

CS/ECE 757 – Computer Networks Fall 2004

Assignment 1: Programming with Sockets in Java

Instructions:

- This assignment is to be completed individually.
- The only help you are allowed are the textbook, the online documentation at www.javasoft.com or www.sun.com, the material on the web which is listed on the CS 757 website, and a book on the Java programming language.
- The grade for this project is weighted as follows:
 - 50% Implementation
 - 20% Functional Tests
 - 30% Write-up

Due Date: September 16, at the beginning of class.

Objective: Implement an *Addressbook* service.

Problem Description:

- The *Addressbook* service is a client-server application. A client sends a query message to the server containing an email address. The server responds to the client with a message that contains the full name which corresponds to the email address.
- A client process *Addressbook_client* submits query messages to the server. The query message contains an email address.
- The server process *Addressbook_Server* runs on a known workstation (say, `cobra.cs.virginia.edu`) and listens on a well-known port number for client requests. When a request comes in, the server looks in a file to find the full name which corresponds to the email address, and sends the full name to the requesting client.
- The following messages format is used for query and response:

Message Type (1 byte)	AString Length (1 byte)	AString (≤ 255 bytes)
--------------------------	----------------------------	--------------------------------

- *Message Type* is set to “Q” (ASCII 81) for queries, and “R” (ASCII 82) for responses.
- *AString Length* is an unsigned 8-bit integer.
- *AString* is a character string with up to 255 characters.

Example (simplified):



Requirements:

- *Addressbook_Server* and *Addressbook_Client* must be able to run on different machines on the Internet.
- The *Address_Client* must be able to access a test server, which is running on iodine.cs.virginia.edu on port 5678.
- The database of the server must be able to handle an arbitrary number of entries.
- The server must be able to process multiple queries. You get full credit if incoming queries are processed sequentially.
- You get 10% extra credit if your server can handle multiple concurrent clients.

Your Task:

1. Implement the *Addressbook* service systems for Unix workstations. Both server and clients are to be implemented exclusively with sockets (Internet domain, stream sockets). The implementation must be done in **JAVA**.
2. Test your client implementation with the server specified in the requirements.
3. Deliver a max 1 page write-up which describes your implementation approach, a description of your test strategy, and known bug list.

Hints:

1. Implement the server process as a background process ("daemon"), which is *listening* on a well-known port number.
2. A client process contacts the server by connecting to the well-known port number on the server machine.
3. When you transmit messages, make sure that you only send the required text string.
4. The server process *accepts* the connection request and then handles the *Addressbook* request. (Note: Alternatively, the server process may *create* a new thred process that handles the request using a new socket (with a number that is different from the well-known port number.) What are the advantages of such a strategy?

Submission Guidelines:

You will submit your homework electronically using the guidelines posted on the website.