Assessment Title: Java 3 Project

Student Name: Lee Say Hon

Student ID: 30003628

Date: 1 December 2020

Table of Contents

[Introduction 3](#_Toc58072878)

[Overview 3](#_Toc58072879)

[Source Code 4](#_Toc58072880)

[Software Specification / Requirements 4](#_Toc58072881)

[UML 5](#_Toc58072882)

[Specification Requirement 5](#_Toc58072883)

[SDLC 6](#_Toc58072884)

[Test table 7](#_Toc58072885)

[Server and client login 8](#_Toc58072886)

[CSV Reader 11](#_Toc58072887)

[Audio Player 13](#_Toc58072888)

[Help File 14](#_Toc58072889)

[Debugging 15](#_Toc58072890)

[Introduction 15](#_Toc58072891)

[Server 15](#_Toc58072892)

[Client 15](#_Toc58072893)

[AudioPlayer 15](#_Toc58072894)

[CSVReader 15](#_Toc58072895)

[Junit Testing 16](#_Toc58072896)

[Introduction 16](#_Toc58072897)

[Result 16](#_Toc58072898)

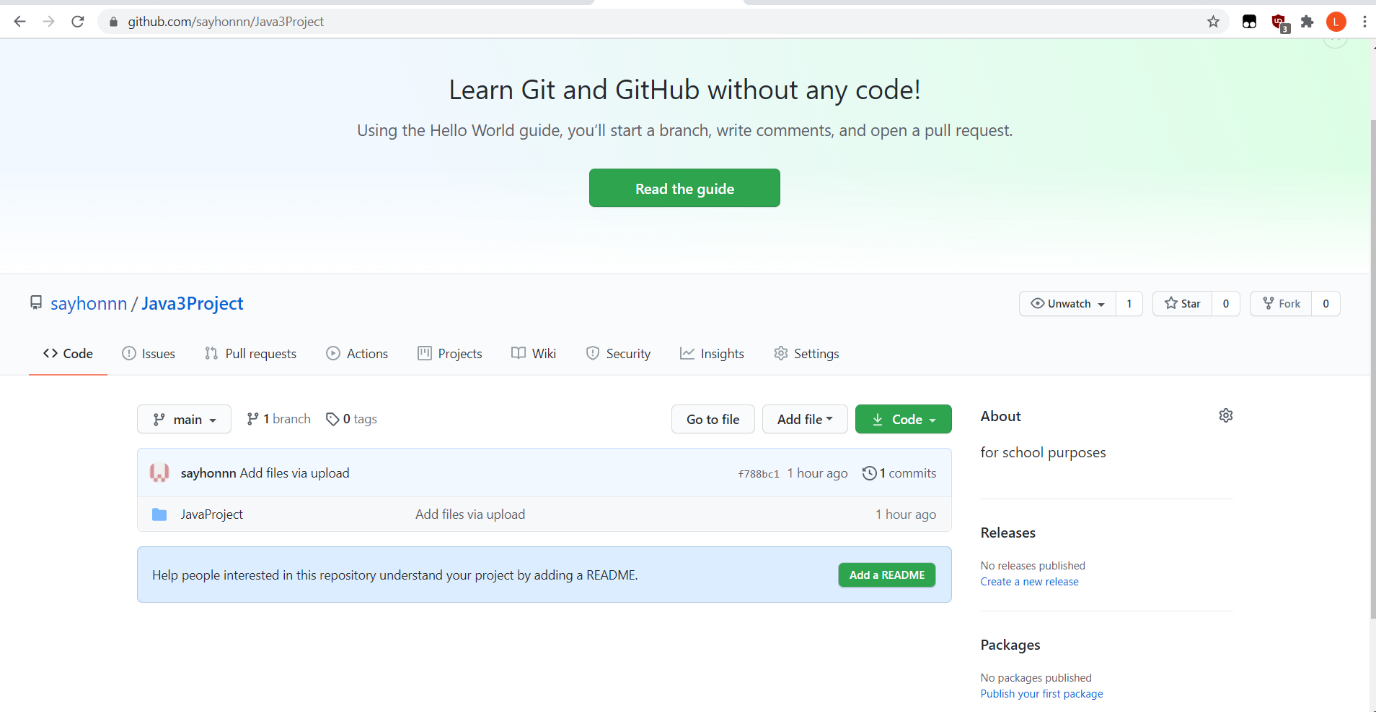
# Introduction

You have been hired as the new programmer by the Jupiter Mining Corporation to produce a test program for the company, this program will be fully documented and tested. With this project you are coming up with a program to show your range of skills and abilities to your new boss. You have been given an example of what your boss is expecting to see the example they have given is an advanced music player that allows the ability to sort and search the songs stored in a binary tree (any sort and search algorithm you select will have to be approved if it is not merge sort and binary search), the GUI should display the sorted track list and highlight and play the searched track, it should save the track list to a csv using a 3rd party library. The music player must load and play files and met the requirements laid out in Question 3.

## Overview

In this documentation, it displays the testing documentation for the project to display all of the functionality that is required from the project have been implemented and working as it should be. I have created a server and client login application for the client to login through the server which will allow them to open a media player and a csv reader.

# Source Code



# Software Specification / Requirements

In this project, you are required to produce a test program which includes the functionality:

- Must contain dynamic data structures (e.g. doubly linked list or a binary tree)

- Must contain hashing techniques

- Must contain sorting algorithm

- Must contain searching technique

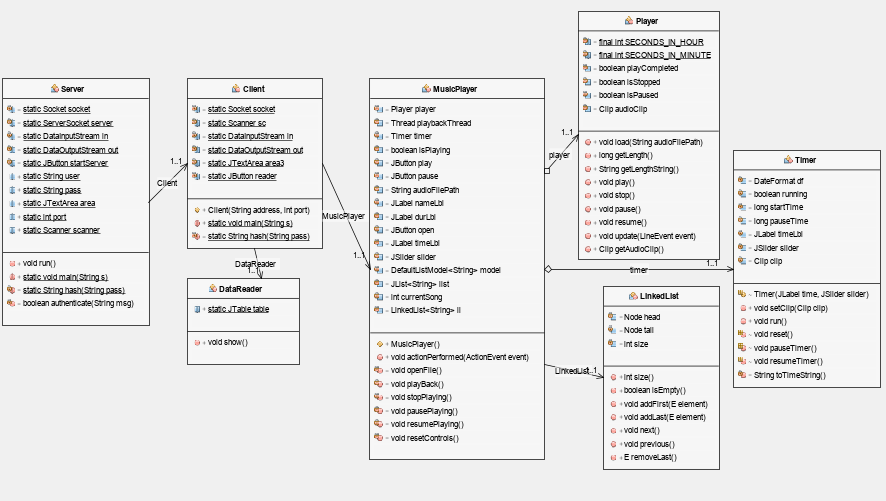
- Must contain 3rd party library

- Must have a GUI

– Must adhere to coding standards

- Must have help files

# UML



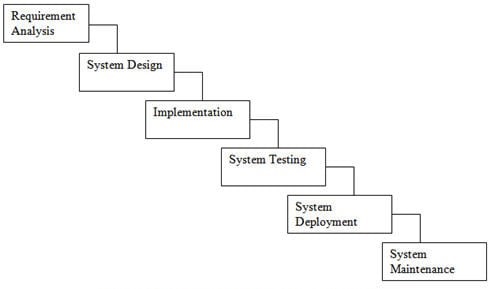
# Specification Requirement

The minimum requirement that is needed in order to run the program and use its features is:

* Pentium 200MHz or greater
* Windows 98
* 128MB of available hard disc space
* VGA graphic
* Windows 64-bits operating system
* 1GB RAM

# SDLC

The SDLC which is chosen for this project is the waterfall model as the requirement for the program was clear and specific at the beginning of the project. If the project has been completed and all the requirements have been met, the system could be deployed for to the client for approval.



