CSE310 Project Documentation

Interest Groups

Fall 2016

Group 28

Annie Courtney

Aaron Lin

Daniel Khuu

Overview

The provided project is completed with all the required functionalities as provided in the specifications except for *logout*. The project is implemented with a multithreaded functionality that allows multiple clients to access the server. The user will be able to log in to the client and display the help menu for the most basic features. For the complex features, the *ag*, *sg*, and *rg* commands as well as all of its subcommands are all functional to browse discussion groups. This allows the user to access discussion groups, subscribed/unsubscribed to these discussion groups, and read posts in a discussion group.

User Documentation

The first thing the user needs to do is to log in to the client. The command will be “login N” in which N is the user ID. If the user is new, N can be any new number; however, if the user wants to access saved data, he or she must log in with the user ID that is associated with that data. The user should see a list of discussion groups in which those that the user is subscribed to has a “X” next to it.

```
1 ( ) comp.programming
2 ( ) comp.os.threads
3 ( ) comp.lang.c
4 ( ) comp.lang.python
5 ( ) comp.lang.javascript
```

After the user is logged in, the user has four options: ag, sg, rg, or logout.

The *ag* command allows the user to see existing discussion groups up to N groups. The command is “ag N” in which N is a number greater than 1. If N is not specified, the default value is 5. With this command, the user will be able to see the first N discussion groups and see if the user is subscribed to these particular groups or not.

While the user is still in “ag” mode, the user is allowed additional functionality. If the user wants to subscribed to one or more of these discussion groups, the user can enter “s X …” in which X is the group ID of the group that the user wants to subscribed to. The user can add in additional group IDs if he or she wants to subscribe to more than one group. An example command is “s 5 9 2”, in which the user wants to subscribe to discussion groups 2, 5, and 9. Another “*ag*” subcommand is “u” in which the user can unsubscribe from a subscribed discussion group. The syntax is exactly the same as subscribing except with the following format: “u X …” Both subcommands will not show anything on the screen, so the user will have to type in the “n” subcommand, which will re-show the list but with two more discussion groups to the list.

To see the next N discussion groups, the user can type in “n N” into the terminal to show the next N groups. If the user just types in “n”, it will show the next two groups in the list by default. If all the discussion groups are displayed, the next ‘n’ command will exit the “ag” mode. To quit out of the *ag* mode manually, the user must enter in “q” into the terminal.

The *sg* command allows the user to see a list of subscribed discussion groups instead of all discussion groups. The number next to each group is the number of new posts. The “u”, “n”, and “q” subcommands from the “ag” command should be functional with all the same functions as with the “ag” command.

```
1. 4 comp.programming
2. 3 comp.os.threads
3. 3 comp.lang.c
```

The *rg* command allows the user to read posts in a particular discussion group. It takes one mandatory argument (*gname*) and an optional argument (*N*). The command should be in the format “rg gname N” in which gname is the name of the discussion group that the user wants to read and N is the number of posts the user wants to display. Each post has an individual post ID in which it may not be numbered numerically as easy as “1, 2, 3, 4, and 5.” Each post also has a marker that displays an *N* next to the post ID to signify if a post is new or not. If a post is new, the *N* will next to the group ID in the list. The following subcommands are now available to the user at this point: [id], r, n, p, and q.

```
10.N Nov 12 19:34:02 How do I program this project?
20.N Nov 11 08:11:34 Java client and Python server error
30.N Nov 10 22:05:47 Best language for client/server program?
40.N 2016-12-13 21:38:01.677 new postttt
50.N 2016-12-13 21:49:26.535 skjahgdkl
```

The [id] subcommand allows the user to display the content of the post. An example of this subcommand is “[1]”, which displays the first two lines of the first post. Here, two more sub-subcommands are now available. The “n” sub-subcommand allows the user to display two more lines of the post by typing in “n” into the terminal. The “q” sub-subcommand lets the user quit the id subcommand by typing in “q” into the terminal.

The r subcommand allows the user to mark a post as read. An example command would be “r 30-50”, which would mean that the posts with post ID #30-50 will be read. After the command is executed, the N marker next to the specified group IDs should be removed after you execute the “n” subcommand.

```
10.N Nov 12 19:34:02 How do I program this project?
20.N Nov 11 08:11:34 Java client and Python server error
30. Nov 10 22:05:47 Best language for client/server program?
40. 2016-12-13 21:38:01.677 new postttt
50. 2016-12-13 21:49:26.535 skjahgdkl
```

The n subcommand allows the user to display the next N posts of the list. An example command would be “n X” in which X is the next number of lines that the user wants to read from the post. If there is no X value specified, the program will print the next two lines of the post by default.

