GUI Applications with I/O
ISTE-121 – Homework 01

Overview

This homework is a continuation of your lab 1. You need to complete lab 1, make a copy of it, and then make the modifications required for homework 1.

You may not have received all the information to do this assignment by the time it is being handed out. The information from the second class of the course, and the lab will help you with this assignment. Some solutions to this are given in the lab, so do what you can until the lab day. Then make sure all your questions are answered for this homework.

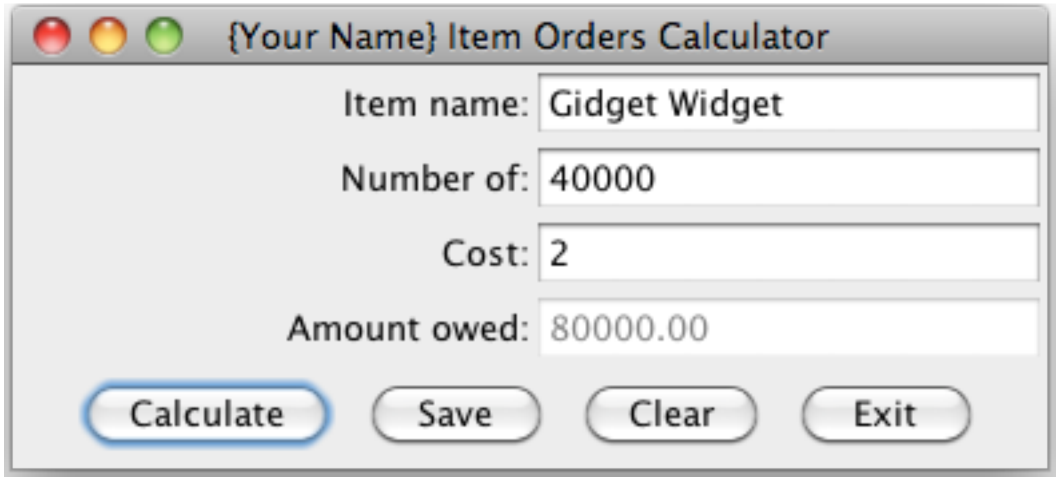
The main class name for homework 1 is OrderSystem. When you get the GUI to look *similar* to the display, that is close enough for this homework.

Part 1: GUI Layout (start with Lab 1 code)

Part 1 is from lab 1: Create a GUI that contains right aligned labels for text fields as shown. Label, field and button object names should be self-documenting representations of the GUI components. Place all of the components in a JPanel, then add the JPanel to the JFrame. This is a common practice, easily allowing moving of the JPanel to another part of the JFrame, should relocation be needed.

Add buttons to control the actions that will take place on the screen. Your screen should now look similar to the screen shot below.

- Have the “Amount owed” field to not allow input.
- Format the amount owed result to have two decimal places. See String class in the Javadocs how to format a number.


{Your Name} Item Orders Calculator
Item name: Gidget Widget
Number of: 40000
Cost: 2
Amount owed: 80000.00

Adding controls for the buttons should start with the easier ones first.

Exit	Exit the program
Clear	Clears the text fields. Can setText() to null or “” (blank).
Calculate	Multiply the “Number of” by the cost per item, place result in the Amount owed field. Formatted to two decimal places. Assume all valid numbers are entered.
Save	Opens a text file for writing, use file name **121Lab1.csv**. Write in comma-separated format, the item’s name, the number of items, cost per item and the calculated amount owned. Each time the user clicks Save, first execute the Calculate code. The calculation code should only appear once in your program.

Part 2: Adding more functionality

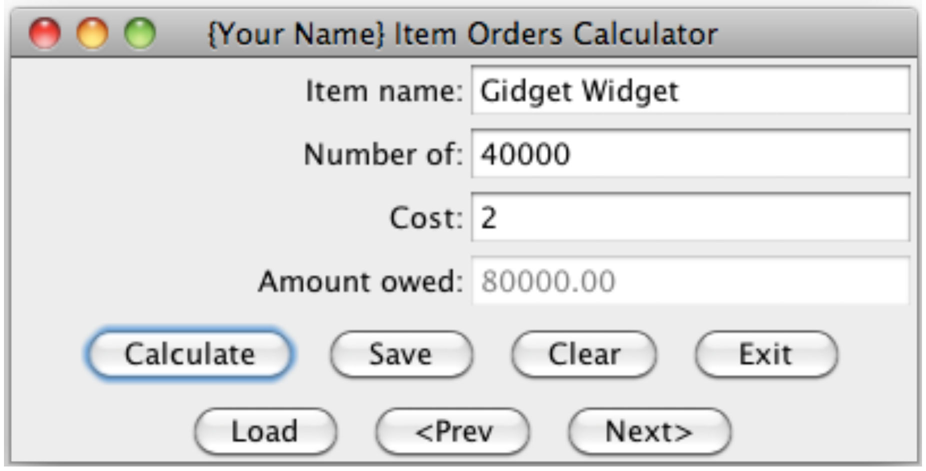
Adding more buttons controls than were in lab 1.

