INDIVIDUAL ASSIGNMENT

JAVA PROGRAMMING

|  |  |  |
| --- | --- | --- |
| **Module Code** | **:** | AAPP004-4-2-JP |
| **Intake Code** | **:** | UCDF2005ICT(SE) |
| **Lecturer Name** | **:** | Tanveer Khaleel Shaikh |
| **Hand in Date** | **:** | 10th April 2022 |
| **Tutorial No.** | **:** | Lab 6 |
| **Member** | **:** | Mannoj A/L Sakthivel |
| **Student ID** | **:** | TP060500 |

Contents

[1. Introduction 3](#_Toc100424751)

[Header Files/ Import Files 3](#_Toc100424752)

[2. Variable 6](#_Toc100424753)

[3. Control Structure 6](#_Toc100424754)

[IF 6](#_Toc100424755)

[IF-ELSE 7](#_Toc100424756)

[4. Looping 8](#_Toc100424757)

[While loop 8](#_Toc100424758)

[For loop 9](#_Toc100424759)

[5. OOP Concepts 9](#_Toc100424760)

[Class 9](#_Toc100424761)

[Object 10](#_Toc100424762)

[Encapsulation 11](#_Toc100424763)

[Generalization 12](#_Toc100424764)

[Constructor 12](#_Toc100424765)

[Get and Set 13](#_Toc100424766)

[Normal Methods 14](#_Toc100424767)

[File Concept 15](#_Toc100424768)

[Write data and Read data 15](#_Toc100424769)

[Search file 16](#_Toc100424770)

[Delete file 17](#_Toc100424771)

[Update file 18](#_Toc100424772)

[6. Sample Output Screens 19](#_Toc100424773)

[7. Additional Features 42](#_Toc100424774)

[Array 42](#_Toc100424775)

[JCalendar 43](#_Toc100424776)

[JTable 44](#_Toc100424777)

[8. Assumptions 45](#_Toc100424778)

[9. References 46](#_Toc100424779)

# Introduction

## Header Files/ Import Files

1. java.io.BufferedReader;
2. java.io.FileReader;
3. java.io.FileWriter;
4. java.io.IOException;
5. java.text.SimpleDateFormat;
6. java.time.LocalDate;
7. javax.swing.JOptionPane;
8. java.io.BufferedWriter;
9. java.io.File
10. java.io.FileNotFoundException;
11. java.util.logging.Level;
12. java.util.logging.Logger;
13. java.awt.event.WindowEvent;
14. java.awt.event.WindowListener;
15. javax.swing.table.DefaultTableModel;
16. java.util.Scanner;
17. Java.io.BufferedReader;

To read text from a character-based input stream, the Java BufferedReader class is utilised. The readLine() function may be used to read data line by line. It accelerates performance. [Javatpoint, 2022]

1. Java.io.FileReader;

To read data from a file, the Java FileReader class is utilised. It, like the FileInputStream class, returns data in byte format. It is a character-oriented class that is used in Java for file processing. [Javatpoint, 2022]

1. Java.io.FileWriter;

The java.io package's FileWriter class may be used to write data (in characters) to files. It is a subclass of the OutputStreamWriter class. [programiz, 2022]

1. Java.io.IOException;

IOException is a kind of exception that programmers use to indicate a failure during Input and Output activities. This is a validated exception. The programmer must subclass IOException and should throw the appropriate subclass of IOException depending on the circumstances. [Kommadi, 2021]

1. Java.text.SimpleDateFormat;

SimpleDateFormat is a concrete class that defines how dates should be formatted and parsed in a locale-sensitive way. It supports date-to-text formatting, text-to-date parsing, and normalisation. SimpleDateFormat enables you to begin by selecting any user-defined date-time formatting pattern. [SimpleDataFormat, 2020]

1. Java.time.LocalDate;

LocalDate is an immutable date-time object that represents a date, which is often represented using the year-month-day format. Additionally, other date parameters such as year-of-year, week-of-year, and week-of-year may be accessed. For instance, the value "October 2nd, 2007" may be saved in a LocalDate. [LocalDate, 2022]

1. Javax.swing.JOptionPane;

javax. swing is a Java API class. JOptionPane has the ability to create a dialogue box that appears on the user's computer's desktop in order to solicit input or show messages to the user.

