Shanghai Jiao Tong University

CS339 Computer Networking

Socket Programming

Chao Gao

5142029014

dimon-gao@sjtu.edu.cn

April 5, 2017

1 Introduction

Recently, we learn something about socket programming on the networking course, and this assignment is a good chance for us to get familiar with the socket programming. Socket is like a door between application layer and transportation layer. And this project is based on TCP protocol.
Recap the requirement.
Implement C/S model:
1) Server listens to a given port(>1024);
2) The client initiates a TCP connection to the server
(hostname or IP address of the server as the input,default port numbers);
3) The client send a request to download a file/text;
4) The server respond with the file/text;
5) The client save the file to local directory.
6) Repeat step 3) 4) 5) until 'esc' is pressed, client tear down the TCP connection.

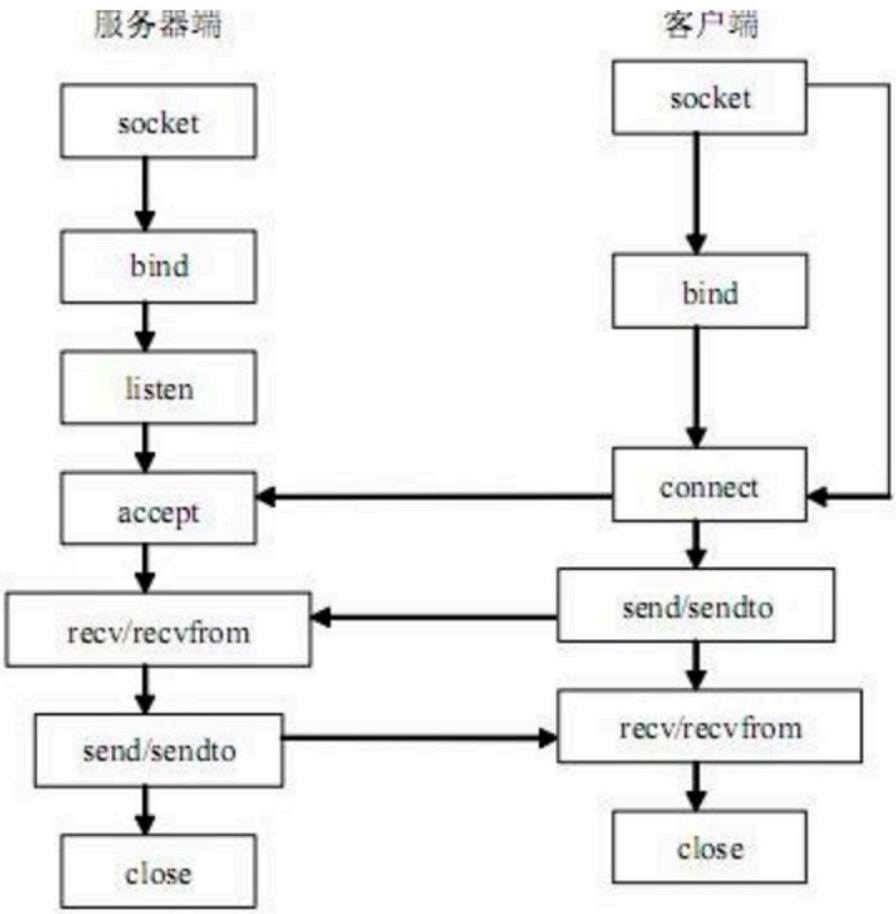
2 File Structure

source ———— source code
runnable ———— compiled java byte codes
demo.mov ———— test video
report.pdf ———— report file

3 Environment

OS: Mac OS X EI Capitan 10.11.4.
Java version: 1.8.0_121.

4 Implementation



4.1 Create socket

For Client
establish the socket by host and port(default "localhost" and 2680)

```
s = new Socket(host, port);
```

For Server
Create the server socket by the same port, listening from the client.

