SESSION 5 CODE TASK SET

TASK SET 1

The Overall Goal of this task set is to create a background for the paper, and then a circle that can be dragged around the canvas. The specific tasks for you to accomplish starting from the template code are:

1. Create a variable for a Raphael rectangle to fill the paper to use as a background; set it's fill attribute.

2. Create a variable for a Raphael circle at the center of the canvas. Set some attributes.

3. Use 'mousedown', 'mousemove', and 'mouseup' events so the user can drag the circle around the canvas.

- Hint: You'll need a variable to remember the state of the mouse.

TASK SET 2

The Overall Goal of this task set is to Click canvas to draw circles with an attribute set by a slider, and to add a way to clear the canvas of the artwork. Specifically:

1. Create a variable called circleRadius, that we'll use to size circles we draw. Initialize it to a value,
2. Draw circles on the paper where mouse clicks are made. (The background rectangle will come in handy here.) Use circleRadius to set the size of the circle when you create them in your callback function.

3. Add a slider to the "<aside>" element in the .html file,

4. Set slider range between [0,1] (so it can be used for anything with scaling)

5. Create a label (using the <label> element) for the slider, styled to look reasonable.

6. Use the slider to change the circleRadius value. You are already using the variable to set the size of new circles, so just changing the value of the variable should affect how future circles are drawn. (Since the slider ranges from [0,1], you'll have to map the slider values to the circle sizes you want. By "map" I mean perform a numerical operation to transform the range of numbers you get from the slider into the range you want your circles to be.)

7. Add a button to the "<aside>" element in the .html file,

8. Use the clear button to remove elements from the paper (calling Raphael's paper.clear())

9. Restore the background and circle elements to the paper after clearing everything, using the paper.put(element) method I provided for you at the top of the template.
(note: you still have the elements available if you named then using a var or let statement so you can pass them to the paper.put method.)

TASK SET 3

The Overall Goal of this task set is to draw a line between each new circle created by user clicks. Specifically:

1. We need to remember the location of the last circle (click) if we want to join a line from it to the new click location. So create a variable named ‘sketchManager’ to remember the position of the last circle drawn. Since a coordinate has both an x and a y value, make the variable an object with an ‘oldX’ and an ‘oldY’ property.

2. Create a drawing method (function) named ‘connect’ as a member of sketchMan that takes the new x and y location where the user clicked as arguments, and draws a line from the last point to the new point. Don’t forget to call the method (where do you want to do that?)

    - Notice that we could have defined ‘connect’ as a function outside of the sketchMan object. It would have looked the same. Are there any advantages to having it be on sketchMan?

3. What happens if you clear your drawing and start again? Can you explain what is happening? Fix it!