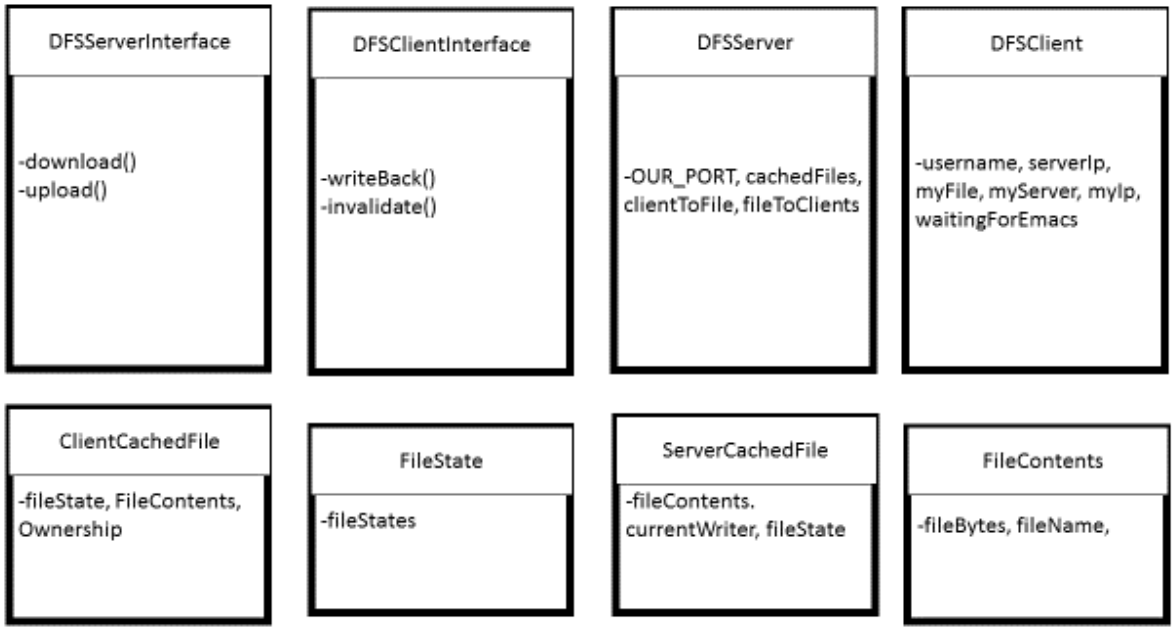
Documentation

Figure 1: Classes



Server

Brief

The server follows the logic descried by the project description; ensuring that all the clients' file-states adhere to logic described in the state transition diagrams. It's how we factored the code that makes our project stand out.

The server relies on the setting of its currently cached file's file-state, represented in the enum FileState.

Flow

Instantiating a server is easy. We use a default port so you only need to pass the server its' own IP as the first argument.

The server is factored out into a collection of functions that worked to simplify debugging of the program and manage the collection of cached files. There're are three main data structures: two hash-maps; a map of client to file, and a map of file to clients; and one vector of cached files.

The cached files are stored as instances of ServerCachedFile which contains an instance of the provided FileContents class as well as a FileState and a string with the current client with write permissions.

Whenever the methods upload or download are called, the state of the ServerCachedFile is updated as well as the hash-maps with references to what client has what file cached. These updates take place after guaranteeing all other clients currently caching that file now have to correct FileState. In the case of ownership change of a file, the server changes the FileState of a cached file to Ownership_Change and blocks. The download() function remains blocked

until the current owner of a file responds to a writeBack() call and uploads a new version of the file.

Client

Brief

Client maintains the program flow dictated by the state transition diagram shown in the project documentation. By factoring out the code into many discrete functions for ease of debugging.

Flow

Opening a client requires two arguments, its' own IP, and the server IP. The port is a constant the client sources from the server class. The client then runs in two discrete stages, a startup, and a main loop.

The startup is executed with the main function. Where here error checking takes place and the client creates an rmi connection to the server as well as adding itself to the rmi.

The second part of the client works in three parts.

First the client gets a file from the server, then writes the file into its' own cache. The client's file state is guaranteed by the server. The client will always download a new version of the file from the server, unless the file state is not invalid and the ownership of the file has not been altered. Before writing reading a file. The client checks if it needs to upload its currently cached file.

Next the client runs emacs.

Finally the client checks if it has a file-state of Release_Ownership, in this case it uploads its current file and changes its file state to Read_Shared.

Other Classes

The two interfaces DFSServerInterface and DFSClientInterface are there to allow the use of the java rmi package.

ServerCachedFile and ClientCachedFile wrap FileContents and hold additional information about a file particular to either the server or the client.

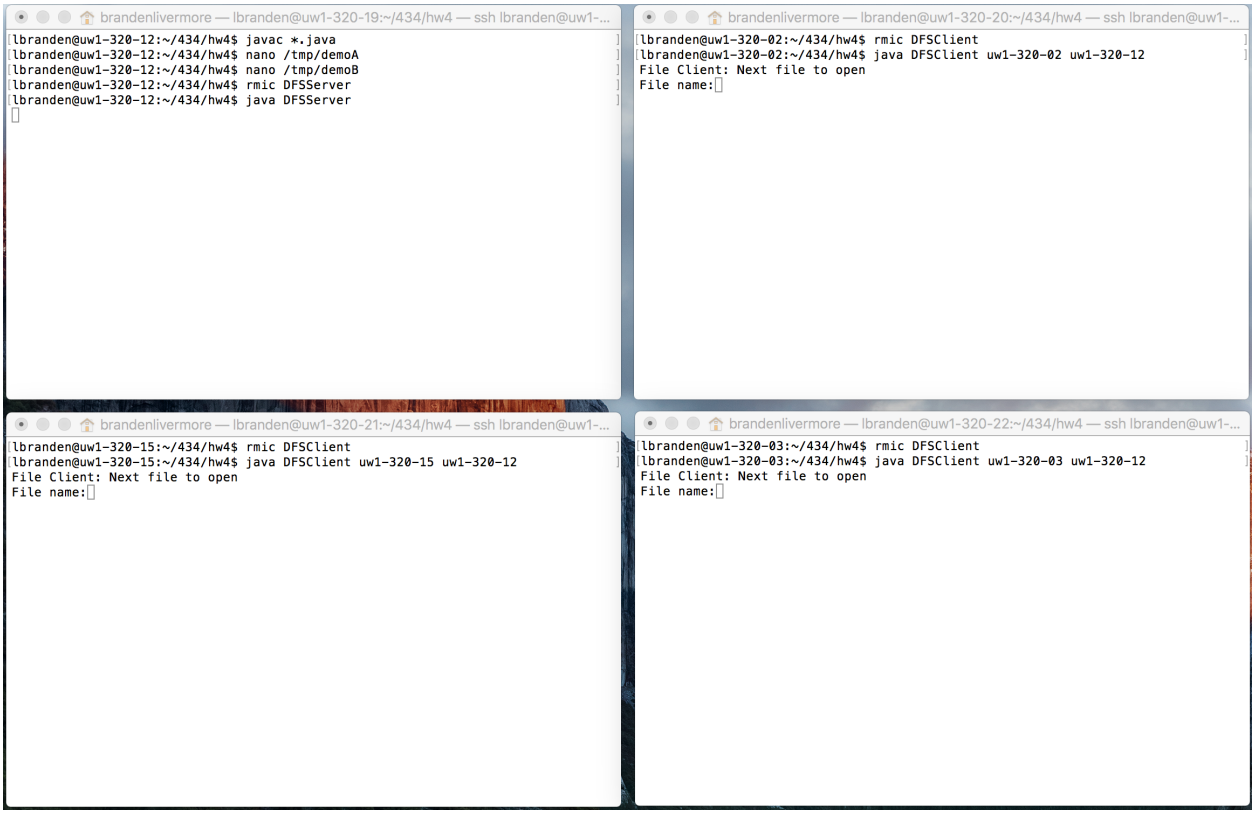
FileContents was provided.

