Homework 2: Swing GUIs

CS 150 with Dr. Sam Schwartz

Due: Sunday, February 22 at 11:59pm via Canvas Upload

1 Purpose

In this homework assignment you will gain experience with creating a non-terminal graphical user interface and integrating it with multiple classes. This assignment will also reinforce object-oriented concepts like scope, encapsulation, access modifiers, and static vs instance variables. It will also start to hint at model-view-controller patterns.

2 Tasks

Part A – Create a Welcome Message UI

Build a simple Welcome App.

Your app should have a `Greeter` class with a method called `getGreeting()`.

Your app should have a separate `View` or `Main` class (which uses `Greeter` as a model) that displays a greeting in a Swing window when a button is clicked.

This app should use JFrame, JLabel, and JButton.

Part B – Build a Login Page

Add an additional button to the app you created in Part A. This button should have the text, "Go to Login" which changes the UI to show a login page.

This login page should utilize an underlying `User` class with private fields. The login page should validate an email/password when the user clicks a "Login" button and subsequently show a success or failure message in the UI.

Part C – Show a Simple Counter App

Upon login success, change the code written in Part B so that the app now shows three buttons, labeled A, B, C. The app should also have four labels: the number of times a user has clicked on A, B, C, and another label showing total clicks across all buttons.

Your app should use static and instance variables in an underlying "model" class, separate from the view, to dynamically display these counts.

Part D — Submit to Canvas

Upload the following five files to Canvas:

- `code.zip` which contains all of your code
- `screenshot-a.png`, or a similar filename, that demonstrates that the code you wrote in Task A is working as specified via a screenshot of your IDE/console.
- `screenshot-b.png`, or a similar filename, that demonstrates that the code you wrote in Task B is working as specified via a screenshot of your IDE/console.
- `screenshot-c.png`, or a similar filename, that demonstrates that the code you wrote in Task C is working as specified via a screenshot of your IDE/console.

3 Grading Criteria

In general I am looking for the elements of validity, readability, and fluency in all code-based assignments. (See more below.)

I tend to dock 5ish points off for each error (although smaller or larger quantities like -1pt or -10pts exist based on the magnitude of the error), and will provide free-form feedback detailing why any points were missed in the comments on Canvas.

Please note that I do not get notifications about replies to my comments on Canvas, so if you have any questions please reach out to me directly.

Validity

Student submitted files which implemented the tasks correctly and handled possible errors gracefully.

Moreover, the student submitted screenshot(s) displaying the code's output.

Readability

The deliverable used professional English and typesetting throughout, and had all methods annotated with JavaDoc.

Fluency

The deliverable was executed in such a way that an experienced practitioner would not find the deliverable "weird-in-a-bad-way" or unduly jarring.