Homework 09 - Cafe1331 Review

Written by: Sumit Choudhury, Dhruv Patel, Maddy Scandlen & Emma Barron

Description and Purpose

Upon realizing that you suddenly have a lot of free time, you decided to open a restaurant called **Cafe1331**. Word spreads fast about your new and popular restaurant. However, potential customers want to know what others thought of your cafe. Oh no, what will you do?

Never fear, JavaFX is here!

In this assignment, you will implement a colorful JavaFX GUI application to represent a review page for your restaurant. You will use various UI controls, layout management, and UI event handling. It will give you good experience working with these kinds of event-drive GUI programs.

Note: You will be learning about event handling while you have the HW available to you. But you can get started with what you already know.

Directions

Install JavaFX!

Follow the instructions on Canvas to properly install JavaFX.

Write a JavaFX Class called CafeReviewPage with the following functionality:

- The title of the window should be "Cafe1331 Reviews".
- There are three main components in the GUI:
  - An area for posted reviews with a minimum width of 350 pixels and minimum height of 450 pixels
  - A place for user to input their review.
  - A button for posting the user's review.
- At the bottom of the GUI, allow a user to input their name, feedback, and color for their review. It should be clear to the user which input field matches name, which input field matches feedback, and which one matches color.
- When the 'Post' button is clicked:
  - If the user did not enter a name, the review should have the name "Anonymous". (For an example, see figure #1 in sample solution section)
  - If the user entered a color, the review's text should be that color. If the user did not input a color or input an invalid color, then do not change the review's text color. The default text color is black. Be sure to catch any exceptions from invalid colors. (For an example, see figure #2 in sample solution section) (Hint: Check out the Color class!)
  - If the user did not enter feedback, do not post. Otherwise, do post.
- Once a review is posted, the input fields should be cleared of any text.
- A review should be added to the page underneath previous reviews.

Notes

(Your solution may look different from the examples!):

- This is a open-ended assignment so BE CREATIVE!
- Make sure that your final submission has proper functionality.

- Chapters 14-16 in the textbook cover JavaFX.
- **There will be 20 points extra credit for going above and beyond on the assignment.** The breakdown of those 20 points is as follows:
  - 5 points for a non-trivial addition to the program's graphics that clearly extends its look
  - 5 points for a non-trivial addition to the program's controls or functionality that clearly extends its capabilities
  - 10 points for a impressive, simply awesome submission (this will be a subjective judgment on what our graders see)

Tips and Tricks

- The Java API is your friend. Use it to your advantage.
- Make sure you are properly able to run JavaFX programs before starting this assignment. Do NOT wait until the last moment to configure your JavaFX!
- Work incrementally. Get a basic UI up and running. Add the buttons, cells for display, etc., piece by piece. Think of which type of layout manager(s) you want to use.
- You will be learning about event handling while you have the HW available to you.

Sample Solution

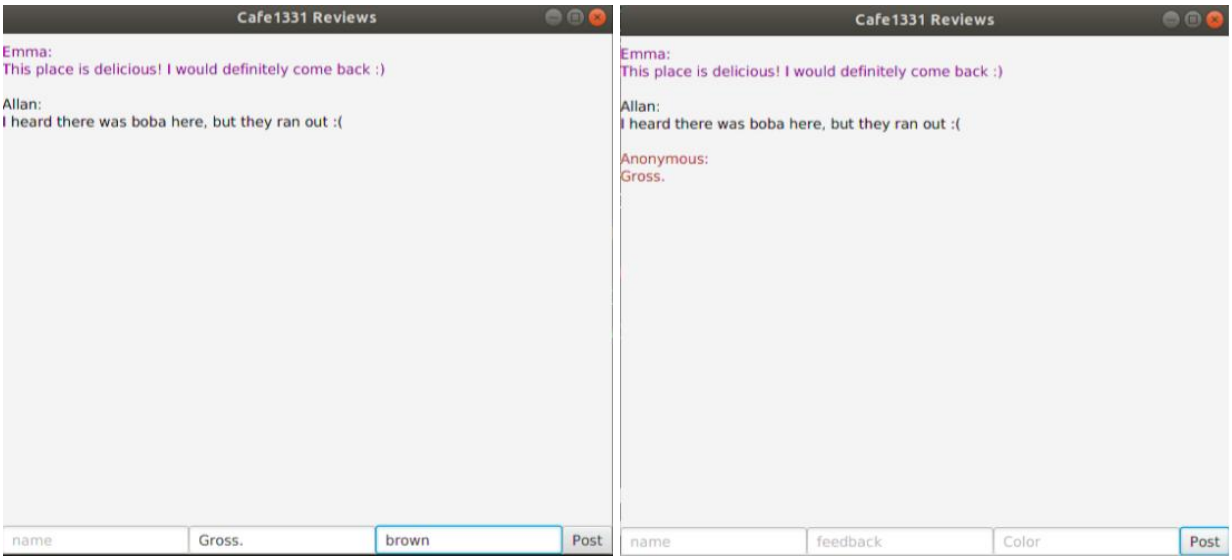

Figure #1. Example of adding an anonymous review. (read left to right).