FileStates is an enum with the a collection of file-states used by both the server and the clients.

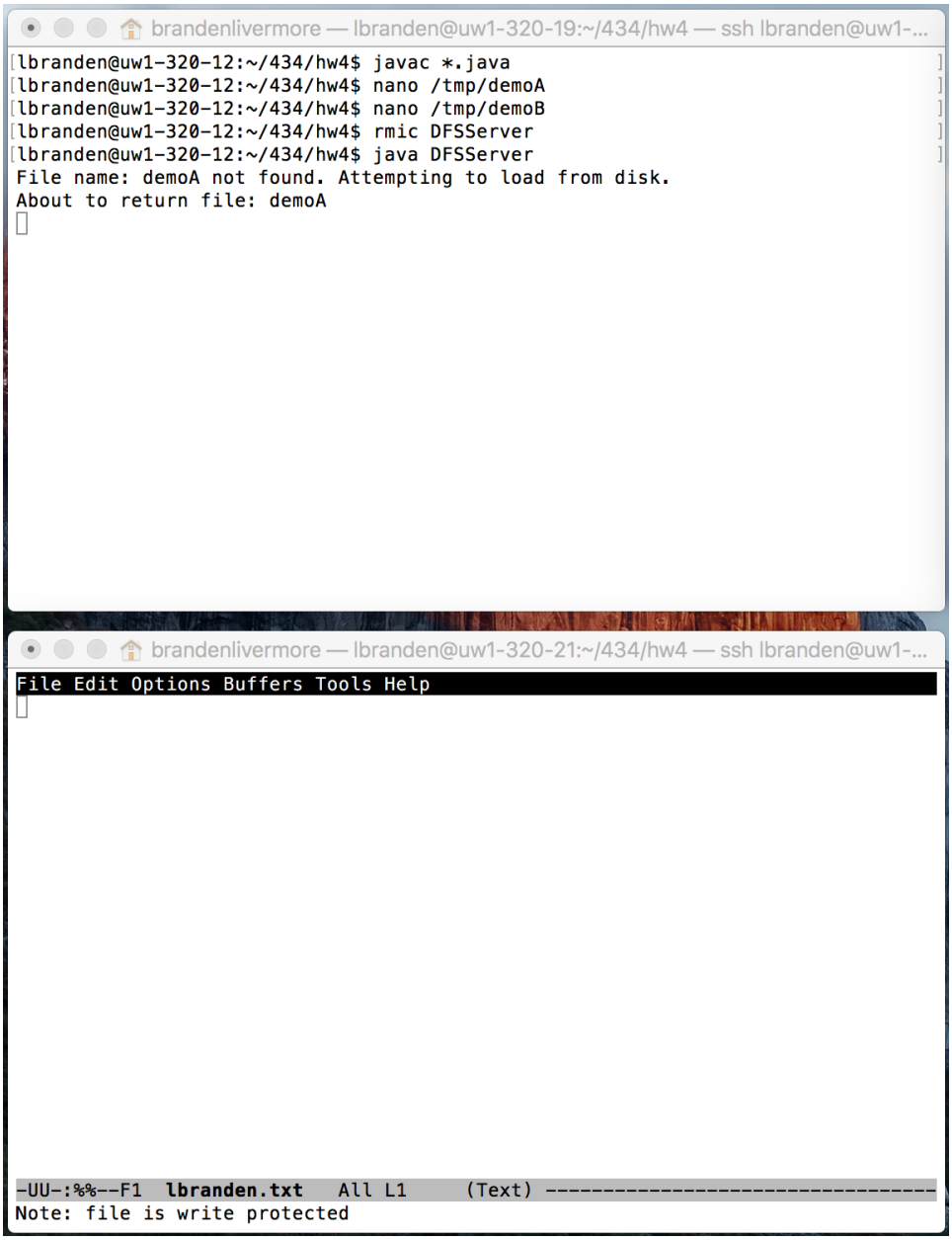
Alex s. Ducken, Branden Livermore: CSS 434 Homework 4 Distributed File System

