CS 580 Client-Server Programming
Spring Semester, 2009
Comments on Assignment 2 Part 1
24 Feb, 2009

Names

```
sendCommandThroughNetworkLayer(args)
char TempChar;
String FormattedString = "";
int portnumber = 8010;
```

Thin client Layer

```
public class SDwitterClient {

        public String userLogin(String userName, String passWord)
{
                        return serverConnection.send("login;screenName:"
                                        + userName + ";password:" + passWord +
";;");
                }
        public String createNewUser(String userName, String passWord) {
                userName = serverConnection.formatString(userName);
                passWord = serverConnection.formatString(passWord);

        if (userNameAvailable(userName)) {
                                return
serverConnection.send("newUser;screenName:"
                                                + userName + ";password:" + passWord +
";;");
                        }
                return "error:Username already in use;;";
        }
```

Not much to test without using Network

Some tests

```
public class SDwitterClientTest {
        @Test
        public void testUserLogin() {
                String result = client.userLogin(uName, uPassword);
                assertEquals(result, "ok:success;;");
                client.quit();
        }

    @Test
        public void testCreateNewUser() {
                assertTrue(client.userNameAvailable(uName));
                result = client.createNewUser(uName, uPassword);
                assertEquals(result, "ok:success;;");
                client.quit();
        }
```

You don't have control over the server, so the second test can only be run once. You need to have control over the server so you can reset it.

You need control over the Server

If you are going to use a server in test

- You need to start it with your tests
- You need it to be in the same initial state

Are you testing the client or the server?

Ask yourself

What code am I trying to test?

A test is not a unit test if

It talks to the database

It communicates over the network

It touches the file system

It can't run correctly at teh same time as any of your other unit tests

You have to do special things to your environment to run it

Michael Feathers

Such test are not Bad

Need to keep them separate

So we have a set of tests that run fast for when we make changes

UpToStream Issues

```
public class UpToStream {
        String server;
        int port;
        public Socket theSocket = null;
        private InputStream fromServer;
        private OutputStream toServer;
        public boolean test;
```

UpToStream Issues

```
public UpToStream(String serverName, int portNumber)
{
        server = serverName;
        port = portNumber;
        try {
                theSocket = new Socket(server, port);
               fromServer = theSocket.getInputStream();
                toServer = theSocket.getOutputStream();
                test = true;
 // this is for testing purposes. test is set to true if the socket connection is established.
// if socket connection is not established, an exception results and the test variable is set to false
              }
              catch (UnknownHostException e)  {
                 test = false;
                System.err.println(e);
              }
              catch (IOException ex)  {
                  test = false;
                  System.err.println(ex);
              }
}
```

UpToStream Issues

```
    public void send(String command)  {
  try {
                toServer.write(command.getBytes());                // sending data to the server.
                toServer.flush();
                test = true;
            }catch (IOException e)  {
                test = false;
                e.printStackTrace();
            }
        }

        public String receive() {
            byte buffer[] = new byte[1000];
            try  {
                fromServer.read(buffer);
            }catch (IOException e) {e.printStackTrace();     }

            String temp = new String(buffer);
            System.out.println("Receiving... " );
            return  temp;
        }
}
```

Issues?

```
protected static String SendMessage(String CurrentMessage) {
    int portnumber = 8010;
    String hostname = "bismarck.sdsu.edu";
    String ServerOutput;
    Socket ServerSocket;
    try {
        ServerSocket = new Socket(hostname, portnumber);
        DataFromServer = new BufferedReader(
                new InputStreamReader(ServerSocket.getInputStream()));
        DataToServer = new OutputStreamWriter(ServerSocket.getOutputStream());
        DataToServer.write(CurrentMessage);
        DataToServer.flush();

        ServerOutput = DataFromServer.readLine();
        return ServerOutput;
    }
    catch(UnknownHostException e) { return "Error in connecting to server!"; }
    catch(IOException e) { return "Error in connecting to server!"; }
}
```

Issues

```
public String uptoChar(char c) throws IOException {
        String result="";
        int readData;
        char readChar;
        while((readData=read())>-1){  //read till EOF
                readChar=(char)readData;
                if(readChar!=c){
                        result=result+readChar;
                }
                else{
                        break;
                }
        };
        return result;
}
```

Reading

```
public String getResponse(){
    return response;
}

public void send(Message message) throws IOException, InterruptedException {
    out.print(message.toString());
    out.flush();

    BufferedReader br = new BufferedReader(new InputStreamReader(rawIn));
    char[] chars = new char[50];
    br.read(chars, 0, 50);

    response = new String(chars);
}
```

This of course does not work.

A bit better, but Still not right

```
int bytesRead = 0;
int bytesToRead=1024;
byte[ ] input = new byte[bytesToRead];
while (bytesRead < bytesToRead) {
    int readSize = in.read(input, bytesRead, bytesToRead - bytesRead);
    if (readSize = -1 ) break;
    bytesRead += readSize;
}
```

We don't know the size so this will not end correctly. What happens if there is more than 1024?

String + verses StringBuilder

```
String result = '';

String result="";
int readData;
...
Some loop
    readData=in.read();
    ...
    result=result + (char) readData;
...
return result;
```

```
StringBuilder result = new StringBuilder(128);

String result="";
int readData;
...
Some loop
    readData=in.read();
    ...
    result=result + (char) readData;
...
return result.toString();
```