CNT 5106 Computer Network Fundamentals

Project 2 Individual Project

Handout 3/2, Due 3/30

Implementation of FTP client and server with multiple threads

1. Description

In this project, you will implement a simple version of ftp client/server software. It consists of two programs: ftpclient and ftpserver. First, ftpserver is started on a computer. It listens on a certain TCP port (such as 4007) and **is capable of supporting multiple clients with different threads**. Then, **two ftpclients** are executed on the same computer where the server runs; the server's port number are supplied in the command line, for example, "ftpclient 4007". The user can issue a command at each of the two clients: "get <filename>", which is to retrieve a file from the server, or "upload < filename>", which is to upload a file to the server.

Testing is carried out in the following sequence (which the TA will do): (a) client 1 will get downloadTestFile1.pptx from the server and write the file to the local disk as newDownloadTestFile1.pptx, (b) client 2 will get downloadTestFile2.pptx from the server and write the file to the local disk as newDownloadTestFile2.pptx, (c) client 1 wil upload uploadTestFile1.pptx to the server, which will write the file to its local disk as newUploadTestFile1.pptx, and (d) client 2 will upload uploadTestFile2.pptx to the server, which will write the file to its local disk as newUploadTestFile2.pptx. The reason to change the name of the file with prefix "new" is to allow you to place all files, including the ftpclient, ftpserver and the test files (downloadTestfile1.pptx, downloadTestfile2.pptx, uploadTestfile1.pptx, and uploadTestFile2.pptx), in the same directory. After testing, the newly created files will not overwrite the original test files.

The test files can be found on Canvas. They are the same as chap1.pptx, chap2.pptx, chap3.pptx, and chap5.pptx but under different names, downloadTestFile1.pptx, uploadTestFile1.pptx, downloadTestFile2.pptx, and downloadTestFile2.pptx, respectively. These files are large. You may need to use a loop the break them into smaller segments and send each segment at a time; in this case, you also need a way to inform the end of transmission.

The implementation does not have to conform to the ftp standard.

2. Programming Environment

Programming language: Java, C, C++, C#, Python
Operating System: Windows, Mac OS or Linux
Programming Tool: Eclipse, IntelliJ, Jcreator, Kawa, Netbeans, … whatever you like.

To use Eclipse, please go through the following list:

1. Download JDK from: https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

2. Download Eclipse from: http://www.eclipse.org/downloads/

3. Here is a link for eclipse tutorial:
http://eclipsetutorial.sourceforge.net/totalbeginner.html

4. Here is a tutorial for socket programming in Java:
http://java.sun.com/docs/books/tutorial/networking/sockets/

3. Code Submission

If you use Java, you will need to submit the following files: server.java, client.java, server.class, client.class, README.txt in a zipped directory, e.g., project2.rar. Please make sure to include server.class and client.class in the submission.

If you use C/C++/C# or Python please put all source files and executables in a zipped directory. Submit the project through Canvas:

1) Go to https://lss.at.ufl.edu/
2) Click “Login to e-Learning”
3) Login with your gator link username/password
4) Go in CNT 5106
5) Click “Assignments” and submit your project

This is an **individual** project. Students must submit their code via Canvas by the deadline. We will run an automatic tool to catch submissions with identical or similar code. There will be no late submissions.