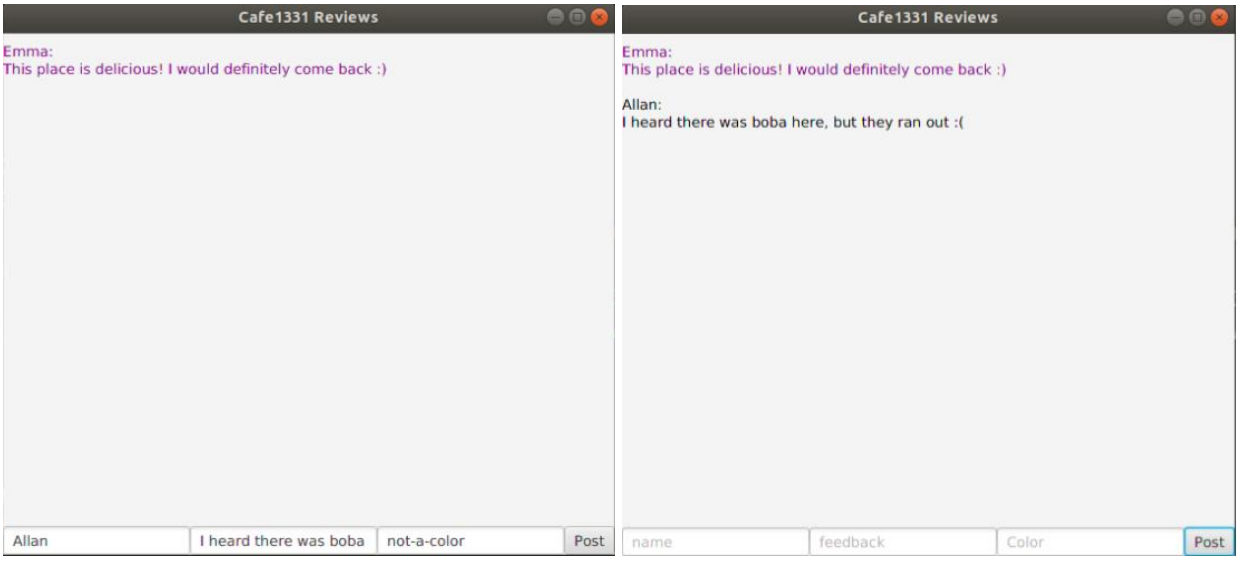

Figure #2. Example of adding an invalid color. (read left to right).

Allowed Imports

You can import anything from the javafx library.

Checkstyle and Javadocs

You must run checkstyle on your submission. The checkstyle cap for this assignment is **20** points. If you don't have checkstyle yet, download it from Canvas -> Files/Resources. Place it in the same folder as the files you want checkstyled. Run checkstyle on your code like so:

```
$ java -jar checkstyle-8.28.jar yourFileName.java
Starting audit...
Audit done.
```

The message above means there were no Checkstyle errors. If you had any errors, they would show up above this message, and the number at the end would be the points we would take off (limited by the checkstyle cap mentioned above). The Java source files we provide contain no Checkstyle errors. In future homeworks we will be increasing this cap, so get into the habit of fixing these style errors early!

Additionally, you must javadoc your code.

Run the following to only check your javadocs.

```
$ java -jar checkstyle-8.28.jar -j yourFileName.java
```

Run the following to check both javadocs and checkstyle.

```
$ java -jar checkstyle-8.28.jar -a yourFileName.java
```

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or breaks our auto grader. For that reason, do not use any of the following in your final submission:

- var (the reserved keyword)

- System.exit
- Runtime.getRuntime.halt
- Runtime.getRuntime.exit

Collaboration

Collaboration Statement

To ensure that you acknowledge collaboration and give credit where credit is due, **we require that you place a collaboration statement as a comment at the top of at least one java file that you submit**. That collaboration statement should say either:

I worked on the homework assignment alone, using only course materials.

or

In order to help learn course concepts, I worked on the homework with [give the names of the people you worked with], discussed homework topics and issues with [provide names of people], and/or consulted related material that can be found at [cite any other materials not provided as course materials for CS 1331 that assisted your learning].

Recall that comments are special lines in Java that begin with `//`.

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- `CafeReviewPage.java`

Make sure you see the message stating "HW09 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the homework. We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

1. Prevent upload mistakes (e.g. forgetting checkstyle, non-compiling code)
2. Provide usage validation (e.g. forbidden imports, reserved keywords, etc)

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files.
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for all official clarifications
