COMPUTER ENGINEERING

4DN4

ADVANCED INTERNET COMMUNICATIONS

Lab – 2:

McNapster

Submitted To:

Professor Ted Szymanski

Submitted By:

Name: Syed Anjoom Iqbal

Student ID: 1063786

MacID: iqbalsa

Email: iqbalsa@mcmaster.ca

Submission Date:

March 31, 2014

COMP ENG 4DN4 2014 Lab1 Report
McNapster

Table of Contents

Objective

The objective of this lab is to implement an FTP server-client system for sharing music and photos. The inspiration of this was taken from the famous *Napster* file-sharing system using a TCP/IP socket connection.

Napster had a major impact in the rapid growth of the internet bandwidth and when it was shut down by the US government then the bubble did burst and many big companies went bankrupt.

Clearly this technology had a significant impact in the history of internet. The objective of this lab is to learn about that technology and implement a simpler version of that.

Introduction

Here is a brief quick overview of the FTP server and client sides.

FTPserver	**FTPclient**
1. Check for correct number of arguments 2. Create SererSocket 3. Start a thread for checking the QUIT command from server terminal 4. As long as the server is running, try to create a client thread whose constructor sits on the blocking accept of the server socket 5. After a connection is established with a client in a thread, keep checking if server is running then sit on a blocking accept of another client thread 6. Based on the command from client, call the appropriate method to handle that action and send relevant information back to client 7. If client closes the connection with BYE command or the connection gets terminated somehow then the server maintains statistics accordingly	1. Check for correct number of arguments 2. Connect to the server 3. Wait for getting a command form user 4. Once a connection has been established then it allows to send commands to the server 5. Handle the user input based on the command 6. If client wants to exit with BYE command then the program closes after closing the socket.

Flow Chart of FTPserver

Please see the `FlowChart_FTPserver.pdf` file in this directory.

Flow Chart of FTPclient

Please see the `FlowChart_FTPclient.pdf` file in this directory.

Java Source Code

Please see `4DN4_lab2\client` and `4DN4_lab2\server` folders in this directory.

Experimental Results

1. When server is started, it shows the valid port range and asks for a port number:

```
Output - FTPserver (run)
run:
Valid range for port number = [1001, 65535]
Enter port number for the server :
```

2. When serer is given port number out of range then the server terminal shows that the port number is not valid and shows the valid port number range again.

```
Output - FTPserver (run)
run:
Valid range for port number = [1001, 65535]
Enter port number for the server : 1000
The port number you have entered is invalid
Allowed range of port number: [1001, 65535]
Next time, please enter a port number within the valid range.
Closing FTPserver application ...

BUILD SUCCESSFUL (total time: 2 seconds)
```

```
Output - FTPserver (run)
run:
Valid range for port number = [1001, 65535]
Enter port number for the server : 65536
The port number you have entered is invalid
Allowed range of port number: [1001, 65535]
Next time, please enter a port number within the valid range.
Closing FTPserver application ...

BUILD SUCCESSFUL (total time: 32 seconds)
```

3. When serer is given port number within the range, then it tells that the argument check has passed and the server has been created in that port. Also it shows in the server terminal that what are the available command(s) from the server terminal.

```
Output - FTPserver (run)
run:
Valid range for port number = [1001, 65535]
Enter port number for the server : 1234
Port number is within valid range: [1001, 65535]
Argument check passed and obtained server port : 1234
Server started at port : 1234

Here is the supported command from FTP Server terminal:
-----------------------------------------------
<COMMAND>      <DESCRIPTION>
QUIT           Exit from the server after closing
               the connection(s) with the
               clients, if there are any.
-----------------------------------------------
```

4. When client is started then it first asks for an IP address:

```
Output
FTPserver (run)   FTPclient (run)
run:
Enter Server IP Address :
```

5. When client gives an IP address then it asks for a port number of server:

```
Output
FTPserver (run)   FTPclient (run)
run:
Enter Server IP Address : a.b.c.d
Enter Server Port Number :
```

6. When an invalid IP and valid port number is given in the client terminal to connect to the server then it throws an exception and tells the client that the server IP address is in wrong format and shows the correct format.

```
Output
FTPserver (run)    FTPclient (run)

run:
Enter Server IP Address : a.b.c.d
Enter Server Port Number : 1234

java.net.UnknownHostException: a.b.c.d

The server IP address entered, is in incorrect format.
Correct format for server IP: <int>.<int>.<int>.<int>
For example: 1.2.3.4
Next Time, please try with a valid server IP address.
Closing application...


BUILD SUCCESSFUL (total time: 2 minutes 14 seconds)
```

7. When a server IP of correct format and port number is in correct but there is no server running in that IP then it shows the NoRouteToHostException and tells the client that there is no server hosted at this destination.

```
Output
FTPserver (run)    FTPclient (run)

run:
Enter Server IP Address : 1.2.3.4
Enter Server Port Number : 1234

java.net.NoRouteToHostException: No route to host: connect

FAILED to connect to server!
There is no route to host : 1.2.3.4:1234
Please try with a IP address where a server is hosted.
Thank you.
_____


BUILD SUCCESSFUL (total time: 23 seconds)
```

