `!`

server: `Your time limit is # secs`

NOTE: if you exceed the time limit, the behaviour of the server is unpredictable.

server: `Move format:  column (letter) + row (number).`

server: `you are dark|white`

`dark` will be assigned to port `9051`.

If you are white (i.e., *id2*):

server: `opponent's move`

server: `XY`

(where `X` is a letter `a-h` and `Y` is a digit `1-8`).

```
repeat
    server: your move
    client: $XY$
    server: opponent's move
    server: XY
until one of the conditions hold:
```

1. if the client proposed an illegal move: the server closes with the message:

   server: `error\n This move is illegal. Bye!`

2. if the game has ended: the server informs about the disk count and the possible winner, and closes the connection

   server: `The game is finished`

   server: `White:  #`

   server: `Dark:  #`

   server: `White won | Dark won | Draw`

   (Possibly the final message will differ, naming the winner.)