

GMIT

Fabio Lelis

G00330441

24 December 2015

Operating Systems I

Client-Server application

What's the task:

Your project is to write a Multi-threaded TCP Server Application which allows multiple client applications to transfer files to and from the server. The client application can use command line input from the user to implement user functions.

The service should allow the users to:

1. Authenticate between the client application and the server application.
2. Copy a selected file from the server. (e.g. get file1.txt)
3. Move a selected file to the server. (e.g. put file1.txt)
4. List all the files in the current directory of the server.
5. Move to a different directory on the server.
6. Make a new directory on the server.

Server:

The current server configuration:

Software: Windows Server 2008 R2 Datacenter SP1 64bit

Hardware: Intel Xeon, 3.50GB of installed memory

Added software: Java 1.8.0_65 and Gitbash

Modifications: port 2004 opened on firewall.

Structure:

The project is organised on five Java files, one txt file, and the users directories.



The file **userslist.txt** is where is store the list of users, their passwords and folders.

The folder **users** contains the subfolders of each user as indicated in userslist.txt.

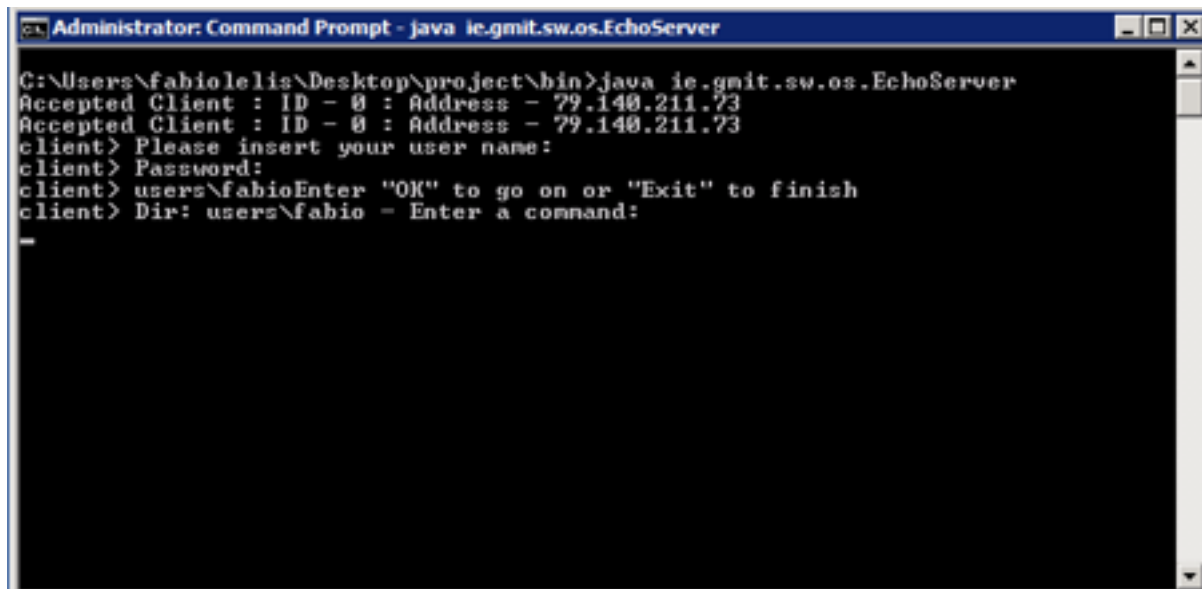
The two main classes are **Requester** and **EchoServer**. Each one playing one side of the server-client workflow. They are in the same project to make easier localhost testing. Unless when the operation involves file transfers, the **Requester** class is basically a chatter, which sends messages that are typed by the user to the server. **EchoServer** gets the message and choose the right response.

The first question from the server is for authentication (this part happens mainly on **Authentication** class). The file userslist.txt is loaded on a hashmap to make the access faster on each new user login. The class **User** is a the representation of a user.