# Test table

|  |  |  |
| --- | --- | --- |
| Test No. | Description | Test Output |
| No. 1 | Ability to start up the server | Figure 1 |
| No. 2 | Ability to connect client to the server | Figure 2 |
| No. 3 | Ability to create new user | Figure 3 |
| No. 4 | Ability to login using the created user | Figure 4 |
| No. 5 | Ability to detect incorrect login details | Figure 5 |
| No. 6 | Ability to open CSV file using third library file to be displayed | Figure 6 |
| No. 7 | Ability to sort CSV according to alphabetical order | Figure 7 |
| No. 8 | Ability to search for records in the CSV File | Figure 8 |
| No. 9 | Ability to add songs into media player | Figure 9 |
| No. 10 | Ability to play songs | Figure 10 |
| No. 11 | Ability to open a Help file | Figure 11 |

## Server and client login

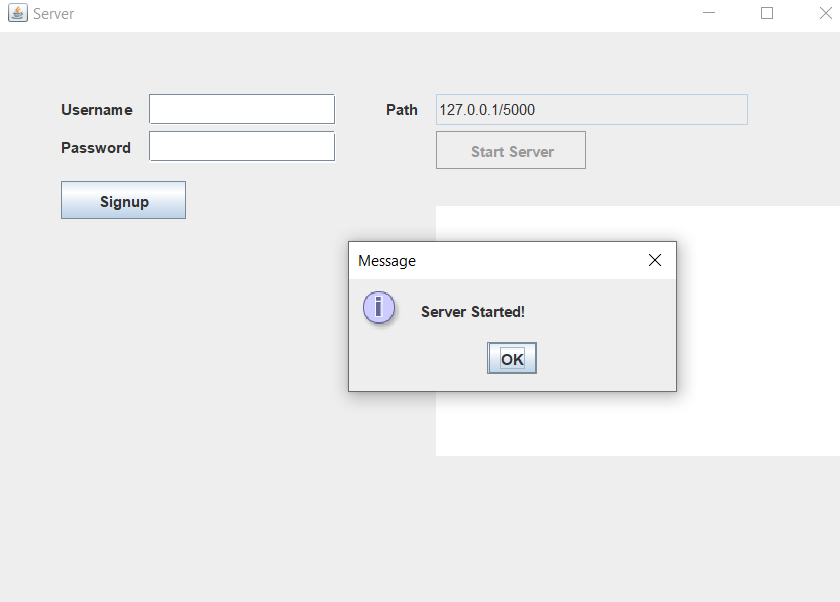


Figure 1

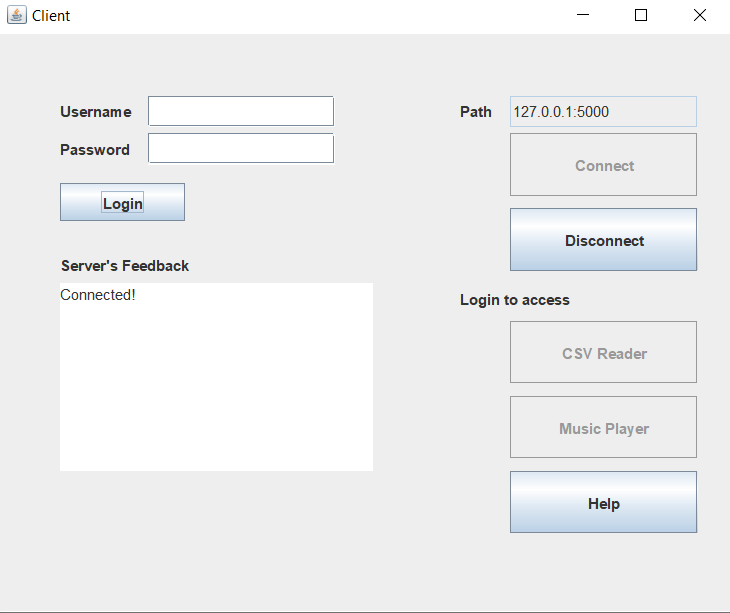


Figure 2

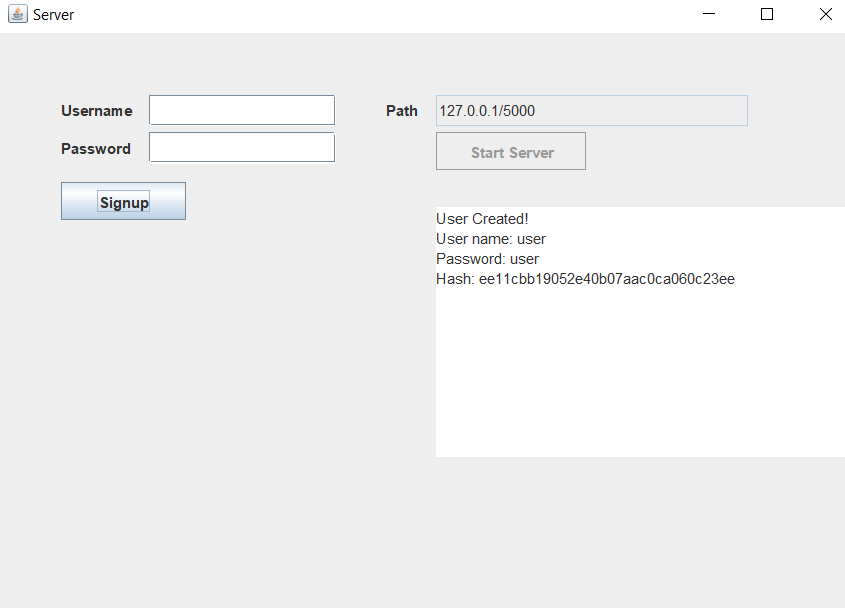


Figure 3

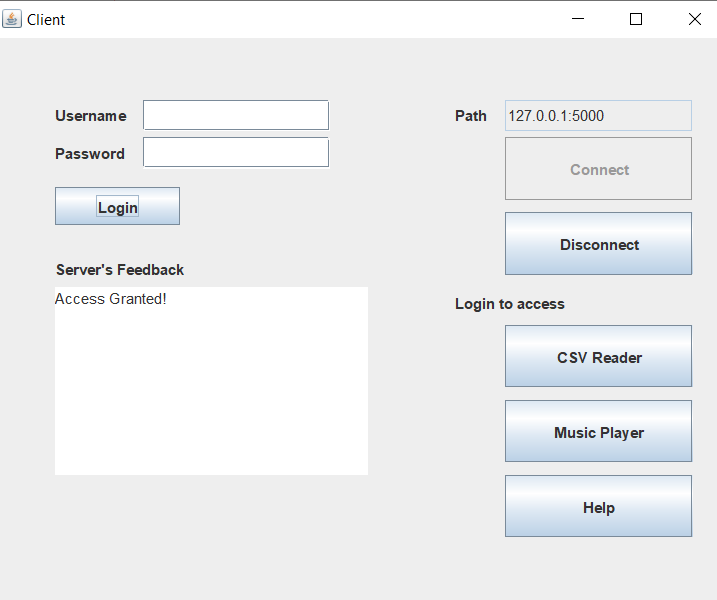


Figure 4

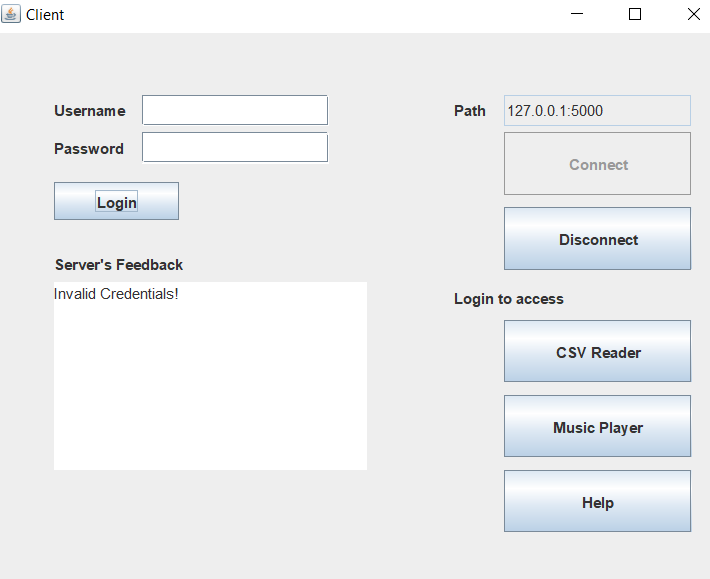


Figure 5

## CSV Reader

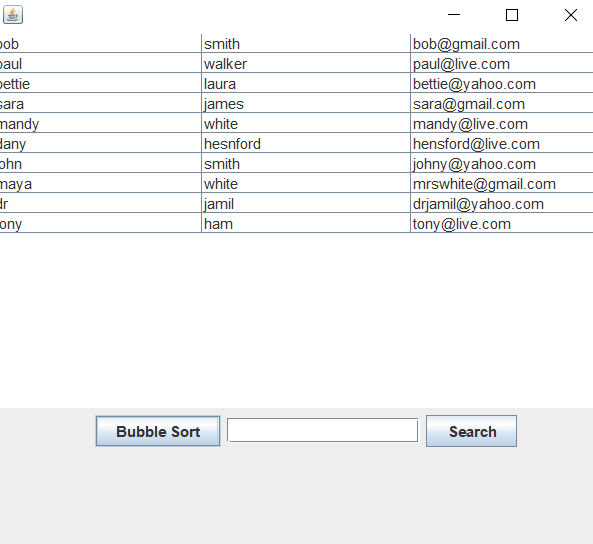


Figure 6

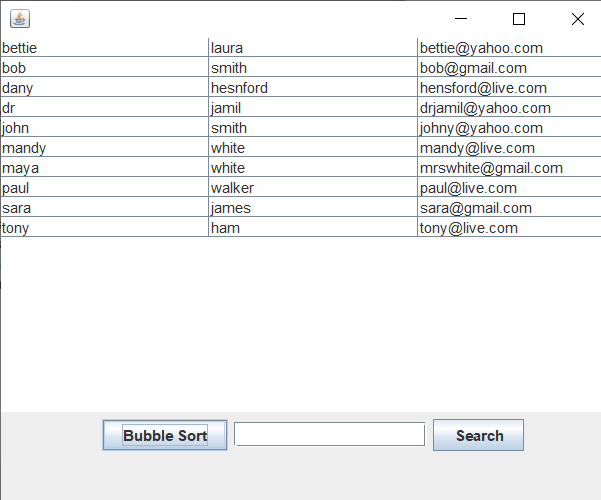


Figure 7

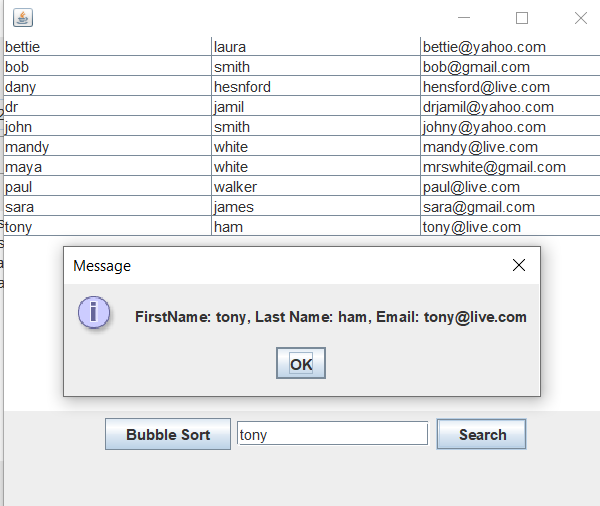


Figure 8

## Audio Player

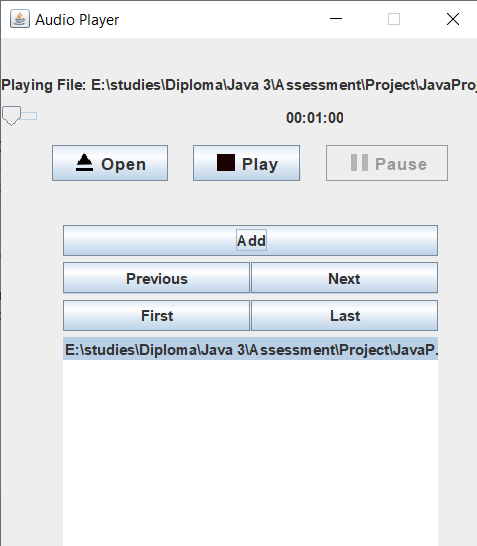


Figure 9

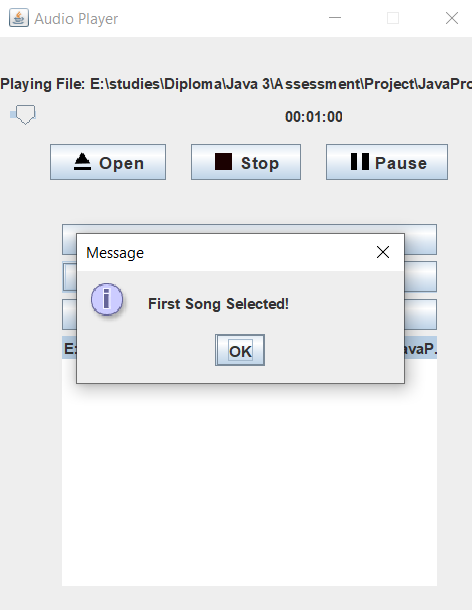


