Running

Compilation

You will need to compile the server before you can run it. Navigate to the extracted jar directory, and run the java compiler

```
cd bbs
javac BulletinBoard.java
```

Launching the Server

Navigate to the extracted jar directory ‘bbs’

```
cd bbs
```

From a shell, type:

```
 //Portnumber must be a valid, and free portnumber on the local machine
 java BulletinBoard [portnumber]
```

Launching a Client

From another shell type:

```
telnet localhost [portnumber]
//There must be a server listening on this port.,
```

You can also launch the java client, in the “client” directory. (See its Readme). The java client works in exactly the same way as the telnet client, and both offer immediate notifications of updates to the bulletin board.

Features

- All changes to the Bulletin board are preserved. (In the bbs.xml file)
- Passwords are not stored in clear-text.
- There is an admin user (‘username: admin, password: admin’) who has additional powers on the board.
- Client input and output are managed in separate threads allowing messages to be broadcast to all connected telnet clients at any time. (Without having to wait for input)
- There are additional commands - see “.commands” and “.adduser”
- Topic names can contain whitespace
- You can set/delete topics by name or ID.

Usage

- Type .commands to see a list of all available commands
- After every “.post” and “.set” command you see the last 5 messages on the selected topic (use “.read” to see them all!)
- Every “.post”, “.create”, “.login” and “.logout” command is broadcast to all connected users so they can see what is happening on the board.
- Users can delete only messages they have posted themselves using “.delete-message”, the admin user can delete any message from any user.

- Users can delete only topics that are empty using “.delete-topic”, the admin user can delete any topic.
- The admin can create new users using the “.adduser” command.

Challenge

- Each client-thread spawned by the server run two threads.(A reader and writer thread), which allows the multi-threaded behavior in both the separate java client, and the original telnet client.
- This allows us to broadcast to all clients at any stage, regardless of whether they are waiting on input or not.
- Updates to posts, log-ins/outs, new-topics and new-users are all broadcasted to connected users.
- The key advantage to offering this to the telnet client as opposed to separate client applications, is that to use this application is that host need only install one piece of software, (the server) and the clients none at all, making it much easier to distribute! (Assuming client computers have telnet installed, which most would).