Output

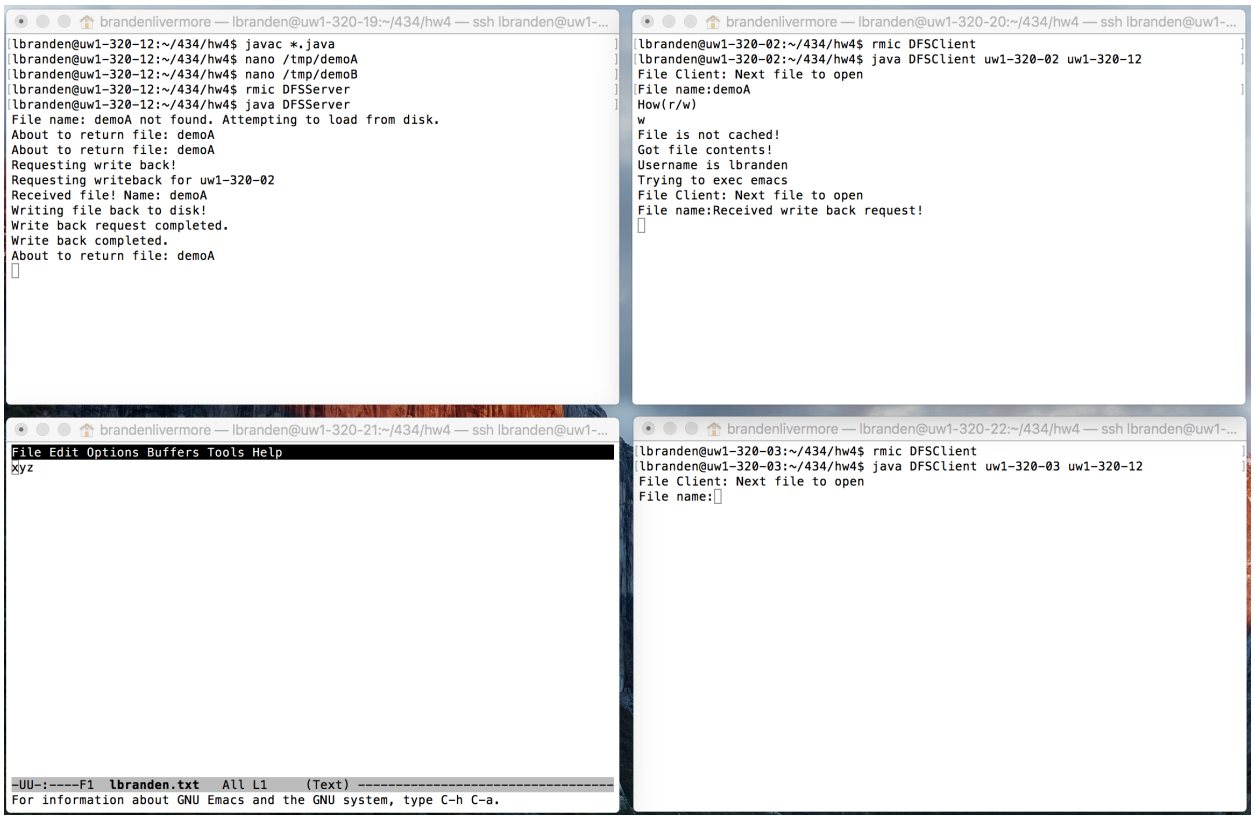
1. Compilation



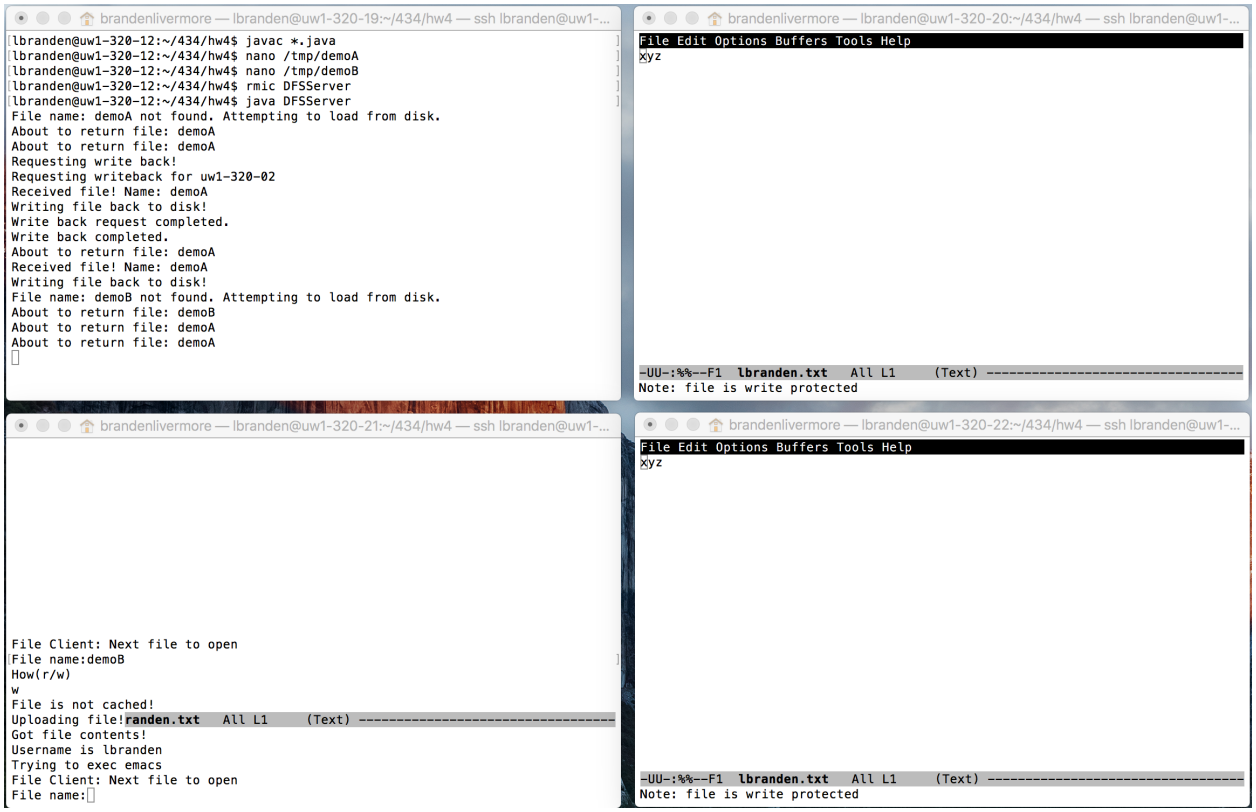
2. File read test



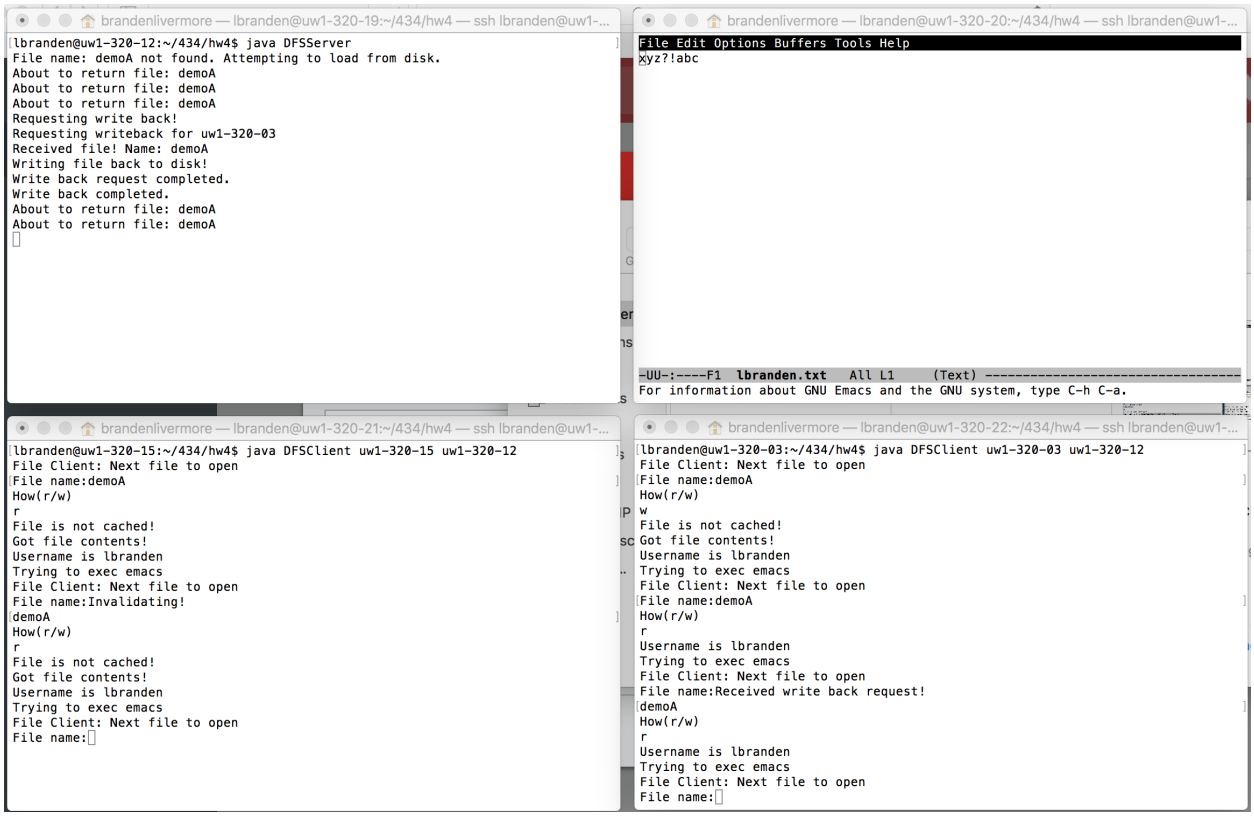
3. File write test



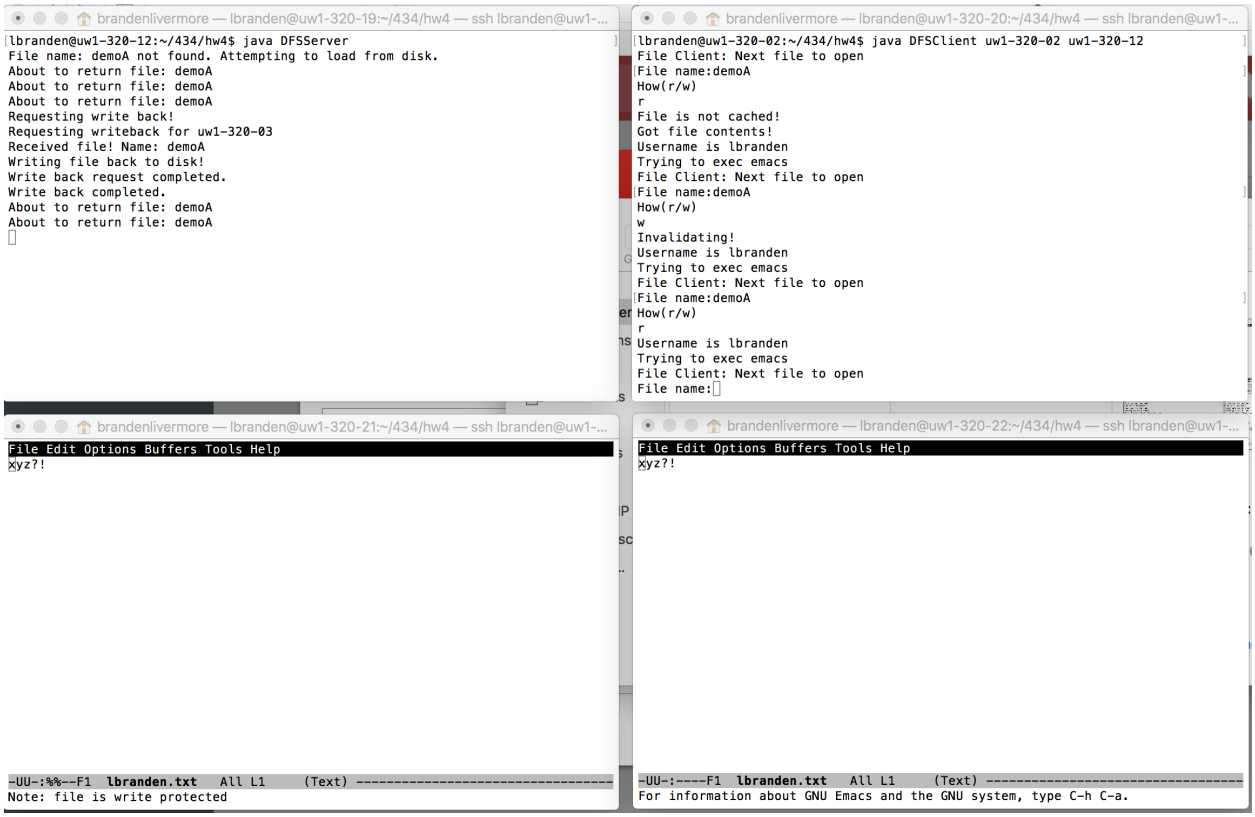
4. File replacement test



5. File writeback test

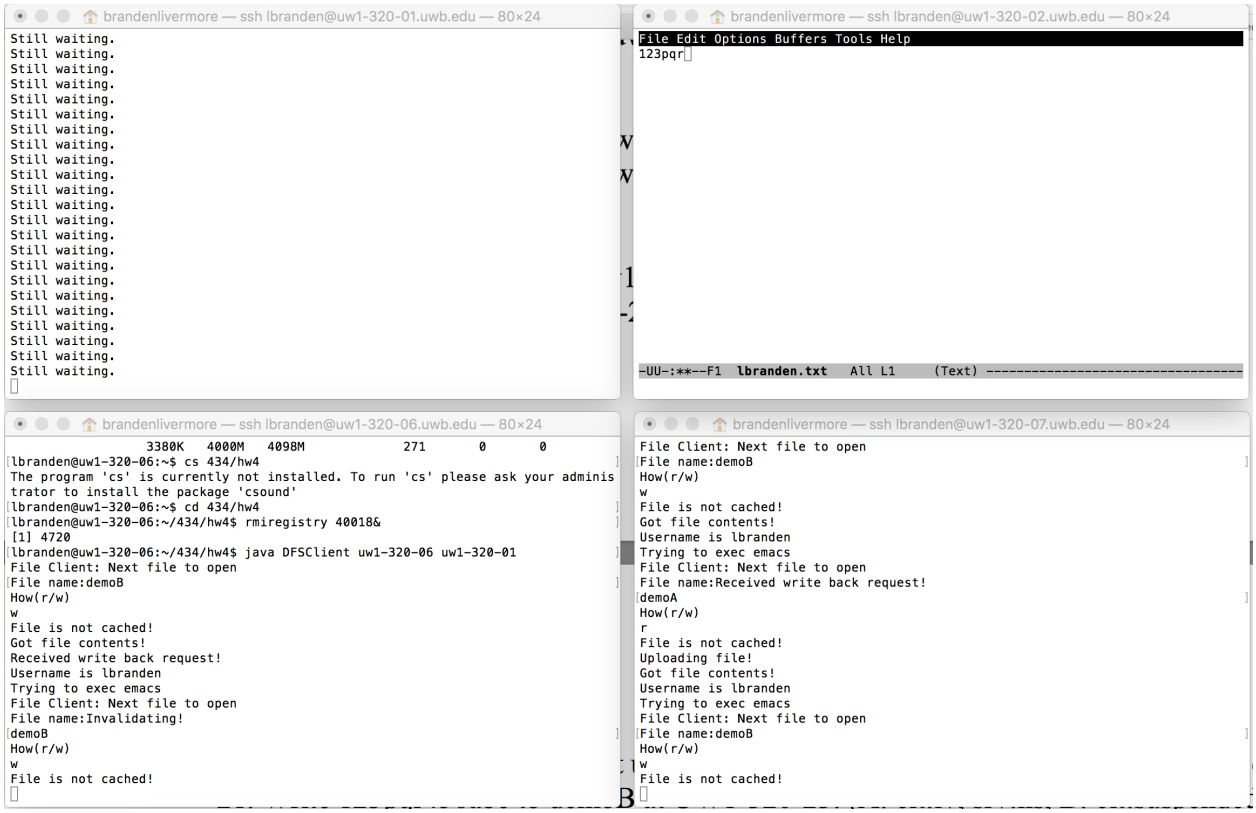


6. Session semantics read test