1. java.io.BufferedWriter;

To provide buffering for Writer instances, the Java BufferedWriter class is employed. It accelerates performance. It derives from the Writer class. Buffering characters are used to write single arrays, characters, and strings efficiently. [BufferedWriter, 2022]

1. java.io.File

File. This package includes all classes necessary for developing user interfaces and painting graphics and pictures. Provides input and output to and from the system through data streams, serialisation, and the file system. [nd, 2022]

1. java.lang.FileNotFoundException - java.lang.FileNotFoundException. This error is generated when an attempt to open a file indicated by a pathname fails. [Maneas, 2021]
2. java.util.logging.Level;

Exceptions, information, and warnings are stored in log files as messages that occur during the execution of a programme. Logging assists a programmer in troubleshooting a programme. Java's java.util package has a logging feature. [Joshi R, 2015]

1. java.util.logging.Logger;

logging. util. java. In Java's logging API, the Logger class is used to record application messages. [Logger in java, 2020]

1. java.awt.event.WindowEvent;

WindowEvent is a class. A low-level event that signals that the state of a window has changed. When a Window object is opened, closed, active, deactivated, iconified, or deiconified, or when the focus is transferred into or out of the Window, this low-level event is triggered. [platform se 7, 2020]

1. java.awt.event.WindowListener;

The window's listener interface. The class that wishes to handle window events either implements this interface (and all of its methods) or extends the abstract WindowListener class (overriding only the methods of interest). The class's listener object is then registered with a window through the window's addWindowListener function.

1. javax.swing.table.DefaultTableModel;

DefaultTableModel class. This is a TableModel implementation that stores the cell value objects in a Vector of Vectors. Warning: DefaultTableModel returns an Object column class. [Defaulttablemodel, 2020]

1. import java.util.Scanner;

The Scanner class is used in the java.util package to read data from various sources such as input streams, users, and files. [Java Scanner class, 2022]

# Variable

Graphical user interface, text, application, email

Description automatically generated

Figure 1: Example Code

As shown from the diagram above, fileStr is a variable. This variable is in charge of making the combo box named Viewver to choose between 2 files based on the combo Box’s input. For example, if the combo Box is set to getSelectedIndex()==0, then the variable’s content is “JungleView.txt” otherwise the variable’s content is “SeaView.txt”. After the content of the variable is chosen, the variable is used in “file” (by importing File), to be further used in the try and catch session for searching for booking based on File name.

# Control Structure

## IF

Graphical user interface, text

Description automatically generated

Figure 2: Example Code

record = read lines in file

br = BufferedReader of file

bw= BufferedWriter of temporary file

The process shown above is to delete the line with the selected ID.

As shown from the diagram above, if the record contains the ID inputted, the if statement will allow the system to “continue”, meaning just to obtain the line where the ID is inputted. The process then breaks out of the if statement to continue writing the record.

## IF-ELSE

**Graphical user interface, text, application

Description automatically generated**

Figure 3: Example Code

The process shown above is the modifying process using JTable, where the new input must be validated accordingly.

As shown from the figure above, if the NameM (modified name) is less than 4 in length, thus, the system will show an error and say that the username is too short. Furthermore, the system will set the input text field to an empty string (“”), for the user to retry. Else if, the username is too long (length of greater than 12), will continue to set the input text field to an empty string. This process is the same for ICM (modified IC number). Finally, the else statement will go through if the ‘if’ and ‘else if’ statements are justified (follows the condition given). The else statement will then write the modified Strings to the file respectively.

# Looping

## While loop

Graphical user interface, text, application

Description automatically generated

Figure 4: Example Code

i is an empty string (String i = “” ;)

As shown from the diagram above, the while loop will go on as long as the file in not empty (null). Thus, in the while loop, empty string (i) + read lines in the file together with line separator (\r\n) . l is equal to BufferedReader of the lines in the file. In conclusion, it is reading the lines in the file. Outside the file, the empty string i will replace the old details with the new details (booking) and thus using FileWriter method.

## For loop

Text

Description automatically generated

Figure 5: Example Code

As shown from the diagram above, the for loop will ensure that the string a will go through the whole array. The array on the other hand, is basically using l, which is reading the lines inside of the file unless null. The string l is then splitting the file using the delimiter (“,”). Inside the for loop however, as long as the string a is equivalent to the word available in the file, the rooomcombo (combo box) will add the roomID which is in the 0th index of the file. In conclusion, the for loop is searching for the word available based on the file to then be later be used inside of an if statement to display the roomid accordingly in the combo box.

