COMP 2613 Assignment 2

The assignment will continue on from assignment 1, and also building on the labs and reinforce the concepts and features of the java framework we’ve learned in weeks seven through ten. You’ll be adding data persistence and a graphical user interface for the Games Information System (GIS).

At the core of this application is your assignment 1 and labs. The assignment is a consolidation and extension of the requirements for the labs.

## Requirements

The design of GIS must follow good object-oriented principles and practices.

Your code must compile and the jar file must run. Compile-time warnings are considered errors and must be eliminated from your code by using appropriate annotation tags.

The main class must be named Gis, and the jar file containing your runnable code must be named <student#>Gis.jar, i.e. A00123456Gis.jar.

As in assignment 1, all activity must be logged to a text file named <student#>Gis.log, i.e. A00123456Gis.log.

Exceptions must be handled such that no stack traces are displayed in the console, but as mentioned above, a message will be logged explaining the cause of the error.

In the second half of COMP 2613 the topics are database, multithreading, graphical user, database, model-view-controller and other design patterns, and network programming. This assignment will touch on many of these.

Remove the printing of the reports to the console and to a text report that we had in assignment 1.

Similar to your labs, the first time GIS is run, the data will be read from personas.dat, players.dat, games.dat, and scores.dat. The data will be stored into four separate tables in a local (derby) database – you can create additional tables if you need them. If collections are used, they are only used to temporarily store datasets. A separate Database class is used to manage the connection to the database. For each of the tables a separate DAO class is required; the DAO classes will contain the table creation, and create, read, update, and delete methods. To re-test the data loading functionality, you will delete the external database files and restart your application.

The application contains a graphical user interface, which will be displayed when the user runs GIS. The UI will always be displayed even if there are errors loading the data. Errors will be written to the application log files and will be messaged to the user in the form of a simple dialog box.

The follow menu must be implemented:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Notes | |
| File | JMenu | Contains the following item |
| Quit | MenuItem | Quits the application |
| Lists | JMenu | Contains the following three items |
| Players | MenuItem | Opens a dialog which contains a list filled with the players first and last names |
| Personas | MenuItem | Opens a dialog which contains a list filled with the gamertags |
| Scores | MenuItem | Opens a dialog which contains a list filled with the scores data |
| Reports | JMenu | Contains the following five items |
| Total | JMenuItem | Prints the total count of each game played in a simple JOptionPane dialog box |
| Descending | JCheckBox-MenuItem | If selected, will indicate the values will be sorted in a descending order; ascending otherwise |
| By Game | JMenuItem | Sorts the report by game name and displays the results in a JTextArea or JList within a dialog box |
| By Count | JMenuItem | Sorts the report by games played and displays the results in a JTextArea or JList within a dialog box |
| Gamertag | JMenuItem | Opens a JOptionPane that allows the user to enter a Gamertag. This Gamertag is used to filter the report output. If an invalid Gamertag is entered then a warning message is displayed. If no Gamertag is entered then the filter is removed. |
| Help | JMenu | Contains the following item |
| About | MenuItem | Displays an information dialog about the application |

All menus and menu items must be accessible via a mnemonic character, i,e, the user can press Alt-F Alt-Q to quite the application, You man need to open the extended properties in WindowBuilder to access the mnemonic setting.

The ‘About’ item must also be directly accessible via an accelerator key, F1. By pressing F1, the About dialog box will be displayed.

As stated above, the Personas menu item will open a dialog that contains a list filled with the Personas. If a row in the list is double-clicked then a new dialog will open with the details for that persona. This dialog will look similar to the dialog that was created in Lab 7. The user can change the first name, last name, email address, or gamertag and update the database if ‘OK’ is pressed. If ‘Cancel’ is pressed, the changes aren’t saved. Either ‘OK’ or ‘Cancel will close the dialog. If any changes are saved to the database and the GIS application is exited, and then restarted, the user will be able to navigate to the item and see that the values have been updated.

You don’t need to implement a details dialog for either the players, games or scores lists, but you will still need to display the players, or scores in a dialog accessed by the ‘Players’ and ‘Scores’ menu items.

From the requirements listed above, you can see that only create, read, and update functionality has been used from the DAOs; you’ll still need to implement delete functionality, though this feature will only be graded based on code inspection. Although create is implemented, you don’t need to create any new records over and above what already exists in the text data files.

If any of these requirements are unclear, make sure you ask for clarification.

## Submission Checklist

I have:

☐ Named my main class Gis

☐ Met all the functional requirements

☐ Followed the java coding guidelines

☐ Run “Source > Format” on my project

☐ Used a file template to add my name & student number to all source files

☐ Used packages; the root package is my student number,   
ex. package a00123456.…;

☐ Used great object-oriented design

☐ Created a runnable Jar file

☐ Included all source code & required resources

☐ Zipped up all my files into a single file named <your student number>.zip,   
ex. A00123456\_assignment2.zip

☐ Submitted my lab before the due date & time

For EACH requirement not followed in the checklist, you’ll lose 1 mark. OUCH!

## Grading

The assignment will be marked out of 10