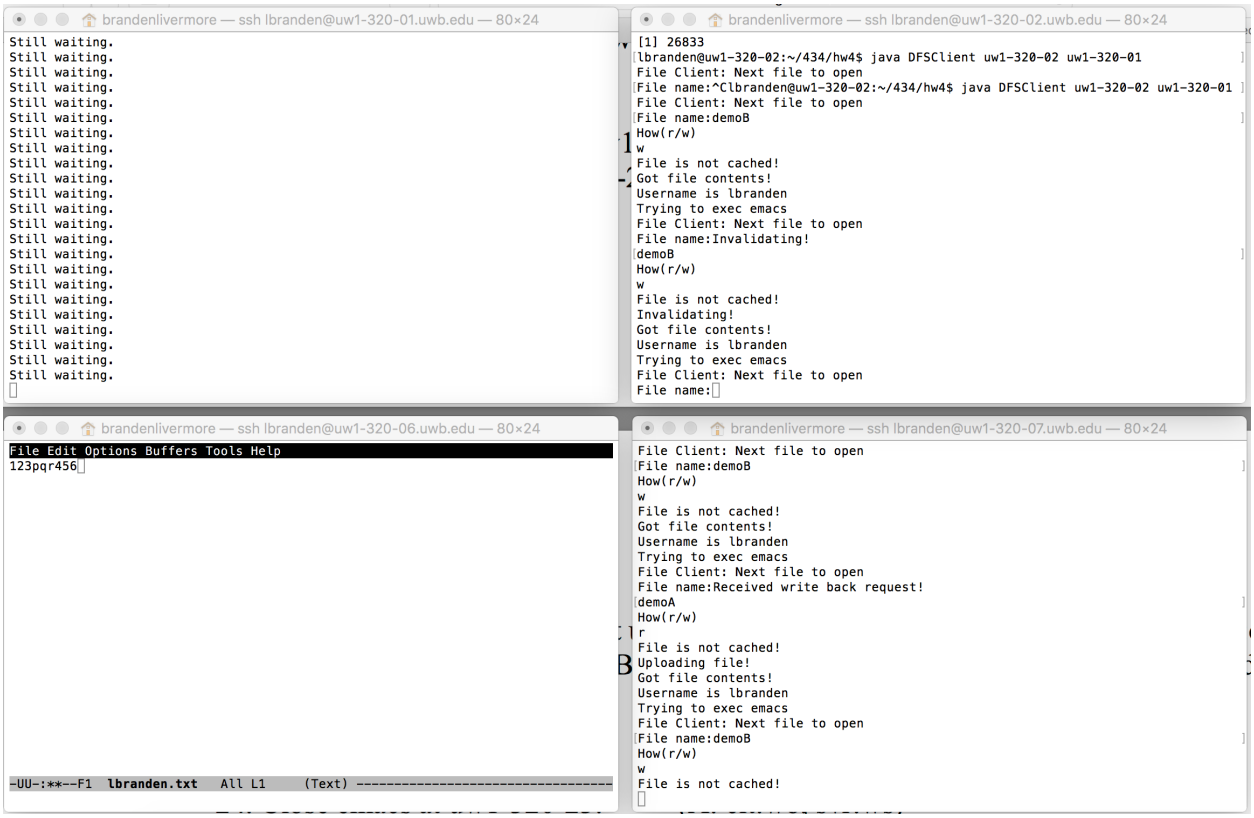


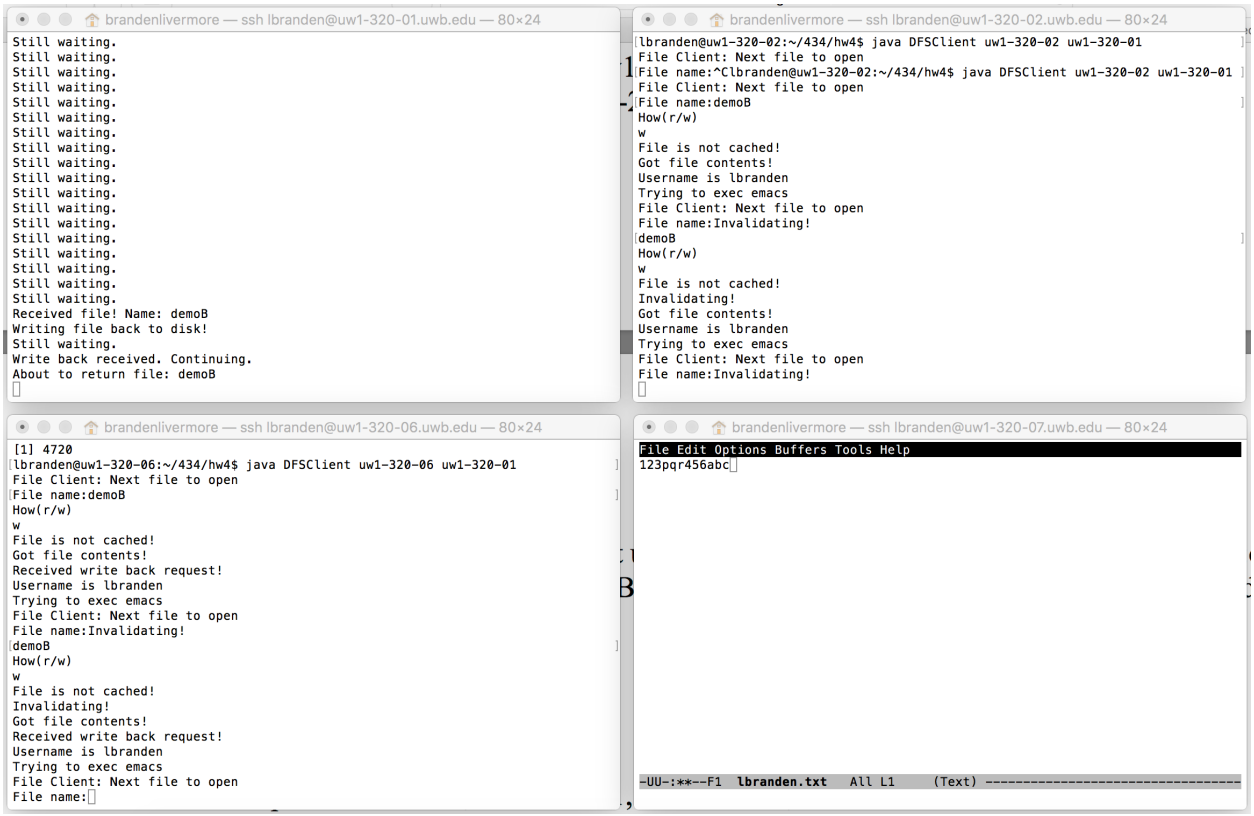
7. Multiple write test

We were a bit confused about this one. At the end of the test, it says to make sure demoB is equal to ‘123’ but after writing to the file and receiving a writeback request, the next client will receive those changes. In this test, all clients but the last one to edit the file would receive the writeback request, so I would expect to see all but the last change in the file. In any case, I included both results where I saved the files (the result is different than what we were told is expected) and results where I did not save the files after modifying (and we get the expected results). The results where we saved the changes come first.