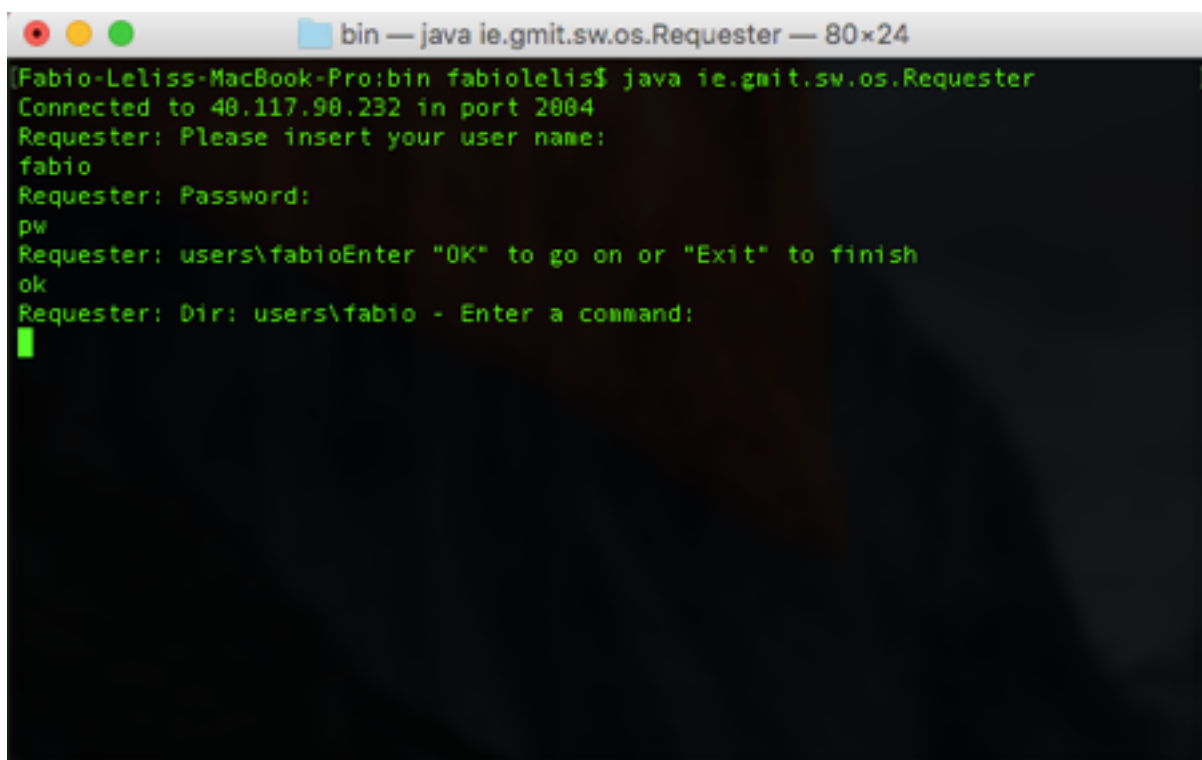
A name and password must be provided by the requester:

The name and password will be checked at the hashmap and if it does exist, the session is started, and if doesn't the user is asked for retry or exit.

If the name user exists but doesn't have a folder, one will be created with his name.
Then if is everything gone ok we have a authenticated session.



