|  |
| --- |
| PACIFIC LUTHERAN UNIVERSITY |
| Laboratory 2 |
| Echo Client Application |

|  |
| --- |
| Sheyla Trudo  3-3-2015 |

Table of Contents

[1. List of Figures 2](#_Toc413756886)

[2. Introduction 2](#_Toc413756887)

[3. Requirements 2](#_Toc413756888)

[4. Design 2](#_Toc413756889)

[Figure 1 3](#_Toc413756890)

[Figure 2 4](#_Toc413756891)

[Figure 3 4](#_Toc413756892)

[Figure 4 5](#_Toc413756893)

[5. Implementation 5](#_Toc413756894)

[List of files 5](#_Toc413756895)

[Lab2ClientServer.java 5](#_Toc413756896)

[EchoClientController.java 6](#_Toc413756897)

[MessagingModel.java 6](#_Toc413756898)

[Client.java 6](#_Toc413756899)

[EchoClientMainView.java 6](#_Toc413756900)

[6. Bibliography 6](#_Toc413756901)

# 1. List of Figures

Figure 1 – UI Design

Figure 2 – Client Side Pseudo Code

Figure 3 – Multi-Threading Pseudo Code

Figure 4 – Client Socket

# 2. Introduction

The purpose of this lab is to become familiar with Java Socket communications. In this lab, the idea is to write the client side implementation of a messenger application. The application is implemented in a simple GUI that allows the user to input and send messages to an existing server. The server then echoes back the message, as well as messages from any other user on the server.

# 3. Requirements

Below is a list of requirements that must be fulfilled by the application.

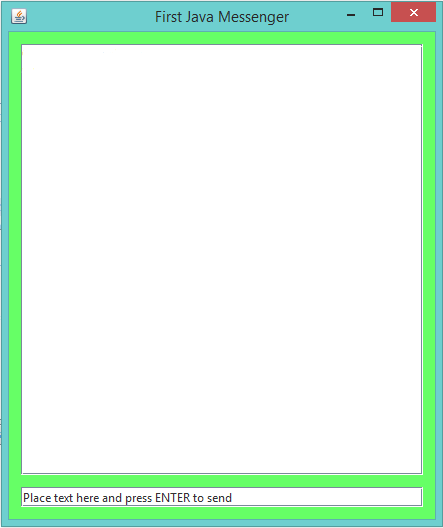
* The app must connect to the specified server
* Use Java sockets and threading
* Allow the user to send messages to a server
* Receive incoming messages from the server

Something extra

* The application uses a GUI interface
* The application logs user messages sent and received from the server into a text file. Also logs any error messages.

# 4. Design

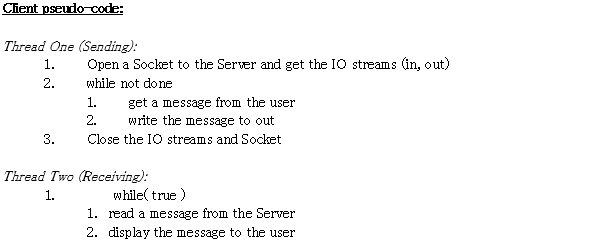
The design of the application includes a simple user interface with a message log that displays messages received from the server as well as error messages. The UI also includes an area for the user to enter and send messages to the user.



## Figure 1

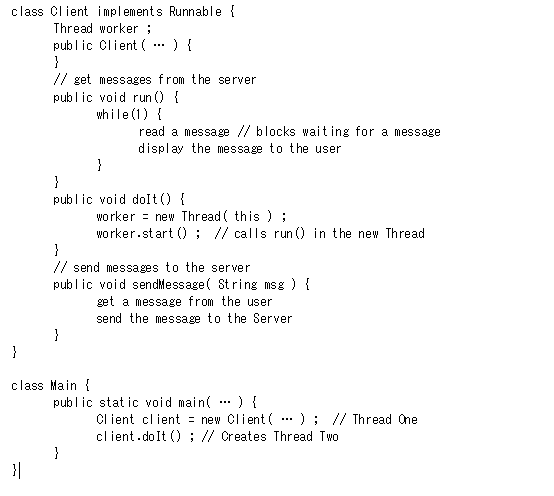
Design of the user interface.

The pseudo code for the client side design was provided by Dr. George Hauser as part of the lab overview.



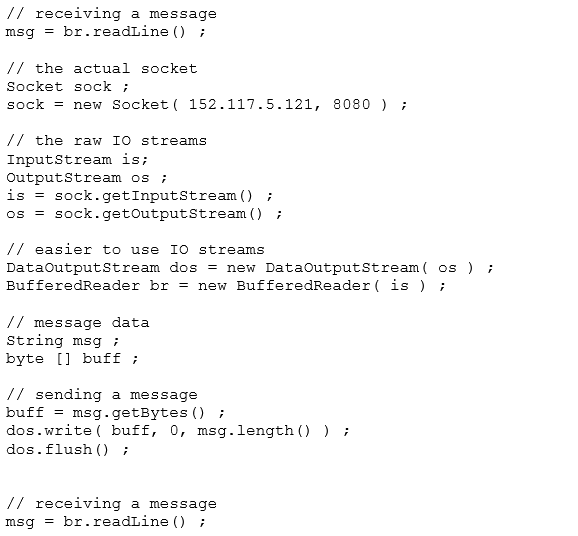
## Figure 2

The client side pseudo code.



## Figure 3

Multiple threading pseudo code provided by the lab overview documents

.

## Figure 4

Example of socket use in Java provided by the lab overview documents.

# 5. Implementation

List of files within the project:

* Lab2ClientServer.java
* EchoClientController.java
* MessagingModel.java
* Client.java
* EchoClientMainView.java

## Lab2ClientServer.java

This particular class is in charge of instantiation and initialization of all objects within the program. It contains the main( String[] args ) class.

Creates

* JFrame clientView
* EchoClientMainView view
* EchoClientController controller
* MessagingModel model
* Client (unnamed)

Sets

* clientView DefaultCloseOperation
* view’s Controller to controller
* controller’s View to view
* controller’s Client to an unnamed object <Client>

## EchoClientController.java

This class is in charge of handling user messages. When the user presses ENTER after entering a message, the controller passes the message along to its set Client object.

The controller begins the call to Client to start listening to server messages and determines how the view should be updated according to the message received. Will also send server messages to the MessagingModel in order to keep them logged in a text file called ‘log.txt’.

When an error is thrown, either by the model or the controller itself, the message is handled by the controller. The controller sends the message to the view telling it to update and sends another message to model, in order to record the message thrown by the exception in the ‘log.txt’ file.

## MessagingModel.java

The main purpose of this class is to record data received from the server into a text file called ‘log.txt’. It may also record any errors thrown by the program.

## Client.java

The Client class acts like a second model in this program. It creates the two separate threads for sending and receiving messages to/from the server. The Java Sockets for the client are contained in this class.

## EchoClientMainView.java

A JPanel that includes methods for handling user input. When the user presses enter after entering a message, the view sends the message to the controller to be handled.

# 6. Bibliography

Dr. George Hauser

For His Expertise and the Following Documents

* + ClientServerProject.docx
  + JavaThreads.docx
  + JavaSockets.docx

Chris Boe

For assisting in errors regarding the Client class and its instantiation

http://www.javabeginner.com/learn-java/java-threads-tutorial

For clarification on the use of threads.

http://www.oracle.com/technetwork/java/socket-140484.html#client

For clarification on the use of Java Sockets.