# OOP Concepts

## Class

Graphical user interface, text

Description automatically generated

Figure 6: Example Code

As shown in the diagram above, the class being used here is the java assignment class. In this class, a main method is used to call the object in the class which is loginoriginal and thus, by calling it, it has to be set visible. This can be done by using the function of swing that is objectname.setVisible(true);

## Object

Graphical user interface, text

Description automatically generated

Figure 7: Example Code

As shown in the diagram above, the object (lg) is being called inside of the main function. As soon as the object is called, the object must be set to visible for it to be displaced in Java swing. In normal cli java however, the object only needs to be called once to execute the programme. In other words, setVisible is not needed in cli using Java.

## Encapsulation

Text

Description automatically generated

Figure 8: Example Code

Text

Description automatically generated

Figure 9: Example Code

As shown from both the diagrams above, the act of encapsulation is being used. This is since I am first declaring a class called BookingFacts and inside of the class setting a private String View. Then since the one of the parameters of the BookingFacts is View, there is a need for BookingFacts.View = View. Finally, using a get method to obtain String of getView() using the return method. Then, as shown in the second diagram, String View is using the getter method which is BookingFacts.getView();

## Generalization

**A screenshot of a computer

Description automatically generated**

Figure 10: Example Code

As shown in the diagram above, the generalization OOP concept is being used. In this case, the PaymentJungle jForm is extending its range to use javax.swing.JFrame to allow the use of all the javax.swing.JFrame functions into the use of PaymentJungle. In other words, if the extends does not exist the JFrame functions will not work thus, there is no GUI used. JFrame is a top-level container that creates an on-screen window. A frame is a basic window that all other components, such as the menu bar, panels, labels, text fields, and buttons, depend on. The JFrame window is the default starting point for almost every other Swing application.

## Constructor

****

**Text

Description automatically generated**

Figure 11: Example Code

As shown from both the diagram above, a constructor method is being used. This is after using a get method from the class BookingFacts(). The bookingFacts bd = new BookingFacts(variables) is being called upon again for future use. This is because the BookingFacts class will be need to be used later for writing the data back in the file. This is the booking process.

## Get and Set

Graphical user interface

Description automatically generated with medium confidence

Figure 12: Example Code



Figure 13: Example Code

Text, letter

Description automatically generated

Figure 14: Example Code

As shown from the pictures above, there is a need to get the variables stated above to be later then used in modifyCustDeet() method. Thus, a get method is used to get the variables and then return it as demonstrated on the third diagram. If there is no get method thus, the BookingFacts() used in the third diagram will not be processed and thus resulting in an unknown variable as the variable is not passed. In conclusion, the get method used is passing data back to the modifyCustDeet() method.

## Normal Methods

****

Figure 15: Example Code

As shown in the diagram above, the normal method used is by calling the modifyCustDeet() method is to execute after it is being called.

Text

Description automatically generated

Figure 16: Example Code

The modifyCustDeet() method will then be executed as soon as it is being called as shown in the second diagram above.

## File Concept

### Write data and Read data

****

Figure 17: Example Code

**Graphical user interface, text, application

Description automatically generated**

Figure 18: Example Code

String i = “”;

As shown in the diagram above, if the combo box chosen is “Sea” then the Sea file will be selected, otherwise the Jungle file will be selected. Next BufferedReader is used to read data from any character stream while FileReader is used to read a file from a disk drive (ssd, hard disk). Declaration String l reads the line of the file. While loop until the while is null (empty). i (empty string) = empty string + l + System.lineSeparator. System.lineSeparator is basically \r\n. l readlines in the file used. Outside the while loop newC string replaces all old details into the mod details. Using FileWriter to write the newC into the file chosen earlier. Lastly, closing the FileWriter(file) and earlier BufferedReader(file). r and w respectively closed.

### Search file

Table

Description automatically generated

Figure 19: Example Code

As shown in the diagram above, after the file is chosen using a combo box. The input text field will then be needed to submit a roomid. Then, a scanner class is being used in this scenario. Perform a while loop to find if there is another line in the input of the scanner given (file used). In the while loop, string line is reading the lines of the file used and trimming the empty spaces (leading and trailing spaces) for better efficiency. If the line contains the roomid inputted, thus use the horizontal data found on that line. Create an array for the data found in the line and by splitting it using the delimiter (“,”). Then, assign a string variable for all the records based on the index found on that line. The record found based on the indexes can then be displayed using setText().