8. When the server ip is correct but port number is not then it catches that and shows some tips to troubleshoot.

```
Output
FTPserver (run)    FTPclient (run)

run:
Enter Server IP Address : 127.0.0.1
Enter Server Port Number : 1111

java.net.ConnectException: Connection refused: connect

FAILED to connect to server!
Quick troubleshooting tips:
Please check the followings
        1. Server is running
        2. Server IP address is correct
        3. Server Port number is correct
Please try to start the client after troubleshooting.
Thank you.
_____________________________________
```

9. When client gives the correct ip and port then it connects with the server and shows the client all the available commands from client terminal and waits for a command from client. Also the server shows that there is a client tread connect to the server.

```
Output
FTPserver (run)    FTPclient (run)

Created & started Thread : Thread-1
Client address           = 127.0.0.1
Client port              = 49175
Thread ID                = Thread-1
Connected Client Count   = 1
File System Directory    = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------
```

Output

FTPserver (run) | FTPclient (run)

```
run:
Enter Server IP Address : 127.0.0.1
Enter Server Port Number : 1234
Connected to server at:
IP Address : 127.0.0.1
Port       : 1234

Here are the supported commands from FTP Client terminal:
-----------------------------------------------------------------
<COMMAND>           <DESCRIPTION>
HELP                Lists all the supported commands and show usage
LIST-ALL            List the files and their sizes from
                    the server's current music directory.
LIST-DETAILS        List the files and their sizes from
                    the server's current music directory.
LIST-NAMES          List the file names from
                    the server's current music directory.
READ <filename>     The client requests the file
                    specified by 'filename' and stores
                    it in it's current working directory.
WRITE <filename>    The client writes the file
                    specified by 'filename' into the
                    server's appropriate directory.
BYE                 Exit from the client after closing
                    the connection server, if one exists.
-----------------------------------------------------------------


Please Enter Command :
```

10. When another client is started and it tries to connect to server with correct IP and/or port number:

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2

run:
Enter Server IP Address : 127.0.0.1
Enter Server Port Number : 1234
Connected to server at:
IP Address : 127.0.0.1
Port       : 1234

Here are the supported commands from FTP Client terminal:
-----------------------------------------------------------------
<COMMAND>           <DESCRIPTION>
HELP                Lists all the supported commands and show usage
LIST-ALL            List the files and their sizes from
                    the server's current music directory.
LIST-DETAILS        List the files and their sizes from
                    the server's current music directory.
LIST-NAMES          List the file names from
                    the server's current music directory.
READ <filename>     The client requests the file
                    specified by 'filename' and stores
                    it in it's current working directory.
WRITE <filename>    The client writes the file
                    specified by 'filename' into the
                    server's appropriate directory.
BYE                 Exit from the client after closing
                    the connection server, if one exists.
-----------------------------------------------------------------


Please Enter Command :
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2

-------------------------------------

Created & started Thread : Thread-1
Client address          = 127.0.0.1
Client port             = 49175
Thread ID               = Thread-1
Connected Client Count  = 1
File System Directory   = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------


-------------------------------------

Created & started Thread : Thread-2
Client address          = 127.0.0.1
Client port             = 49176
Thread ID               = Thread-2
Connected Client Count  = 2
File System Directory   = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------
```

11. When a 3rd client is started with correct IP and Port number:

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

run:
Enter Server IP Address : 127.0.0.1
Enter Server Port Number : 1234
Connected to server at:
IP Address : 127.0.0.1
Port       : 1234

Here are the supported commands from FTP Client terminal:
-----------------------------------------------------------------
<COMMAND>           <DESCRIPTION>
HELP                Lists all the supported commands and show usage
LIST-ALL            List the files and their sizes from
                    the server's current music directory.
LIST-DETAILS        List the files and their sizes from
                    the server's current music directory.
LIST-NAMES          List the file names from
                    the server's current music directory.
READ <filename>     The client requests the file
                    specified by 'filename' and stores
                    it in it's current working directory.
WRITE <filename>    The client writes the file
                    specified by 'filename' into the
                    server's appropriate directory.
BYE                 Exit from the client after closing
                    the connection server, if one exists.
-----------------------------------------------------------------


Please Enter Command :
```

Output

FTPserver (run) | FTPclient (run) | FTPclient (run) #2 | FTPclient (run) #3

```
-------------------------------------

Created & started Thread : Thread-1
Client address           = 127.0.0.1
Client port              = 49175
Thread ID                = Thread-1
Connected Client Count   = 1
File System Directory    = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------


-------------------------------------

Created & started Thread : Thread-2
Client address           = 127.0.0.1
Client port              = 49176
Thread ID                = Thread-2
Connected Client Count   = 2
File System Directory    = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------


-------------------------------------

Created & started Thread : Thread-3
Client address           = 127.0.0.1
Client port              = 49177
Thread ID                = Thread-3
Connected Client Count   = 3
File System Directory    = C:/Users/Anjoom/Desktop/server/pub/MUSIC/

----------------------------------------
```

12. Now if the client-1 sends 'LIST-ALL' command to the server then it gets the correct files with their sizes as show from terminal as well. The server shows the received and handled command from client in which thread.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Anjoom>cd Desktop\server\pub\MUSIC\

C:\Users\Anjoom\Desktop\server\pub\MUSIC>dir
 Volume in drive C is WINDOWS_7
 Volume Serial Number is 7C27-CBD3

