Group 05: ZHANPENG HE, MANISH PATEL, JOHN NELSON

Makefile:

Torrent file should be in same path as the RUBTClient.

We also have a makefile that can be used for compilation and run.

- Extract the file
- Compilation: : make
- Run: : make run
- Downloaded file will stored as downloadedFile.mov
- Remove .class Files: make : cleanClass
- Remove Downloaded Files : make cleanDownload
- Remove everything : make cleanAll

Work Split:

ZHANPENG HE	MANISH PATEL	JOHN NELSON
listner.java	QuitMainThread.java	peer.java
RUBTClient.java	message.java	RUBTClient.java
ConnectionToPeer.java	pieces.java	
peer.java	RUBTClient.java	
makefile		

New Classes Added In Phase 2:

Listener.java

This is a class set up the server socket and listen on a certain port. Any incoming connection will be handled in this thread and a connectionToPeer object will be created to start the communication with peer. then, All these object will be added to the main thread. There is a method named shutdown to prevent the listener from listening incoming connection.

QuitMainThread.java

This thread class runs in the main thread and waits for user input so that the program is able to quit normally after quitting, all the piece downloaded will be saved.

Modified Existing Classes In Phase2:

RUBTClient.java

It is the main client class this project. The program is designed to use a command-line argument the name of the .torrent file to be loaded and the name of the file to save the data to Opens the .torrent file and parse the data inside and decodes the data Send an HTTP GET request to the tracker at the IP address and port specified by the Torrent File object. We get the tracker response which is the HashMap from which we get peer list and other peer information. Then it connects to the specific peer mentioned in the instructions via the Peer class by using connectionToPeer class. Then, download the piece of the file and verify its SHA-1 hash against the hash stored in the metadata file. All communication is done over TCP.

Then we download chunks of the file and reassemble the file on our end.

Also, it gets the messages from connectionToPeer then handles all the messages and decide what message to send to the peer.

Cleaned code then we submitted for phase 1 and added some code snippets and methods.

A method named updateChokedPeers updates chocked peers

A method named storeTempPieces determines the size needed to store all bytes and stores the pieces we have downloaded into the Downloaded.txt file for the next downlaod It won't be called if the download is finished.

A method named loadDownloadHistory loads the pieces from the Downloaded.txt file that were downloaded from previous download

A method named processFreePeers processes peers that are currently free and send them a request to download from.

A Method milliTomin is just obvious conversation from milliseconds to seconds.

Pieces.java

Piece is a data object that stores information on a piece of the file. Pieces serves as a temporary storage area while Peer downloads the data for the piece. It has the concept of slices, which are the segments of the Pieces that are the proper size for transmission over the network.

Cleaned code then we submitted for phase 1 and added some necessary changes

Peer.java:

Peer represents a connection to peer that is responsible for downloading pieces. It’s field has connectionTopeer object. It keep track of the state of the peer. Also, let main thread know which pieces should be downloaded from this peer.

Cleaned code then we submitted for phase 1 and added some necessary changes

connectionToPeer.java:

It builds a socket to peer with the ip address and port specified in tracker response and send handshake to peer. Once handshake is one then it begins to communicate with the peer. It is responsible for getting the different types of messages and pass them to the main thread so, main thread can handle them. Then main thread will use its method to

send different types of messages to peer. Since TCP data is received as a byte stream, it reassembles the data into discrete message. These message are then received as a byte stream, it reassembles the data into discrete messages.
Cleaned code then we submitted for phase 1 and added some necessary changes
A method named sendPieces sends pieces to other peer that we have already have.

Message.java

This is a model that we used to pass messages from connection thread to the main thread then, main thread will handle them and decide what message to send to peers.
Cleaned code then we submitted for phase 1 and added some necessary changes