CS 4773
Object Oriented Systems
Fall 2020

Assignment 3: A drawing program sans drawing

Due: Tuesday October, 26th by midnight

DESCRIPTION:
In this assignment, you will design and build a program that stores an arbitrary sequence of commands that manipulates and uses a drawing list. A drawing list is used to produce a graphical scene, e.g., [create a red rectangle, move it to coordinates (5,5), create a blue triangle, move it to coordinates (10,2), draw the red rectangle, draw the blue triangle]. While it would be fun to actually draw the shapes that we will store and manipulate in our list, we shall instead focus only on the creation and maintenance of the drawing list. NOTE: actually implementing drawing via JavaFX is worth 10 bonus pts.

The commands that your program will be able to process are:

CREATE RECTANGLE [w] [h]

- [w] is the required width of the rectangle
- [h] is the required height of the rectangle
- The origin of a created rectangle is the upper left hand corner at (0,0)
- The color of a created rectangle is Red

CREATE CIRCLE [r]

- [r] is the required radius of the circle
- The origin of a created circle is the center at (0,0)
- The color of a created circle is Blue

SELECT [any shape from 1..n]

- Selects a shape to use with future commands MOVE, DRAW, COLOR, DELETE
- n is the total number of shapes
- Currently selected shape defaults to a “no shape” value
- Calling SELECT on a shape that does not exist in the list will produce the following error message “ERROR: invalid shape for SELECT”

MOVE [x] [y]

- Moves the currently selected shape's origin to the specified (x,y)

DRAW

- Outputs the currently selected shape information to the console. E.g.,

```
Rectangle, Color: Red, Origin: (5,5), Width: 10, Height: 7
Circle, Color: Blue, Origin: (10,9), Radius: 12
```

COLOR [c]

- Sets the color of the currently selected shape.
- [c] is one of the following valid colors:
  - `Red, Blue, Yellow, Orange, Green`

DELETE

Removes the selected from the scene
Sets the currently selected shape to a "no shape" value

DRAWSCENE

- Draws all shapes in the scene using their current origins and colors
- Output for each shape is the same as using the DRAW command

---

UNDO

- "Undoes" the most recent command in the drawing list
- UNDO can be called after UNDO, which undoes the next most recent command in the drawing list
- Undoing a SELECT command sets the currently selected shape to the previously selected shape (which could be "no shape")
- Undoing a MOVE command restores the selected shape to its previous position
- Undoing a COLOR command restores the previous
- Undoing a CREATE command deletes the shape
- Undoing a DELETE restores the previously deleted shape to the scene. It should also restore the currently selected shape to the restored shape
- Undoing DRAW or DRAWSCENE has no effect

- The UNDO command is not stored on the drawing list (i.e., you cannot UNDO an UNDO)

You will start with an empty drawing list. Each command is processed as it is received, which means the shapes in the scene will be modified as certain commands are processed. Note that UNDO can undo the previously processed command, so you will need to come up with a design to accommodate that functionality and you need to use the Memento pattern to provide the necessary support for undoing the current configuration of shapes (don't worry: memento handles this beautifully!).

Your program does not need to process keyboard input, but you are welcome to include that functionality AS LONG as it is also able to process command files like the included examples.

Your object-oriented design must include the Strategy, Command, and Memento patterns. You also need to create and include a UML class diagram that includes all of the classes in your project and indicates where the 3 patterns are implemented.

Use our clean-coding standards. Keep your class sizes <= 200 lines of code.

DELIVERABLES:

1. Name your project CS4773Assignment3
2. Put a PNG or JPG copy of your UML class diagram in your project and call it assignment3UML.png or .jpg. Please clearly identify your strategy, memento, and command patterns
3. If you worked on a team, be sure to include the team member names in the comments area of BB when you submit
4. Archive the project to a zip file called assignment3.zip and submit the resulting zip file to Blackboard.

LATE POLICY:

-10 pts for every day late

RUBRIC

30 pts	All functionality implemented correctly
30 pts	All patterns implemented correctly
20 pts	Domain objects and responsibilities are **well-designed**
10 pts	Code is **readable**. No Javadoc is required.
10 pts	UML class diagram
+10 pts	Program really draws shapes using JavaFX