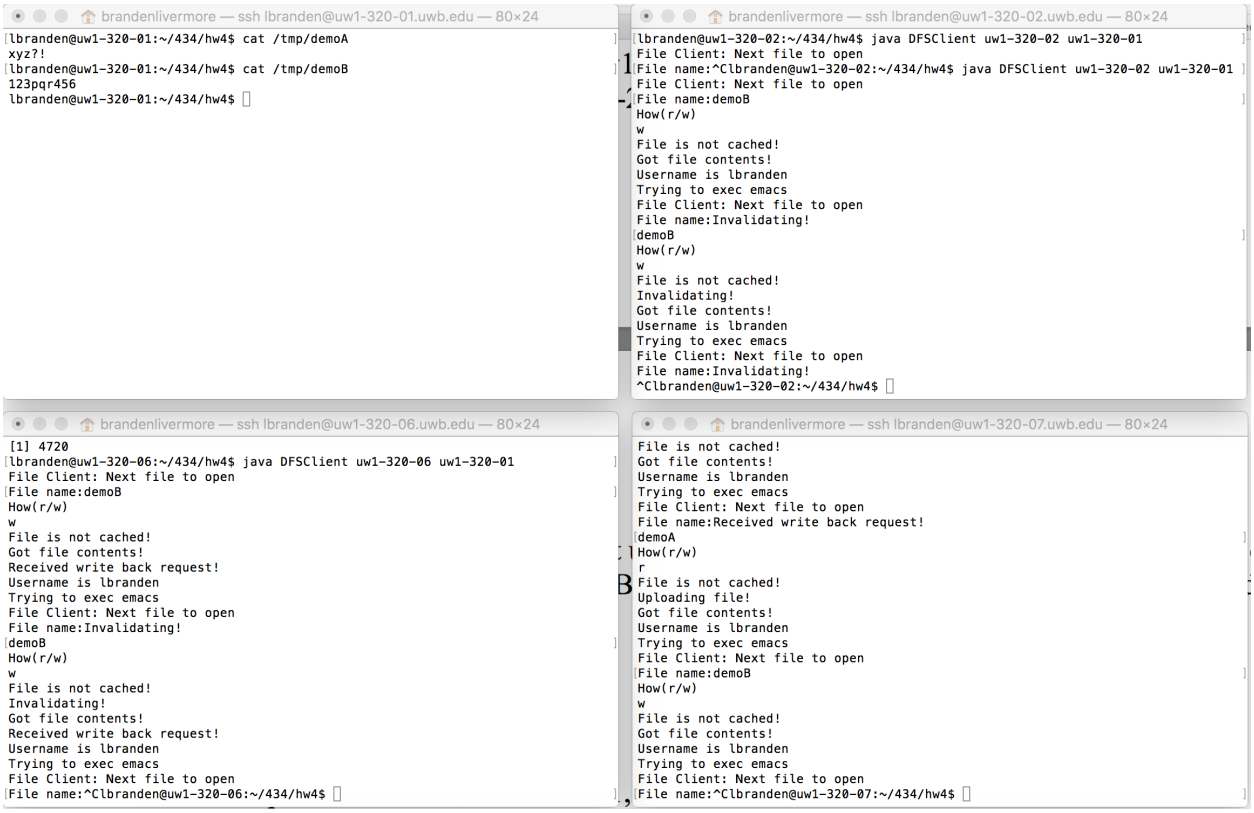


Alex s. Ducken, Branden Livermore: CSS 434 Homework 4 Distributed File System





Alex s. Ducken, Branden Livermore: CSS 434 Homework 4 Distributed File System



brandenlivermore — ssh lbranden@uw1-320-01.uwb.edu — 80×24

```
lbranden@uw1-320-01:~/434/hw4$ cat /tmp/demoA
xyz?!
lbranden@uw1-320-01:~/434/hw4$ cat /tmp/demoB
123pqr456
lbranden@uw1-320-01:~/434/hw4$
```

brandenlivermore — ssh lbranden@uw1-320-02.uwb.edu — 80×24

```
lbranden@uw1-320-02:~/434/hw4$ java DFSClient uw1-320-02 uw1-320-01
File Client: Next file to open
File name:^Clbranden@uw1-320-02:~/434/hw4$ java DFSClient uw1-320-02 uw1-320-01
File Client: Next file to open
File name:demoB
How(r/w)
w
File is not cached!
Got file contents!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:Invalidating!
demoB
How(r/w)
w
File is not cached!
Invalidating!
Got file contents!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:Invalidating!
^Clbranden@uw1-320-02:~/434/hw4$
```

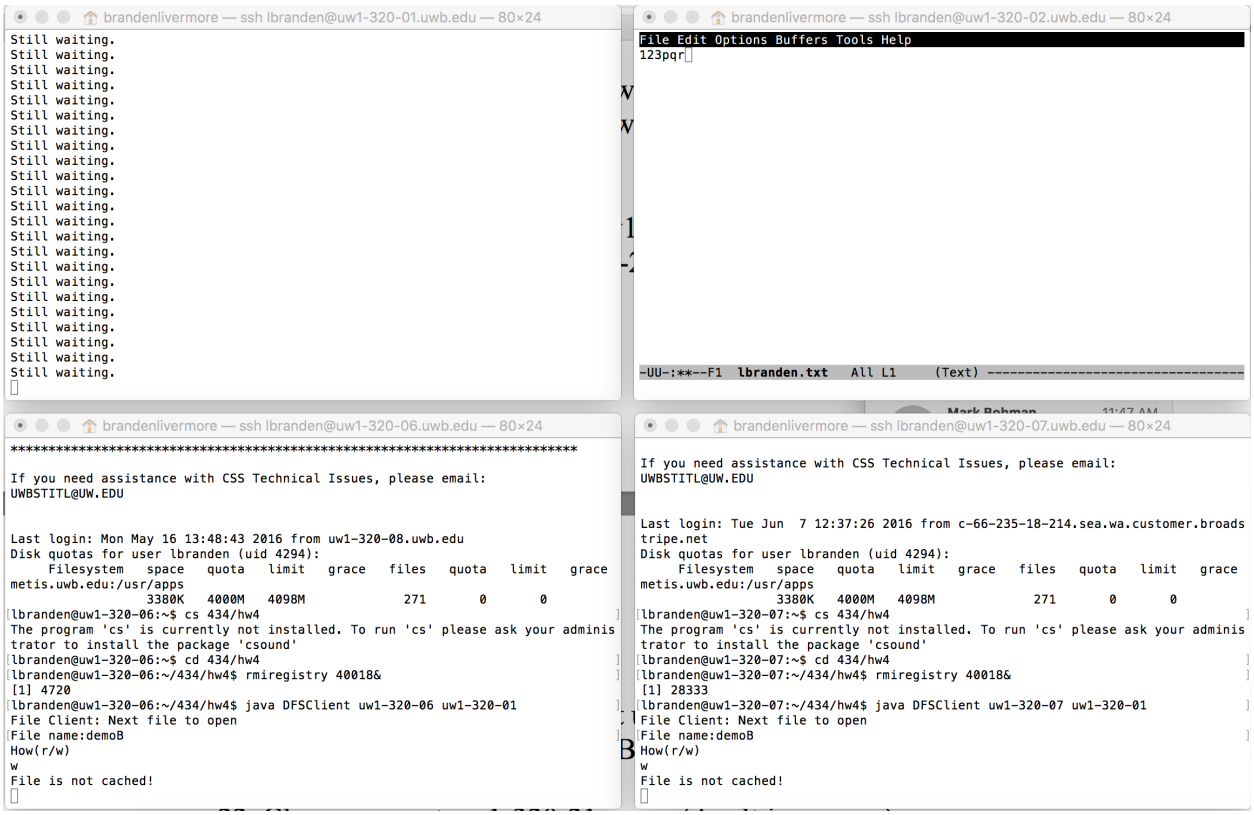
brandenlivermore — ssh lbranden@uw1-320-06.uwb.edu — 80×24

```
[1] 4720
lbranden@uw1-320-06:~/434/hw4$ java DFSClient uw1-320-06 uw1-320-01
File Client: Next file to open
File name:demoB
How(r/w)
w
File is not cached!
Got file contents!
Received write back request!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:Invalidating!
demoB
How(r/w)
w
File is not cached!
Invalidating!
Got file contents!
Received write back request!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:^Clbranden@uw1-320-06:~/434/hw4$
```

