Lab 3 - Say Cheese!

David B. Levine
Computer Science Department
St. Bonaventure University

Objective

This lab gives you practice with 2-dimensional arrays and shows another application containing multiple implementations of an *interface*.

General Idea

The goal of this lab is to gain more experience writing multiple implementations of an interface. Our basic data structure will be a (group of) two-dimensional array(s) that represent an image. All of these will exist within the context of a larger image manipulation program.

Instructions:

1. Start Eclipse on your machine. Make sure that you are in your own workspace. Download the file *SingleImages.zip* and save it to your Desktop. Uncompress it (again on your desktop). Now choose "File:Import" from the Eclipse menu. Import the "Existing Project" named "SingleImages" into your workspace. (Use the Desktop\SingleImages as your root directory and make sure that the "Copy project into workspace" box is **not** checked.) Run the program ImageTestBed. Maximize the screen and then play with the images. Some are stored in the SingleImages / Images folder now found on your desktop. For each button in the program, list its name and a description of its effect on the image(s).

2. Answer the following questions:

    a. Within the package named CS132Images, how many files are there?
    b. Which file contains the interface? How do you know?
    c. Which files are apparently unused by the program (at this point in time)?

3. Open the file `ImageVerticalInverter.java` and look at it. Open the file `ImageTestBed.java` and scroll down to the createButtons method (about line 65). After verifying that your project has an `ImageHorizontalInverter.java` file, figure out how to add a button for this to the interface. Do so and run the program. Describe the effect of your new button. Looking at the code, explain why this is so.

4. Modify the ImageHorizontalInverter class to do the right thing. After testing it to your satisfaction, print out the listing of the class. Be sure that it contains your name at the top. (Dr. Levine no longer deserves credit for the amount of code he wrote here.)

5. Next create a new class that will implement the ImageTransformer interface. Name this class ImageDarkener. (It should be part of the auxilliary package.) An ImageDarkener should darken the image by reducing all color values by 1/3. [To do this first double the value and then divide it by three.] The ImageNegator class can be used as a pattern for your class. Add an appropriate button to ImageTestBed and test your code, including showing your result to your instructor. Once you are confident that it works, print the listing. Now darken an image, copy it from right to left, and then repeat a few times. Describe the behavior that you observe.

6. Next write an ImageBrightener class. An ImageBrightener brightens the image by increasing the color values. Each value should increase by half the distance between the value and 255. Thus, 155 would go to 205 while 205 would go to 230 and so on. Once again add the button, test the code and print the listing. Now, starting with a "clean" image, Darken it and then Brighten it. Do you get the same image back? Explain.

7. Last, we create a ImageRemoveBlue class. We will do this in several steps. First create it and add a button to the user interface so that it leaves the image in tact. Next, add a private static method named assignConstantSingleChannel to the class that assigns a constant color value to each cell of the three arrays. Have the removeBlue method assign the constant 128 to each channel. What happens to the image? Explain. Add a second parameter to the assignConstantSingleChannel to pass the constant color value that will be assigned to the channel. Then assign the value 255 to the red channel, 128 to the green channel and 0 to the blue channel. What happens to the image? Explain. Finally, modify removeBlue to leave the red and green channels unchanged and assign 0 to the blue channel. This should accomplish the task. Once you are confident that it works, print the listing.

8. EXTRA CREDIT ONLY. Create, and install an ImageSepiaTone. Once you are confident that it works, print the listing.

Hand in:

Hand in a cover page, a document answering the questions from Steps 1, 2, 3, 5, 6, and 7 and the printouts from Steps 4, 5, 6, and 7. If you completed the extra credit, include printouts from Step 8.

Due Date:

The report is due at class time on Monday, February 3

Assignment Type (see Academic Practices and Policies Document):

Help Policy in Effect for This Assignment: Group Project with Limited Collaboration

In particular, you may discuss the assignment and concepts related to the assignment with the following persons, in addition to an instructor in this course: any member of your group; any St. Bonaventure Computer Science instructor; and any student enrolled in CS 132.

You may use the following materials produced by other students: materials produced by members of your group.