CSC 469 Program #1 – YOUTP

Design your own application-layer protocol and then code it up.

1. Overview

Program #1 is worth 8 points or 8% of your final grade. It's due in 2 weeks on Tuesday January 23, 2007. There are, however, intermediate due dates… read on!

The edumacational goals of program #1 are to tune your understanding of:

1. Application-layer protocols
2. The client-server paradigm
3. Coding up sockets in Java
4. And learning Java for most of you (um… yikes!)

Good luck!

2. Description

Program #1 actually slices into three steps (and three due dates):

1. Design your protocol (**due Thu Jan 11**)… we'll compare protocols in class Thursday, OK? (Yes, that's a rhetorical question)
2. Code up a server and test client for your protocol (**due Thu Jan 18**)
3. Code up a client for someone else's protocol (**due Tue Jan 23**)

YOUTP

First, you'll want to design your very own application-layer protocol to do something. "Protocol for what?" you may ask. "I don't care," I might reply. Something. This is not hard; it's fun and creative. So have fun… and be creative. You can use any of the application-layer protocols in Chapter 2 as a model: HTTP, SMTP, FTP, etc. The choice of acronym for your protocol will be crucial to this project. Ha! Also, don't forget to pick a port number over 1024 that you like.

Let's set a minimum of 3 commands for any protocol, not counting "hello" and "goodbye", if you even bother doing something like that. The description of your protocol does **not** have to be fancy… just enough for someone else to understand. This is your first RFC.

Coding it up

OK, you designed YOUTP, showed it to the world, so now let's code up a server for the protocol and a test client. Please make your client graphical, using Java's Swing GUI, just to get our feet wet.

A client for THEOTHERGUYSTP

As a last step, you'll have to code up a small GUI client that uses someone else's protocol. We'll do this randomly (oh, RANDTP, that's interesting) and exchange emails so that you can swap code, support files, and complaints if (when?) stuff doesn't work as advertised. Obviously, you'll borrow liberally from the client you wrote for your own protocol testing.

3. Grading

Create a `program01` folder on your k: drive for your deliverables. The deliverables for Program #1 are:

- **Your protocol design** – Describe your protocol in a Word document or web page, whatever. I can't imagine it will be more than a page or two.
- **Your code** – server, client and client for the other guy's protocol
- **Your documentation** – use Javadoc to create a tree of web pages documenting your classes, their methods and data fields
- **Your tests** – Describe somewhere (you can do it in your README below) the severe test conditions under which you placed your software
- **`README.txt`** – Create a README file describing the state of your program… what works, what doesn't, where I can find things, how you liked the "other guys" protocol, etc.

Um, about your code, please follow the coding guidelines defined in my "Dry Technical Notes" at: DTN #3 - Java Coding Guidelines. As this is our first program, it seems likely that you would have questions, so speak up!

4. Hints

First, there are some weird, simple protocols out there in the "real" world, like daytime and echo. Heck, even FTP's kind of weird/archaic as HTTP can do all that stuff now anyway. So, here are some weird protocol ideas just off the top of my head:

- your favorite quotes or more generally lists of your favorite things (that's what I'm doing so I'd actually stay away from it),
- statistics from your favorite local sports team,
- your DVD collection or sort of a IMDBTP,
- a calculator (?),
- a calendar or scheduler or something

Shoot guys, you're more creative than the old man river over here… so hack away!

The IP address `127.0.0.1` is always reserved for your local host. You can use this when initially debugging your code. When running both server and client from the same machine, you may want to open a Dos/`cmd` window in which to run the server.

You can determine your PC's IP address by running `ipconfig` in your `cmd` window.