### Delete file

**Graphical user interface, text, application, email

Description automatically generated**

Figure 20: Example Code

As shown in the diagram above, two files are being used. One is the original while another is a tempt file. The original file being used to Read while the tempt file being used to write. While loop the record in the original file unless null (empty). If record contains ID continue. Write the other records into the tempt file (not the searched ID record). Replace the search ID record with a roomID which haven’t been booked into the tempt file. Close both the files. Delete the original file. Lastly, rename the tempt file to the original file name. In conclusion, the ID that was inputted was replaced to an non booked room pattern.

### Update file



Graphical user interface, text

Description automatically generated

Figure 21: Example Code

As shown in the diagram above, BufferedReader and FileReader to be used as according to the file. String l to readlines in the file. While loop the contents in the file while not null (empty). String newC to replace the contents of the old string details with the modified string mod. Use FileWriter to write the new String mod to the chosen file. Close both the r value and the w value. Display the ModifySea() again to show the implemented changes.

# Sample Output Screens

Graphical user interface

Description automatically generated

Figure 22: Login Page

In this page, the user is supposed to login. The staff is required to enter the correct username and password (hard coded) to avoid intrusion of the system.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 23: Correct Details

Details are correct

Graphical user interface, text, application

Description automatically generated

Figure 24: Wrong Details

Either username or password is wrong

Graphical user interface, diagram

Description automatically generated

Figure 25: Main Page

Main Page (Log out button goes back to login page)

Graphical user interface, website

Description automatically generated

Figure 26: Delete Booking

Delete Booking Page (can choose between jungle or sea). Delete button deletes the record in the txt file. Back button goes back to home page

A picture containing graphical user interface

Description automatically generated

Figure 27: View Booking

View Booking Page (can choose between Jungle or sea). Back button goes back to home page

Graphical user interface, text, application

Description automatically generated

Figure 28: View Booking Jungle

If view button is clicked (jungle).

Graphical user interface, text, application

Description automatically generated

Figure 29: View Booking Sea

If view button is clicked (sea).

Graphical user interface

Description automatically generated

Figure 30: Search Booking

Search Booking Page (can choose between jungle and sea). Back button goes back to login page

Graphical user interface, application

Description automatically generated

Figure 31: Random roomID

If random roomID inputted.

Graphical user interface

Description automatically generated

Figure 32: Search Jungle Booking

Example using a jungle roomID.

Graphical user interface

Description automatically generated

Figure 33: Search Sea Booking

Example using SeaID.

Chart, treemap chart

Description automatically generated

Figure 34: Booking

Booking Page (Back button goes back to main page).

Chart

Description automatically generated with medium confidence

Figure 35: Jungle RoomID

Based on View. The roomID changes (Jungle)

Chart, treemap chart

Description automatically generated

Figure 36: Sea RoomID

Based on View. The roomID changes (Sea)

Shape

Description automatically generated

Figure 37: Number of Days

Number of days is minimum 1 or maximum 5 only.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 38: Username too long

If username more than 12 in length ().

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 39: Username too long

If username less than 4 in length ().

Graphical user interface, text, application

Description automatically generated

Figure 40: Invalid IC number

If IC-number is not using 12 integers in length ().

Graphical user interface, text, application

Description automatically generated

Figure 41: Invalid phone number

If phone number is not 10 digits.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 42: Invalid Email

If email doesn’t contain “@” and “.com”.

Graphical user interface

Description automatically generated with medium confidence

Figure 43: Modify Sea Booking

Modify Sea View (Back button goes back to main page).

