Michael Kuczkuda

Project 2

Change Log

1. Instead of having a global SimpleDrawing called draw, ChartBuilder extends the Class SimpleDrawing. This allows the extended class to override the methods needed to use, componentResized and mouseClicked. The extension also allows the ChartBuilder to create the chart itself.
2. Changed the Nested Class structure to a Super class Structure. I made a class called Shape which has three methods, draw, erase, and checkPoint all of which are meant to be override by each shape but does nothing if not. This class also states that each constructor needs at least one point(an x and y) and a SimpleDrawing. I then moved the Nested Classes from ChartBuilder and made them their own classes, that extends Shape. Shape is also default so that only the package can see it. All subclasses will also be default so that nobody outside the package has access to them besides ChartBuilder.
3. Changed the two different ArrayLists to a single List of type Shape. Originally I had two different Lists for each shape respectively so that they didn't interfere with each other but adding the functionality of click to delete cause that to be inefficient, especially when we were creating even more shapes. Since the Shape has the methods draw, erase, and checkpoint, it was quite easy to use this new List for each shape.
4. Changed the functions eraseSquare, eraseLastSquare, etc to just a erase, and eraseLast. Since the new list was of Type Shape is was easier to access all of the shapes to delete rather than shape specific.
5. Change my line algorithm to the Bresenham Line algorithm. My algorithm didn't allow for diagonal lines before but they were needed for some of the new shapes(Arrow, Diamond, Parallelogram, Triangle). The Bresenham Algorithm Allowed for this.
6. Added the Function checkpoint algorithm to Square, to check the clicks. I created the Algorithm to go through the list and call this function for each shape, and if it returns true erase and delete that shape. I then made the functional version of the Square which checks within the Square as well as the lines themselves.
7. Created the class of Circle which extends Shape, that makes a circle using the radius and midpoint. The algorithm used is an algorithm from javascript(webGL) changed to be functional with this program. I also added a drawCircle function to ChartBuilder which has the same arguments of the constructor minus the SimpleDrawing, which is the command this is used instead. The Circle is created then added to the list. checkPoint checks a box around the Circle.
8. Create three new constructors that allow for changing the size of the grid as well as change the foreground color and background color. This allows for the user to easily create new charts of whatever type but also keeps the simple no input for the standard one with 600 by 600 with a gray background and black foreground.
9. Created the class of Diamond, which extends Shape. Uses 4 lines to create the shape using the middle point, width, height. The constructor calculates the points then draws the shape automatically. I also added a drawDiamond function to ChartBuilder which

has the same arguments of the constructor minus the SimpleDrawing, which is the command this is used for it. The Diamond is created then added to the list. checkPoint checks a box around the Diamond.

10. Created the class of Triangle, which extends Shape. Uses 3 lines to create the shape using the point of point in the direction it is pointing, width, height, and string stating which side it is facing. The constructor calculates the points depending on the direction stated then draws the shape automatically. I also added a drawTriangle function to ChartBuilder which has the same arguments of the constructor minus the SimpleDrawing, which is the command this is used for it. The Triangle is created then added to the list. checkPoint checks a box around the Triangle depending on its type.
11. Created the class of Parallelogram, which extends Shape. Uses 4 lines to create the shape using the point of the top left corner, width, height, a shift to say how far from a standard square it is, and the type of shift(If horizontal or vertical). The constructor calculates the points depending on the kind of shift and whether the shift is positive or negative then draws the shape automatically. I also added a drawParallelogram function to ChartBuilder which has the same arguments of the constructor minus the SimpleDrawing, which is the command this is used for it. The Parallelogram is created then added to the list. checkPoint checks a box around the Parallelogram depending on its type.
12. Created the class of Arrow, which extends Shape. Uses 3 lines to create the shape using the middle point, width, height. The constructor calls the draw function which calculates the points then draws the shape automatically. I also added a drawArrow function to ChartBuilder which has the same arguments of the constructor minus the SimpleDrawing, which is the command this is used for it. The Diamond is created then added to the list.
13. Created the function redraw in ChartBuilder which goes through the list and redraws each shape in the list. This method is private because it will only be used by the componentResized function.
14. Created the function componentResized which checks for when the window resizing and only calls redraw. Redraw broke because of the program calling it before the list has time to initialize causing a call out of bounds. To fix this I added a private int called count which starts at zero and increases when a shape is drawn but decreases when a shape is erased. I also changed the loop in redraw from a foreach to a for. This fixed the issue and once the new drawing.jar was added, everything worked.
15. Created the checkPoint functions in Arrow and Line. Arrow just checks each the three lines using the lines check. The line checks by going through the algorithm which drew it and instead of turn that point on, looks 10 pixels around it to see if the click was there. I noticed a bug, all of the points were off from my clicks because I used e.getScreenPosition instead of e.getPoint which caused the numbers to be off. Once fixed all of the checkPoint functions worked properly.
16. Lastly created the Test class and tested all of the above(again due everything being tested while being created).