Homework #6 - Avoidance of Data Races

Ryan Magdaleno - CS342 - Fall 2023

Instructions for Reference are below my answers.

Please note that I removed strange whitespace as this file was hard to read initially. The original code is in the same order, no logic was changed, simply the whitespace. My final styled version can be found in ServerUNMODIFIED.java. Use that to reference the lines I am talking about in my answers below. My fixes are in Server.java.

1 - Why is it not thread safe? You must point out lines of code as well as describe why they are not thread safe.

On line 15 this file defines and initializes an ArrayList of ClientThread objects:

ArrayList<ClientThread> clients = new ArrayList<ClientThread>();

The ClientThread class extends the Thread class so it's essentially a thread class that concurrently runs on its own thread. The issue arises that multiple concurrently running threads can access this ArrayList. First on line 62, the ClientThread method updateClients is one thread unsafe section. This method iterates over the clients ArrayList and sends the parameter string to their ObjectOutputStream (out). This is not a Synchronized method, multiple threads can access/modify this list concurrently causing unsafe thread behavior. The lines where this function is run are on line 82, 88, 92.

Another section of code starts in the run method of the ClientThread class. This method again modifies the clients ArrayList by removing its own instance or (this) from the clients ArrayList. The call occurs on line 93:

clients.remove(this);

This method occurs during an exception but the reason why this is not thread safe would
be because of the first issue where it’s accessing the clients list via the updateClients method,
having a remove could cause thread safety issues.

2 - What issues could arise if it is not fixed?

The issues that could arise would be most likely the ConcurrentModificationException or just undefined behavior that the programmer most likely did not intend for.

3 - How would you fix it using what you learned about synchronization?

Now look at the Server.java file which has my modifications. I will list here what changes I made and why. On line 17 I removed the original clients ArrayList. Instead of the standard ArrayList we are now using a thread-safe List from the Collections class.

List<ClientThread> clients = Collections.synchronizedList(new ArrayList<>());

On line 67 I added a synchronized block for our new clients list. This ensures that
if a thread is modifying or accessing the client list, it must release its lock
on the client list before any other threads can do their operations on the client
list. On line 95 I added another synchronized block where the clients list is the object
to be thread protected. Similar reason as to why on line 67, this time it's for
the removal of (this) from clients.

Homework #6 Instructions:

The server code (in the Server.java file) in the code sample for the GUI Server, posted here on BB, is not thread safe. In a PDF, you will answer the following questions:

1) why is it not thread safe? You must point out lines of code as well as describe why they are not thread safe.

2) What issues could arise if it is not fixed?

3) How would you fix it using what you learned about synchronization? To answer this question, you will edit the Server.Java file from the GUI Server code example posted here in BB. You must write lines of code or fix lines of code that already exist in the file for full credit.

While you can alter lines of code in the file and add code to the existing file, you may not restructure the existing file or delete significant portions from it. You also may not change the behavior and intended behavior of the program.

Submit both the PDF and edited Server.java (NOT the whole GUI Server example) file as a single PDF file or zip them together. Name it netid + Homework6. For example, I would have a submission called mhalle5Homework6.zip.