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 44: Modify Username too long

Modified username too long

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 45: Modify Username too short

Modified username too short

Graphical user interface, text, application

Description automatically generated

Figure 46: Modify IC number invalid

IC number not in 12 digit format

Table

Description automatically generated

Figure 47: Modify Sea View

Modifying process of roomid 2006

Graphical user interface

Description automatically generated

Figure 48: Name and IC modified

Yayato changed to Gaurav

IC number 020130.. changed to given input

A picture containing graphical user interface

Description automatically generated

Figure 49: Payment Sea

Payment Page (back button goes back to main page)

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 50: Random roomID inserted

Random roomID inputted

Graphical user interface

Description automatically generated with medium confidence

Figure 51: Payment Sea (correct ID)

Roomid 2010 is selected (as shown the deposit is still rm100)

Graphical user interface

Description automatically generated with medium confidence

Figure 52: Receipt for Sea

Receipt page (shows the roomID receipt of number 2010)

Graphical user interface, text

Description automatically generated

Figure 53: Booking Details Modified

Room 2010 payment changed from rm100 to rm1185

Graphical user interface, table

Description automatically generated

Figure 54: Modify Jungle

Same logic goes for modify jungle (back button goes to home page)

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 55: Modified Username too long

Modified username too long

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 56: Modified Username too short

Modified username too short

Graphical user interface, text, application

Description automatically generated

Figure 57: Invalid Modified IC Number

IC number must be 12 digits

Graphical user interface

Description automatically generated

Figure 58: Modify Jungle (ID 1009)

Modifiying details in room 1009

Graphical user interface, table

Description automatically generated

Figure 59: Name and IC modified

As shown above, the room name and IC is successfully modified.

johnny to johnson

020130.. to 050120301779

Graphical user interface

Description automatically generated with medium confidence

Figure 60: Payment Jungle

Payment jungle page (back button to go back to home page)

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 61: Random roomID Inputted

If random roomID is inputted

Graphical user interface

Description automatically generated

Figure 62: Payment Jungle roomID 1009

Process of payment for roomID 1009

Graphical user interface

Description automatically generated

Figure 63: Receipt for room 1009

Receipt for room 1009

Graphical user interface, text

Description automatically generated

Figure 64: Payment modified part 1

Graphical user interface, text, application, email

Description automatically generated

Figure 65: Payment modified part 2

Payment changed for room 1009 from rm100 to rm 1185.

# Additional Features

## Array

**Text

Description automatically generated**

Figure 66: Example Code

As shown in the diagram above, an array method is used to print out all the items inside the file into an array. In this example. String [] array will create an array to put the items from the file into an array. The array is then split by using the delimiter (“,”). Thus, then a for loop is used to use a variable (string “a” to go through all the items inside the array). If string “a” reads a line with the word “Available”, the String “a” will immediately display the roomid of the line with the string “available” into the combo box “rooomcombo”.

## JCalendar

A screenshot of a computer

Description automatically generated with low confidence

Figure 67: Example Code

As shown from the diagram above, JCalendar is being used

Text, letter

Description automatically generated

Figure 68: Example Use of sdf

In the code section , I am just implementing SimpleDataFormat java function with year-month-day format. Thus, by using this method, I will also implement the sdf format into the CheckInDates text field. Therefore, when the user click on the CheckInDates text field, the user can see the work of JCalendar being shown onto the screen as shown on the first diagram. After the use of JCalendar, the input of the CheckInDates will be stored in a text file once the process is complete.

## JTable

Table

Description automatically generated

Figure 69: Example Code

jTable to modify the name and ic number of the jungleView.txt file.

Text

Description automatically generated

Figure 70: Example use of JTable

The diagram above shows the work of jTable. All the string variables shown above, is created to allow the Roomies table to obtain information from the txt file.

Chart, scatter chart

Description automatically generated

Figure 71: Overiding process

The modified input will then be used to overwrite the old information by stating the old information (String details) and modified information (String mod). The process involved in overwriting the file information is the .replaceAll(details,mod) java function.

Graphical user interface, text, application

Description automatically generated

Figure 72: Use of pointer in JTable

As shown in the diagram above, first I declare the text file used. Then declare string l. r variable to BufferedReader and FileReader the file. Initialiation of int count = 0. While loop as long as r is reading the lines of the of as long the file is not null (empty). Array i uses the delimiter (“,”) to split the data in the file and to put it into an array i. String variable at the i [index of file]. After obtaining the data, create a new object model to send the data in the file to the table Roomies. Add row to the jTable according to the string variables mentioned earlier.