brandenlivermore — ssh lbranden@uw1-320-07.uwb.edu — 80×24

```
File is not cached!
Got file contents!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:Received write back request!
demoA
How(r/w)
r
File is not cached!
Uploading file!
Got file contents!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:demoB
How(r/w)
w
File is not cached!
Got file contents!
Username is lbranden
Trying to exec emacs
File Client: Next file to open
File name:^Clbranden@uw1-320-07:~/434/hw4$
```

Results where the modifications were not saved:



brandenlivermore — ssh lbranden@uw1-320-01.uwb.edu — 80×24

```
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
Still waiting.
```

brandenlivermore — ssh lbranden@uw1-320-02.uwb.edu — 80×24

```
File Edit Options Buffers Tools Help
123pqr
-UU-:**--F1  lbranden.txt   All L1     (Text) ----------------------------------
```

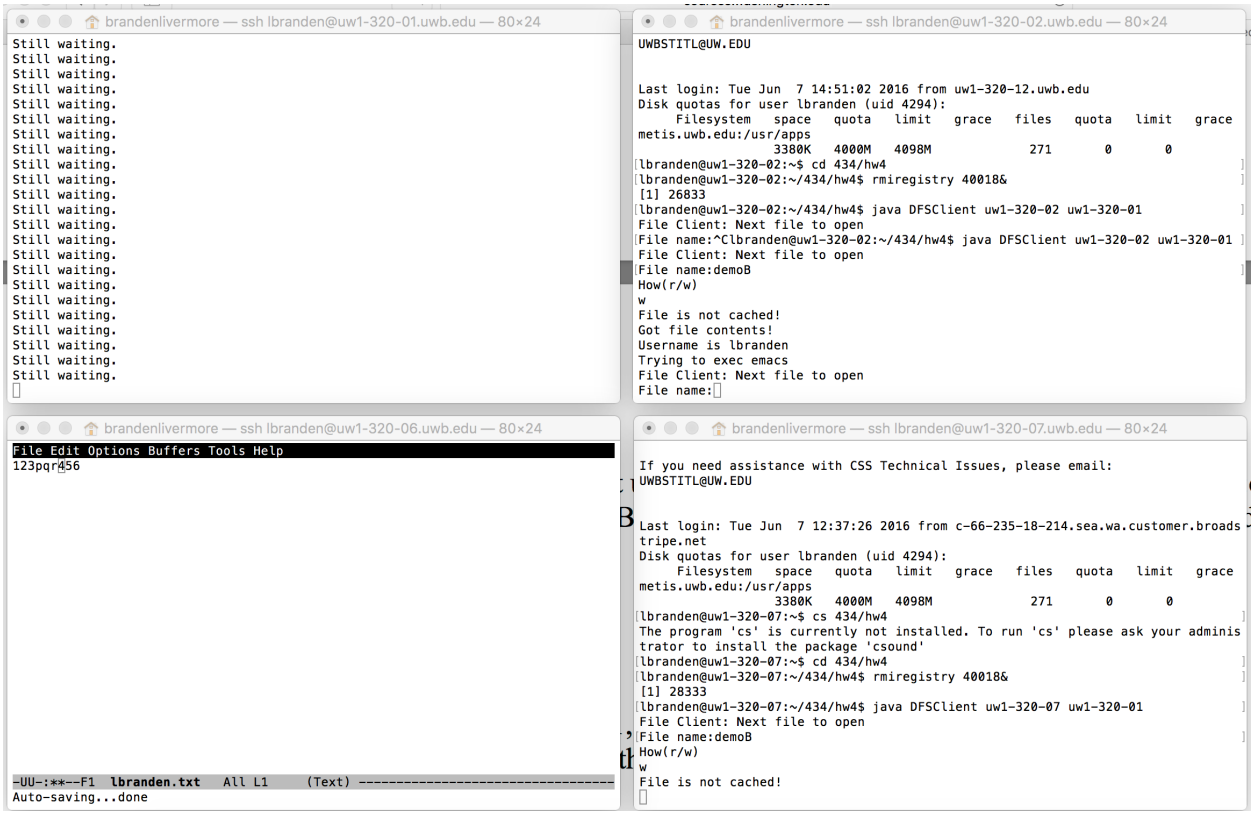
brandenlivermore — ssh lbranden@uw1-320-06.uwb.edu — 80×24

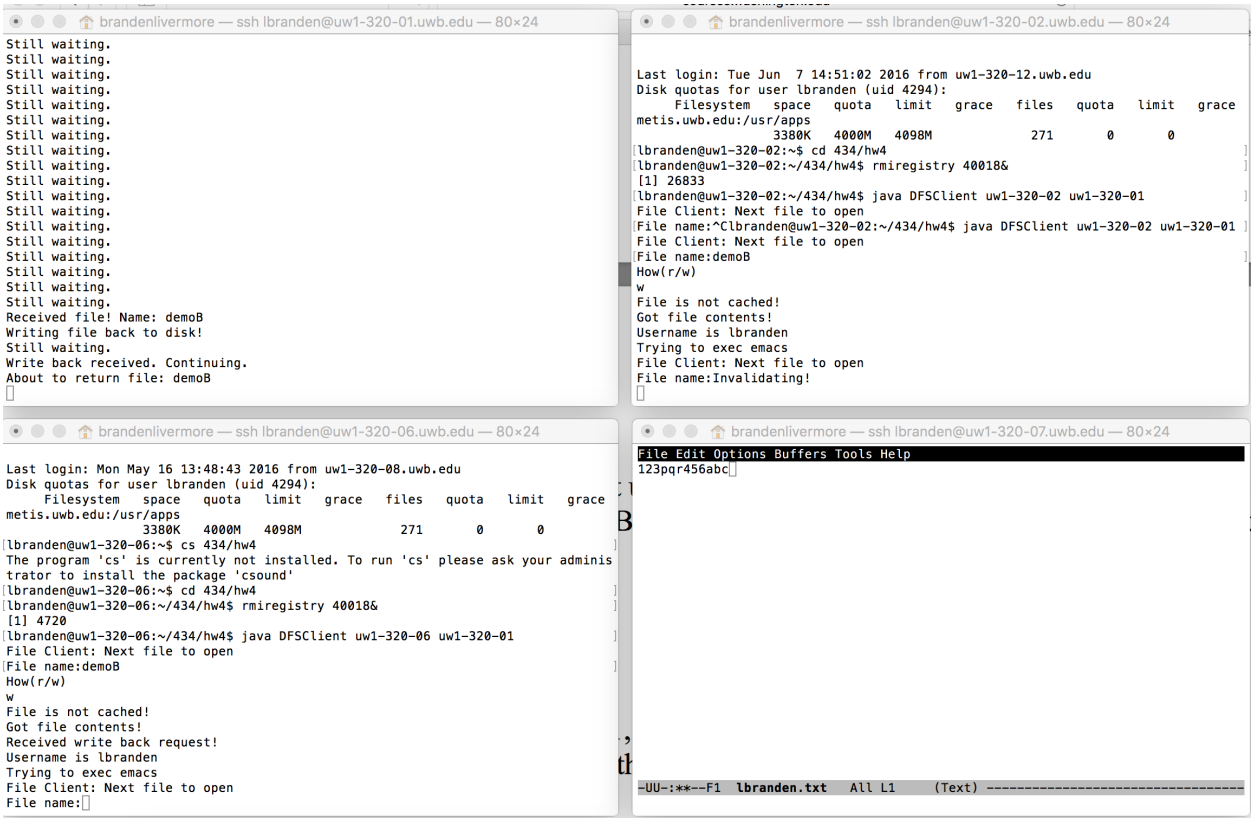
```
*******************************************************************************

If you need assistance with CSS Technical Issues, please email:
UWBSTITL@UW.EDU


Last login: Mon May 16 13:48:43 2016 from uw1-320-08.uwb.edu
Disk quotas for user lbranden (uid 4294):
     Filesystem   space   quota   limit   grace   files   quota   limit   grace
metis.uwb.edu:/usr/apps
                  3380K   4000M   4098M             271       0       0
lbranden@uw1-320-06:~$ cs 434/hw4
The program 'cs' is currently not installed. To run 'cs' please ask your adminis
trator to install the package 'csound'
lbranden@uw1-320-06:~$ cd 434/hw4
lbranden@uw1-320-06:~/434/hw4$ rmiregistry 40018&
[1] 4720
lbranden@uw1-320-06:~/434/hw4$ java DFSClient uw1-320-06 uw1-320-01
File Client: Next file to open
File name:demoB
How(r/w)
w
File is not cached!
```