When accepted, create a new socket establishing a connection to the socket in the client side.

```
final ServerSocket ss = new ServerSocket(2680);
try {
    System.out.println("server start...");
    Socket socket = ss.accept();
    System.out.println("connect ok");
}
```

4.2 Send file name

This part is implemented in the client side.
Read file name from user.

```
BufferedReader inFromUser = new BufferedReader(new
            InputStreamReader(System.in));
file = inFromUser.readLine();
```

Send file name by socket.

```
DataOutputStream dos = null;
dos = new DataOutputStream(s.getOutputStream());
dos.writeBytes(file + '\n');
```

4.3 Receive file name and find the file

Thie part is implemented in the server side.
Receive file name.

```
BufferedReader inFromClient = new BufferedReader(new
        InputStreamReader(socket.getInputStream()));
String fileName = inFromClient.readLine();
```

Find the file in the server side.

```
File file = new File(fileName);
FileInputStream dis = new FileInputStream(file);
```

4.4 Send and receive the file

For server, send the file.

```
byte[] buf = new byte[4096];
int len = 0;
DataOutputStream dos = new DataOutputStream(socket.
            getOutputStream());
dos.writeLong((long) file.length());
dos.flush();
while(true) {
    int read = 0;
    if(dis != null) {
        read = dis.read(buf);
    }
    if(read == -1) {
        break;
    }
dos.write(buf, 0, read);
```

For client, receive the file.

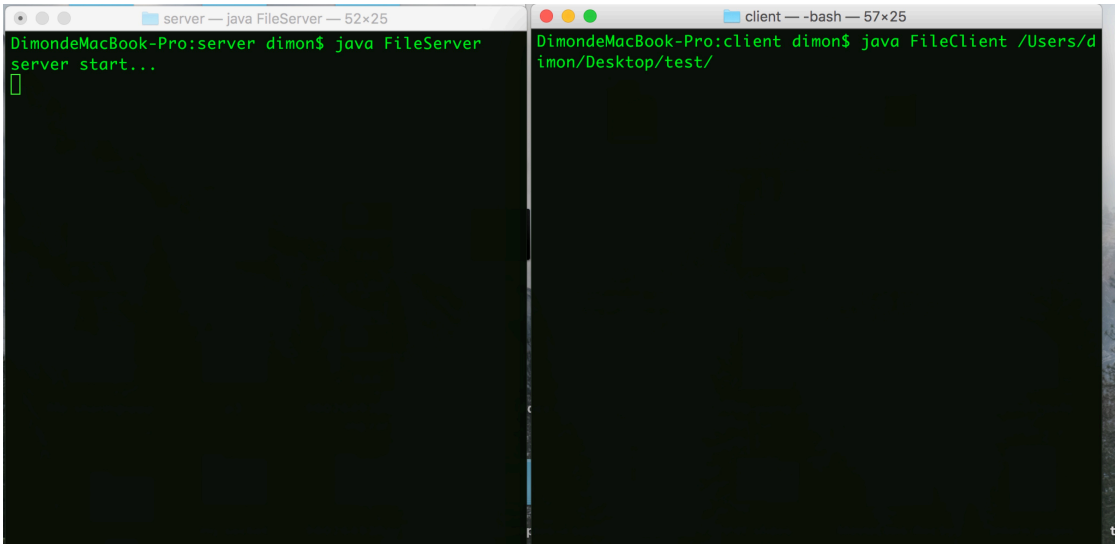
```
long len = dis.readLong();
byte[] buf = new byte[4096];
int read = 0;
int totalRead = 0;
int remaining = (int)len;
while((read = dis.read(buf, 0, Math.min(buf.length,
```

```
                remaining))) > 0) {
        totalRead += read;
        remaining -= read;
        System.out.println("read " + totalRead + " bytes.");
        fileout.write(buf, 0, read);
    }
```

