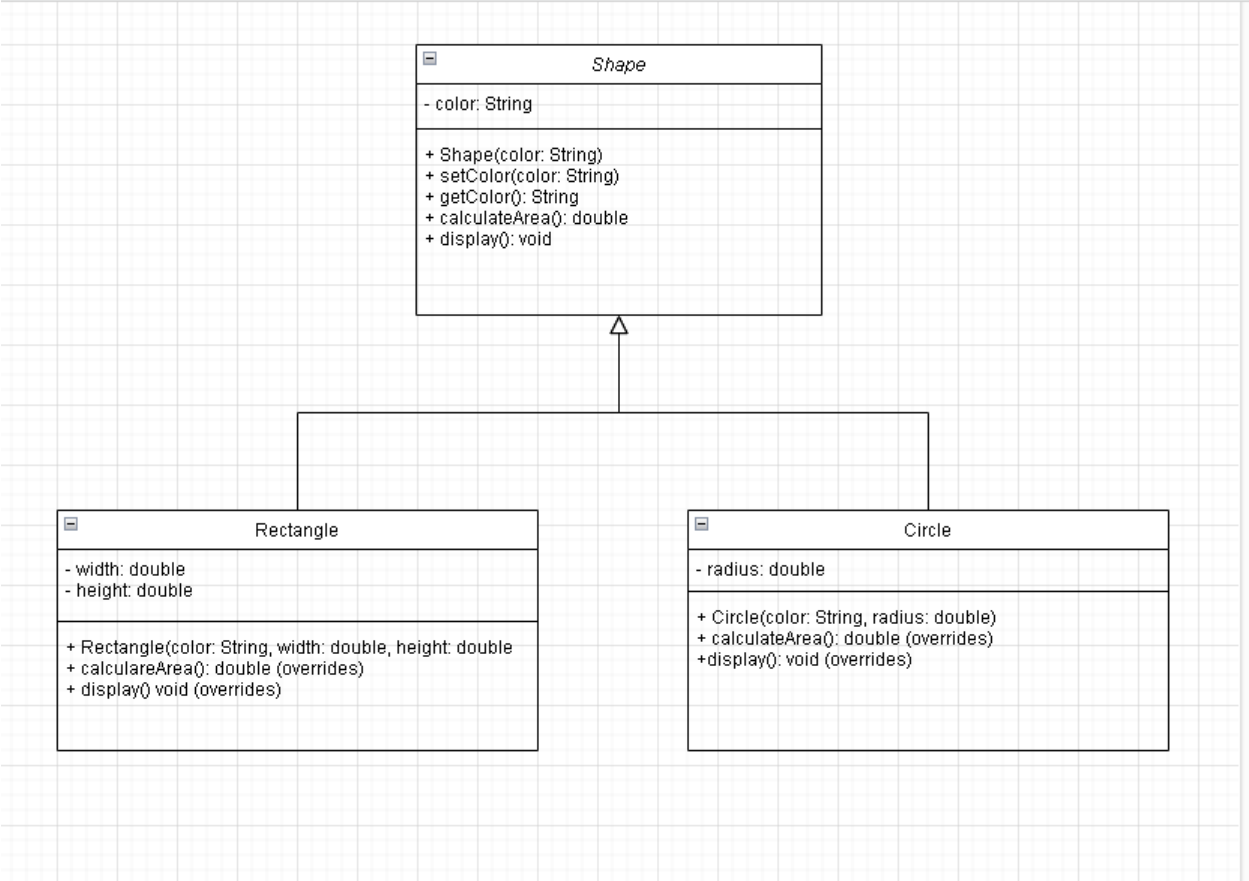
Ryan Richardson

Project 03



Code Analysis

The code used in this example follows many of the modularity guidelines. It displays inheritance, encapsulation, and polymorphism. Methods such as calculateArea() and display() override the shape class for each child class performing the necessary calculations. The attributes are still correctly inherited from the parent class such as color. There are getters and setters which seem necessary in this situation. Methods could be set to private to avoid usage outside its public class. All methods are relevant to the classes and their attributes. All methods are encapsulated in the class such as get and set color, while other methods are still used but overridden by the child classes using their attributes. Each class has one main responsibility

with methods that act on it, manipulating their instance variables. This code can be used on any shape and still have all methods be relevant.