# Assumptions

1. I assume my system works fine and the user is able to login by providing correct login id and password
2. Most input text fields are validated; thus, the system will work fine if an input causes an exception
3. Payment will be done once only for each customer
4. Modification of booking details is only for name and IC number of the customer.
5. Payment will be (should be done) straight after booking.
6. Rooms can only be used again after deletion/completion of “rented status” to “available status”
7. Only after payment, the customer can see their receipt.
8. RoomID can only be searched one at a time
9. Only one staff can access the system
10. RM100 will be taken as deposit payment before pull payment
11. The staff can help book customer bookings, delete, view and search bookings. Lastly, the staff can do modification of bookings and payment for customer.

# References

*Java FileReader class - javatpoint*. www.javatpoint.com. (n.d.). Retrieved April 9, 2022, from https://www.javatpoint.com/java-filereader-class#:~:text=Java%20FileReader%20class%20is%20used,for%20file%20handling%20in%20java

*Java FileWriter class*. Programiz. (n.d.). Retrieved April 9, 2022, from https://www.programiz.com/java-programming/filewriter#:~:text=The%20FileWriter%20class%20of%20the,It%20extends%20the%20OutputStreamWriter%20class

Kommadi, +B. (2021, October 12). *Java.io.ioexception - how to solve Java IOEXCEPTION*. Examples Java Code Geeks. Retrieved April 9, 2022, from https://examples.javacodegeeks.com/java-io-ioexception/#:~:text=java.-,io.,subclass%20based%20on%20the%20context

SimpleDateFormat (Java Platform SE 7 ). (2020, June 24). Retrieved April 9, 2022, from https://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html#:~:text=SimpleDateFormat%20is%20a%20concrete%20class,patterns%20for%20date%2Dtime%20formatting

LocalDate (Java Platform SE 8 ). (2022, January 5). Retrieved April 9, 2022, from https://docs.oracle.com/javase/8/docs/api/java/time/LocalDate.html#:~:text=LocalDate%20is%20an%20immutable%20date,be%20stored%20in%20a%20LocalDate%20

*Java bufferedwriter class - javatpoint*. www.javatpoint.com. (n.d.). Retrieved April 9, 2022, from https://www.javatpoint.com/java-bufferedwriter-class#:~:text=Java%20BufferedWriter%20class%20is%20used,arrays%2C%20characters%2C%20and%20strings

Uses of class java.io.file (java platform SE 8 ). (2022, January 5). Retrieved April 9, 2022, from https://docs.oracle.com/javase/8/docs/api/java/io/class-use/File.html#:~:text=File,-Packages%20that%20use&text=Contains%20all%20of%20the%20classes,for%20painting%20graphics%20and%20images.&text=Provides%20for%20system%20input%20and,serialization%20and%20the%20file%20system

Maneas, S.-E. (2021, October 12). *Java.io.filenotfoundexception - how to solve file not found exception (with video)*. Examples Java Code Geeks. Retrieved April 9, 2022, from https://examples.javacodegeeks.com/java-io-filenotfoundexception-how-to-solve-file-not-found-exception/#:~:text=solve%20the%20java.-,io.,denoted%20by%20a%20specified%20pathname

Joshi, R. (2015, December 17). *Java.util.logging example*. Examples Java Code Geeks. Retrieved April 9, 2022, from https://examples.javacodegeeks.com/core-java/util/logging/java-util-logging-example/#:~:text=Logging%20is%20used%20to%20store,util

*Logger in java - java logging example*. JournalDev. (2020, December 10). Retrieved April 9, 2022, from https://www.journaldev.com/977/logger-in-java-logging-example#:~:text=and%20logging%20configurations.-,Java%20Logger,messages%20in%20java%20logging%20API

Windowevent (java platform SE 7 ). (2020, June 24). Retrieved April 9, 2022, from https://docs.oracle.com/javase/7/docs/api/java/awt/event/WindowEvent.html#:~:text=Class%20WindowEvent&text=A%20low%2Dlevel%20event%20that,or%20out%20of%20the%20Window

Defaulttablemodel (java platform SE 7 ). (2020, June 24). Retrieved April 9, 2022, from https://docs.oracle.com/javase/7/docs/api/javax/swing/table/DefaultTableModel.html#:~:text=Class%20DefaultTableModel&text=This%20is%20an%20implementation%20of,a%20column%20class%20of%20Object%20

*Java Scanner class*. Programiz. (n.d.). Retrieved April 9, 2022, from https://www.programiz.com/java-programming/scanner#:~:text=The%20Scanner%20class%20of%20the,%2C%20users%2C%20files%2C%20etc