

CS 5700 Homework 1: Shapes Report

The first homework assignment was to create a shapes library using OO software development. We were supposed to extend the software to incorporate multiple shapes while using principles of abstraction and modularity. Before I dive into the code, I will first have to explain that I did not take this class because of a requirement, but more for the learning experience I hope to gain through the class. I have not used much Java or C sharp before this class, other than the brief amount that was required of me several years ago. I hope to continue the fast-paced learning that I have begun to undertake as I progress through this class.

As I worked through this assignment, I became aware of how useful UML diagrams can be. UML diagrams not only allow you to break down the design of your code, but they allow you to see how different pieces of your code will interact. This was new to me because while I have briefly used UML before, I have never really had the chance to step back and understand why people use it. I also begun to understand why it is necessary to continuously go back and update these diagrams. When I first started this assignment, I drew everything out on a whiteboard and tried to stick to that exact plan through the entirety of my code. I soon found that this was not very efficient and then had to go back and update this information as I put it into a UML diagram. Also, I created this diagram for the requirements of the class. Looking back, I realize how it would have been much easier to create a Shapes class that incorporated Points and Lines into one class. Then, I would have constructed an interface like I did, but through these two classes, I could have constructed all 2D shapes. I would have done this by using points and lines to create a center point and radius for the circles and ellipses, and then done the same for the vertices and sides of a polygon class. The polygon class would have had n number of lines constructed that connected with one another at shared vertices without crossing paths. Looking at things in retrospect however, always seems simpler than the path taken.

The path that I chose was to use the point class, and use it along with the General Shapes interface, to construct all shapes. I quickly realized the similarities between Ellipses and circles, creating two radii that then would be forced to similarity when being used by the Circle Class. I did the same thing with the Rectangle and Squares Class, except interchanging length for height and width. This allowed me to use the Point class to construct a rectangle class and an ellipse class, with very little code reuse in order to add a square and a circle class. Now by coding this way, I realized that I had restricted myself to rectangles and squares, instead of creating a polygon class that would have allowed me to easily create a triangle class. As shown in my diagram, I used the line class to create the triangles, with the point vertices interconnected.

Another new thing to me was using test cases. I had never used a system such as Gradle or Maven before. While writing these test cases, I began to understand how test cases really work and how simple and yet effective they can be. In the future, I plan to use them much more frequently in my coding and I hope to learn things such as PIT to do mutation test checking as I have been told that integrated programs like these add almost fool proof testing when used correctly.

Overall, I learned a lot from this first assignment although I struggled through the majority of it. Unfortunately, I had to miss the last two classes due to a trip home to Pennsylvania for family related purposes. Knowing this set me back, I look forward to working hard to catch up and improving my software development skills in the coming weeks. I hope to have a better grasp on the next assignment and to gain further knowledge and experience in the design aspect of coding as well as the actual software programming itself.

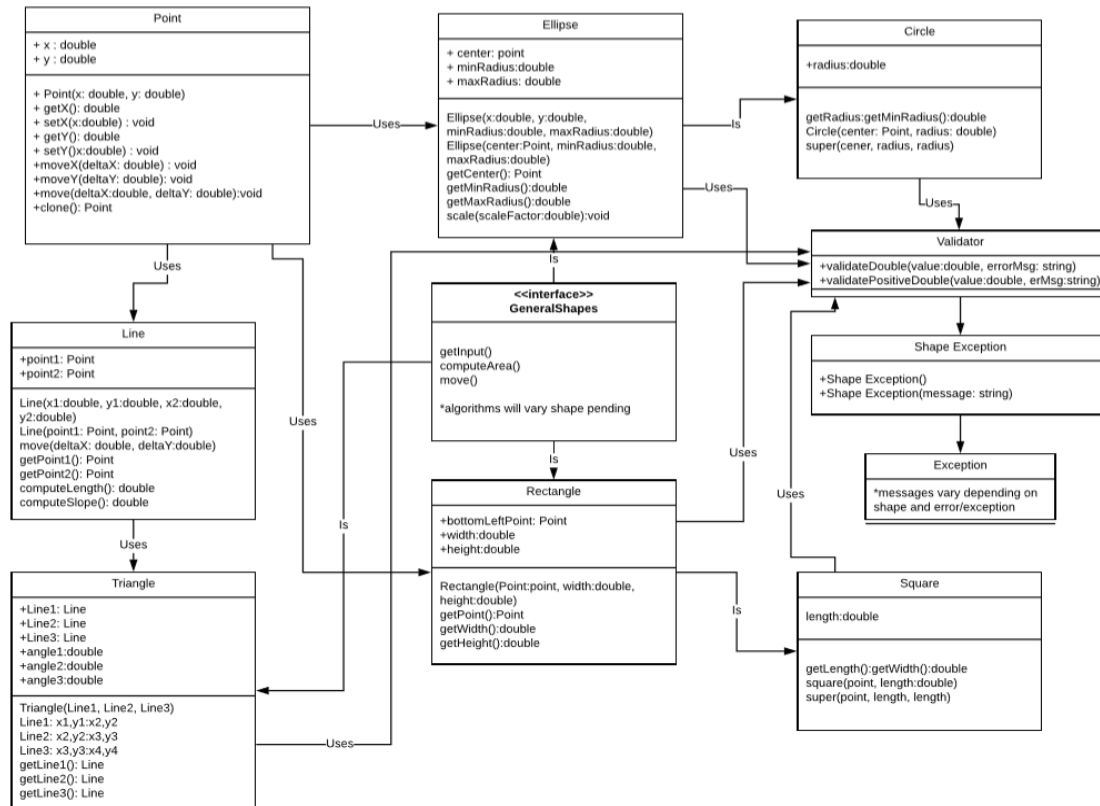


Figure 1: UML of Shapes Library Diagram