Advanced – command, which ‚steals‘ the I/O

In the attached code in src/main you will find a simple TCP chat client. It is created by HOSTNAME, PORT of a chat server, InputStream and PrintStream.

- It connects to HOSTNAME:PORT via TCP.
- On line input from a given InputStream it writes to the said server.
- On line input from the server it writes to the given PrintStream.
- The class MainStandaloneChat illustrates how that works as a standalone client

Extend your command interpreter with the command

```
chat-connect <hostname> <port>
```

When called, this command ‚steals‘ the I/O of the command interpreter, i.e. the program should behave just like a chat client started by MainStandaloneChat.

If ‚exit‘ is written, the ChatClient is destroyed and the command interpreter should continue to work as usual.

Notes:

- The MainChatInCmdInterpreter class illustrates an ugly, partially working decision. You can start from that.
  - Refactor,
  - Add documentation
  - Add unit tests
- In order to test your client, you can use the attached server
  - Install node.js (google it)
  - Then in cmd run it like
    ```
    C:\Downloads\chat-server>node server.js
    ```
  - This would start the chat server on localhost:8080