Figure 10

## Help File

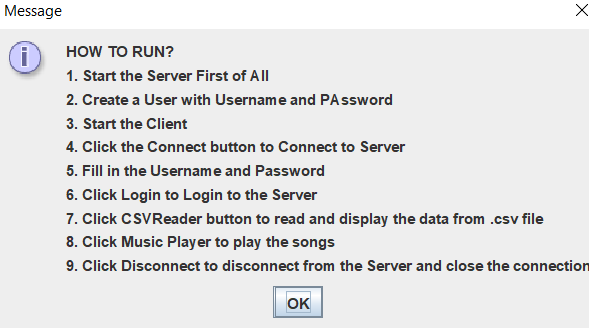


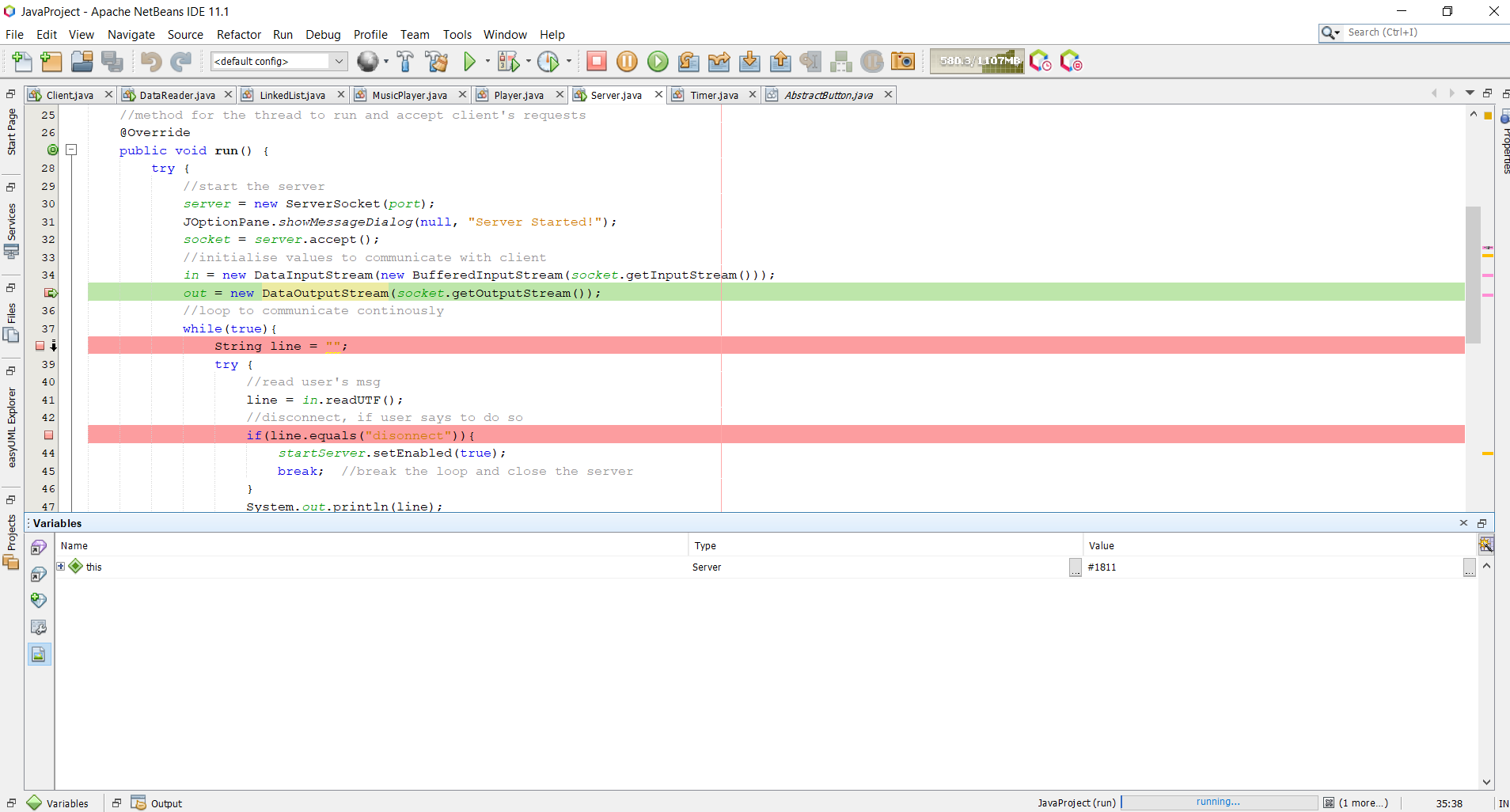
Figure 11

# Debugging

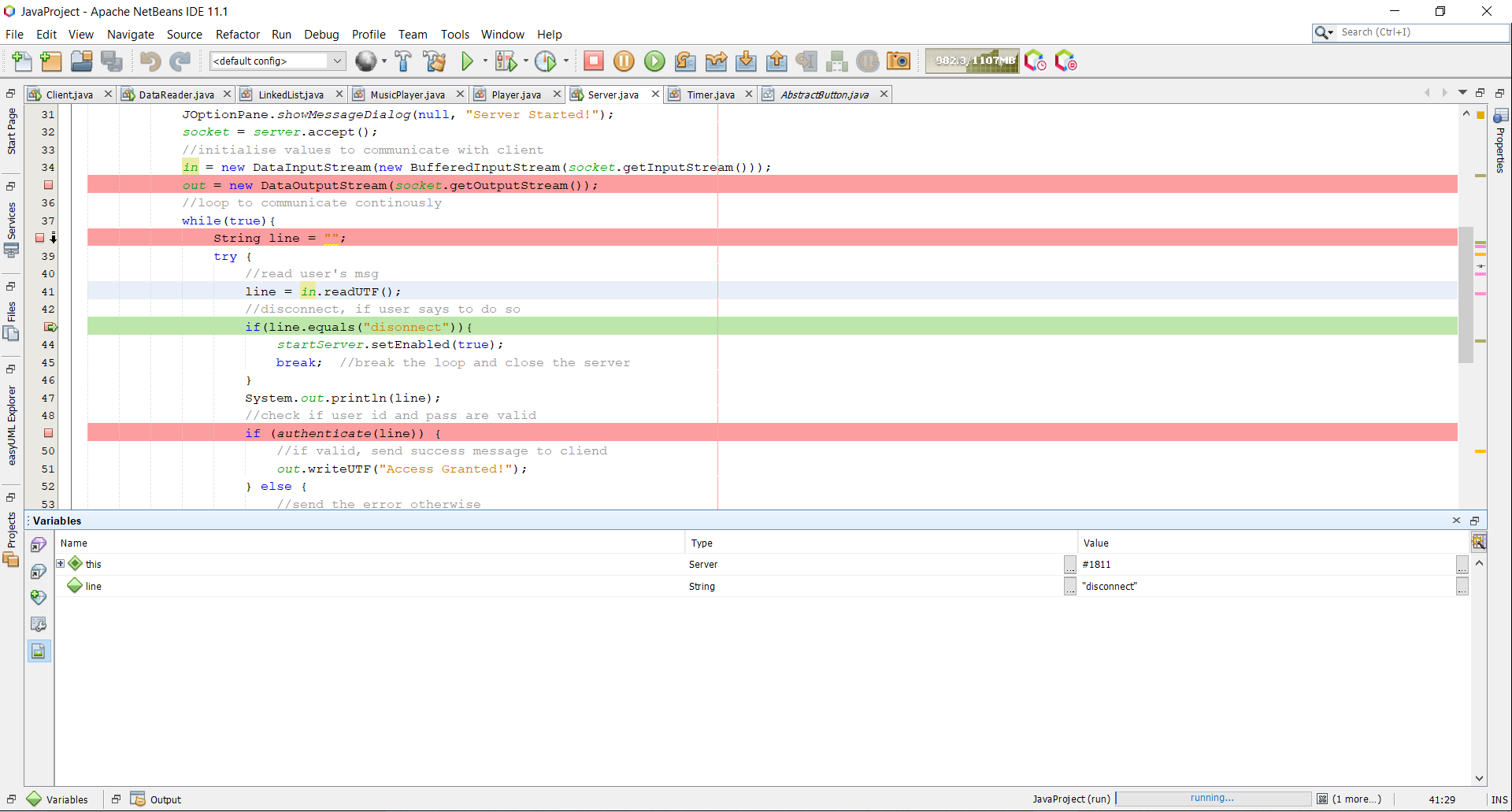
## Introduction

Debugging is a very important process in programming, it is a step which need to be taken when a program is being build and before it have been released to the client. it is a process in part of building a application in order to help programmers find and identify the source of the problem, and the programmers can determine a way to work around the problem.

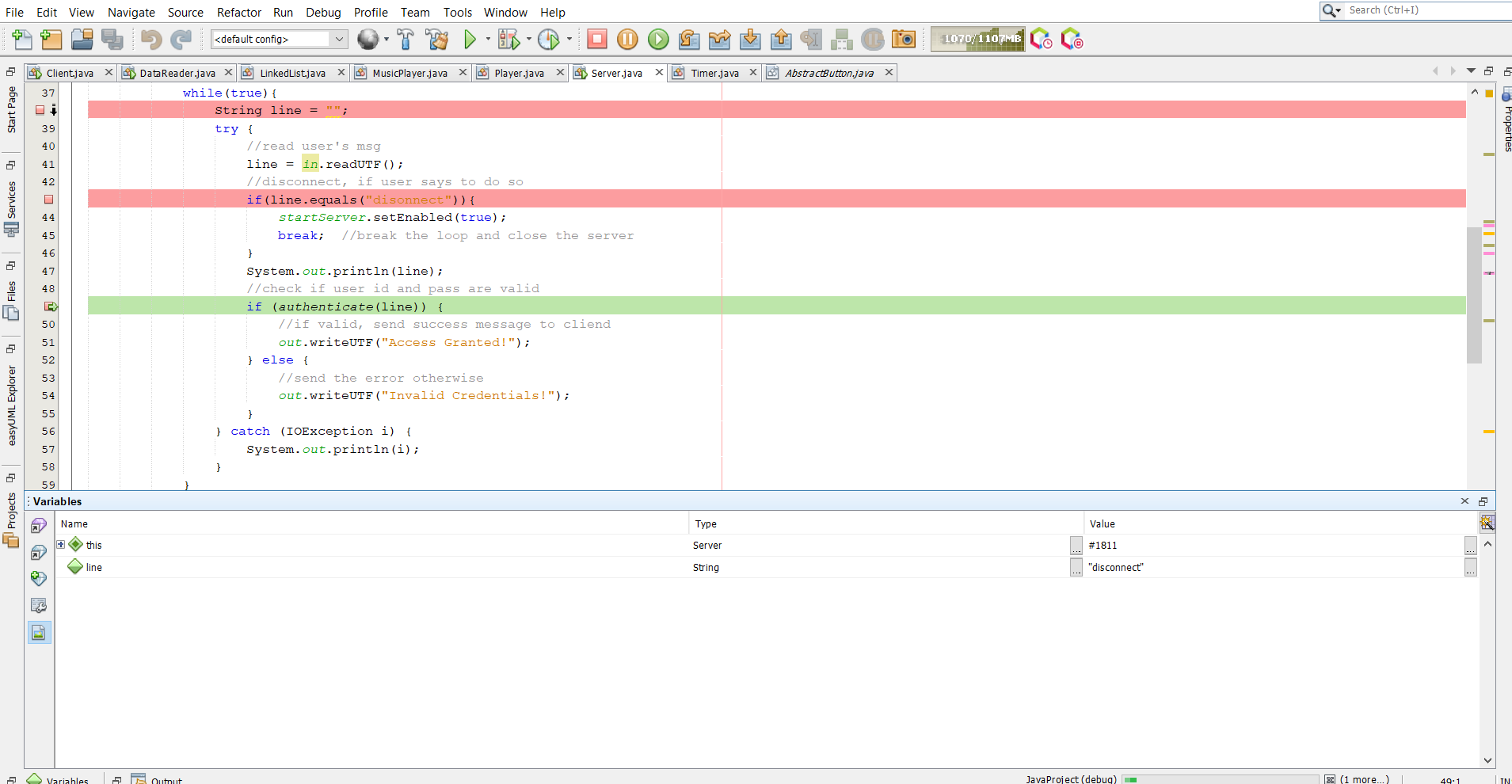
## Server



This is the debugging screenshot of starting up the server.

