ACtivity 3 30019812

Java 3

Contents

[Introduction 1](#_Toc58071578)

[Test Table 2](#_Toc58071579)

[Debugging 3](#_Toc58071580)

[Test proof 1: 3](#_Toc58071581)

[Test proof 2: 3](#_Toc58071582)

[J-unit Testing 4](#_Toc58071583)

[UML Diagrams: 5](#_Toc58071584)

[Client UML 5](#_Toc58071585)

[Server UML 5](#_Toc58071586)

[Source Control 5](#_Toc58071587)

[Functions 6](#_Toc58071588)

[Reference images 7](#_Toc58071589)

# Introduction

I was hired to develop a server client login where the client is able to login and access a media player. The client should then be able to make use of the functions in the media player allowing them to play songs, load songs from a csv file, sort these songs as well as save them to a new csv file.

I was asked to follow certain standards and requirements given by the client. For sorting I made use of a bubble sort method instead of the initial merge sort, for hashing I made use of the SHA-256 method, for searching I made use of a simple Linear search method. I made use of Linked lists as a data structure and finally for a 3rd party library I made use of OpenCSVReader to be able to read csv files and display them in the list box of the media player. All methods were separated by comments that explain what that method does

The source control used was Github and it will contain both the j-unit test program in initial programs as well as the test documentation and all the libraries used.

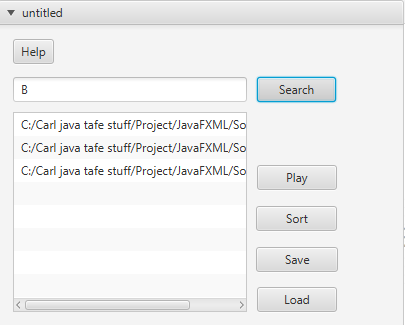
# Test Table

|  |  |  |
| --- | --- | --- |
| What is being tested | Test | Did it pass |
| Username and password connection | Ref 1 | Pass |
| Incorrect Username and password connection | Ref 2 | Pass |
| Play Button song | Ref 7 | Pass |
| Search song in the list | Ref 5, 6 | Pass |
| Save to csv | Ref 8, 11 | Pass |
| Load from csv file | Ref 3 | Pass |
| Sort the list in the list box | Ref 4 | Pass |
| Server receives hashing for password | Ref 1 | Pass |
| Incorrect song search | Ref10 | Pass |
| Help File | Ref 9 | Pass |

# Debugging

## Test proof 1:

Testing to see if the Search button does select the song from the list and plays it?

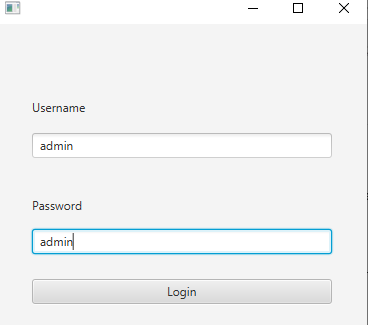


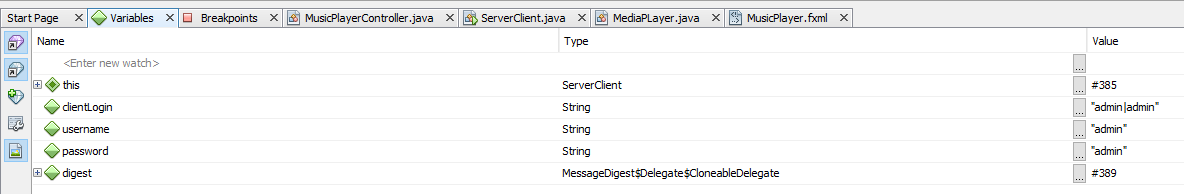


As you can see, I’m searching for the third song in the list in the csv file and therefore receiving the correct song to play.

