HW4 – Alternative A – Command Pattern

Estimated time: 24-30 hours

Objectives

- Become familiar with applying the Command with Undo and Redo features.
- Gain more practice applying other patterns.
- Learn to considering performance when selecting between design alternatives.
- Improve unit testing skills to reduce complexity and improve performance.
- Strengthen UML modeling skills.
- Strengthen programming skills in area of user interfaces.

Overview

In this assignment, you will build a basic drawing program using the Shapes library the you built in HW1 and extended in HW3.

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 give the drawing a title
   1.2. The user should be able to modify the title at any time

   Note: a drawing can be thought as composite shape

2. The user should be able save a drawing to a file.
   2.1. If the drawing has not been saved before, then the program should ask the user for the path of the file to create.
   2.2. Even if the drawing has been saved before, the program should include as “Save as” option that allows user specify a new path.
   2.3. When specifying a path for a file, the program should allow the user to browse the file system to select a directory and file name.
3. The user should be able load a previously saved drawing through a menu option or hotkey.
   3.1. If the user has unsaved work in a current drawing, the program should warn the user before loading the request drawing and give the user a chance to either save the current drawing or discard unsaved changes.
   3.2. The program should allow the user to browse the file system to select a path.
4. The program should present the user with a palette containing the following types of shapes: ellipse, circle, rectangle, square, triangle, line, and embedded pictures.
5. The program should present the user with a canvas on which the drawing will be rendered.

6. The user should able to select any component-type from a palette and place it on the drawing, using reasonable gestures or commands.

7. The user should be able to select one or more existing shapes in the drawing and perform any of the following actions using reasonable gestures or commands.
   7.1. The user should be able to delete selected shapes.
   7.2. The user should be able to move selected shapes.
   7.3. The user should be able to copy the selected shapes to a paste buffer.
8. The user should be able to paste the shapes from the paste buffer to the drawing.
9. The UI should allow multiple gestures (keystrokes, mouse movements, and button clicks) for some of the actions. For example, the user should be able to hit the delete key to delete selected shapes as well as clicking on a trash can icon.
10. The program should keep a history of actions (commands) that the user performs to since the drawing was created or last opened.
11. The user must be able to undo previous drawing actions in reverse order, all the way back to the initial drawing creation or opening of the drawing.
12. The user must be able to redo undo action.

Instructions

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

1. Study the requirements. Ask questions to clarify, if necessary.
2. Design your system.
   2.1. Look for **good** opportunities to apply the design patterns.
   2.2. Use the patterns to help manage the inherent complexity of the problem and eliminate accidental complexity, if it seems appropriate. Don't feel like you must use every pattern studied to date. Only use a pattern if makes sense. However, there should be an obvious case for an application of the Command pattern.
   2.3. Be sure to separate your GUI Layer from your Application Logic Layer.
3. Implement and test the classes in your Application Logic Layer.
4. Implement your GUI.
5. Use ad hoc testing to do testing system.

Submission Instructions

Zip up your entire solution, including test cases and sample input files, in an archive file called CS5700_hw4a_<*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 design documented with UML class and interaction diagrams	20
A working implementation, with good abstraction modularity, encapsulation	25
Effective use of appropriate patterns	30
Reasonable unit-test cases	30
Reasonable ad hoc system testing	15