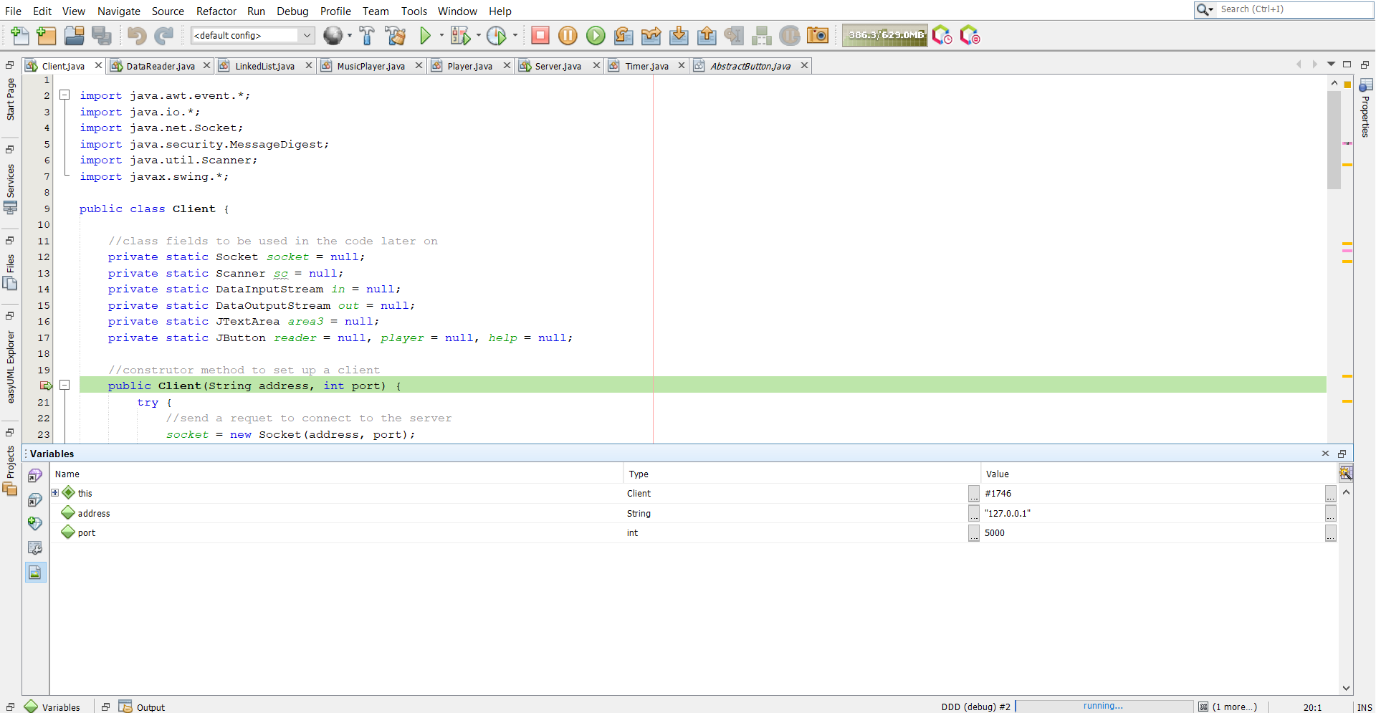


This is the debugging screenshot of the ‘if’ statement at the server class when disconnect was clicked on the client class.

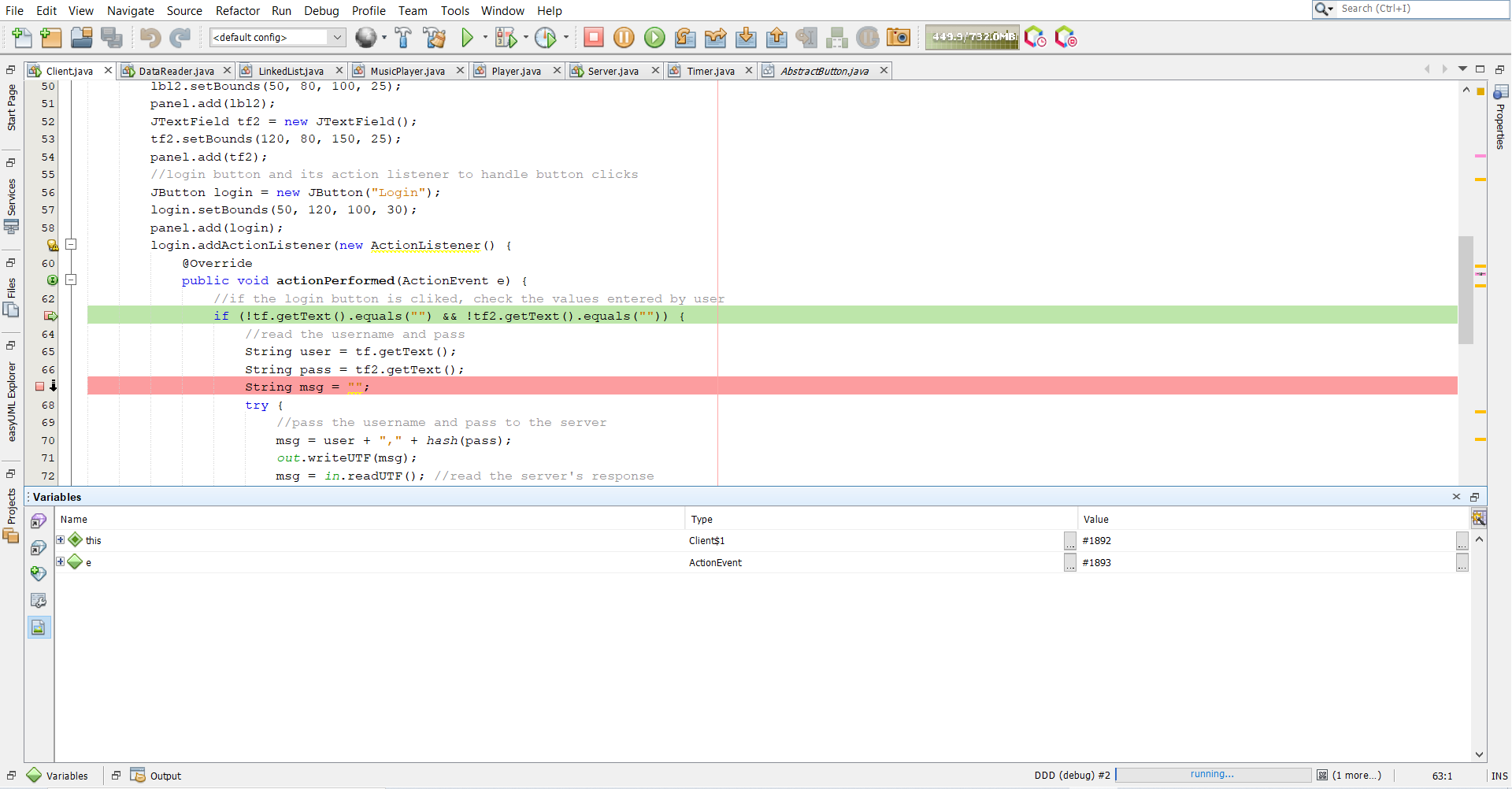


This is the debugging screenshot of the ‘if’ statement of the server class when the client class is trying to connect to the server.

## Client

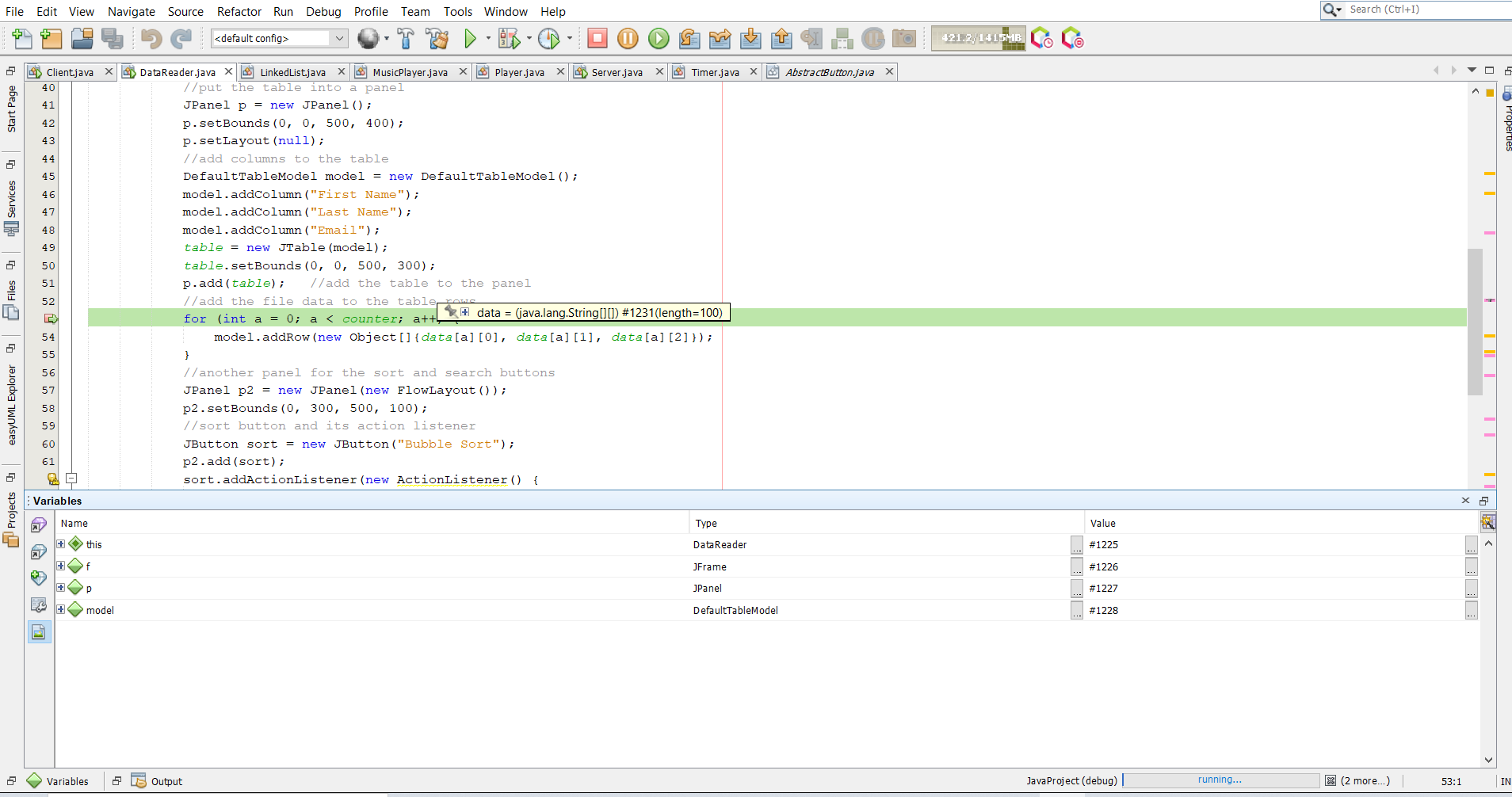


This is the debugging screenshot of the ‘if’ statement of the client class when connect was clicked.

