Workshop 5: Chat Client & Server

Dr. Mirela Damian
Assigned: April16, 2013
Due: **April 23, 2013**

Description

In this programming workshop, you will further your understanding of operating systems in general and network programming specifically as you create a chat server and client. The server will accept any number of client connections (within reason), and will coordinate the receiving and distributing of chat messages. The client will connect to the server, and will send messages to the server and receive and display all replies from the server. You may work alone or with one other classmate, and each person is expected to contribute equally to the effort.

Project Specification

- Copy echolab.tar from the **/tmp** directory on **tanner** into your **csc2405** directory (if not already done so)
- Enhance the echo server and client to meet the following specifications.
  - The server starts and listens on a specified port. The client connects to a specified host and port.
  - For each client connection, the server spawns a thread to interact with the client.
  - The client sends a request to join to the server.
    Protocol: JOIN name (Example: JOIN Melissa)
  - The server receives a request to join from the client, and adds that client to a list of clients involved in the current chat session.
  - The server receives a request to exit the chat session from the client, and removes that client from its list of clients involved in the current chat session. **Protocol: EXIT**
  - The client and server have additional commands available to them:
    - **HELP – returns a list of available commands**
    - **WHO – returns a list of names of those who have joined in the chat**
  - All other message sent by the client and received by the server are then redistributed to all clients involved in the current chat session.
  - When displayed, the message is of the form:
    Name: Message (Example: Melissa: Hello world)
  - A text-based log file is kept of all traffic going to and from the server and the client.
  - Make sure you protect critical sections with semaphores.

What to Hand in

Create a text file named "readme" (not "readme.txt", or "README", or "Readme", etc.) that contains:

- Your name (or names of your team members) and the workshop number.
- A description of whatever help (if any) you received from others outside of class.
- An indication of how much time you spent doing the assignment outside of class.
- Your assessment of the workshop: Did it help you to learn? What did it help you to learn? Do you have any suggestions for improvement? Etc.
- Any information that will help us to grade your work in the most favorable light. In particular you should describe all known bugs.

Descriptions of your code should not be in the readme file. Instead they should be integrated into your code as comments.

Your readme file should be a plain text file. Don't create your readme file using Microsoft Word or any other word processor.

Leave the readme file and your C source code in your account. Do not make any changes to these files after the deadline for the assignment. You don't need to turn any paper copies in this time.

Grading

100	Total points possible
20	Program compiles successfully and runs
40	Server functions properly, handles multiple clients
40	Client functions properly, handles all communication with the server correctly