CISC 322 Fall 2017 Group Project

Version 4.0 17-09-09 13:02

This is a **draft** that may change in response to student questions

Table of Contents

1 Overview

A major deliverable for CISC 322 is a group project wherein you will work within an existing framework to implement a project of your choosing. See the onQ site for deadlines, and the course website for where the project fits into the course grading scheme. There is also a component for presenting your final software, in the first two lectures of the last week of classes.

1.1 Learning Outcomes

This project evaluates aspects of the following course learning outcomes listed on the course website:

- work effectively in a group setting
- communicate effectively verbally and in writing
- recognize and describe the various architectural styles and design patterns

Phase 1 evaluates the additional learning outcome (a specific case of "communicates effectively")

- Make tradeoffs among alternative technical decisions and justify their choice.

1.2 Team Formation

You will form a team of 3 people. Team formation will take place in the third week of term after the add-drop period. Since the learning outcomes require group work, it is not appropriate to do the work individually. If an individual or pair are left out of group formation, I will assign them to existing 3-person groups.

Ideally all team members should be familiar with Java and at least one person on your team should already be familiar with Swing. Anyone not familiar with Java should make up the missing knowledge with the help of their teammates, and consult relevant sections of the Java Swing tutorials at

http://docs.oracle.com/javase/tutorial/uiswing/

The material in lectures on the *Observer* (or *Publisher/Subscriber*) design pattern (currently scheduled

for Week 7) may be helpful in understanding *Listeners* but will certainly not be enough to familiarize you with Swing.

2 Project Phases

The project has several phases. Each requires a combination of software and prose. You should write the documentation for each phases as you finish it, and combine Phases 2-4 into a single coherent report at the end. In your final report you may need to extract, combine, and rewrite material from earlier reports, but some material should remain in separate sections corresponding to the phases, to describe your progression through them.

You are responsible for setting reasonable intermediate deadlines well before the due dates (for which see http://onq.queensu.ca). You can begin Phase 1 immediately; you can begin Phase 2 as soon as you finish Phase 1, and Phase 3 as soon after that as you complete the appropriate readings.

2.1 Phase 1. Choosing a Project

All projects must use the existing editor framework (see the separate handout) to implement a system that allows users to read, write, and modify some moderately complex type of data. A huge number of possibilities fit these simple criteria. The existing framework handles text files; last year's project handled CSV files. You should ideally pick a data type for which packages already exist to do much of the work; for example, Java has an extensive image handling library.

The documentation for this phase must include some discussion of tradeoffs. A tradeoff study considers more than one alternative for a design decision (preferably three or more), documents the strengths and weaknesses of each alternative, and justifies a single choice among them. For example, if you pick a data type for which several packages exist, you could discuss how each package suited the framework and your intended editing functionality. Alternatively, you could discuss what criteria led you to pick one particular data type from several possibilities you considered.

The main grading criteria are thoroughness of your analysis of alternatives and quality of the justification for your choice.

Include an appendix that lists the functionality you intend to provide in Phase 4. This will not be graded; we will use it to discuss whether your plans are of appropriate complexity (neither too difficult nor too easy). We will negotiate an order of implementation of the features to make sure you have something suitable to deliver in Phase 4.

Hand in this document separately in the appropriate dropbox.

2.2 Phase 2. Displaying The Data.

In this phase you will demonstrate a prototype that shows you can read and display your chosen type of data. You may use any means appropriate for getting this functionality implemented quickly, and need not use Swing yet (but it would be a good idea to do so if it doesn't take undue amounts of time). You need not be able to write data yet (but it would be a good idea to do so). Do not include features for editing the data. You may or may not be able to reuse much of this work in Phase 3.

Hand in a screenshot of your program displaying some simple but non-trivial data.

2.3 Phase 3. Integration with Editor framework.

The the course directory on CASlab contains an `editor` subdirectory with a `.jar` file containing an editor framework and Java source code for a simple text editor (`.txt` files) that uses it. Change the text editor classes to interact with the framework to open and save your chosen type of file. It is best if you understand the Factory Method and Abstract Factory design patterns before doing this.

Document

- The design patterns you used in your implementation.
- The classes you introduced, and how they interact with each other and with existing classes. Use an appropriate combination of prose and UML diagrams.
- What other design and implementation decisions you made in interfacing with the framework.

Hand in a screenshot of your editor displaying some simple but not trivial data.

2.4 Phase 4. Adding Functionality

Implement at least four actions that modify the existing data. They must be features not directly provided by the Swing components you used to display it; for example, in the text editor, inserting and deleting text were already provided by `JTextArea` and so wouldn't count.

3 Peer Evaluation and Individual Grades

Phases 1 and 4 will require each of you to submit (in the indivdual dropbox on the OnQ site) a `.txt` file reporting your evaluation of how well each group member (including yourself) contributed to the overall project. Include a brief description of why you gave the evaluation you did. Other document formats are not acceptable. The peer evaluation scheme is in a separate handout.

Especially strong or weak peer evaluations will modify the "group grade" for the individual. Thus a 4/5 peer evaluation will result in the individual getting 80% of the group grade for that deliverable.

4 What to Hand In

For Phase 1, hand in

1. a single `.pdf` file with your report, in the group dropbox and
2. a `.txt` file with your peer evaluation, in the individual dropbox.

Other formats will not be accepted.

For each of Phases 2 and 3 hand in a single `.jpg` file with a screenshot of your software displaying a simple instance of the data type you are dealing with. These will not be graded, but failing to hand in a conforming screenshot will incur a 5% penalty to the Phase 4 grade.

For phase 4, hand in (in the group dropbox)

1. A single executable `.jar` file with the final working version of your software. It must run correctly on CASlab with the default classpath via a suitable `java -jar` command (which you should document). The grader will ensure there is an unchanged copy of the framework `.jar` file in the directory used for testing.
2. A `.pdf` file with a combined report from phases 2-4.

3. A `.zip` file with all your new and edited `.java` source code.

In the individual dropbox, hand in a `.txt` file with your peer evaluation.

Each `.pdf` file must have a title page naming the project and all group members, an "active" table of contents (where clicking on the ToC takes you to the appropriate section), and numbered sections (and subsections, if you have any).

Deviating from these instructions will be considered failure to submit.