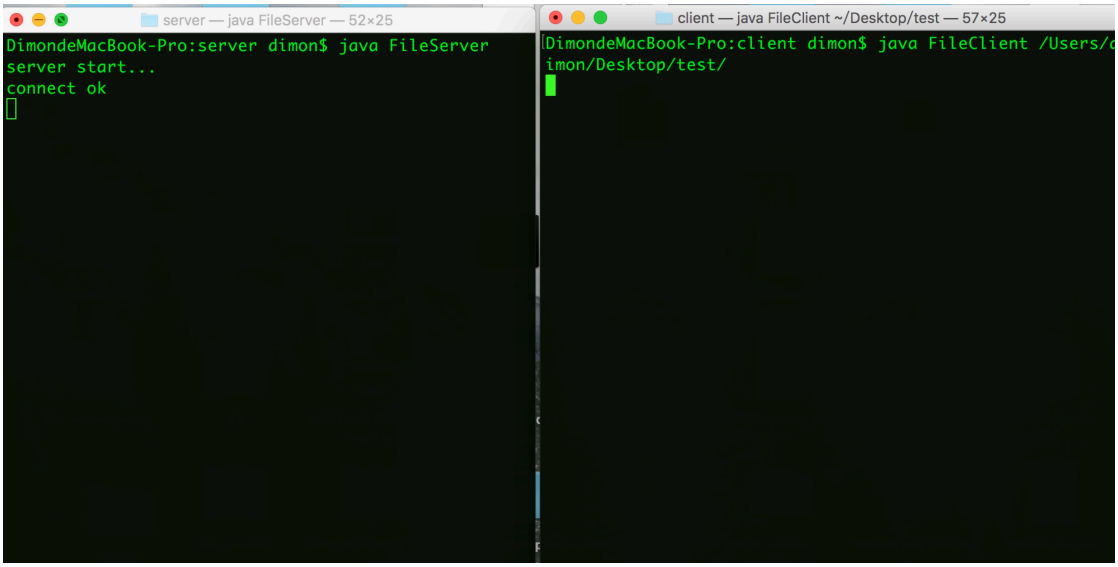
5 How to run

Open two terminals, cd to the server directory and client directory
Execute the programs. (for client can decide the file savepath, default is /Users/dimon/Desktop/)

Server creates socket and listen from client.

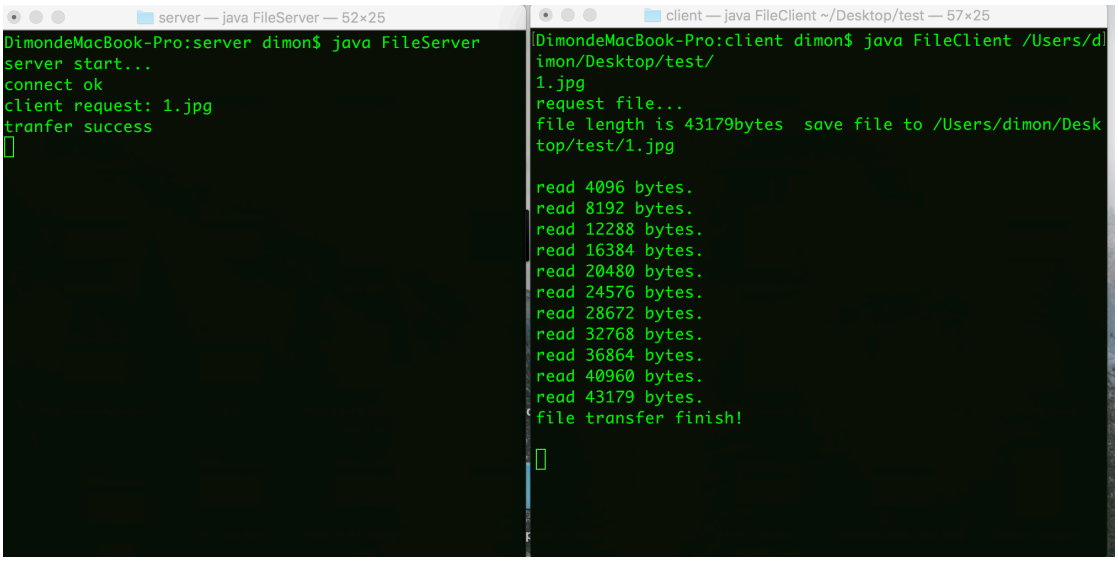


Client creates socket and establishes the connection.

