**SER422 Spring 2016 Lab EXTRA CREDIT (50 points) Intro to NodeJS**

**Assigned 3/21/16, due 3/29/16 at 11:59pm via online submission to Blackboard**

In this extra credit lab you will do a shortened version of Lab 1 in NodeJS. Recall in Lab 1 you wrote a client-server calculator program that calculated a running total. In that lab I gave you a number of requirements, but for this extra credit lab you only have to do the following:

1. (15) Make a client socket program that can be invoked on command line: “node calcclient.js <client> <cmd> <val>” where
   1. <client> is a string client identifier
   2. <cmd> is one of the following:
      1. “a” – add <val> to the running total for that client
      2. “m” – subtract (minus) <val> from the running total for that client
      3. “s” – set (assign) the running total for that client to <val>
      4. “q” – cause the server program to exit
   3. <val> is any integer value

The client program should print out whatever value it receives back from the server and terminate. If an error occurs, it should output a message “Error connecting to server” and then exit.

1. (20) Make a server socket program that can be invoked on the command line “node calcserver.js” with the following constraints:
   1. The server should listen on port 3000
   2. The server should remain robust in the face of bad inputs from a client, meaning it should not fail and exit, but instead print to the screen “Invalid request specification”
   3. The server should be able to accept the <client> id, <cmd>, and <val> sent from the client and perform the specified computation. It should return the new running total for that client (if there are no errors) for the <cmd>s “a”, “m”, and “s”.
   4. In the case of “q” the server should close all open connections to clients immediately, dump the client ids and running totals to the screen, and exit.
2. (15) The server should introduce the following manipulations in its event queue:
   1. If the <client> id received is “ASU” then the <cmd> should only happen after a delay of 30 seconds
   2. If the <client> id received is “UA” then the <cmd> should happen immediately, before any other work on the event queue.
   3. If the <client> id received is “NAU” then the <cmd> should execute normally and a custom event should be emitted of type ‘lumberjack’.
   4. Create a custom event listener for events of type ‘lumberjack’ that outputs to the screen “I saw a lumberjack!”

Note we have not gone over basic socket connections in NodeJS, but they are pretty obvious from the API documentation (<https://nodejs.org/api/net.html>). Further, I have pushed a simplesockserver.js and simplesockclient.js example to get you started. These examples also show how to process command-line arguments. NodeJS has a simple TCP socket example in its tutorial.

Also note there is no file persistence requirement; that is the server may maintain the individual client running totals any way it wants to in memory and does not have to persist them to disk in between server invocations.

**Submission:**

You are to implement your solution **by yourself**, no partners or outside help for this lab! Submit via a zipfile named <asurite>labEC.zip on Blackboard by the due date. You should have only 2 files, calcclient.js and calcserver.js.

NO LATE SUBMISSIONS ACCEPTED!!!