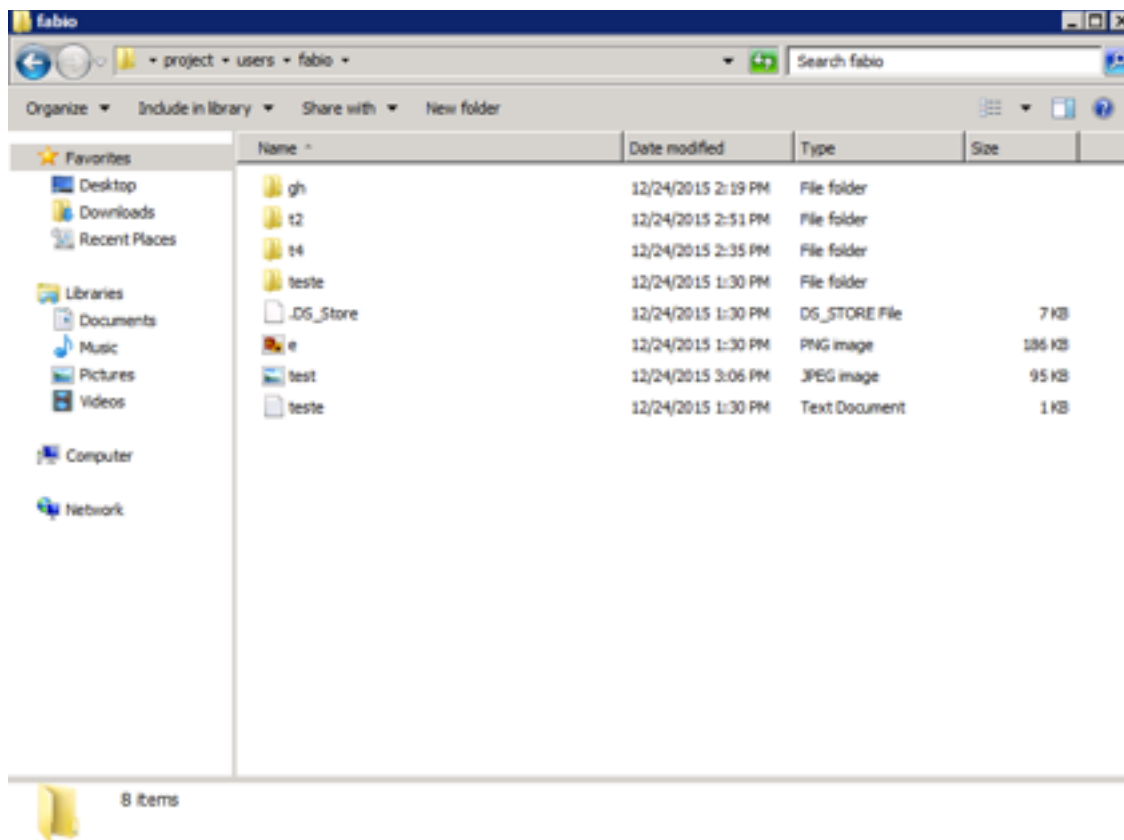
```
Administrator: Command Prompt - java ie.gmit.sw.os.EchoServer
C:\Users\fabiolelis\Desktop\project\bin>java ie.gmit.sw.os.EchoServer
Accepted Client : ID - 0 : Address - 79.140.211.73
Accepted Client : ID - 0 : Address - 79.140.211.73
client> Please insert your user name:
client> Password:
client> users\fabioEnter "OK" to go on or "Exit" to finish
client> Dir: users\fabio - Enter a command:
-
```



```
bin — java ie.gmit.sw.os.Requester — 80x24
(Fabio-Leliss-MacBook-Pro:bin fabiolelis$ java ie.gmit.sw.os.Requester
Connected to 40.117.90.232 in port 2004
Requester: Please insert your user name:
fabio
Requester: Password:
pw
Requester: users\fabioEnter "OK" to go on or "Exit" to finish
ok
Requester: Dir: users\fabio - Enter a command:
█
```

Commands:

Now you can run six types of command and, in this example you have this files:



The class **CmdExecutor** is used to run the following options:

LS list the files and folders in current directory.

```
bin — java ie.gmit.sw.os.Requester — 80x24
[Fabio-Leliss-MacBook-Pro:bin fabiolelis$ java ie.gmit.sw.os.Requester
Connected to 40.117.90.232 in port 2004
Requester: Please insert your user name:
fabio
Requester: Password:
pw
Requester: users\fabioEnter "OK" to go on or "Exit" to finish
ok
Requester: Dir: users\fabio - Enter a command:
ls
Requester: Files:
File .DS_Store
File e.png
Directory gh
Directory t2
Directory t4
File test.jpg
Directory teste
File teste.txt

Enter "OK" to more commands or "exit" to finish
█
```