The p subcommand allows the user to publish a new post to the discussion group by typing in “p” to the terminal. The program will prompt for a subject line first, and then the contents of the post. The user will keep writing to the post until the user types in a period by itself. The program will exit out of that mode. The user can then type in the n subcommand to see the next post with the unread marker next to it. The following image shows an example post.

```
Subject: new postttt
Author: test7
Date: 2016-12-13 21:38:01.677

lalallsdfa ksjd aklsdjhfaksjldfh ksjdfha
ksdfaksdjfhakslalkdjfsdflsjkfasdasdjkfg
n
sksldfjhgsljdfhglgssdkjfgslfgsjfkg
n
q
exited post reading state
```

The q subcommand will let the user exit out of the *rg* command by typing in “q” into the terminal.

System Documentation

First, the user must run both the client and server programs. The client is coded in Java, and the server is coded in Python 2.7. Both must run at the same time for the project to function. The client can be run by entering the client source folder and entering the command “javac client/*.java” into the terminal. The server can be run by opening a terminal, changing to the server directory, and entering the command “python threadedserver.py”. The user can connect the client to the server by entering “x 12001” in which x is the hostname the server is running on. An example input will be “allv25.all.cs.stonybrook.edu 12001.” After this is complete, the program is ready to go.

Both client and server are implemented using the TCP protocol. There is a static server port (12001) that is always the same throughout uses. The concurrent process is handled in the server using threads. The program uses JSON files and text files to save data of all discussion groups, their posts, and if it’s subscribed or not. The client uses a separate linked list to save the list of all groups, to save the list of all subscribed groups, and to save the posts that is read in a discussion group. The maximum number of clients that the server could handle is around 100 clients. There is no limit of the size of the post of any discussion group. If you type in *n* for any of the subcommands by itself, the default value is 2. If the user wants to read the next number of posts/groups by a specified number, he or she should type in “n X” in which X is the next number of posts/groups.

Testing Documentation

A couple of peripheral files were included in the files along with the program. To use these, the user will have to put it into the same folder as the server program. A test case for the program is as follows:

```
1. start the "server" on one machine, say allv23, it may output a PortNumber
   start one client on the same machine as the server
   start another client on another machine, say allv25
   (clients execute "client ServerName PortNumber" to start)

2. At client 1, type the following commands:
   login Viresh # login with user id Viresh
   help         # display all commands with a brief description and syntax of
each


   ag 5         # list all existing groups, 5 at a time. Some may be
subscribed
   s 1          # subscribe to an unsubscribed group, say #1 in the current
list (page 1)
   n            # show next five groups (page 2). Initially must have 15-20
groups
   s 2 3        # subscribe to two unsubscribed groups in page 2
   q            # exit from ag command


   sg 5         # list all subscribed groups, 5 at a time. Three shown, maybe
more
   u 2          # unsubscribe from the 2nd group in the current page
   n            # list next 5 groups if there are any. Pick a group, say
comp.lang.c
   q            # exit from sg command


   rg comp.lang.c 5 # reading the post titles in comp.lang.c, 5 posts at a
time
                # unread posts first, read ones after. Not reverse
chronological is fine

   4            # display the 1st page content of the 4th post in the current
post list
   n(*)         # display the content of this selected post until the end of
post
               # (*) is not part of a command. It only means press n multiple
times
   q            # exit from reading the post. The post list before '4' is
shown
```

```
   r 1-2        # mark the first two posts in this page as read
   n            # show the titles of the next 5 posts

   p            # make a post to this group. Prompted for post subject line
   (subject)    # enter post subject line. Prompted for post content
   (content)    # enter post content. It should have at least two paragraphs
   .            # enter a dot by itself on a line at the same time as Aich
                # show post list, two new posts  shown on top as unread

                # interact with server, say, by press n, to see Aich's 2nd
new post

   q            # Exit from rg command

3. While client 1 is running, at client 2,
   login Aich

   ag           # shows all discussion groups available, one page at a time
   s 3          # subscribe to the 6th group in the current page of groups
   s 5          # subscribe to the 8th group in the current page of groups
   u 2          # unsubscribe a group in the current page of groups
   q            # exit from ag

   rg comp.lang.c # start to read the group comp.lang.c
   p            # post to the same group, at the same time as Viresh.
                # See if both can see two new posts after
   p            # make another new post, see if the new post can be seen
                # by way of an alert at Viresh
   q           # exit from rg
```