Final Project

Jpaint

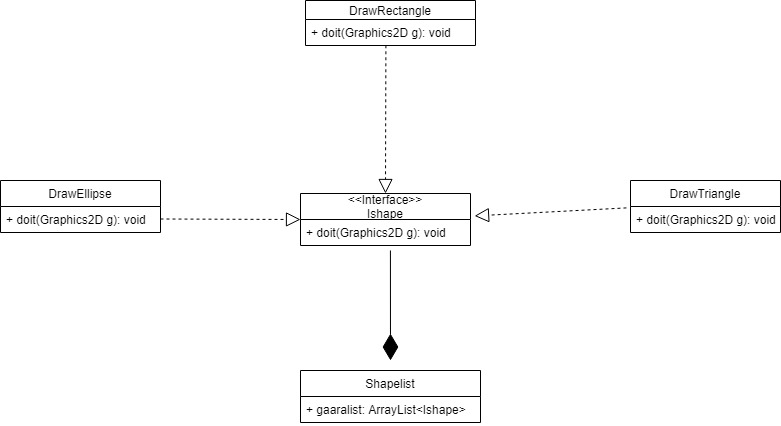
1. **List of missing features, bugs, extra credit, and miscellaneous notes.**

I implemented most of the features except outlining shapes when selected and drag to select, group and ungroup. My undo and redo is buggy, it works fine for creating a shape and partly for delete and paste , I request to check it after done verifying all the other features. I’ve added random print statements in between methods to check which part was working please don’t mind that.

1. **Design Patterns used**
2. Strategy Pattern

The strategy pattern is a behavioral software design pattern that enables selecting an algorithm at runtime. Instead of implementing a single algorithm directly, code receives run-time instructions as to which in a family of algorithms to use.

Classes and interfaces involved – Ishape, DrawRectangle, DrawTriangle, DrawEllipse, Shapelist

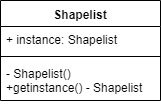


1. Singleton Pattern

The singleton pattern is a software design pattern that restricts the instantiation of a class to one "single" instance. This is useful when exactly one object is needed to coordinate actions across the system.

I used this on shapelist and colorenum(the next one) function used to map colors to shape, since I found I only needed to create these once and other classes can use this by calling upon this instance.

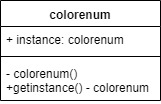
Classes involved - Shapelist



1. Singleton Pattern

For colorenum

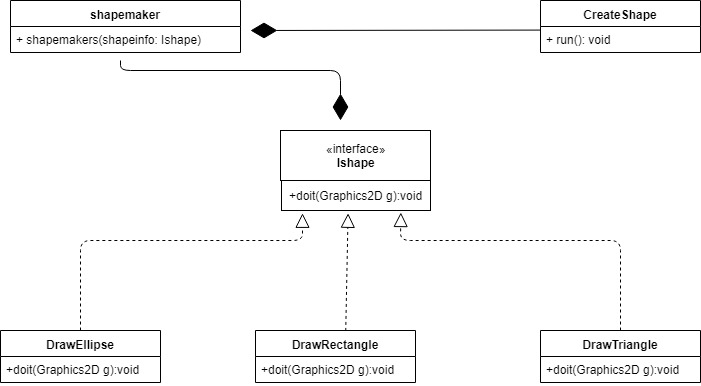
Classes involved - colorenum



1. Factory Pattern

The factory method pattern is a creational pattern that uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created.

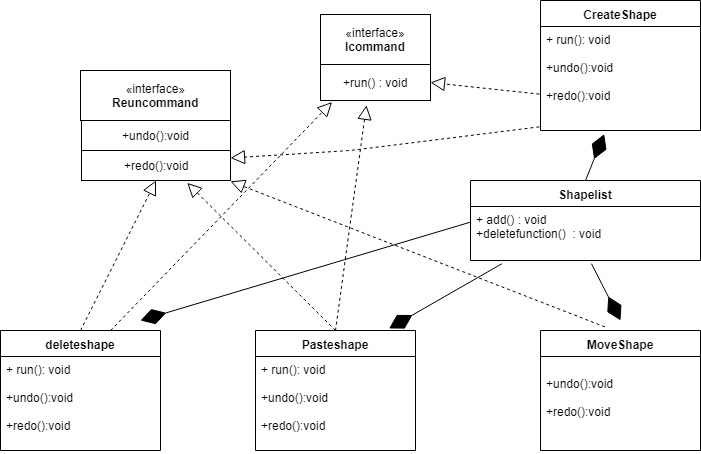
Classes and interfaces involved – Ishape, DrawRectangle, DrawTriangle, DrawEllipse, shapemaker, CreateShape



1. Command Pattern

the command pattern is a behavioral design pattern in which an object is used to encapsulate all information needed to perform an action or trigger an event at a later time. This information includes the method name, the object that owns the method and values for the method parameters.

Classes and interfaces involved – Icommand, Reuncommand, CreateShape, Shapelist, deeteshape, Pasteshape, MoveShape



1. **Success and failures**

The project was really challenging. Trying to implement sprint 4 was the hardest part. Sprint 3 was easier since when I was implementing moving a shape, I accidentally didn’t put a clear function on my Shapelist which resulted in 2 shapes appearing at once when moving which was basically a copy paste function, so I added the same code to copy and paste and edited the moving function. For some dumb reason my redo command in delete is not even getting called, I’m pretty sure I’m missing something minor but couldn’t figure it out. I was focused on getting the features first that made it really hard to refactor the code for adding more design patterns at the end as it was too big of a task at the end. I tried so much for group and ungroup but couldn’t implement it, I felt it was too complicated or I was missing something, which is a big regret. Overall, I’m happy on what I learnt during this project.