CO871 2013/14 Assessment 2

Introduction

The aim of this assessment is for you to demonstrate that you can design and build a Graphical User Interface in Java, using Java Swing. You will add a GUI to the Fotoshop application that you developed in Assessment 1; you should build on the code that you wrote for this first assessment, though you are permitted to change that code.

One of the things that we are testing in this assessment is that you are familiar with details of how coding in Swing works. Therefore, you *must* write the code directly, i.e. not use some kind of GUI-builder. If you have any queries about this then please ask me.

Submission

You should submit, via Moodle, a zip file containing the pdf document discussion in task 1 of the specification, and a NetBeans, Eclipse or BlueJ project implementing the remainder of the parts. The deadline is 23:00 on 2nd December 2013.

Specification

1. Read through the remainder of the specification. Write a document that contains a design for the GUI. This should consist of a sketch of the GUI, a justification for why you have implemented each of the parts of the solution in the way you have, and a description of the extension that you have implemented for the final part. This should be at most two pages and be submitted as a pdf file.

[10 marks]

2. Create a Swing window. Create some way of displaying an image—call this the *current image*—in the window. Then, create some way of loading and saving images into the program, e.g. through a file choice dialogue or via text boxes. You are allowed to use the file chooser in Swing for this.

[15 marks]

3. Create some way of applying filters and transformations to the current image. This could be a series of buttons, or menu options. You should implement at least the *mono* and *rot90* commands.

[10 marks]

4. Implement a filter that adjusts the brightness of the image (there is no need to do this in a very sophisticated way). Add a slider to the interface that adjusts the brightness of the current image.

[10 marks]

5. Create some way to display a list of the filters that have been applied to the image since it was loaded (a “breadcrumb trail”). Then, implement some means of choosing from the list, so that the current image reverts to the image in the state that it was at that point. There are a number of ways of doing this; adding the filters to a list is one way to do this.

[15 marks]

6. Add some buttons to the interface that carry out a sequence of filters. You should be able to customise *which* sequence of filters, and be able to add a name to this sequence, which will appear either on the button or in a label underneath it.

[20 marks]

7. Choose a substantial addition to the interface, and implement it. By “substantial” I mean something like tasks 5 or 6 above. I am happy to advise on whether a proposed addition is of the right scale (but, not give you ideas for this—you must come up with your own idea).

[20 marks]

Total Marks: 100.

This assessment is worth 50% of the total marks for this module.

Plagiarism and Duplication of Material

We want you to evidence that you can program. Therefore, the work you submit must be your own, unaided effort. We will run checks on all submitted work in an effort to identify possible plagiarism, and take disciplinary action against anyone found to have committed plagiarism; this can include penalties that go beyond the work on this assessment.

There are a large number of common patterns in Swing programming, and so obviously there are no penalties for using these patterns.

Generally, using a line or two from elsewhere is fine; using a few lines is okay if you acknowledge the source of the lines; using more than six lines, is generally unacceptable.

Some guidelines on avoiding plagiarism:

• One of the most common reasons for programming plagiarism is leaving work until the last minute. Avoid this by making sure that you know what you have to do (that is not necessarily the same as how to do it) as soon as an assessment is set. Then decide what you will need to do in order to complete the assignment. This will typically involve doing some background reading and programming practice. If in doubt about what is required, email C.G.Johnson@kent.ac.uk.

• Another common reason is working too closely with one or more other students on the course. Do not program together with someone else, by which I mean do not work together at a single PC, or side by side, typing in more or less the same code. By all means discuss parts of an assignment, but do not thereby end up submitting the same code.

• It is not acceptable to submit code that differs only in the comments and variable names, for instance. It is very easy for us to detect when this has been done and we will check.

• Never let someone else have a copy of your code, no matter how desperate they are. Always advise someone in this position to seek help from their class supervisor or lecturer. Otherwise they will never properly learn for themselves.• It is not acceptable to post assignments on websites, and we treat such actions as evidence of attempted plagiarism, regardless of whether or not work is paid for.

You are reminded of the rules about plagiarism that can be found in the Stage I Handbook. These rules apply to programming assignments. We reserve the right to apply checks to programs submitted for assignment in order to guard against plagiarism and to use programs submitted to test and refine our plagiarism detection methods both during the course and in the future.