brandenlivermore — ssh lbranden@uw1-320-07.uwb.edu — 80×24

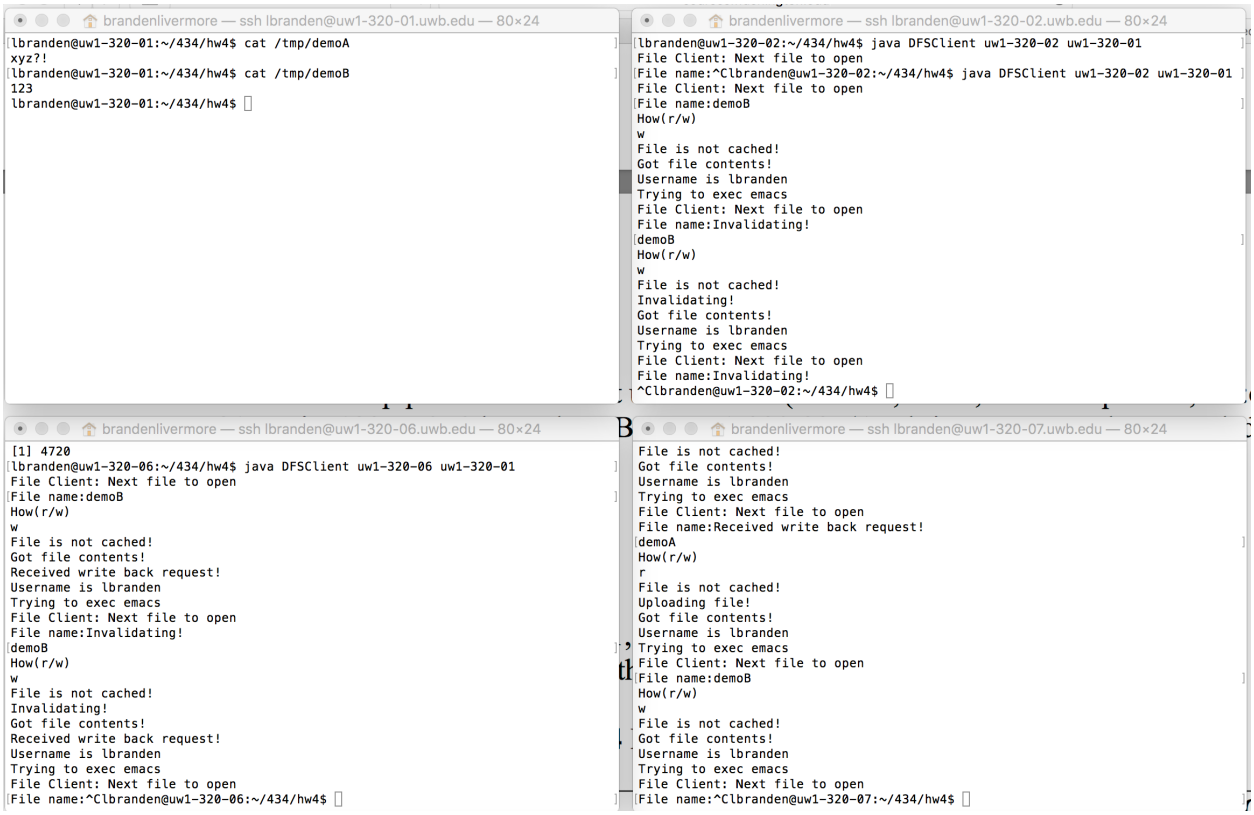
```
If you need assistance with CSS Technical Issues, please email:
UWBSTITL@UW.EDU


Last login: Tue Jun  7 12:37:26 2016 from c-66-235-18-214.sea.wa.customer.broads
tripe.net
Disk quotas for user lbranden (uid 4294):
     Filesystem   space   quota   limit   grace   files   quota   limit   grace
metis.uwb.edu:/usr/apps
                  3380K   4000M   4098M             271       0       0
lbranden@uw1-320-07:~$ cs 434/hw4
The program 'cs' is currently not installed. To run 'cs' please ask your adminis
trator to install the package 'csound'
lbranden@uw1-320-07:~$ cd 434/hw4
lbranden@uw1-320-07:~/434/hw4$ rmiregistry 40018&
[1] 28333
lbranden@uw1-320-07:~/434/hw4$ java DFSClient uw1-320-07 uw1-320-01
File Client: Next file to open
File name:demoB
How(r/w)
w
File is not cached!
```

Alex s. Ducken, Branden Livermore: CSS 434 Homework 4 Distributed File System





Alex s. Ducken, Branden Livermore: CSS 434 Homework 4 Distributed File System



Discussion

Functional Improvements

Improving our program would start with exception handling, currently we handle exceptions only generally and usually terminate to process. A more robust design would handle individual exceptions and in general not terminate the process. We would factor out the enums into two different groups, one for the server and one for the client. Our ServerCachedFile is not properly factored and does not use getters and setters. Writing our comments in a standard for that can auto generate documentation would also be a considerable improvement.

Performance Improvements

Our program preforms well and completes the test. Making use of multiple threads for the server could allow for handling more clients. Where one thread handles downloads, and another handles uploads. Using semaphores and interrupts instead of spin locking the on ownership_change would likely yield considerable improvements.

Working with emacs proved difficult and while a solution was found, there is likely a more efficient way to handle spinning up emacs.