Syntax: ls

CD changes the current folder

Syntax: cd subfolder; cd ..; cd ../folder; etc












```
bin — java ie.gmit.sw.os.Requester — 80x24
[Fabio-Leliss-MacBook-Pro:bin fabiolelis$ java ie.gmit.sw.os.Requester
Connected to 40.117.90.232 in port 2004
Requester: Please insert your user name:
fabio
Requester: Password:
pw
Requester: users\fabioEnter "OK" to go on or "Exit" to finish
ok
Requester: Dir: users\fabio - Enter a command:
cd t2
Requester: Dir: users\fabio\t2 - Enter a command:
```

MKDIR creates a folder

Syntax: mkdir subfolder; mkdir ./newfolder; mkdir ../newfolder; etc



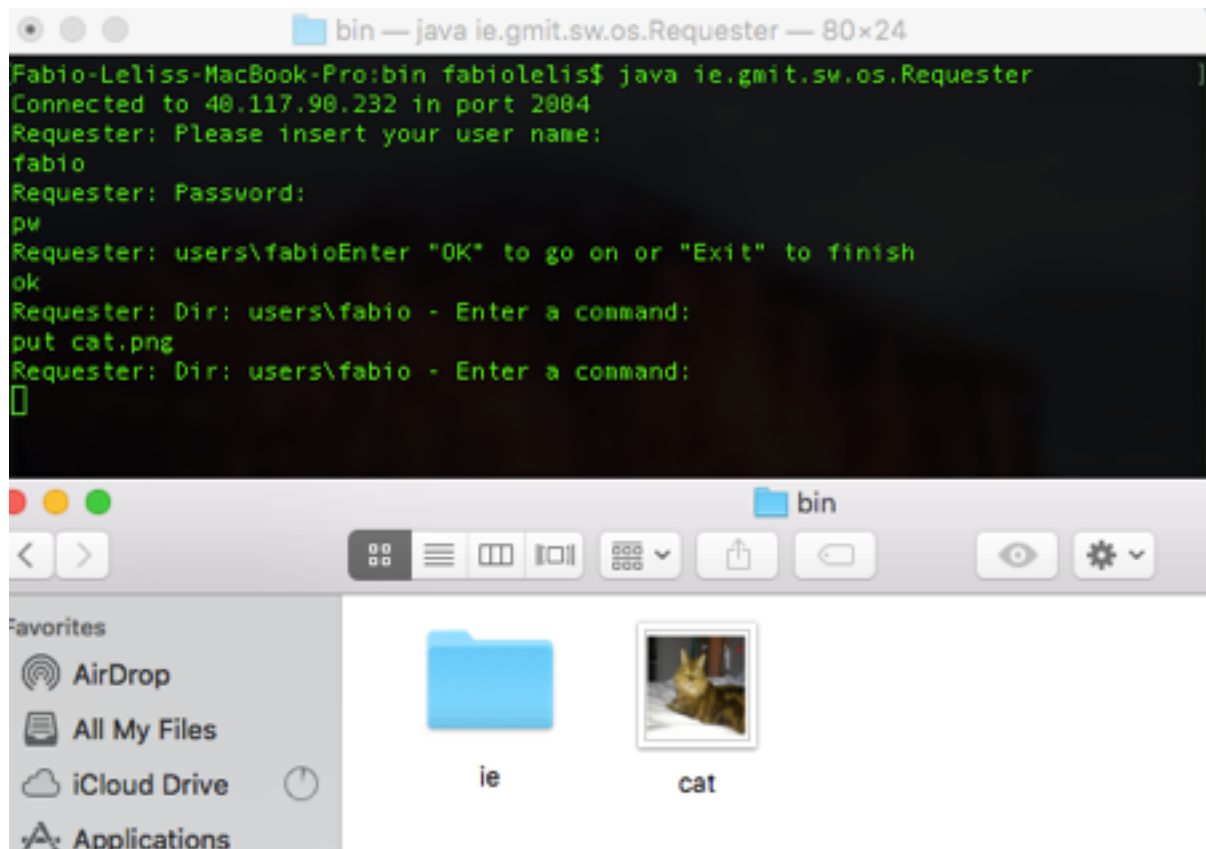
```
bin — java ie.gmit.sw.os.Requester — 80x24
[Fabio-Leliss-MacBook-Pro:bin fabiolelis$ java ie.gmit.sw.os.Requester
Connected to 40.117.90.232 in port 2004
Requester: Please insert your user name:
fabio
Requester: Password:
pw
Requester: users\fabioEnter "OK" to go on or "Exit" to finish
ok
Requester: Dir: users\fabio - Enter a command:
mkdir testmk
Requester: Dir: users\fabio - Enter a command:
```