 Directory of C:\Users\Anjoom\Desktop\server\pub\MUSIC

27/03/2014  03:50 PM    <DIR>          .
27/03/2014  03:50 PM    <DIR>          ..
20/03/2014  07:45 AM                 7 01-01-test-text.txt
17/03/2008  08:40 AM           402,121 MUSIC-braveheart.mp3
17/03/2008  08:24 AM           609,672 MUSIC-desprado.mp3
27/03/2014  03:50 PM           464,896 MUSIC-funky_town.mp3
17/03/2008  08:15 AM           431,880 MUSIC-tokey_drift.mp3
               5 File(s)      1,908,576 bytes
               2 Dir(s)  69,362,413,568 bytes free

C:\Users\Anjoom\Desktop\server\pub\MUSIC>
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : LIST-ALL
--> processListCommand START

LIST-ALL :
_________________
01-01-test-text.txt          7        Bytes
MUSIC-braveheart.mp3         402121   Bytes
MUSIC-desprado.mp3           609672   Bytes
MUSIC-funky_town.mp3         464896   Bytes
MUSIC-tokey_drift.mp3        431880   Bytes
_________________

<-- processListCommand DONE

Please Enter Command :
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

----------------------------------------

Thread-1 : command = LIST-ALL    Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
```

13. Now if the client-3 sends 'LIST-DETAILS' command to the server then it gets the correct files with their sizes as show from terminal as well. The server shows the received and handled command from client in which thread.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Anjoom>cd Desktop\server\pub\MUSIC\

C:\Users\Anjoom\Desktop\server\pub\MUSIC>dir
 Volume in drive C is WINDOWS_7
 Volume Serial Number is 7C27-CBD3

 Directory of C:\Users\Anjoom\Desktop\server\pub\MUSIC

27/03/2014  03:50 PM    <DIR>          .
27/03/2014  03:50 PM    <DIR>          ..
20/03/2014  07:45 AM                 7 01-01-test-text.txt
17/03/2008  08:40 AM           402,121 MUSIC-braveheart.mp3
17/03/2008  08:24 AM           609,672 MUSIC-desprado.mp3
27/03/2014  03:50 PM           464,896 MUSIC-funky_town.mp3
17/03/2008  08:15 AM           431,880 MUSIC-tokey_drift.mp3
               5 File(s)      1,908,576 bytes
               2 Dir(s)  69,362,413,568 bytes free

C:\Users\Anjoom\Desktop\server\pub\MUSIC>
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : LIST-DETAILS
--> processListCommand START

LIST-DETAILS :

_________________
01-01-test-text.txt          7        Bytes
MUSIC-braveheart.mp3         402121   Bytes
MUSIC-desprado.mp3           609672   Bytes
MUSIC-funky_town.mp3         464896   Bytes
MUSIC-tokey_drift.mp3        431880   Bytes

_________________

<-- processListCommand DONE

Please Enter Command :
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Thread-1 : command = LIST-ALL     Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
Thread-3 : command = LIST-DETAILS        Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
```

14. When the client-2 send 'LIST-NAMES' command to the server then it gets the correct files names only without their sizes as show from terminal as well. The server shows the received and handled command from client in which thread. This takes much less time compared to 'LIST-ALL'

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\Anjoom>cd Desktop\server\pub\MUSIC\

C:\Users\Anjoom\Desktop\server\pub\MUSIC>dir
 Volume in drive C is WINDOWS_7
 Volume Serial Number is 7C27-CBD3

 Directory of C:\Users\Anjoom\Desktop\server\pub\MUSIC

27/03/2014  03:50 PM    <DIR>          .
27/03/2014  03:50 PM    <DIR>          ..
20/03/2014  07:45 AM                 7 01-01-test-text.txt
17/03/2008  08:40 AM           402,121 MUSIC-braveheart.mp3
17/03/2008  08:24 AM           609,672 MUSIC-desprado.mp3
27/03/2014  03:50 PM           464,896 MUSIC-funky_town.mp3
17/03/2008  08:15 AM           431,880 MUSIC-tokey_drift.mp3
               5 File(s)      1,908,576 bytes
               2 Dir(s)  69,362,413,568 bytes free

C:\Users\Anjoom\Desktop\server\pub\MUSIC>
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : LIST-NAMES
--> processListCommand START

LIST-NAMES :

_________________
01-01-test-text.txt
MUSIC-braveheart.mp3
MUSIC-desprado.mp3
MUSIC-funky_town.mp3
MUSIC-tokey_drift.mp3
_________________

