CISC 3120

Exercise 5

Extended Chatter

Overview

In this exercise:

- You will gain experience writing socket code.
- You will continue to gain experience using the Java API documentation to understand how to use these libraries.

Requirements

This is a pretty simple application, in concept. Clone the repository at http://github.com/bc-cisc3120-s17/class23-code and import the project called Chatter. This application implements a simple chat application across a Socket connection. It's better than the book's chat application, in that the users can actually exchange messages. But it's also quite rigid: the users *must* take turns, so each one must wait for the other's chat before they can write another message.

Run the application and get a sense of how it works. Open two console views on Eclipse, then run the server application first, then the client application. Note that when the client-user types something, it is immediately sent to the server, and then the server-user must type something.

Once you understand the behavior, look at the code. Notice that once the socket connection is made, both sides follow the same "protocol."

Once you understand the code, implement an extension to the protocol: if the "current chatter" types a chat which ends in three periods ("..."), then instead of the other chatter getting their turn, the current chatter gets to type another chat. This can go on as long as the current chatter wants to keep talking (technically speaking, this allows the other chatter to *bogart the mic*). When the current chatter types something that doesn't end in three dots, then the other chatter gets their turn (or turns). Note that these three dots should have no spaces between or after them.

Other Things to Think About

If you get that working quickly, then start writing a GUI version of the program. (This is not required, but a good exercise.) As with the current version, you'll need one GUI application for the server side, and one for the client. Think about two things:

1. The console-based application makes it pretty clear to the user when it's "OK to type." How can take advantage of the GUI to make the user aware, in a natural way, of the same thing?
2. What would it take to relax the chatter rules altogether (so that either user could type whenever desired, and the text would immediately appear for the other user)?