NS Lab 2B - Chat Room

Hao Zhu (h.zhu@uva.nl)
Rex Valkering (rexvalkering@gmail.com)
Sam Ansmink (sam.ansmink@student.uva.nl)
Koen Koning (koenkoning@gmail.com)

Hand-in time (submit to blackboard) by November 13 11:59CET
Total points: 15

Abstract
This assignment focuses on client/server architecture. You will learn how to multiplex sockets to handle multiple simultaneous connections and how to combine socket I/O with other operations.

This lab must be done individually.

Preparation

For this assignment you must use Python 2.x. It is already installed on the lab computers; otherwise you can download it from http://www.python.org. If you do not know Python, learn it. It is a very simple language.

You can find examples on socket programming in Python in your textbook, section 2.6.

Socket module documentation: http://docs.python.org/library/socket.html

Select module documentation: http://docs.python.org/library/select.html

Assignment

The main assignment is to program a server for a chat room that can handle multiple clients at the same time. You also have to program a client application to communicate with the server.

In lab 2 you learned how to make a server that can accept one client at a time. In this lab, you will use the **select**() function from the select module to deal with multiple sockets at the same time. The **select**() function determines which sockets are ready to read from, ready to write to, or have errors.

On some platforms such as Linux there are more efficient alternatives to select(), but these are not cross-platform. Therefore you must use select(). Conceptually there is no difference, as they all have the same functionality.

Submission

Submit you working code in the following files:

- Lab2bserver-<yourname>.py (task 1)
- Lab2bclient-<yourname>.py (task 2)

Where <yourname> is <last name plus first letter of first name>, for example lab2b-koningk.

You must also write your full name and student number at the top of the files (in comments).

Protocol

The client and server communicate using a simple line-based text protocol where clients send commands to a server. Commands are strings terminated with a single \n (newline character).

The server does not send commands to the client, so the client simply prints out whatever the server sends.

You must implement all of these commands:

Command	Result
/nick <user>	Sets the client's nick name, unless it is already taken. The server must also notify all other users of the name change. Initially clients have no name, in which case the server should choose a name for them.
/say <text>	Every user receives the chat message.
/whisper <user> <text>	Only the specified user receives the chat message.
/list	The user gets a list of all connected users.

For example, the client might send the following commands. The server must parse the commands and take appropriate action.

```
/nick alice
/list
/say hello world
/whisper bob boo!
```

When a client connects or disconnects, the server notifies all users.

Task 1 – Server (10 pts)

Program a chat room server according to the protocol described above using **select**() to deal with multiple clients.

Use the **lab2bserver-yourname.py** file as a base.

Tips:

• If a listening socket is readable, it means a client is trying to connect.

• If a socket is readable, but **recv**() returns no data (zero length string), it means the other side has disconnected.

Task 2 – Client (5 pts)

Program a chat client to communicate with the server using the protocol described above. Instead of using a command line interface, you must use the simple GUI interface provided by **gui.py**. You cannot use blocking socket I/O as that would block the GUI event loop. Therefore you should use the **select**() function here as well. Alternatively you can set the socket to non-blocking.

The reason for using a GUI instead of the command line is that the client must do multiple things at once: Receiving/displaying incoming messages as well as letting the user enter text, and that is not as easily handled by a command line.

Use the **lab2bclient-yourname.py** file as a base.