<-- processListCommand DONE

Please Enter Command :
```

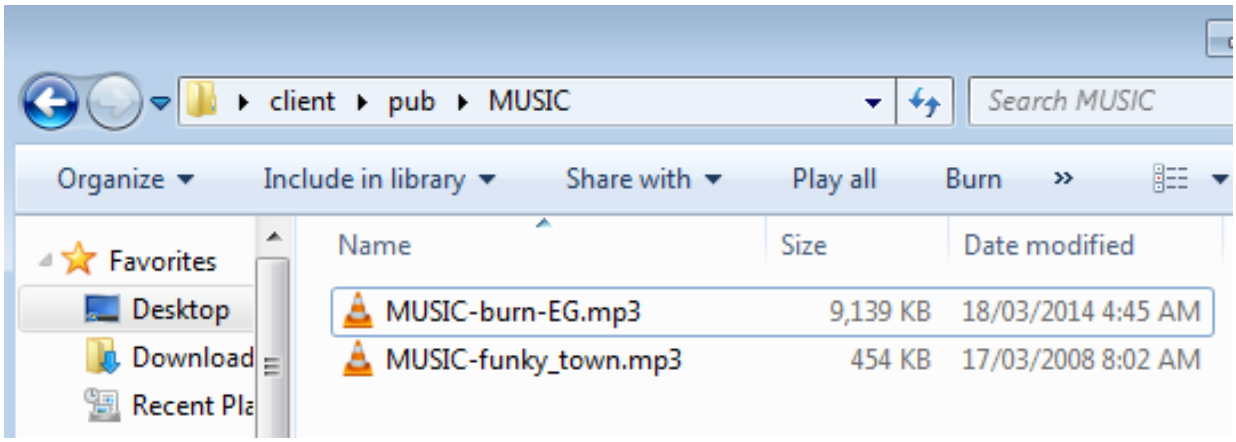
```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Thread-1 : command = LIST-ALL    Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
Thread-3 : command = LIST-DETAILS        Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
Thread-2 : command = LIST-NAMES  Connected Client Count  = 3
-->
handleListNamesCommand START
handleListNamesCommand END
<--
```

15. When the client-3 passes ‘HELP’ command then it displays all the available commands in the client terminal but there is nothing in the server terminal.

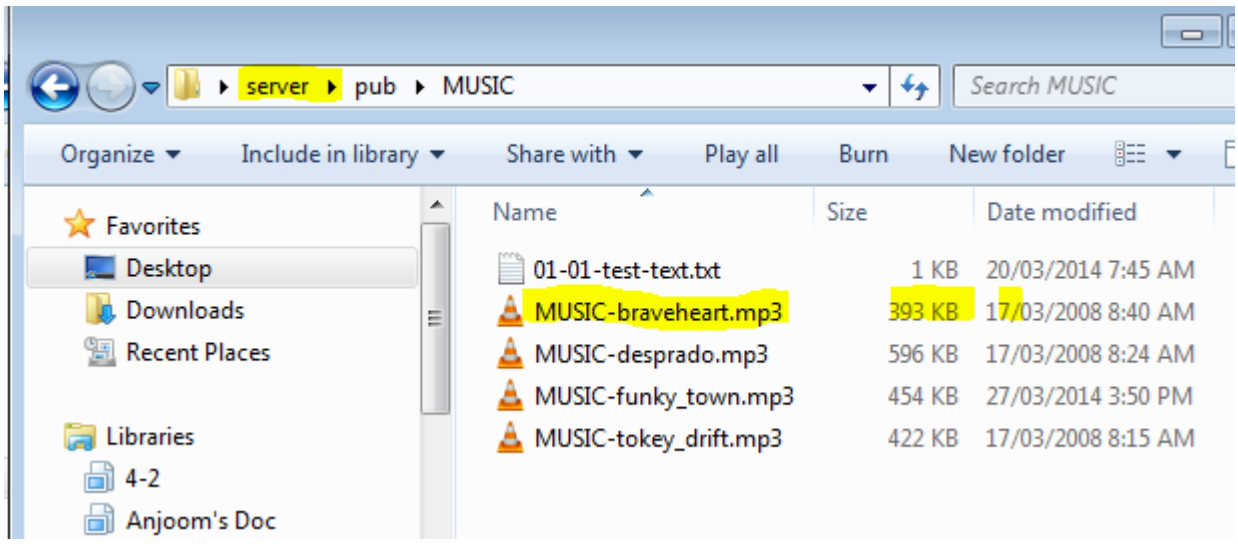
```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : help

Here are the supported commands from FTP Client terminal:
-----------------------------------------------------------------
<COMMAND>           <DESCRIPTION>
HELP                Lists all the supported commands and show usage
LIST-ALL            List the files and their sizes from
                    the server's current music directory.
LIST-DETAILS        List the files and their sizes from
                    the server's current music directory.
LIST-NAMES          List the file names from
                    the server's current music directory.
READ <filename>     The client requests the file
                    specified by 'filename' and stores
                    it in it's current working directory.
WRITE <filename>    The client writes the file
                    specified by 'filename' into the
                    server's appropriate directory.
BYE                 Exit from the client after closing
                    the connection server, if one exists.
-----------------------------------------------------------------


