Warren Shenk & Ian Stainbrook
CS 3251 – Fall 2013

Project 4 Architecture Description

- Client/server architecture uses a multithreaded server to handle multiple clients on the same incoming socket.
- On the server, the primary structure used to represent music files is the list_item, which represents a hash-filename pair. These are used to determine which files are present on the server (LIST) and compare them to what is present on the client (DIFF). The list_item struct, the "array" that encapsulates it, and related operations are defined in utils.c.
- On the client, the list_item struct is mimicked in Java with ListItem and ListItemArray classes.
- Server and client communicate with each other using a well-defined protocol. One host sends a specifically-formatted message to the other and expects a specifically-formatted response. Details of this protocol are as follows:
  - For the LIST command, the client sends the server the string "LIST". In response, the client expects the sequence of bytes <count><hash1><filename1>...<hash n><filename n>
    - count is a 4-byte integer representing the number of items to follow
    - filenames 1-n are 257-byte character sequences, each containing a null-terminated filename of at most 256 characters
    - hashes 1-n are 16-byte MD5 hashes of file contents
  - The DIFF command is handled client-side, so no communication is required. The client makes use of a utility function to compare an authoritative file set (the server's) to another file set (the client's).
  - For the PULL command, the client sends the server PULL<count><hash1>...<hash n>
    - count is a four-byte integer representing the number of hashes to follow
    - hashes 1-n are 16-byte MD5 hashes of file contents
    - The client will send the server the hashes of all of the files it does not possess. The server may in turn send the client all, some or none of these files depending on the data cap (see below).
  - Responding to a PULL command, the server sends the client <count><filename1><filesize1><file1>...<filename n><filesize n><file n>
    - count is a four-byte integer representing the number of files to follow
    - filenames 1-n are 257-byte character sequences, each containing a null-terminated filename of at most 256 characters
    - filesizes 1-n are four-byte integers representing the size of each respective file
    - files 1-n are the byte representations of the files themselves
  - To set a cap for data transfer, the client sends the sever "CAP " appended with a four-byte integer representing the cap in megabytes. The server should respond with "CAPOK". From then on, in choosing which files to send to the client after a PULL request, the server will consider the total size of the message to be sent (including metadata) and, if necessary, limit the songs sent to only the most popular based on the Play Count field of the iTunes XML file.
  - The LEAVE command is handled client-side, so no communication is required. The client closes its connection to the server (which in turn ends the server thread that was handling that particular client).