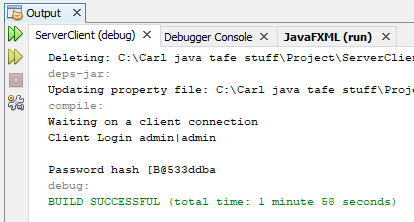
## Test proof 2:

Testing to see if the hash code is the exact password it is connected to and not linked to a different one?



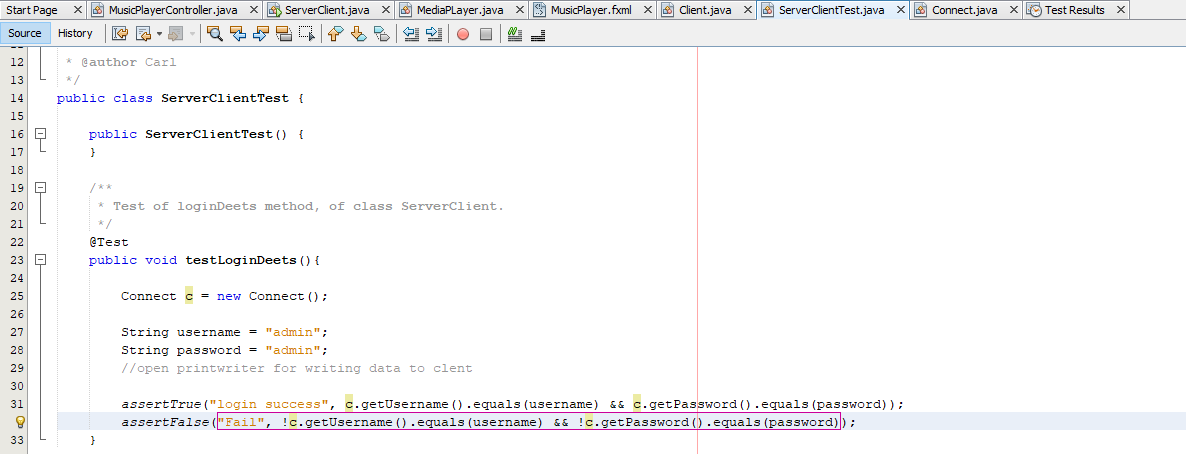


It selects the password “admin” and steping further into it, it compares to see if the hash for admin in the login from the client end matches the hash for admin.

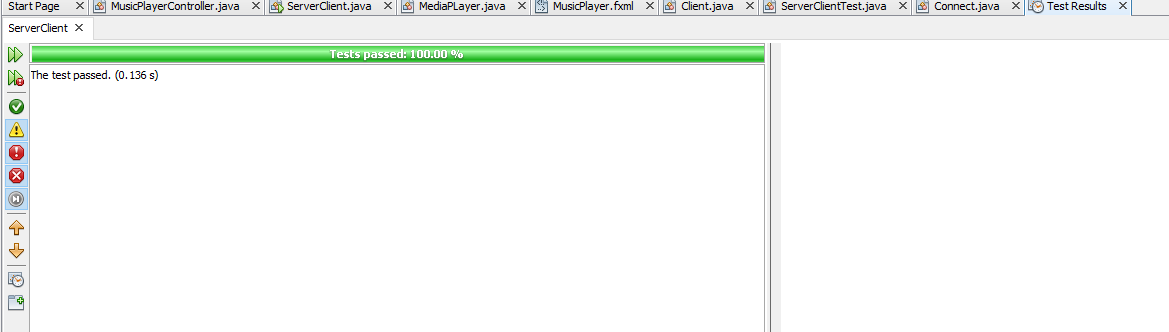


Therefore, I am able to say that the test is true.