Please Enter Command :
```

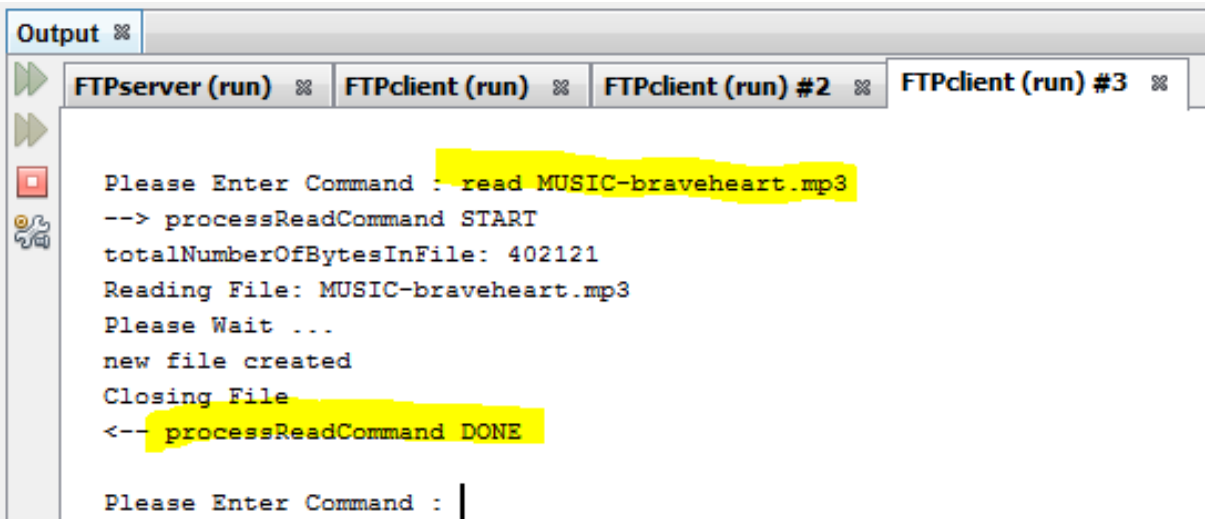
16. When the client-3 wants to read a file and passes 'read MUSIC-braveheart.mp3' command then reads the file from server and store it in the client directory.

Client directory before read:



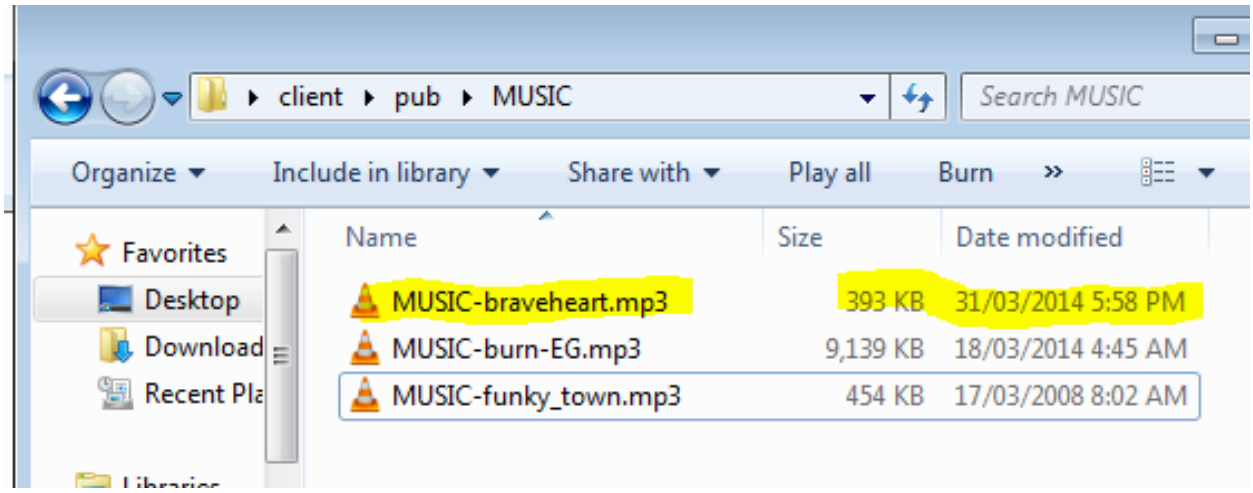
Server directory:





