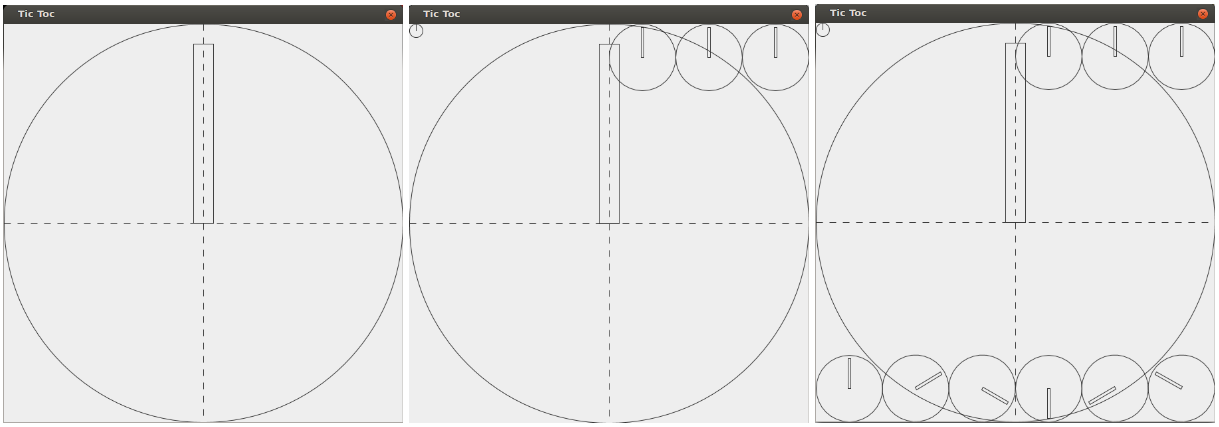

Stage 1 (left). Stage 2 (center). Stage 3 (right). Note that the dotted lines are just for reference and are drawn for your by HourTimerComponent.

HourTimer

Read over all these instructions carefully. Make sure you understand completely what functionality you have to implement before you start coding. Ask if any part of the instructions are unclear.

Implement the HourTimer class. The HourTimer demonstrates a circular timer that can represent any number of minutes (like a clock with only a minute hand).

A few details about how to draw the HourTimer:

- The minute hand's length is 0.9 times the radius of the HourTimer (RATIO_OF_HAND_LENGTH_TO_RADIUS).
- The minute hand's width is 0.1 times the radius of the HourTimer (RATIO_OF_HAND_WIDTH_TO_RADIUS).
- Initially the HourTimer should have it's minute hand pointing vertically upward (representing 0 minutes).

Stage 1 The HourTimer should be able to be constructed with no parameters. In that case it is to be drawn centered at the point 300,300 (DEFAULT_CENTER_X and DEFAULT_CENTER_Y). Its default radius is 300 (DEFAULT_RADIUS).

Stage 2 You'll need to uncomment the stage 2 code in HourTimerComponent.

Add a 3 parameter constructor. Use the example Stage 2 code in HourTimerComponent to infer what the parameters ought to be. When you're finished, the clocks should be able to be drawn in different places and at different sizes.

Stage 3 You'll need to uncomment the stage 3 code in HourTimerComponent.

Finally, add rotation. Implement the setTime function that takes a number of minutes as a parameter. When this is set the minute hand should be drawn rotated to the appropriate number of minutes, like the minute hand of a clock (e.g. 30 causes the hand to be drawn vertically downward, 45 causes the hand to be drawn horizontally to the left, etc.).