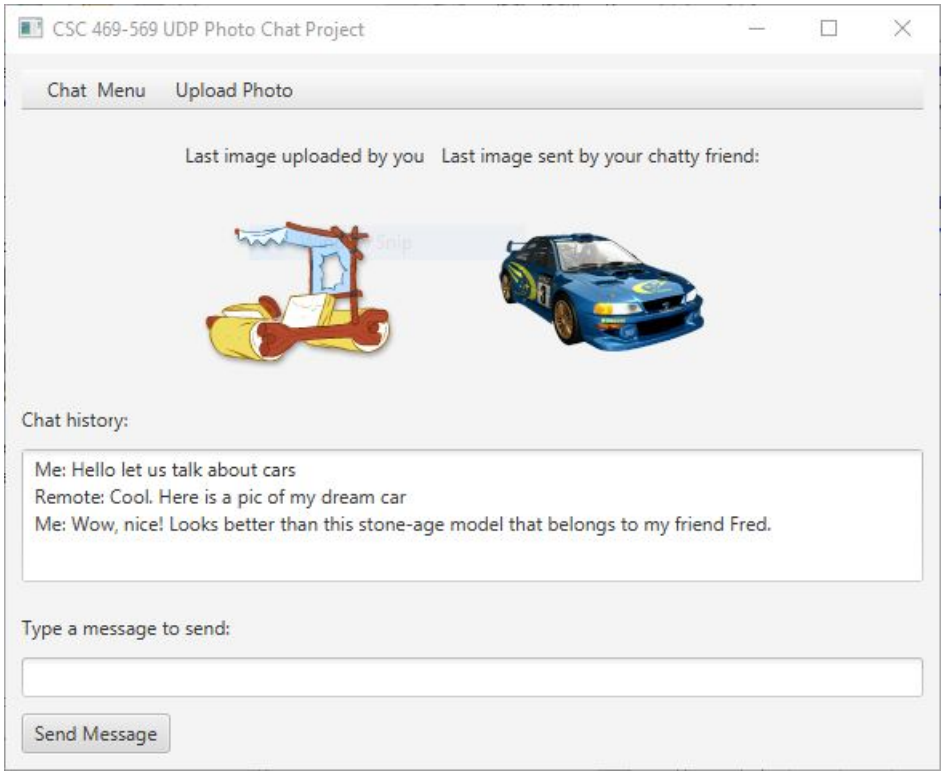
CSC 469 UDP GROUP PROJECT
JAVAFX NETWORKED CHAT APPLICATION WITH PHOTO EXCHANGE

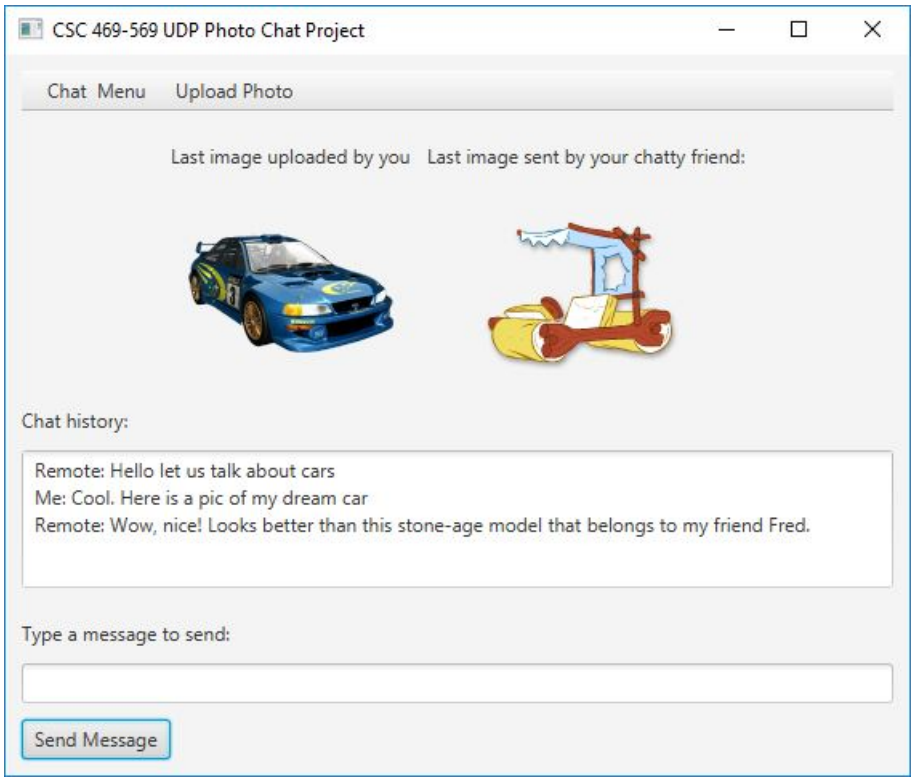
G. MUGANDA
DEPARTMENT OF COMPUTER SCIENCE
NORTH CENTRAL COLLEGE

1. UDP Photo Chat Project

Write a networked chat application that uses UDP for its transport layer. The application will allow two users to exchange text messages as well as photo images. Below are two screen shots of what the user interface should look like.

For this assignment, you have the option of working in groups of up to 3 people. If you elect to work in a group, you must appoint a group leader who will email me the names of the people in the group, with a cc to all members of the group.





The application must allow users to communicate without freezing up the user interface.

2. Clients, Servers, Port Numbers

This is a true peer to peer application, with neither of the two communicating parties playing a role different from the other. Each side will have a `DatagramSocket` that runs at port specified by the user when the application starts. This should be done by an event handler for a `Connect` menu item. This handler will ask the user, in order, for three items of information:

(1) The IP address of the remote chat partner,
(2) The port number of the remote chat partner,
(3) The port number to use for the remote socket.

Using different port numbers for the remote and local chat participants allows the application to be tested on a single machine.

3. Menus

The application is to have two menus:

(1) Chat Menu: this menu has two menu items:
   (a) Connect: brings up a series of three dialogs to solicit the local user for (1) the IP address of the remote partner, (2) the port number of the remote partner, and (3) the port number for the local socket. Once this information has been collected, the local socket is set up.
   (b) Exit: this menu item will cause a *quit* protocol command to be sent to the other side, and then will terminate the application.
(2) Upload Photo: This menu item will bring up a JavaFX file chooser to allow the user to select a file containing an image to send to the other side. The listener for this menu item will display the image locally and send a copy of the image to the remote side.

