COSC1295 Advanced Programming

Assignment 2 - Semester 1, 2019

Submission due date: 23:59pm Tuesday 4th Jun 2019

IMPORTANT: For more details about submission and demo, please read the section Submission Details and Demos at the end of this assignment specification.

Academic Integrity

The submitted assignment must be your own work. For more information, please visit http://www.rmit.edu.au/academicintegrity.

Plagiarism is treated very seriously at RMIT. Plagiarism includes copying code directly from other students, internet or other resources without proper reference. Sometimes, students study and work on assignments together and submit similar files which may be regarded as plagiarism. Please note that you should always create your own assignment work even if you have very similar ideas.

Plagiarism-detection tools will be used for all submissions. Penalties may be applied in cases of plagiarism.

Overview

You are required to use Java SE 8.0 and JavaFX to develop a Graphical User Interface (GUI) for the ThriftyRent rental vehicle management program created in Assignment 1.

This assignment is designed to help you:

1. Enhance your ability to build a Graphical User Interface using JavaFX
2. Practise implementation of various GUI event handlers and handling exceptions
3. Read from and write to a database using Java JDBC technology
4. Incorporate text file handling in your program to import and export data

NOTE:

- This assignment is of a size and scope that can be done as an individual assignment. Group work is not allowed.
- A more detailed marking rubric and demo schedule will be provided closer to submission.

General Implementation Requirements

- All information displayed to the user and all user interactions should be done via the GUI. There must be no Console input and output.
- You are free to create your own GUI layouts as long as your layouts are clear and meet the requirements shown in the following sections. Marks might be deducted for very poor GUI design.
- Any user inputs via the GUI should be validated.
- You must not use 3rd-party GUI components which are not built by you.

- Marks will be allocated to proper documentation and coding layout and style. Your coding style should be consistent with standard coding conventions shown in the General Implementation Requirements section of Assignment 1.
- Your programs will be marked with Java SE 8.0. Make sure you test your program with this setting before your final submission.

Task Specifications

NOTE: Carefully read the following requirements. In addition, regularly follow the Canvas assignment discussion board for assignment related clarifications and discussion.

Packages and Organisation of Code

You must use the following packages to separate your code into sets of related classes and interfaces:

- ***view***: contains all your GUI classes.
- ***controller***: contains all your listener classes.
- ***model***: contains the main business logic of your application, all your classes to store and process data (Vehicle, Car, Van...), all exception classes and all database and file handling classes.
  You can use sub-packages inside the packages shown above.

Data Generation

You are required to generate data for 15 vehicles, including 10 cars and 5 vans. Each vehicle needs to have 2 or 3 completed rental records with random information such as customer ids, rent, return dates, rental fees and late fees that you are free to choose as long as they are reasonable.

For this assignment, each vehicle will have a corresponding image. Each vehicle image must have a moderate size (from 100KB to 250KB). Click here for an example of such an image. You must keep all images in a folder named **images** which is a direct sub-folder of your assignment project. If a vehicle has no corresponding image, a default image with the phrase "No Image Available" should be used (please see this example).

Exception Handlings

Creating Custom Exception Classes

In assignment 1, methods such as rent(...), returnvehicle(...), performMaintenance(), completeMaintenance(..) have boolean as return types to indicate whether the corresponding operation is executed successfully or not. In assignment 2, you are required to modify all those methods so that all of them will return void instead of boolean and will throw custom exceptions. You are required to create custom exceptions such as RentException, ReturnException, MaintenanceException, InvalidIdException…. to represent various exceptional cases that can occur when those methods are executed. All of those custom exception types should allow appropriate error messages to be specified when exception objects are created.

Generating and Propagating Exception Objects

To use your custom exceptions effectively, you will need to look for areas in your program where you need to handle various cases such as when there is an invalid vehicle id, or when a vehicle cannot be rented or returned, or when there is an invalid user input… Then you should generate appropriate exceptions and then throw those exceptions back to the appropriate class to be handled.

Handling Exceptions

Those various exception objects should then be allowed to propagate back to the appropriate class (i.e. those exception should not be caught locally within the class that generates the exception).

All exceptions will need to be caught and handled in a appropriate manner by displaying a dialog box via the GUI to the user, showing the error message contained within the various exception objects that have been propagated up from the relevant method call.

Using Database For Data Persistence

Every time your program is executed and terminated, data will be read from and save to a database. For this assignment, you should use an embedded HSQLDB (shown in the lecture) or SQLite database. You must keep all database files in a folder named **database** which is a direct sub-folder of your assignment project.

You must create at least two tables in your database. One table named **VEHICLES** to store details of rental vehicles, and another table named **RENTAL_RECORDS** to store details of rental records. In each table, you must store details of your vehicles and records in separate columns in those tables with appropriate data types, rather than just storing the string return by calling toString methods from those vehicle and rental record objects.

NOTE: No need to store images directly in the database. Only image file names should be stored as text in the database. All images will still be kept in the **images** folder which is a direct sub-folder of your assignment project.

