Assignment 2 Milestone 2 – Database Manager

Dean Zeller, Dr. Krishna Kambhampaty
CS201
Fall 2022

Objective The student will implement a Java program to manage the database created in milestone 1 through a text menu interface.

Background – Interface

All programs need some sort of interface to communicate with the user. For Milestone 1, the test items were entered manually through the Java tester. This is fine for an early testing of the program, but a programmer should always keep usability in mind. The text-menu interface, while simplistic, is an effective way to create an interface with the program. Naturally, there are far more efficient user interfaces, but this is an excellent starting point.

Basic Requirements

1. Read over the interface example, to understand what this assignment entails.
2. Create a java file named `ComicCollectionInterface.java`, named appropriately for your selected item.
3. Instantiate the database from Milestone 1 within the interface file. This will serve as the collection the interface will modify.
4. Create the text interface to do the following commands:
    a. A – Add new: Accept attribute values from the user to add a new item to the collection.
    b. D – Delete: Display a numbered list of the items, and allow the user to select one for deletion.
    c. P – Print collection: Print the entire collection in an organized table.
    d. F – Filter collection: Filter the collection for a specific target, showing only those items that match the filter parameter.
    e. S – Sort collection: Sort the collection based on one of the fields, specified by the user. Students may use any sorting algorithm (bubble, insertion, etc…) or tool available to sort the items (Java `sort` command).
    f. Q – Quit program: Exit the program.
5. Set up the necessary methods and interface to call the appropriate method when requested by the user. Each letter should execute the appropriate method.
6. Complete the method for `printCollectionTable`. It should print the collection in a neatly formatted table, similar to the example below. Modify the output so it has a header and nicely formatted output in a table.

```
Title                #   Writer                Artist           Condition     Value
-------------------------------------------------------------------------------------------
Great Lakes Avengers   1  Zac Gorman            Will Robson      VF            $4.50
Jessica Jones         10  Brian Michael Bendis  Michael Gaydos   NM            $6.00
Avengers              34  Mark Waid             Mike Del Mundo   G             $0.50
```

7. Complete the method for `printCollectionList`. It should print the collection as a vertical list instead of a horizontal table. Use print formatting to align all fields nicely.
8. Submit your work to the marmoset server.
9. Show your work to the instructor in class.

TextInterface.java

```
import java.util.ArrayList;
import java.util.Scanner;

public class TextInterface
{
    private ArrayList<String> acceptableChoices;
    private String choice;
    private Scanner input;

    public TextInterface()
    {
        this.input = new Scanner(System.in);
        this.acceptableChoices = new ArrayList<String>();
        acceptableChoices.add("A");
        acceptableChoices.add("T");
        acceptableChoices.add("S");
        acceptableChoices.add("Q");
        this.choice = "Z";
    }

    public void displayMenu()
    {
        System.out.println("The MENU...");
        System.out.println("A -- Choose A");
        System.out.println("T -- Choose T");
        System.out.println("S -- Choose S");
        System.out.println("Q -- Quit");
        System.out.println("Anything else -- error");
    }

    public void userChoiceInput()
    {
        System.out.print("Enter choice => ");
        String response = this.input.nextLine();
        while (true)
        {
            if (this.acceptableChoices.contains(response))
            {
                System.out.println("Thank you for your selection.");
                this.choice = response;
                break;
            }
            System.out.println("Sorry, you must enter an acceptable selection.");
            System.out.println("Selections available:  "+this.acceptableChoices);
            System.out.print("Enter choice => ");
            response = this.input.nextLine();
        }
    }

    public void executeUserChoice()
    {
        if (choice.equals("A"))
            this.chooseA();
        else if (choice.equals("T"))
            this.chooseT();
        else if (choice.equals("S"))
            this.chooseS();
        else if (choice.equals("Q"))
```

```
            this.chooseQ();
        else
            System.out.println("Error in User choice " + this.choice);
    }

    public void chooseA()
    {
        System.out.println("Hey!  You chose A!  An A for aeffort!");
    }

    public void chooseT()
    {
        System.out.println("Hey!  It's T-Time!");
    }

    public void chooseS()
    {
        System.out.println("Hey!  Look at that S-car go!");
    }

    public void chooseQ()
    {
        System.out.println("Quitting time, I guess.");
    }

    public void repeatUntilQuit()
    {
        this.choice = "Z";
        while(!this.choice.equals("Q"))
        {
            this.displayMenu();
            this.userChoiceInput();
            this.executeUserChoice();
        }
        System.out.println("Exiting program.  Goodbye!");
    }
}
```

TextInterfaceTester.java

```
public class TextInterfaceTester
{
    public static void main(String args[])
    {
        TextInterface ti = new TextInterface();
        ti.repeatUntilQuit();
    }
}
```