HW3 – Reducing Complexity with Appropriate Use Design Patterns

Estimated time: 16-20 hours

Objectives

- Become more familiar with UML modeling
- Become familiar with applying the Singleton, Factory, Flyweight, Command, and Undo patterns
- Improve unit testing skills to reduce complexity and improve performance
- Strengthen programming skills in area of user interfaces

Overview

In this assignment, you will build a simply drawing program that allows a user to paint a scheme with instance of different kinds of objects.

Functional Requirements

1. The user should be able to create a new drawing at any time, through either a menu option or a hotkey.
   1.1. The user should be able to select an initial background color or image that fills the entire drawing space.
2. The user should able to create instance of various likes of objects and place them on the drawing. You are free to decide what drawings your program with make and therefore what kinds of objects your program will support. For example, you example, you program could be for painting landscapes, drawing cartoons, creating floor plans. For landscapes, some of the kinds of objects that you might support would trees, bushes, and flowers. Your program must allow the user to create and places at least six different kinds of objects on drawing.
3. The user should be able to select an existing object on the drawing and perform some action on that object.
   3.1. The user should be able to move the selected object on the drawing
   3.2. The user should be able to remove the selected object from the drawing
   3.3. The user should be able to change certain some properties of the object, like size or color
   3.4. The user should be able to duplicate the selected object
4. The UI should allow multiple gestures (via keystrokes or mouse movements and clicks) for most actions.
5. The user must be able to undo previous drawing actions in reverse order, all the way back to the initial drawing creation action.
6. The user should be able to save a drawing and re-open it later.
7. The user should be able to export a drawing as an image file.

Instructions

To build this system, you will need to do the following:

1. Decide what you'd like your drawing program to be for, e.g. landscapes, people, blueprints, or even UML diagrams.
2. Design your system. Look for **good** opportunities to apply the singleton, factory, flyweight, command, and undo patterns. Also, don't forget about the other patterns that you've learned, like strategy and observer. Keep your design simple!
3. Implement and test the application logic, e.g. your classes for the various types of objects that your drawing can contain, classes that load, save, or export drawings, and the drawing itself. Your unit tests for these components must be executable and thorough
4. Implement your UI
5. Test your software at the system level using ad hoc testing methods.

Submission Instructions

Zip up your entire solution, including test cases and sample input files, in an archive file called CS5700_hw3_<*fullname*>.zip, where fullname is your first and last names. Then, submit the zip file to the Canvas system.

Grading Criteria

Criteria	Max Points
A clear and concise designs consisting of UML class, interaction, and state diagrams	20
A working implement, with good encapsulation, abstractions, inheritance, and aggregation, and appropriate use patterns study in class to date	40
Thorough executable unit test cases for core application logic	30
Reasonable systems test using ad hoc methods	10

Optional Extra Credit

Criteria	Max Points
Allow for more than 6 different kinds of objects, where some are specializations of others.	5
Allow the user to create a more sophisticated terrain or world, instead of a simple background color or image	15
Make the terrain and drawing objects 3D and render the drawing in perspective	30