COSC1295 Advanced Programming

Assignment 2 - Semester 2, 2018

Submission due date: 11:59 PM on Sunday 14th of October (end of Week 12)

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 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 FlexiRent rental property management program created in Assignment 1.

This assignment is designed to help you:

1. Develop your skills of using various Java collection classes and exception handling
2. Enhance your ability to build a Graphical User Interface using JavaFX
3. Practise implementation of various GUI event handlers
4. Read from and write to a database using Java JDBC technology
5. 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 will be provided closer to submission.
- There will be **no demo in Week 12**, so you have maximum time available to complete this assignment. Check the due date of this Assignment carefully to avoid late penalty.

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 programs with this setting before you make the 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 controller classes
- ***model***: contains all your classes to store and process data (RentalProperty, Apartment, PremiumSuite...) 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 property, including 10 Apartments (with random number of rooms and locations of your choices) and 5 Premium Suites. Each property needs to have 2 or 3 completed rental records with random customer ids.

For this assignment, each property will have a corresponding image and a long description which is from 80 to 100 characters.

Each property image should 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 property has no corresponding image, a default image with the phrase "No Image Available" should be used.

Collections and Exception Handlings

Collections

In this assignment, you are required to used various collections in the Java Collection Framework such as ArrayList, Set and HashMap to store data instead of using fixed-length array as in assignment 1.

Creating Custom Exception Classes

In assignment 1, methods such as rent(...), returnProperty(...), performMaintenance(), completeMaintenance(..) have boolean as return type 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. As a consequence, you are required to create custom exceptions such as RentException, ReturnException, 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 property id, or when a property 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 generated the exception).

All exceptions will need to be caught and handled in a appropriate manner by displaying a message via the GUI to the user 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 opened 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 **RENTAL_PROPERTY** to store details of rental properties, and another table named **RENTAL_RECORD** to store details of rental records. In each table, you must store details of your properties and records in separate columns in those tables with appropriate data types, rather than just storing the whole toString representations.

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.

Graphical User Interface (GUI)

All user interaction with the Flexirent system will be done via the GUI. Users should be able to click buttons, select menu items, choose options from combo boxes to perform all functionalities described in Assignment 1 such as add, rent and return a property as well as perform and complete property maintenance.

Main Program Window

- This is the first window users will see when running your program.
- This window should contain a menu bar or menu pane 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 users click a button or select a menu item, new windows can be opened to allow users to perform corresponding functionalities.

- The centre area of this window will contain a scrollable list of rental properties managed by your program. Each list item should be implemented by 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 a suggestion of how the list should look like.
    - In this list, each list item provides an overview of a rental property with an image of that property, property type, status... and a button that users can click on to go to the detail view of that property to perform more functionalities related to that property (described below)
- Search and filtering capabilities: your main window should also contain various combo boxes, allowing users to filter the property list by type (Apartment, Premium Suite) or by number of bedrooms (1, 2, or 3 bedrooms) or by status (Available, Rented, Maintenance) or by suburb.

Property Detail Window

- When the user selects a property in the property list of the Main Program Window, your program should display this Property Details Window to allow users to see all details of the property which has been selected, including the property image and long description.
- This window must have a scrollable list to display complete rental records of that property.
- This window must have buttons which allow users to perform various activities related to this property such as rent, return, maintenance and complete maintenance.
- This window must still keep the main menu bar or menu pane as mentioned in the Main Program Window description shown above, allowing users to perform common activities such 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 other JavaFX User Interface (UI) controls to implement your graphical interfaces for functionalities such as add property, return property, 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 your program.
- ~~There will be GUI Demo in the tutorial in week 11 or week 12 (details will be announced later)~~
- There will be **no demo in week 12**, so you have maximum time available to complete this assignment. Check the due date of this Assignment carefully to avoid late penalty.

Using Text Files for Exporting and Importing Data

Your FlexiRent GUI program must allow users to export and import all rental property and rental record data to and from text files. One way to implement this feature is by adding Export Data and Import Data menu items in the main menu bar or menu pane of your program GUI.

When working with data stored in text files, you must follow the format as shown here.

When the user chooses to export all data, you should export all data in your program 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 importing data, 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 property and rental record data from that text file. If a property 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 in the format shown above.

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

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

You must submit a zip file of your project via Canvas.

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

This assignment has a lab demo component in Week 11 or 12. Details will be posted later on the course announcement.

Submission due date: by 11:59 PM on Sunday 14th of October (end of Week 12)

You should compress all your source code into one zip file and submit only that zip file (no RAR or 7-Zip).

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