CS 349: Computer Networks

Project : Socket Programming

Shrinivas Acharya
10010164
Computer Science and Engineering

Procedure for Executing Code

1. Change to the directory where the 10012338 (or wherever the compressed folder is uncompressed).
2. Use the following to compile all the codes simultaneously
   $make all
3. This would create 4 executables in the working directory
   server
   server_select
   server_threaded
   client
4. Now open another terminal.
5. In one of the terminal, run any one of the servers executables created
   1. for iterative server run $./server
   2. for select enabled server run $./server_select
   3. for threaded server run $./server_threaded
6. In another terminal change the working directory to 10012338 and run the client executable using one of the examples given below.

Assumptions

1. Date is assumed to be in the format mmddyy.
2. Time is assumed to be in the format hhmm.
3. Addition and removal of events will not affect the already running getall command because the number of enteries has already been exchanged between the server and client.
4. I have assumed the port number to be 9310, although it can be changed in any of the server by changing the #define line at the starting of the code
5. Addition, Removal and Updation are synchronised in the sense that two processes cannot perform any two out of these operations simultaneously, as is expected.
6. The events are stored using a data structure and thus are stored only as long as the server is up and running. Once the server shuts down, the events are also lost. This could have been avoided by using file to store events.

Sample Input and Output

1. add

   ```
   $./client localhost 9310 rajat add 080813 0800 0900 Tutorial
   ```

   Client outputs: Event added successfully and quits

2. remove

   ```
   $./client localhost 9310 rajat remove 080813 0800
   ```

   Client outputs: Event removed successfully and quits

   If record does not exist

   Client executes the following

   ```
   ./client localhost 9310 rajat remove 080813 0900
   ```

   Client outputs: Event could not be removes and quits

3. update

   ```
   $./client localhost 9310 rajat update 080813 0800 0900 Exam
   ```

   Client outputs: Event updated successfully and quits

   If event does not exist

   Client executes the following

   ```
   ./client localhost 9310 abhishek update 080813 0900 1000 Tutorial
   ```

   Client outputs: Event could not be updated and quits

4. Get with start time

   ```
   $./client localhost 9310 rajat get 080813 0800
   ```

   Client outputs

   ```
   Event Details:
   Date: 08/08/13
   Time:
           From: 800
           To: 900
   Event: Tutorial
   ```

   If the record does not exist

   ```
   $./client localhost 9310 rajat get 080813 0900
   ```

   Client outputs: Event could not be retrieved and quits.

5. Get with no start time

   First add some other events to the date

   ```
   $./client localhost 9310 rajat add 080813 0900 1200 Classes
   ```

   ```
   $./client localhost 9310 rajat add 080813 1400 1700 Lab
   ```

   ```
   $./client localhost 9310 rajat get 080813
   ```

   Client output:

   ```
   Event #1:
   Date: 08/08/13
   Time:
               From: 800
               To: 900
   Event: Tutorial

   Event #2:
   Date: 08/08/13
   Time:
               From: 900
               To: 1200
   Event: Classes

   Event #3:
   Date: 08/08/13
   Time:
               From: 1400
               To: 1700
   Event: Lab
   ```

6. Getall

   Client executes the following

   ```
   ./client localhost 9310 rajat getall
   ```

   Client output

   ```
   Event #1:
   Date: 08/08/13
   Time:
               From: 0800
               To: 0900
   Event: Tutorial

   Event #2:
   Date: 08/08/13
   Time:
   ```

```
        From: 0900
        To: 1200
Event: Classes

Event #3:
Date: 08/08/13
Time:
        From: 1400
        To: 1700
Event: Lab
```

Each of the events is displayed after a delay of 2 seconds each

Note: I have used locks for synchronisation of add, remove and update events in the threaded server.