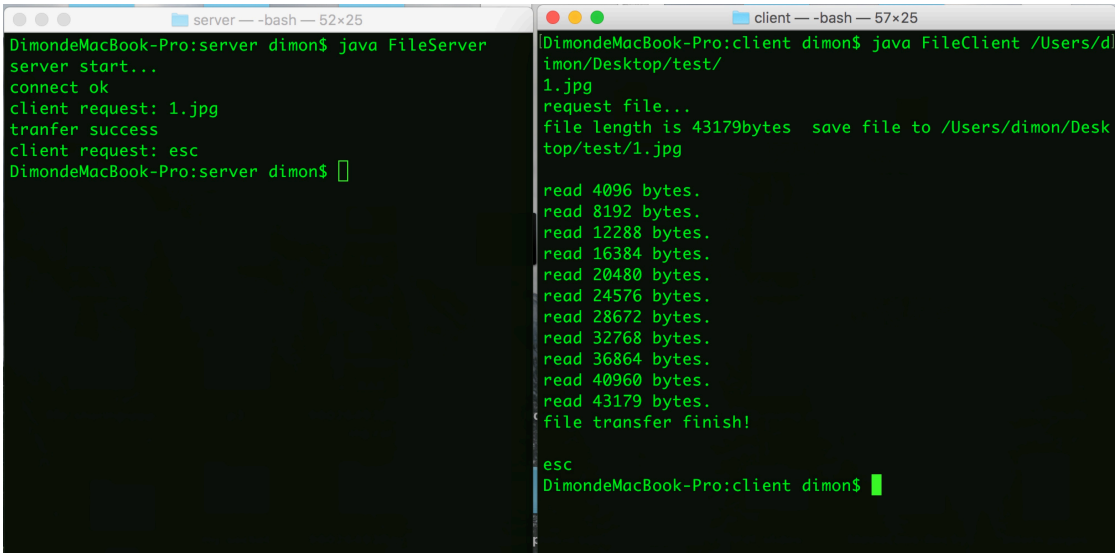


Type the file name. (the file should already in the server side, otherwise may cause problems).

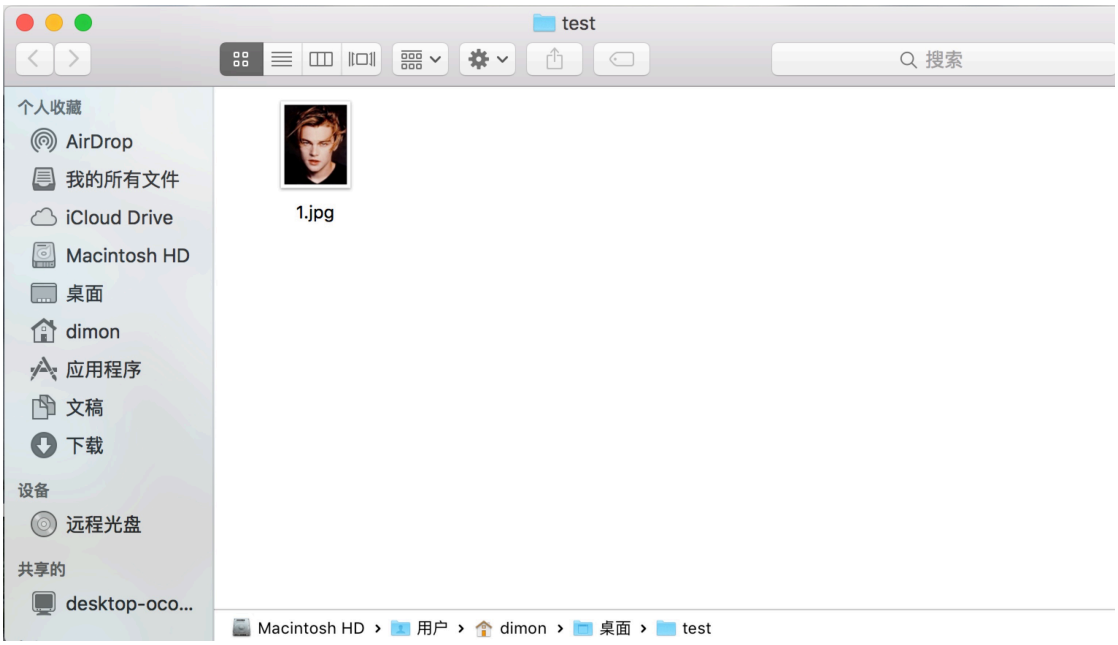
File tranfer.



Type esc to tear down the TCP connection.



Test result, file has been the correct folder.



Since the screenshots are not intuitive, so I made a simple demo to show the

process, you can find and watch it in my submission.

6 Problems and experience

6.1 whole process

Experience is the best teacher. Through this assignment, I get to know more about the process of TCP socket communication and have a clear thought. So that I can program without too much problems.
For the client side.
1. Read file name from user.
2. Send file name to server.
3. Receive the file from server.
For the server side.
1. Receive the file name and find the file.
2. Send the file to client.

6.2 File operations

For reading and writing file, I have some problems. I searched for some sample codes, and find out I can use **FileOutputStream** and **FileInputStream**. And here I define a buffer for larger files, this way helps to use less space and enhance program robustness.

6.3 Multi-thread for server

Because the project requires us to have a loop for the C/S model, and loop exits when 'esc' is typed, we have to consider multi-thread programming. At first, I just check 'esc' for the situation to exit the loop. However, server always executes only once and exits out automatically. Finally I figure out that for server, we should use threads to receive the request from the client side.

6.4 Weakness

This program still has some defects. The file which client wants to download must already be in the server directory, I do not figure out how to deal with the exception. And the file can not be too large, I use int to store the size of file.

7 Acknowledgement

Thanks a lot to Prof.Shen for the impressive teaching about socket programming and the detailed guidance about this assignment.