H05_SmileFrownRaphel

Scalable Vector Graphics (SVG) with Raphael

Learning outcomes:

a. Using state variables to remember information between calls to callback functions.

b. SVG graphics

1. paths
2. Bezier curves
3. event listeners
4. function with arguments
5. animation (with Raphael library)

Preparatory Remarks:

- Make a new directory in your class projects directory for this homework assignment.
- Download the starting template code (SmileFrownRaphael) from the session resources.
- This is a bare-bones template, but it already includes the raphael library and sets up the Raphael paper for you.
- You will save yourself LOTS of time if you learn what a Bezier curve is before starting in on coding anything. You don't need to do the math, but understand it **well** conceptually and graphically. It is a core concepts in computer graphics (and really not difficult once you see it. Google for explanations and interactive demos.

Assignment Instructions

1. Create a background rectangle using Raphael with the dimensions of the canvas already in the dimX and dimY variables in main.js
2. Draw a smile using a 'Bezier' curve (using the 'Q' command in the path string - search for making Bezier curves! - The path commands (M, L, Z, Q, etc) are the same in Raphael as they are in raw SVG). Do this by creating a "mouth" variable and initializing it to a path from Raphael.
3. Create a toggle button on the html page that visually indicates what state it is in by changing the text displayed on it (e.g. changes between "up" and "down", or "smiling" and "not smiling") and by switching between two images, as well.
   a. Hint: you will have to use a "state" variable outside the button's listener callback function to remember the button's state between event callbacks.
4. Use the toggle button to toggle the mouth between a smile and a frown. In your button 'click' listener, check the button's state with an 'if' statement to control whether you are going to make a smile or a frown.
5. Create a drawMouth function that takes 2 arguments, bx and by, and creates the mouth path string by combining the values passed in with the values you are using for the endpoints (or corners) of your mouth. Call the drawMouth function in your button event listener.
6. Animate the mouth drawing between smiles and frowns so the mouth "morphs" slowly between expressions using Raphael's animate method. Add a 3rd parameter to drawMouth and use it to determine the duration of the animation.
7. Draw eyes using ellipses.
8. Animate the eyes so they are "closed" when the mouth is frowning and open when the mouth is smiling.

. Hint: use the ellipse attribute ry (vertical radius) to do this.

9. Draw a small dot at the bezier point that determines the shape of the mouth (the 'Q' coordinate in your path). It should animate along with the mouth in drawMouth.
10. Make the dot draggable
    a) hint: use listeners for mousedown, mouseup, and mousemove events. Create and set a "state" variable (eg 'draggingDot') outside of your listener callback

functions in order to keep track of the mouse state between mousemove callbacks

  b) another hint: listen for mousedown on the dot, but mouseup and mousemove anywhere on the div where the paper lies - this will make sure you don’t “lose” the dot if you crag your mouse too quickly.

  c) In the listener for mouse moves, use an ‘if’ statement to check the state of your ‘dragging’ variable.

11. As the dot is dragged, redraw the mouth using the dot position as the Bezier point (an only move the dot and redraw the mouth if you are in the “dragging” state).

Give yourself some time for this assignment, and also have fun and experiment a bit as you do it!