```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Thread-3 : command = READ          Connected Client Count  = 3
-->
handleReadCommand START
rdFile = 'MUSIC-braveheart.mp3'
totalRdByteCount = 402121
handleReadCommand END
<--
```

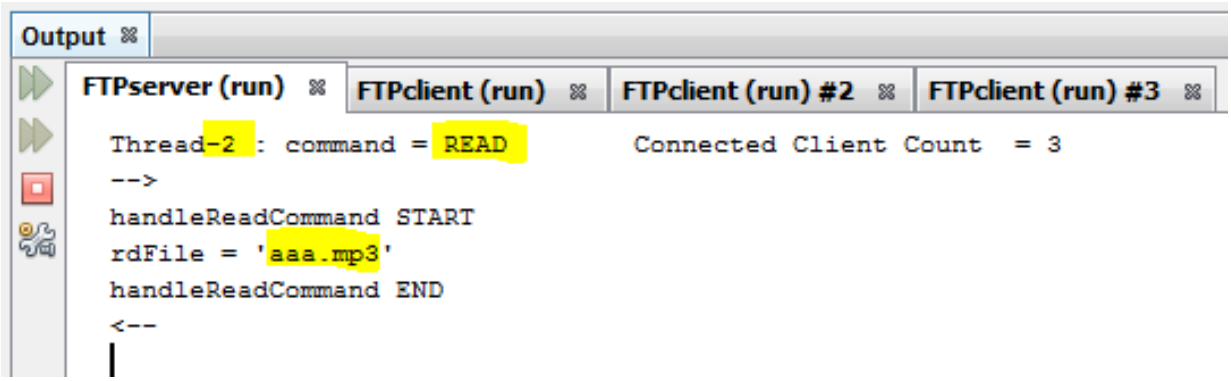
Client directory after read:



And the file plays similarly the one in the server.

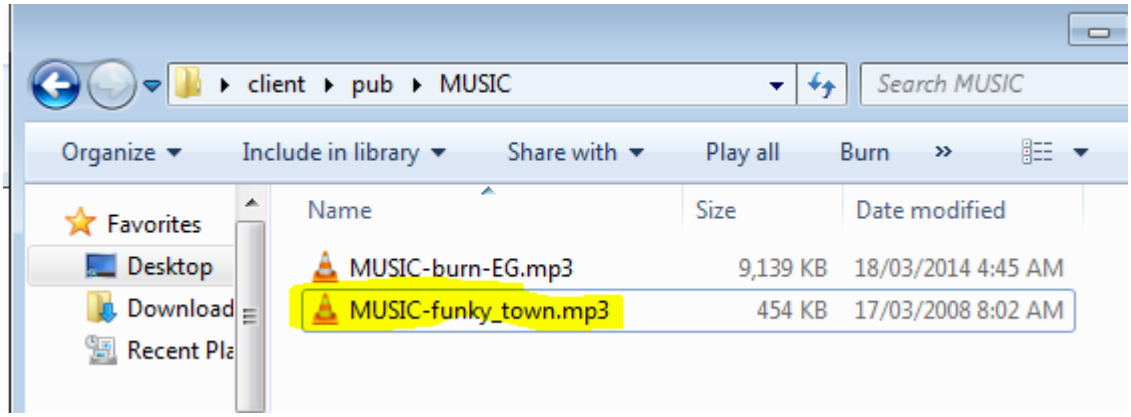
17. When the client-2 wants to read a file that does not exist and passes 'read aaa.mp3' command then it shows error message saying that there is no such file in the server.

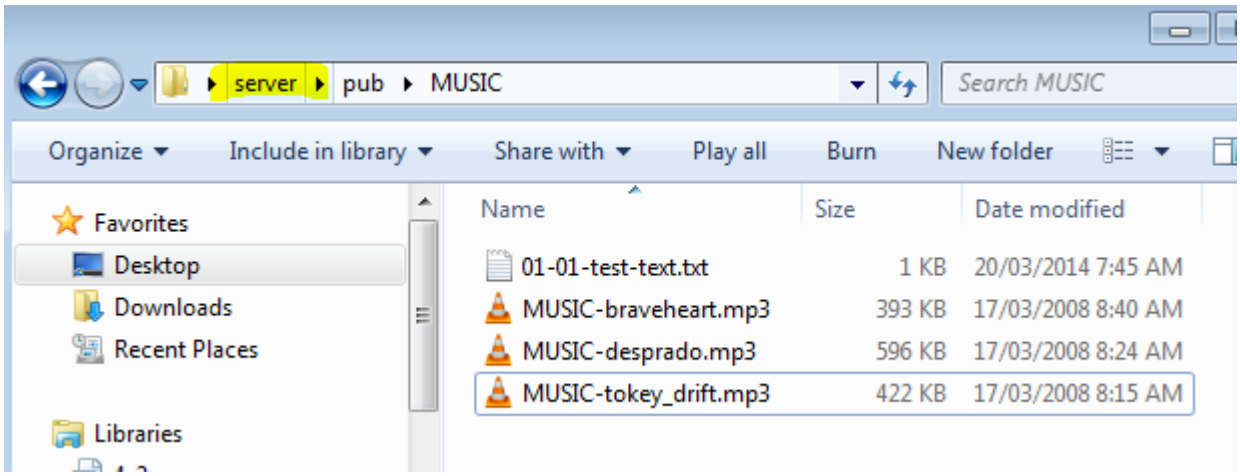
```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : read aaa.mp3
--> processReadCommand START
In the server, there is no such file named: aaa.mp3
<-- processReadCommand DONE