For instructions about using HSQLDB with Eclipse, please visit this page
https://tinyurl.com/y7hty8tg

Graphical User Interface (GUI)

All user interaction with the ThriftyRent system will be done via the GUI. Users should be able to click buttons, select menu items, choose list items from drop-down lists to perform all functionalities described in Assignment 1 such as add, rent and return a vehicle as well as perform and complete vehicle maintenance.

Main Program Window

- This is the first window a user will see when running your program.
- This window should contain a menu bar from which users can execute the main functionalities of your program, such as import, export data, quit the program and other main functionalities described below. When a button is clicked or a menu item is selected, new windows or dialog boxes should be displayed to allow users to perform corresponding functionalities.
- The centre area of this window will contain a scrollable list of rental vehicles managed by your program. Each list item displays information about a rental vehicle using various JavaFX User Interface Controls

such as ImageView, Label, Buttons…, rather than just a text area showing the output of getDetails() method. Click here for an example of how the list of rental vehicles should look like.

  - In this list, each list item provides an overview of a rental vehicle with an image of that vehicle, vehicle type, status... and a button that users can click on to go to the detail view of that vehicle to perform more functionalities related to that vehicle (more details are in the Vehicle Detail Window section shown below)

- Search and filtering capabilities: your main window should also contain 4 drop-down lists, allowing users to select list items from those lists to filter the main vehicle list by type (car, van) or by number of seats (4, 7 or 15 seats) or by status (Available, Rented, Maintenance) or by make (Honda, Toyota…)

Vehicle Detail Window

- When the user selects a vehicle in the vehicle list shown in the Main Program Window, your program must display this Vehicle Details Window to allow the user to see all details of the selected vehicle, including a larger image of that vehicle.
- This window must have a scrollable list to display complete rental records of that vehicle.
- This window must have buttons which allow users to perform various activities related to this vehicle such as rent, return, maintenance and complete maintenance.
- This window must still keep the main menu bar as mentioned in the Main Program Window section shown above, allowing users to perform common activities such as import, export data, quit the program and other main functionalities that you see reasonable.
- Users should be provided with a way to return to the Main Program Window from this Detail Window

Other GUI Requirements

- You are encouraged to explore and use various JavaFX User Interface (UI) controls to implement your graphical interfaces for functionalities such as add vehicle, return vehicle, maintenance and complete maintenance. My suggestions for such UI controls are the Dialog class and its subclasses such as TextInputDialog, ChoiceDialog and Alert classes in the javafx.scene.control package.
- All user inputs via the GUI must be validated.
- All error messages and messages from Exception objects should be displayed in the GUI using JavaFX Alert classes in the javafx.scene.control package. Do not output any error message to the Console.
- Important functionalities such as Save to Database, Import, Export, Quit should always be available in the main menu bar your program.

Using Text Files for Exporting and Importing Data

Your ThriftyRent GUI program must contain Export Data and Import Data menu items in the main menu bar to allow users to:

- export all vehicle data of your program and corresponding rental records to a single text file.
- import all vehicle data and corresponding rental records from a given text file.

Those text files must follow a pre-defined format as shown here.

When the user selects the Export Data menu item, your program should export all data to *a single text file* named **export_data.txt**, which will be saved in a folder chosen by the user. (Hint: this feature can be implemented in your GUI program by using the DirectoryChooser class in the javafx.stage package).

When the user selects the Import Data menu item, your GUI program must display a FileChooser window (hint: FileChooser class is located in the javafx.stage package) to allow the user to select a text file also in the format as shown above. Your program must be able to read rental vehicle and rental record data from that text file. If a vehicle has no corresponding image, a default image with the phrase "No Image Available" should be used.

NOTE: You don't need to export images. All images will still be kept in the **images** folder in your assignment project. Only image file names should be exported as shown in the format shown above.

Other Requirements

- Although you are not required to use more than one class per task, you are required to modularise classes properly. No method should be longer than 50 lines.
- You should aim to provide high cohesion and low coupling.
- You should aim for maximum encapsulation and information hiding.
- Your coding style should be consistent with Java coding conventions (http://www.oracle.com/technetwork/java/codeconventions-150003.pdf)
- You should comment important sections of your code remembering that clear and readily comprehensible code is preferable to a comment.
- Your programs will be marked with Java SE 8.0. Make sure you test your programs with this setting before you make the submission.

Submission Details and Demos

You must compress all files in your project (including source code files and other resources such as database files and images) into one zip file and submit only that zip file (no RAR or 7-Zip).

You must submit the zip file of your project in the Assignment 2 page on Canvas.

You can submit your assignment as many times as you want prior to the due date. The latest submission will be graded.

Submission due date: 23:59pm Tuesday 4th Jun 2019

IMPORTANT: you are required to **come to a demo session** starting in Week 13 (the duration of each demo session is 10 minutes). During the demo, you will download your final submission from Canvas and run your program on your computer and demonstrate the main features of your program to a tutor.

NO MARK will be given if you fail to come to a demo session.

Assignment rubric and demo session schedule will be announced on Canvas closer to the submission date.

Please do NOT submit compiled files (*.class files), or you will get zero. You will get zero if the submitted code cannot be compiled.

THE END