# J-unit Testing



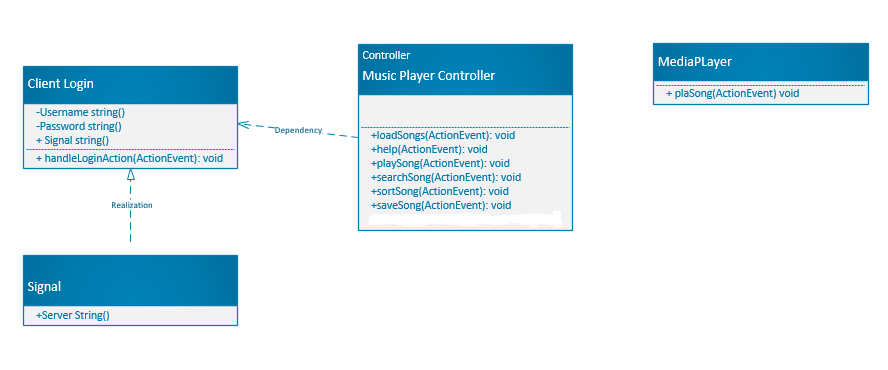
Result for both tests



As you can see I ran a test to see if the file will fail at the password and username not being equal to the connect class getUsername and getPassword and will pass at the password and username being equal to the connect class getUsername and getPassword.

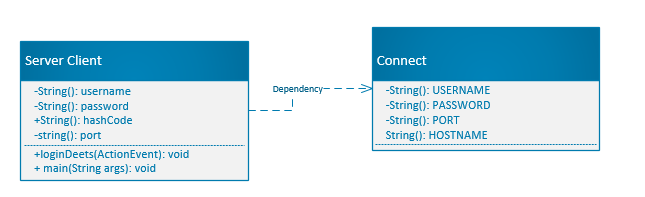
# UML Diagrams:

## Client UML



Updated from the prior documentation. As you can see a new class was created called mediaPlayers and a double Mouse click button was added as well to allow music to be played when double clicked in the list View.

## Server UML



# Source Control

# Functions

Login:

This allows the user to access the media player and play songs.

Hashing:

The server receives the user login details and hash code connected to the password.

Sorting:

If the list box is unsorted, by clicking the sort button the user is able to sort the songs in alphabetical order.

Search:

If the list box contains quite a bit of songs, the user doesn’t have to scroll through each of them, instead he is able to type it in the textbox and click “find” once the song is found, the media player will then play that song.

Save:

This allows the user to make use of the 3rd party library to save the songs in the list box to a new csv file called “NewSongs.csv”.

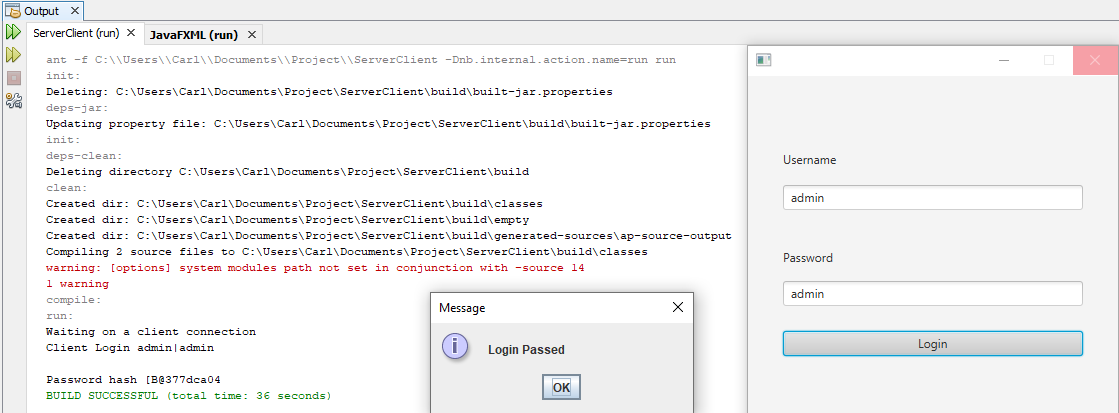
Load:

The user is able to populate the songs1.csv file and play the songs that are in the file.