Please Enter Command :
```



18. When the client-1 wants to write a file and passes 'write MUSIC-funky_town.mp3' command then writes the file from the client to the server directory.

Before write:





Output

FTPserver (run) | FTPclient (run) | FTPclient (run) #2 | FTPclient (run) #3

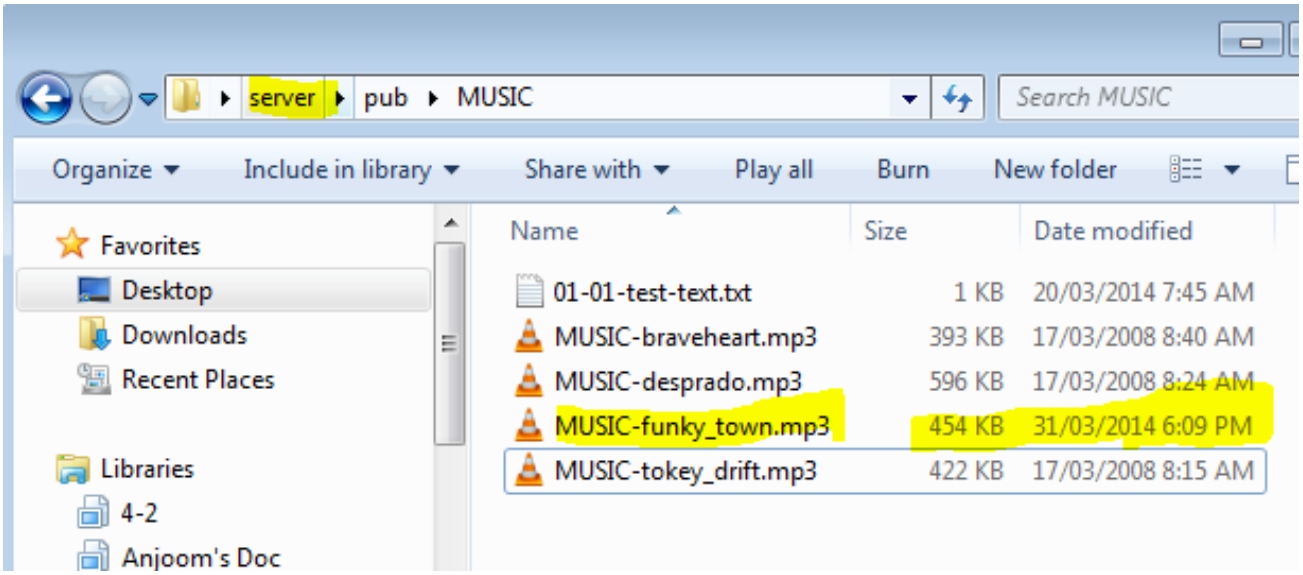
```
Thread-1 : command = LIST-NAMES  Connected Client Count  = 3
-->
handleListNamesCommand START
handleListNamesCommand END
<--
Thread-1 : command = WRITE       Connected Client Count  = 3
-->
handleWriteCommand START
received wrFileName: 'MUSIC-funky_town.mp3'
totalNumberOfBytesInFile: 464344
Writing....
newFilePath = 'C:/Users/Anjoom/Desktop/server/pub/MUSIC/MUSIC-funky_town.mp3'
totalBytesWritten: 464344
Closed new file
handleWriteCommand END
<--
Thread-1 : command = LIST-ALL    Connected Client Count  = 3
-->
handleListDetailsCommand START
handleListDetailsCommand END
<--
```

Output

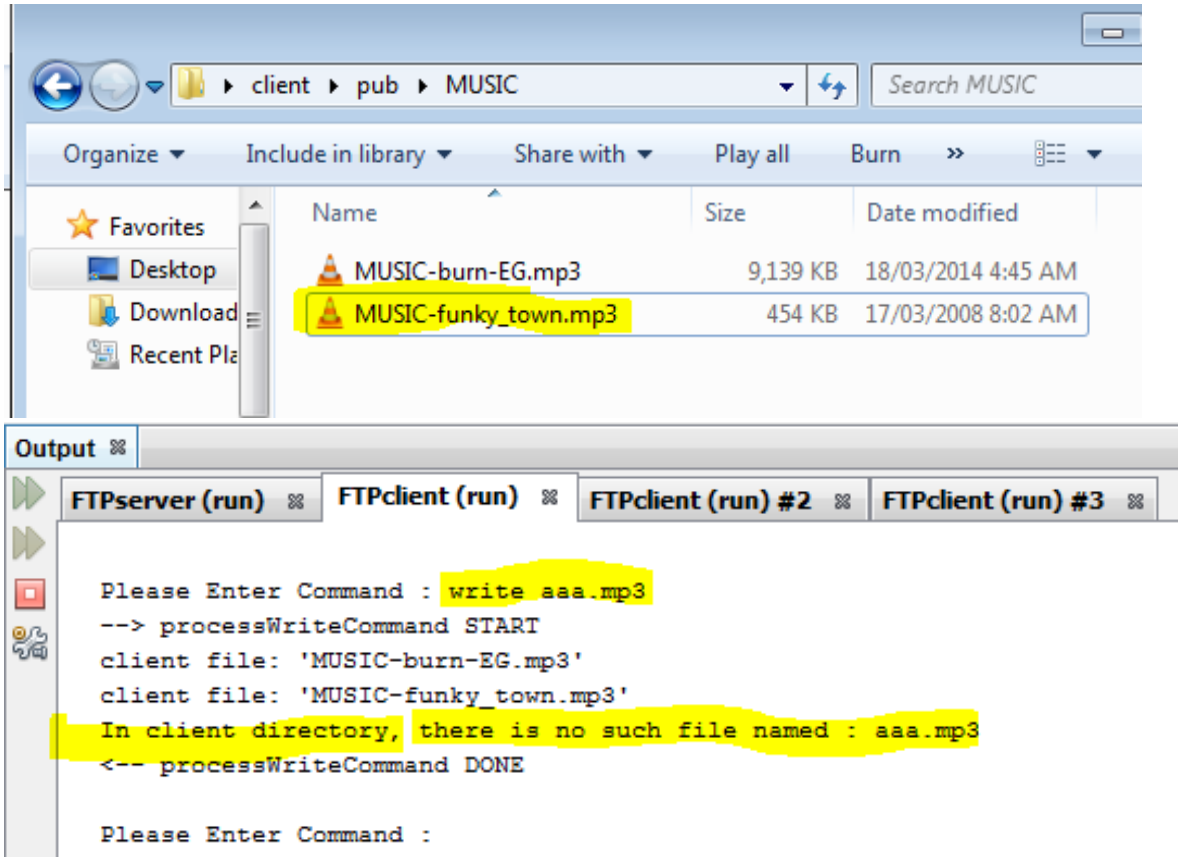
FTPserver (run) | FTPclient (run) | FTPclient (run) #2 | FTPclient (run) #3