| | | | | |
|---|-----------|--------------------|---------------|--------|
|  | gh | 12/24/2015 2:19 PM | File folder | |
|  | t2 | 12/24/2015 2:51 PM | File folder | |
|  | t4 | 12/24/2015 2:35 PM | File folder | |
|  | teste | 12/24/2015 1:30 PM | File folder | |
|  | testmkdir | 12/24/2015 3:40 PM | File folder | |
|  | .DS_Store | 12/24/2015 1:30 PM | DS_STORE File | 7 KB |
|  | e | 12/24/2015 1:30 PM | PNG image | 186 KB |
|  | test | 12/24/2015 3:06 PM | JPEG image | 95 KB |
|  | teste | 12/24/2015 1:30 PM | Text Document | 1 KB |

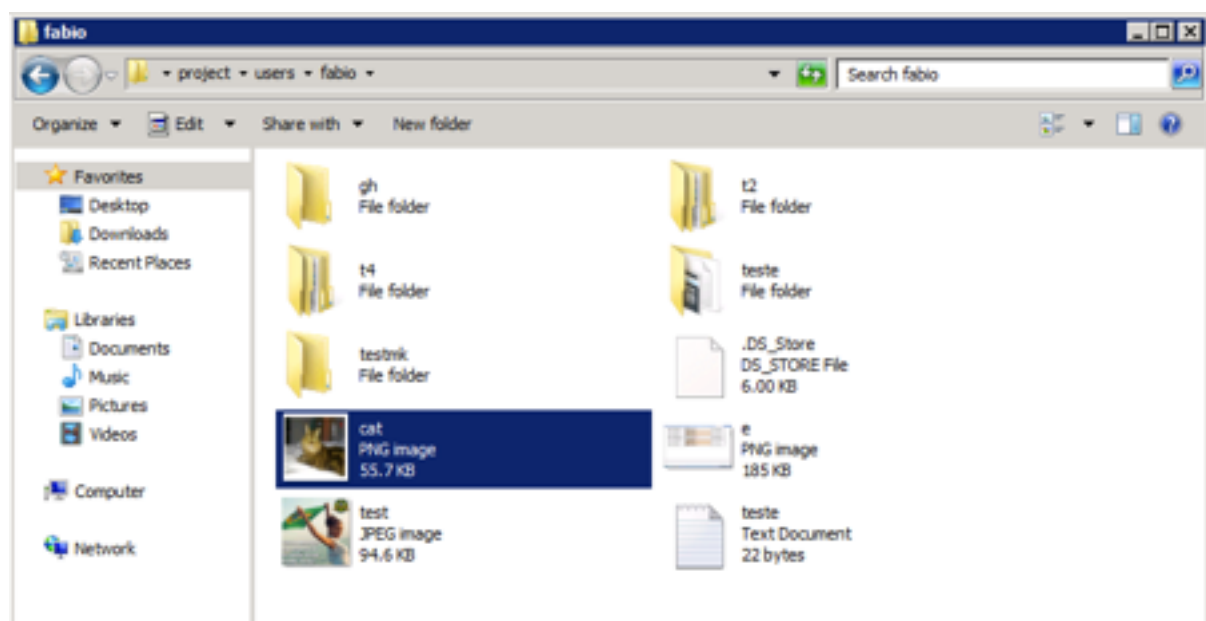
PUT takes a file on the same directory as Requester on client and puts it in the user folder on the server.