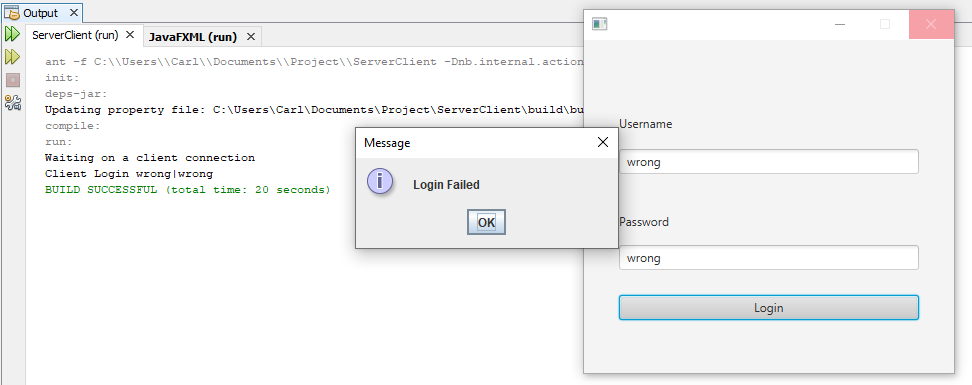
Play:

User clicks to play to play the first song in the sorted listbox.