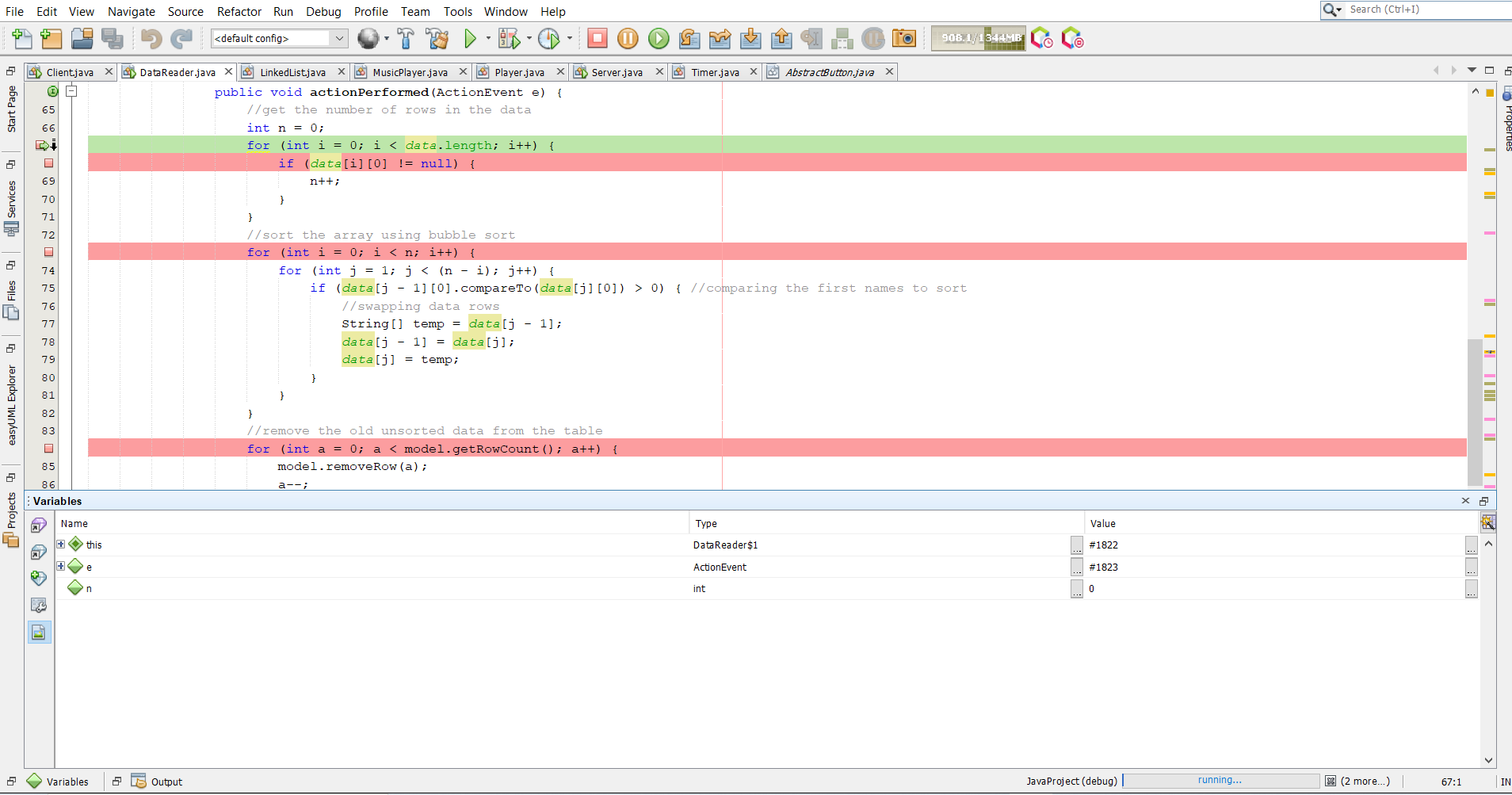


This is the debugging screenshot of the ‘if’ statement of the client class when trying to login to the server.