Syntax: put test.jpg

Client:

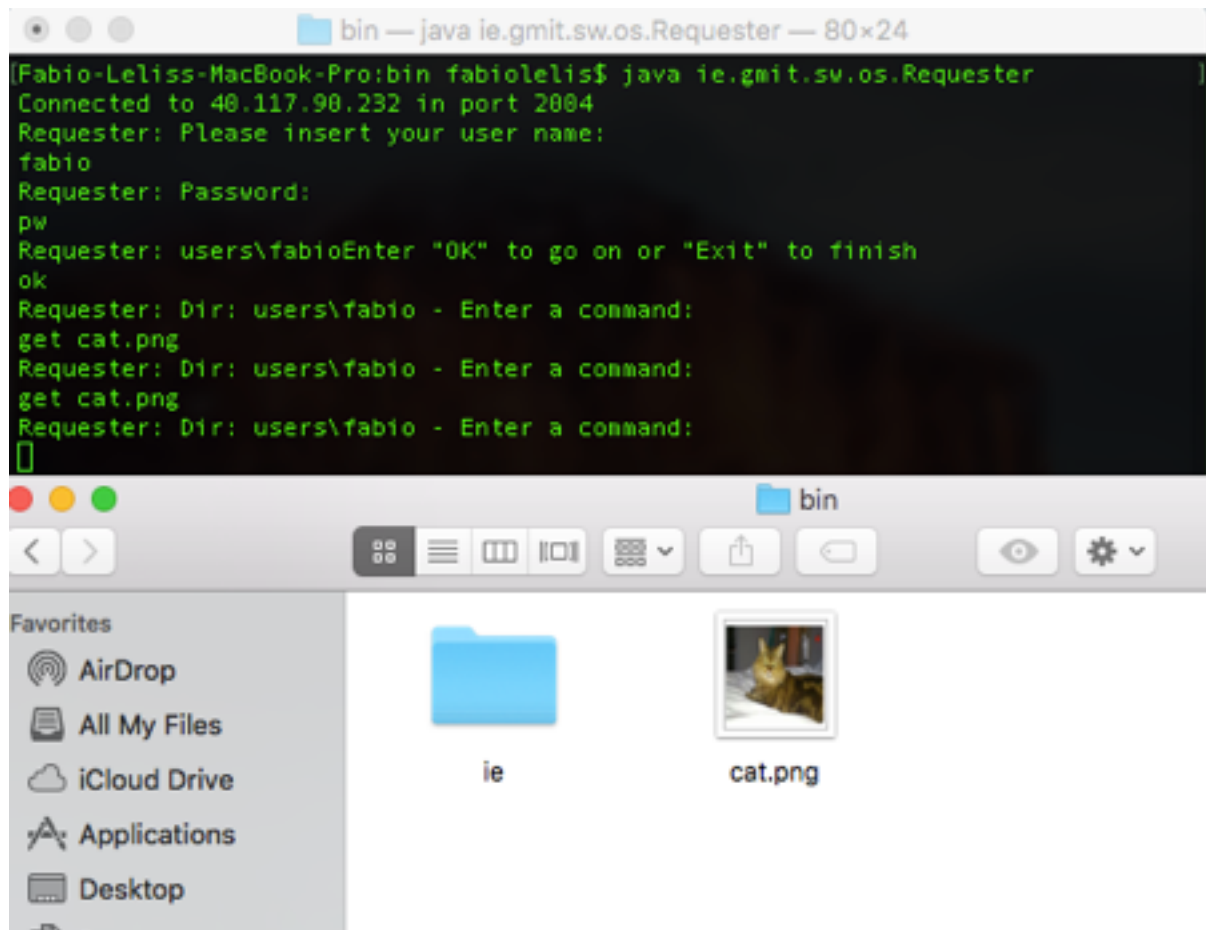


Server:



GET inversely, takes a file in user folder, on server, and send to the same directory as Requester.

Syntax: get test.jpg



EXIT finishes the thread logging out the session.

Syntax: exit