Writing good code. When adding new functionality, think atomic methods. Create methods to do specific tasks. Call these methods from the different actionPerformed. Such as you should have done for the Calculate button’s actions, which needed to be called from both the Calculate and Save buttons.

Load, Next> and <Previous: We want to add controls that will load the lab 1 CSV file so we can step forward or backward through the file by pressing the next or previous buttons. This requires these controls:

- Load – Reads the set file into the program for use later. Displays the first record in the file
- Next – Displays the next record in the file
- Prev – Displays the previous record in the file

See requirements on the next page for Load, Next and Previous buttons.

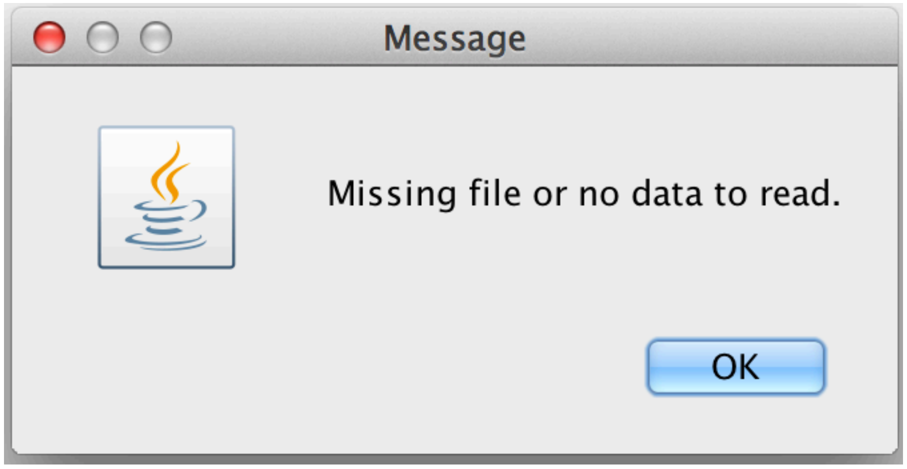

{Your Name} Item Orders Calculator

Item name: Gidget Widget
Number of: 40000
Cost: 2
Amount owed: 80000.00

Depending on how you wrote the original program you may have to make adjustments to setting the size of the JFrame or moving components to different areas of the frame layout.

Load, Next, Previous:
The Load method is to return an integer of the count of items read into the program. As records are read, store all records in memory for later use. If no records are loaded because of some error or the file does not exist, a negative one, -1, is returned. This error is reported by using the JOptionPane class to display the following message as shown.
After loading all the records, display the count of how many records were loaded.

```
JOptionPane.showMessageDialog( null, message );
```


Message

Missing file or no data to read.

The Next method displays the next item in the collection of orders you read. When the last order is displayed, clicking the Next will show a message that there is no more data in the next direction.

The Prev method displays the previous item in the collection of orders you read. When the first order is displayed, clicking Prev will show a message that there is no more data in the previous direction.

Your program must catch and handle each separate exception, showing an appropriate message using the JOptionPane popup messages. This means you need to catch more than 'Exception'.

HInt: You may replace the JOptionPane *null* with the Load button object, next or previous button object based on which button was last pressed. This centers the message over the button that was just pressed, making seeing the error message more visible to the user. This is vitally important when using multiple monitors.

Use the String's split method. Do not duplicate any significant amount of code.

Note: The opening and closing of files should be kept within the methods required. Files should not be left open through the whole program.

Bonus:

The buttons need more explanation than the text on them. Let’s add a tool tip to the button. To find how to do this, look at the JButton class and search for a method that will set the tool tip text for a button. Text to add should be something meaningful like for the Save button: “Calculate and save order”.

R·I·T	Rochester Institute of Technology Golisano College of Computing and Information Sciences Department of Information Sciences & Technology

ISTE-121 Homework 1 gradesheet   Name: ____________________________
GUI with IO

Item	Possible Points	Earned Points
GUI displays as expected	15	
Code to handle events is modularized into methods	15	
Calculate – only one calculation in the code Save – calculates then saves in a CSV file Clear – clears all fields Exit – makes sure file is closed and exists the program	10 from Lab 1	
Load: • File/data does't exist message appears when expected • Load method returns int value (-1 or record count) • Display number of records loaded at end of load • Handles all possible errors, even if error not displayed	20	
Previous: • Determines if there is a previous record to show, if not display a message • Displays previous information from the list each time Prev is clicked	20	
Next: • Determines if there is a next record to show, if not display message • Displays next information from the list each time Next is clicked	20	
Bonus: Tool tips	+5	
Deductions: • Unnecessary duplication of code (-5) • Not following coding standards, naming coventions • Missing javadocs for class/method headers, comments in other parts of code.		
Total:	100	

Comments: