10-point assignment
For this assignment we will explore a simple version of how animation works. The assignment requires that you utilize both if instructions and inheritance.

To begin download the initial folder. Notice that this program will not use any supplied classes excepting ThreeButtons. (i.e., you are not permitted to use the Line.java or Oval.java classes.)

The central work of this assignment consists of implementing a class called Wheel. Your Wheel class must inherit JComponent and override the paint() method in order to draw a wheel-like image filling the JComponent's bounding rectangle. Each Wheel object appears as follows:

1) It has a colored rim (determined by calls to setBackground()) that is 3 pixels thick.
2) It has seven light gray, 3 pixel thick, spokes that are uniformly spaced around the rim and forming a 7-pointed star. (See image on the right.)

If written properly, the only methods you need include in your Wheel class are the paint() method, a roll() method and the constructor method. You can rely heavily on inherited methods. Here are some hints for coding the paint() method:

The following instruction should be included in paint() in order to set all subsequent drawLine and drawOval methods to draw with a line thickness of 3 pixels. Note that this assumes the Graphics parameter is called g:

```java
((Graphics2D)g).setStroke(new BasicStroke(3, BasicStroke.CAP_BUTT, BasicStroke.JOIN_BEVEL));
```

Drawing lines is best done by using the line's angle, call it a. (Java presumes that 0 degrees points directly to the right and angles increase in a counterclockwise direction.) In order to calculate the X and Y coordinates of any point on a circle you can use the following formulas that assume an angle of a and a circle radius of r:

```java
X == cos(a) * r + r
Y == sin(a) * r + r
```

Recall that the Math class supplies cosine and sine functions. Also recall that these methods work in radian measure, so you will want to call Math.toRadians(a) in order to convert an angle, a, from degree measure into radian measure.

The roll() method is supposed to make the wheel object appear to roll by some number of degrees. Rolling involves causing all lines to be drawn properly for the new angle and also relocating the whole wheel in the direction of travel. In order to move the wheel by the proper distance you should remember how to calculate the circumference of a circle; the distance of travel is given by the circumference / (a/360) assuming the wheel rolls by a degrees.
Of course, you will still need to write a Driver class. HINT: It is better (and easier) to write as much of the behavior as possible in the Wheel class and just call its methods from Driver.) As the program begins executing all that is visible is a ThreeButtonFrame colored black. The three buttons must function as follows:

**LEFT**
The first time that LEFT is clicked it must create a red Wheel that is 100 pixels in diameter and located in the upper left corner of the window. The second click of the LEFT button creates a green Wheel that is 200 pixels in diameter and located in the upper left corner of the window. The third click creates a blue Wheel that is 300 pixels in diameter and located in the upper left corner of the window. Clicks of this button after the third do nothing.

**MID**
Clicking the MID button at a time after the red Wheel has been created causes the red Wheel to increase its diameter by 10%. If the red Wheel has not yet been created and the MID button is clicked, then nothing should occur (no error messages).

**RIGHT**
Clicking the RIGHT button causes all three Wheels to roll to the right along the top of the window. (Any Wheel that has not yet been created cannot roll and your code must not produce any error messages if one or more Wheels have yet to be created.) The amount of roll will vary for each Wheel; this degree amount is fixed for each Wheel at the time it is instantiated and should be selected randomly from 2 through 10 degrees of rotation. Also, when any Wheel rolls off the window (is totally invisible), then it is replaced in the upper left corner of the window. You might want to experiment with the sample programs to see precisely how this rolling behaves.

**for three additional points...**
Modify your program so that for the rolling so instead of simply rolling across the top of the window that it rolls around the window. (i.e., when a Wheel reaches a corner of the window it begins to roll down/up/across the next border.) There is a posted example for this behavior.

**to submit your solution...**
Email your solution in the form of an attachment to riley@cs.uwlax.edu. Please compress the entire folder into a single zip file and attach this zip file to the email.

**Due date:** Nov. 12, 2010