Computer Networks Semester Project

The application written uses client and server classes to send a file, implementing Socket programming in Java.

Message:

I created a text file, Message.txt, in the project folder to be sent via the server.

Implementing the TCP protocol:

On the client side I establish a port number and get the IP address of the device using Java’s LocalHost grabber. The client sends a request file message: filename= “Message.txt”. This stored into an output stream buffer, and sent through the opened socket.

The server is initialized with the same port number as the client, and is in a while loop when accepting information from that port. It receives the client’s message, stores it into a buffer and reads the contents. I then store the filename sent as a file object, and check if that file exists in the server’s domain. If it does not exist, the server sends a return message back to the client: File (requested filename) does not exist. If the file is in the server’s system it calculates a checksum of the file. The desired file’s checksum is calculated from the Checksum and CRC32 Java APIs and returned to the server. The server sends an acknowledgment message to the client: File (given filename) exists; Checksum is (calculated file checksum). The server then sends the contents of the requested file to the client.

The client reads the socket input and deposits the file into a different text file ie. receivedFile.txt. Finally, there will be the received file in the client’s folder and the program exits.

Implementing the UDP protocol:

For this file transfer, I created a FileEvent class to store all data of the requested file. This also included methods to get the filename from the pathname and get the accurate byte count of the file.

The client sends the path of the requested file to the server through Datagram packets. Again I established a port number and IP address through Java’s LocalHost API. Once the path, byte length, address, and port were packaged appropriately, I called the Socket function send.

The server is set up to receive messages from the open port. It receives the client’s desired path, creates it into a FileEvent object, and obtains the filename. It then checks if the file exists and if it does not exist, it returns a message: Path sent is not in system. If it does exist, the server gets the file data and writes it back to the client as an output stream.

The client receives this stream and packages it into a file, then deposits the said file into the destination path desired.