```
Please Enter Command : list-names
--> processListCommand START

LIST-NAMES :

_________________
01-01-test-text.txt
MUSIC-braveheart.mp3
MUSIC-desprado.mp3
MUSIC-tokey_drift.mp3
_________________

<-- processListCommand DONE

Please Enter Command : write MUSIC-funky_town.mp3
--> processWriteCommand START
client file: 'MUSIC-burn-EG.mp3'
client file: 'MUSIC-funky_town.mp3'
wrFileSize = 464344
Writting...
totalRdByteCount = 464344
<-- processWriteCommand DONE

Please Enter Command : list-all
--> processListCommand START

LIST-ALL :

_________________
01-01-test-text.txt             7        Bytes
MUSIC-braveheart.mp3            402121   Bytes
MUSIC-desprado.mp3              609672   Bytes
MUSIC-funky_town.mp3            464896   Bytes
MUSIC-tokey_drift.mp3           431880   Bytes
_________________

<-- processListCommand DONE

Please Enter Command :
```

After write:



19. When client-1 tries to write a file 'aaa.mp3' that does not exist then the client goes through all the files in client directory and when it does not find it the it tells the client that the file does not exist. Server does not see this command.



20. Client-2 gives BYE command then it closes the client application and closes the corresponding thread in the server and decrements the connected clients count.

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : bye
Client closed connection sucessfully!
BUILD SUCCESSFUL (total time: 41 minutes 15 seconds)
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Thread-2 : command = BYE          Connected Client Count  = 3
-->
handleByeCommand START
Thread-2 is closing ...
BEFORE: Connected Client Count = 3
AFTER:  Connected Client Count = 2
THREAD CLOSE: SUCCESSFUL
handleByeCommand END
<--
```

21. When client-3 is stopped forcefully by clicking the red stop button in Netbeans then the corresponding thread closes in the server safely.

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : BUILD STOPPED (total time: 18 seconds)
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

java.net.SocketException: Connection reset

SocketException:
Connected Client Count  = 1     Thread-3 : Socket has been closed due to termination of client
```

22. Now if the server is closed with QUIT command then server closes but the clinet-1 does not crash immediately.

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

QUIT
Server Command received:QUIT
Number of connected clients when server is getting closed = 1
All client connections has been closed
SERVER CLOSED SUCCESSFULLY
BUILD SUCCESSFUL (total time: 3 minutes 28 seconds)
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

-----------------------------------------------------------------------------

Please Enter Command :
```

23. Now if the client-1 tries to send any command then it will close the client application safely after throwing a connection reset exception as the server is not running.

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : list-all
--> processListCommand START

java.net.SocketException: Connection reset by peer: socket write error

BUILD SUCCESSFUL (total time: 5 minutes 17 seconds)
```

24. When a big file is being transferred and the server is closed with QUIT command from server terminal then the program catches that exception and gives some useful message. Also it cleans up the partially transferred file. For example, when a client is writing a big file to server with 'write MUSIC-burn-EG.mp3'

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Please Enter Command : write MUSIC-burn-EG.mp3
--> processWriteCommand START
client file: 'MUSIC-burn-EG.mp3'

java.net.SocketException: Connection reset by peer: socket write error

SocketException: Connection got terminated or Server is no longer running.
BUILD SUCCESSFUL (total time: 6 minutes 37 seconds)
```

```
Output
FTPserver (run)   FTPclient (run)   FTPclient (run) #2   FTPclient (run) #3

Thread-1 : command = WRITE        Connected Client Count  = 3
-->
handleWriteCommand START
received wrFileName: 'MUSIC-burn-EG.mp3'
quit
Server Command received:QUIT
Number of connected clients when server is getting closed = 3
All client connections has been closed
SERVER CLOSED SUCCESSFULLY
BUILD SUCCESSFUL (total time: 6 minutes 21 seconds)
```

Customization

1. Extensive error handling has been implemented
2. Three more client commands has been implemented : HELP, LIST-NAMES, LIST-DETAILS.
    a. HELP: Lists all the supported commands and show usage
    b. LIST-NAMES: List the file names from the server's current music directory without the size. So this command returns the file list much more faster than the LIST-DETAILS or LIST-ALL which need to compute the size of each file.
3. All the available commands are displayed in both server and client
4. The program has been made interactive with appropriate messages for better experience

Issue/Problem Encountered

While designing the java application initially I had all the fields and methods declared as static which was giving me error when I have more than 1 client. This made sense as static variables have only one memory for all the instances of the clients.

To fix this, I have removed the static keyword from all the fields and methods of the client. Now every time a client is launched it will create a new object in different memory space. So there will not be any conflict.

On the other hand, I have left most of the server fields and methods as static as we are only launching one FTP server for this lab.

Demo information

TA: Maryam Razaee

Date: Thursday March 27, 2014

Time: 3:30 pm

Conclusion

This was an interesting and useful lab. This helped a lot clearing the socket file transfer concepts, the file operations in java, the error handling and many other important concepts.

This gives the understanding of an aspect of internet which had significant impact in the history of internet.

THE END