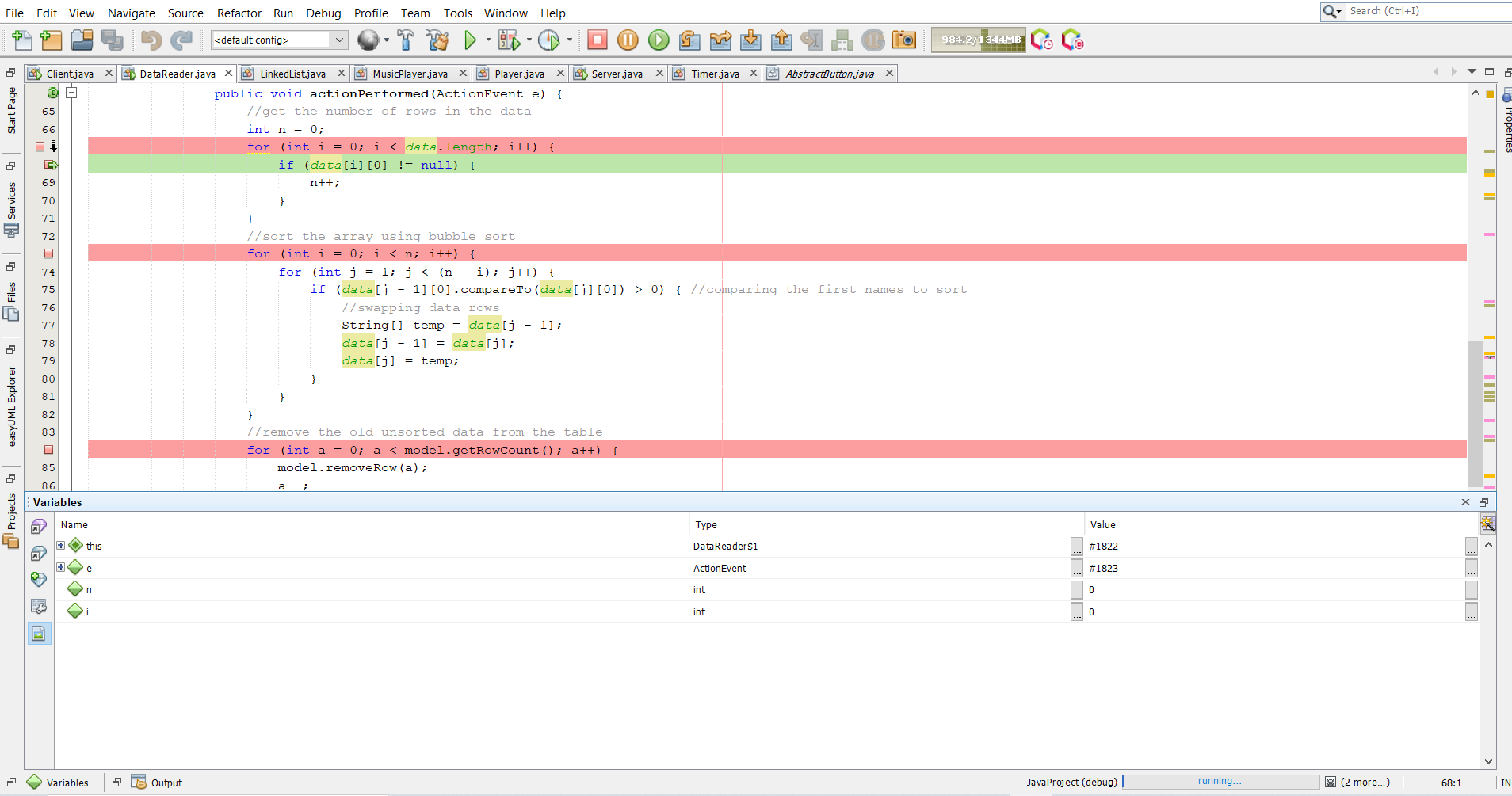
## Data Reader



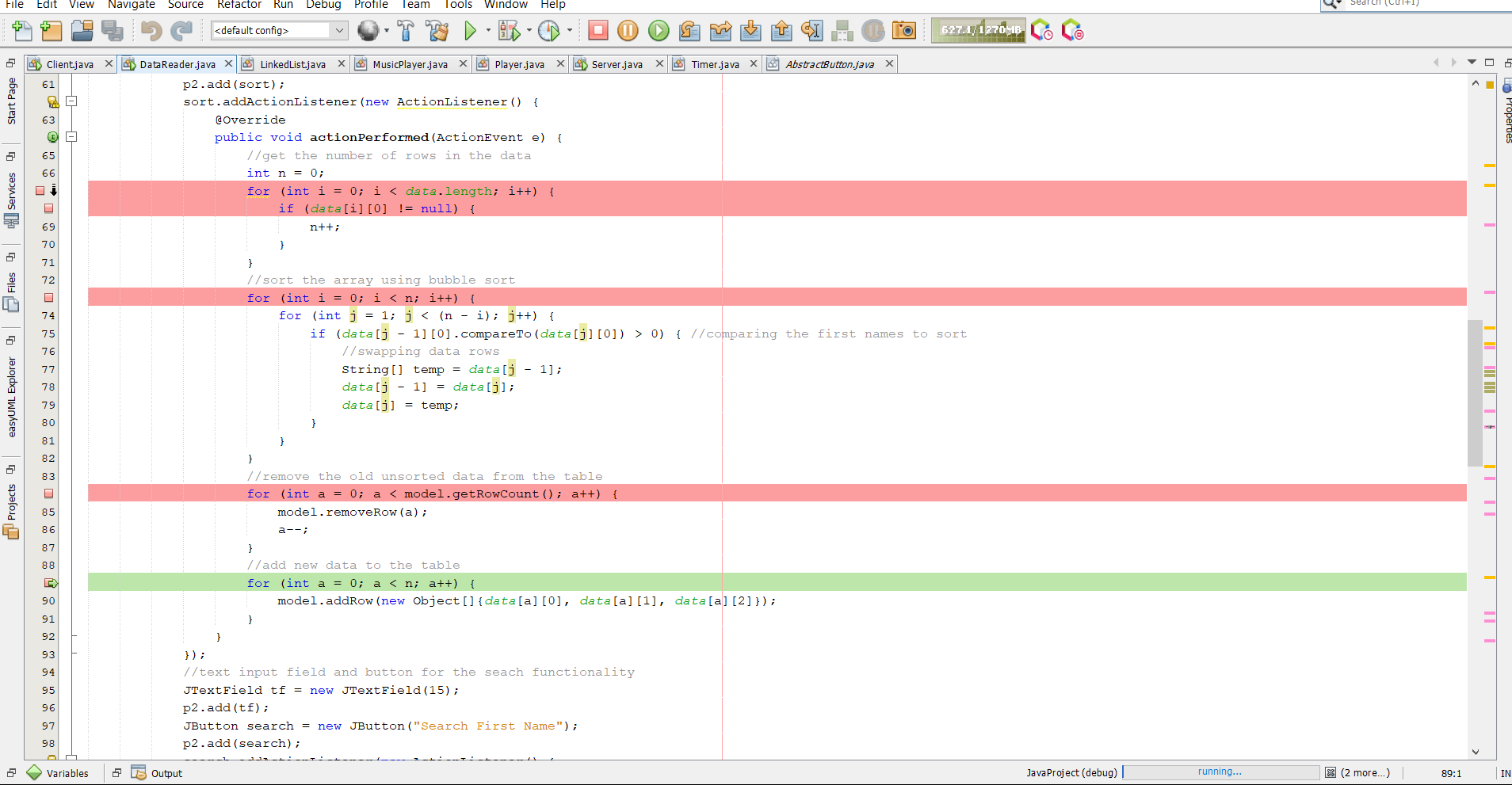
This is the debugging screenshot of the ‘for’ statement in the DataReader class when the CSV reader button have been clicked.



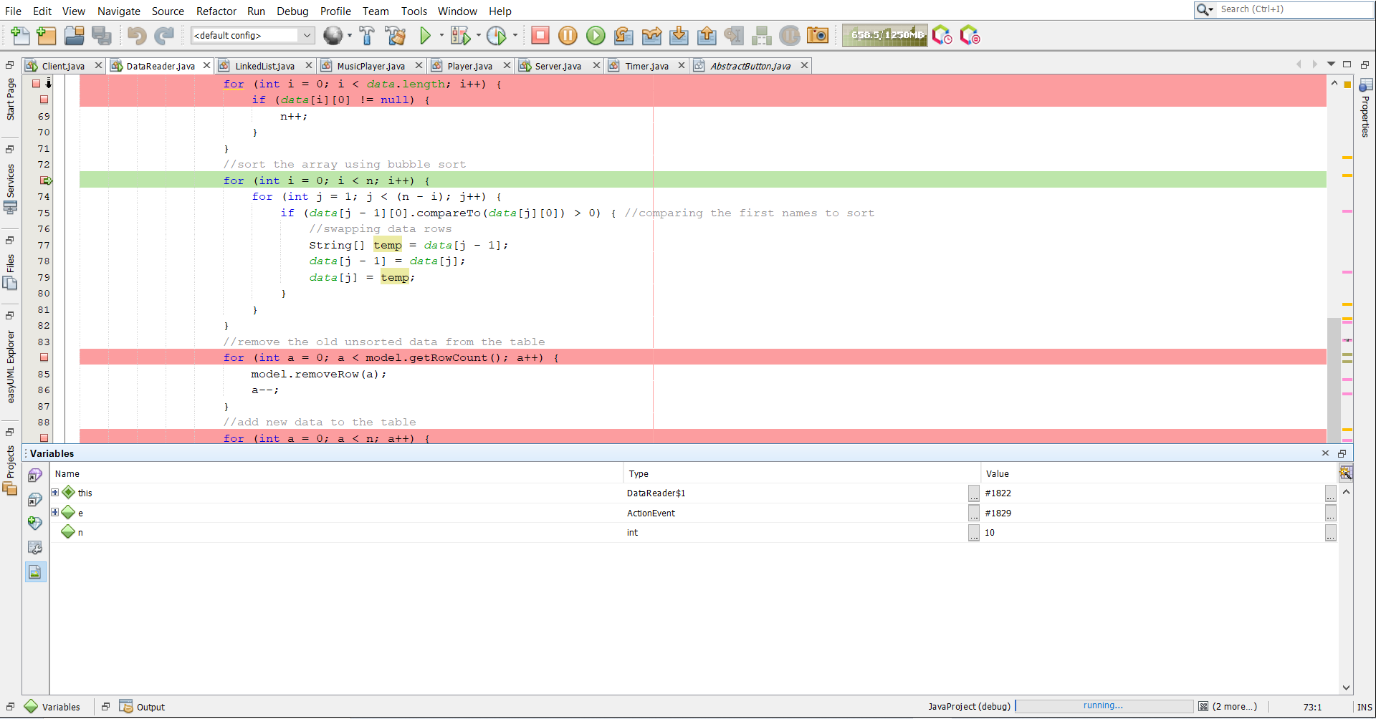
This is the debugging screenshot of the ‘for’ statement when the bubblesort button have been clicked in the DataReader class.



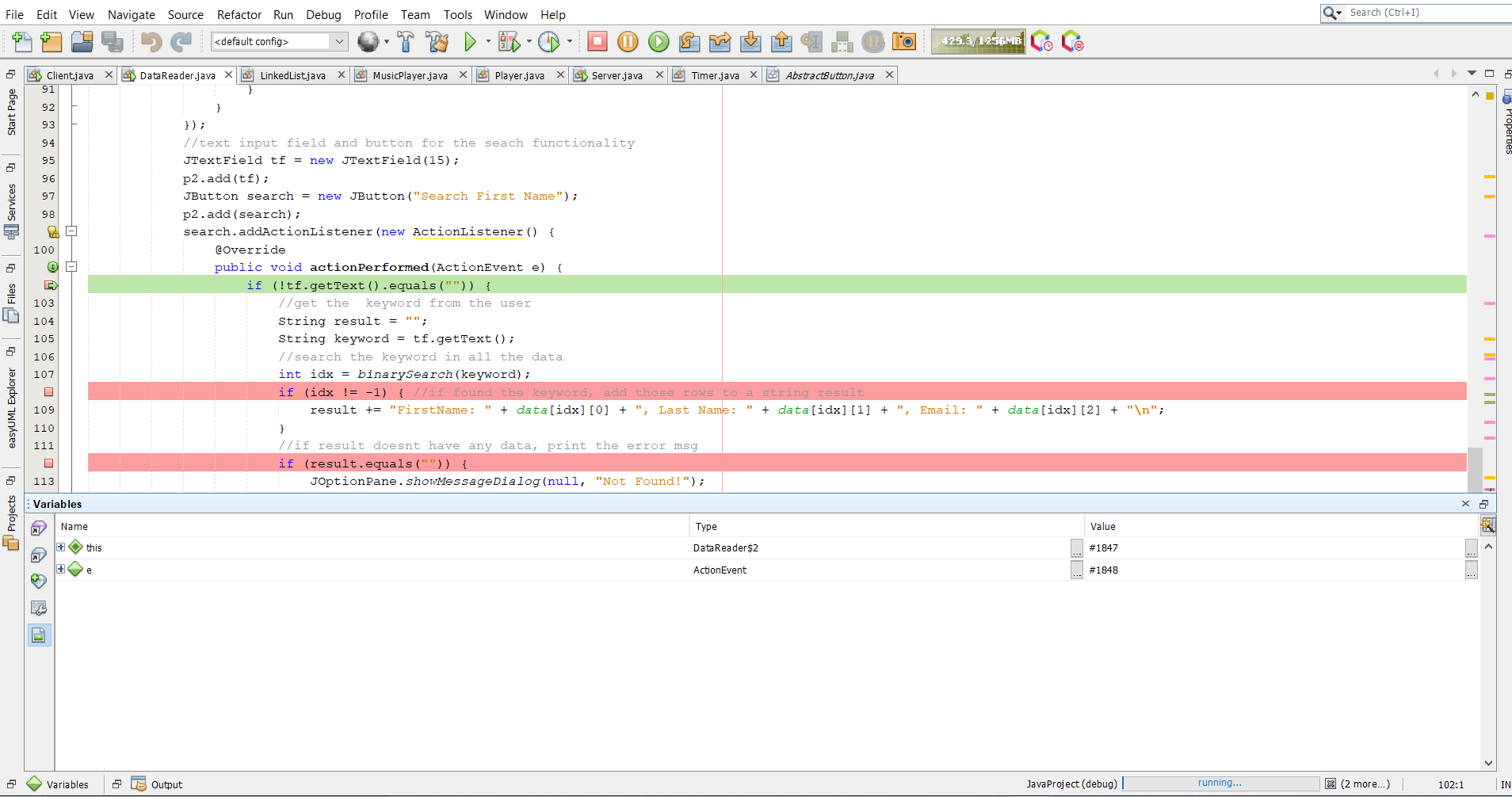
This is the debugging screenshot of the ‘if’ statement when the bubblesort button have been clicked.