4. The Application Protocol

In addition to using UDP, you must use a binary protocol with the following protocol commands. Square brackets are used to delimit parts of a protocol command. Within the square brackets, components in angled brackets `< >` are syntactic variables, while components with no brackets are literal strings.

(1) `[chat][<message>]` : This command consists of two strings encoded using the UTF-8 character encoding. This command is used to send a chat text message.
(2) `[quit]` : This command consists of a single string encoded in UTF-8 format.
(3) `[photo] [<image-size>] [<image-bytes>]` : This command is used to send a image. The literal word `photo` encoded in UTF-8 is followed by a long integer (expressed in binary) that represents the size of the image to be transferred in bytes. The final component in this command is the sequence of bytes that comprise the image.

5. Caveat

This project is only meant as an introduction to UDP Programming. Your project is only required to work correctly for images of modest size, so that the IP datagram carrying the image is not likely to be fragmented en route to its destination. Nor is the application expected to recover from lost, corrupt, or reordered packets.

I am providing a collection of small images that you can use in testing the project.

6. Hints

Perhaps the quickest way to get started is to start with the code base for the previous project and modify it.

To represent the images, look at the JavaFX `Image` and `ImageView` classes. Basically, you create an *Image* object from a file or from a URL source. To display the image, it must be wrapped in an `ImageView` object.

An `ImageView` object is a `Node` object and can be displayed in a GUI by adding it to a pane.

To write and read the protocol commands, you need to use objects and methods of the `DataInputStream` and `DataOutputStream` classes.

Because `DatagramPacket` objects are built around byte buffers, you will also find use of the `ByteArrayInputStream` and `ByteArrayOutputStream` classes useful.

Due Date: Monday of Week 10.