Student: Eduardo de Oliveira Castro
Blazer ID: edc
Date: 02/11/2015

Lab #05

Problem)

Use the AnalogClockIcon and AnalogClockTester classes and uses it to create a real functional analog clock that updates itself in realtime.

Procedures and Solution)

The first demanded part was to draw the hands inside the clock with the actual time of the computer. For this I used the Calendar class to get the real time, calculated the right angle for each one of the three hands(one for seconds, one for minutes and one for hours) using the following functions:

$$\text{secondsHand} = \text{second} * 360/60$$
$$\text{minutesHand} = \text{minutes} * 360/60 + \text{secondsHand}/60$$
$$\text{hoursHand} = \text{hours} * 360/12 + \text{minutesHand}/12$$

We have 60 different seconds for 360 different angles(the total angles in a circle) and that's why we made the division 360/60 and multiply by the second value, that will return the angle that the seconds hand should be. The same principle is used with the minutes hand but it also moves according to the seconds hand, so we need to get this angle and divide by 60(because there are 60 seconds and 60 different angle possibilities) and sum it to the previous angle calculation to discover in which point between two "minutes spots"(there are 12 different ones in a regular clock, like 1 and 2) the hand will be. For the hour we also use the same principle but instead of 60 possibilites we have 12 and we also need to discover in which point between "hour spots" the hand it is, so we take the minutesHand and divide it by 12.

With this numbers, and the coordinates from the line calculated we just need to call the rotatePoint method to have the right end Point2D instance that will be used to create the shape that will be drawn with the beginning Point2D instance. The same is done for all three hands.

With this procedures we have a regular clock but not an animated one, it just shows the clock at that exact time and stops. So now the next step is to animate it, for this the only necessary step was to add the "jLabel.repaint();" line inside the timer anonymous class assigned to the Timer variable with 1000ms defined as the "ticker time", so in every second the code will run that command inside the anonymous class again that it is demanding the JLabel to be recreated and every time it gets the new actual time so we have the animated clock.

Problems)

The first encountered problem was at the provided seconds function that was wrong, at the instructions it says to divide the seconds number by 12 but the right value is 60, since we have 60 different possible angles for the seconds hand.

The second encountered problem was with the Calendar instance, since I was creating it at the beginning of the AnalogClockIcon class it was only taking the time one time, so the ticker method worked but it was always redrawing the clock with the same time, since the algorithm was not calculating it again. After fixing it and moving the creation of the instance for inside the methods that calculates the hours, minutes and seconds the problem was solved and the clock working as supposed.

Conclusion)

This was my first experience working with GUI and drawings with pure Java and it was pretty interesting, I suppose that games uses similar procedures as the ticker to "transform code into life/real game". I have already seen that Android apps uses similar structures.