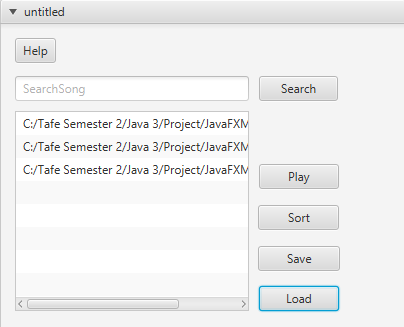
# Reference images



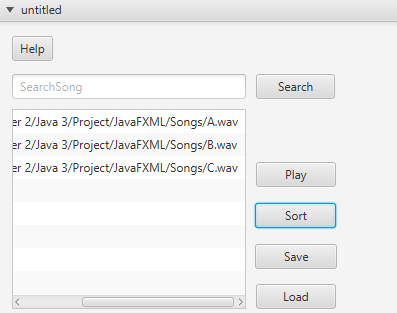
Ref 1



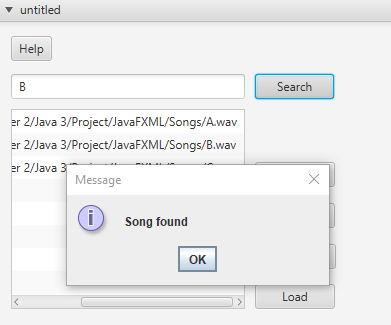
Ref 2



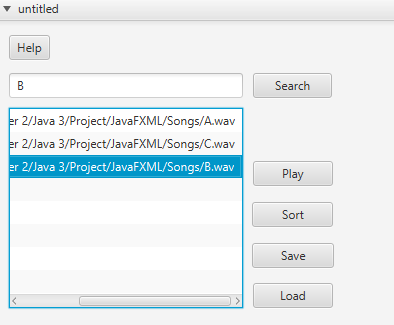
Ref 3



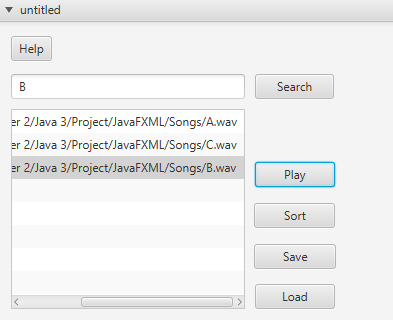
Ref 4



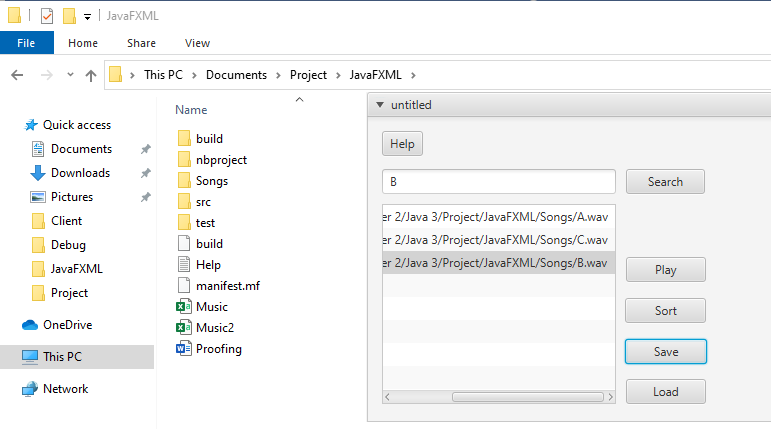
Ref 5



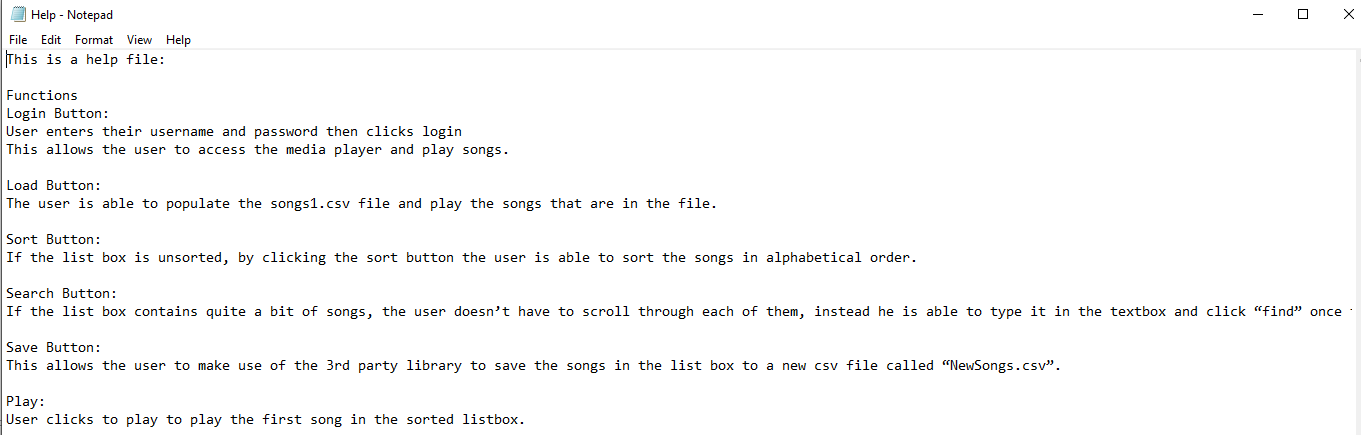
Ref 6



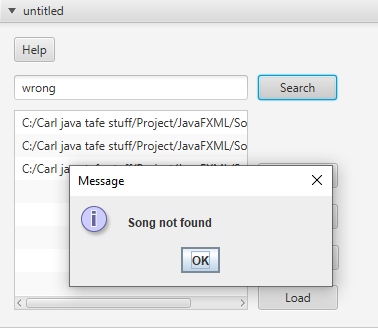
Ref 7



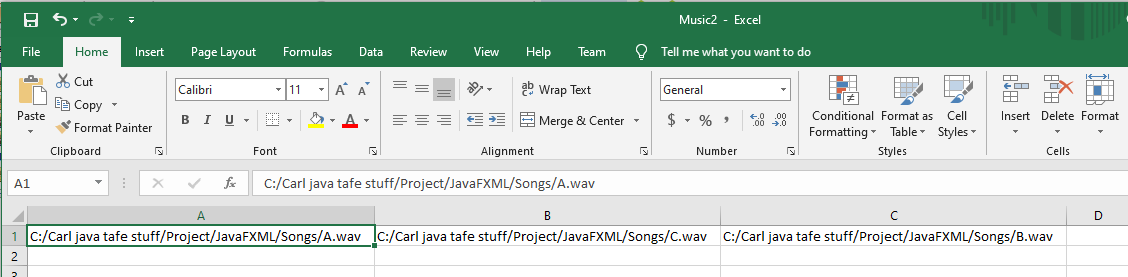
Ref 8



Ref 9



Ref 10



Ref 11