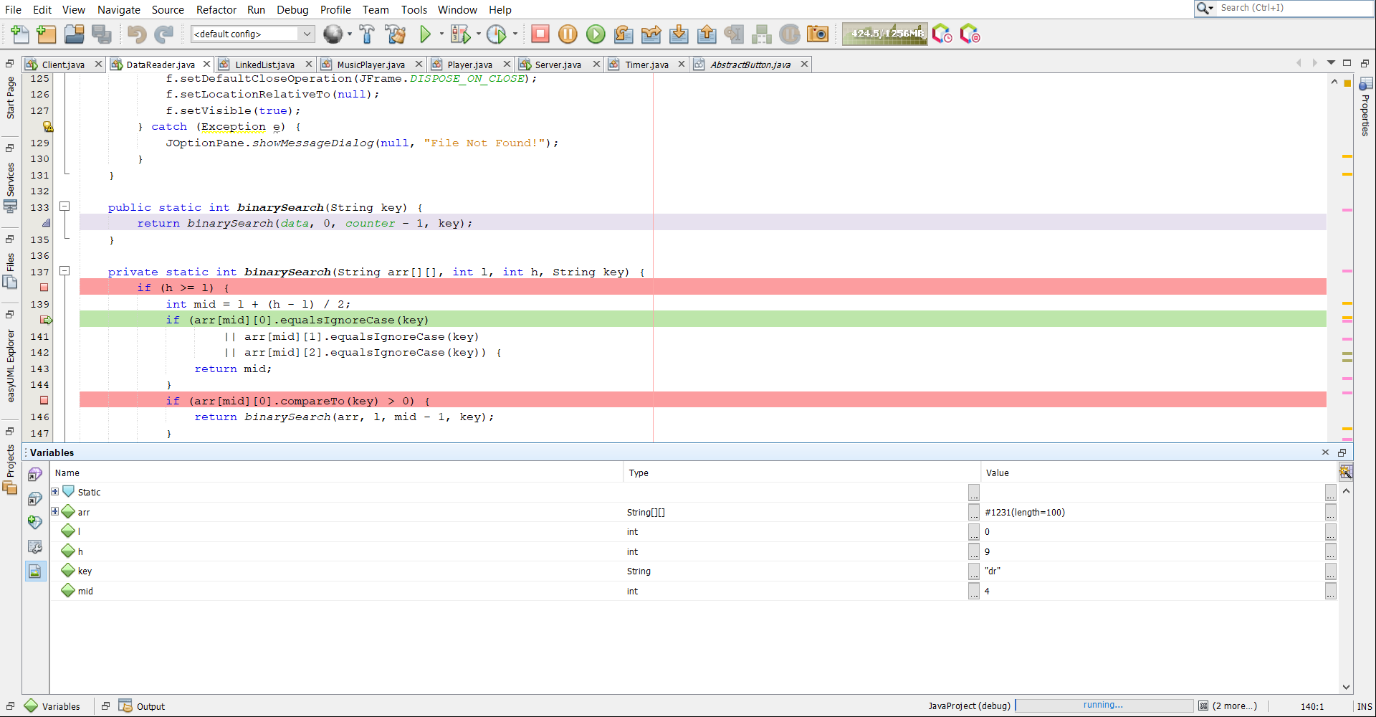
This is the debugging screenshot of the ‘for’ statement to replace the sorted data in the table.



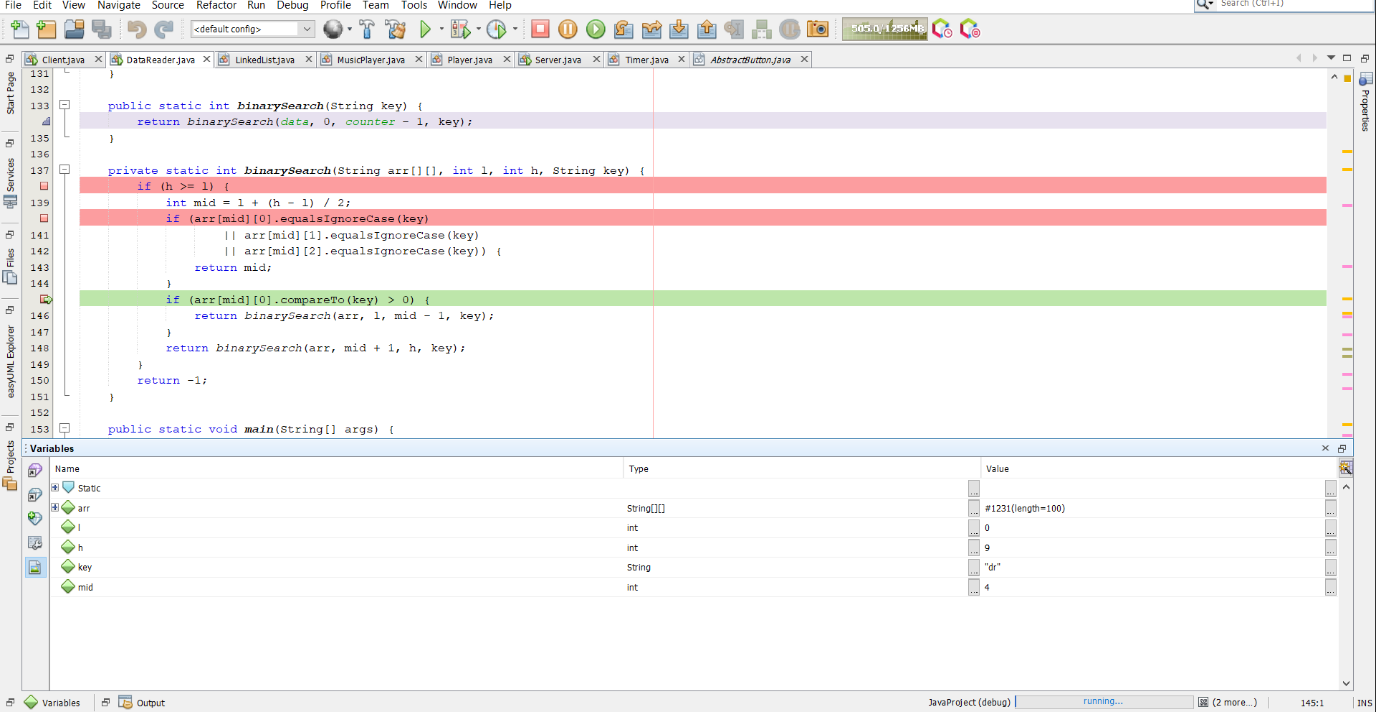
This is the debugging screenshot of the ‘for’ statement for the bubblesort method.



This is the debugging screenshot of the ‘if’ statement when search button have been clicked



This is the debugging screenshot of the ‘if’ statement for binary search method.



This is the debugging screenshot of the ‘if’ statement to search for the object in the list.

# Junit Testing

## Introduction

jUnit is a method contained in a class that is used for testing. The class will be called as a ‘Test class”. To define that a the method in the class is used for testing, it will be annotated with a ‘@Test’ annotation.

## Result

