IN-CLASS ACTIVITY 5:

Version #1: ***(Start file: example.js)*** Your task is make the sprite walk across the canvas from right to left. We will start the man off-canvas at position canvas.width+20 and walk to position -35 with the fixed horizontal position (of the top of the sprite) at the canvas.height -115.

a) Start by adding the sprite by setting the sprite source to './running-sprite-sheet.png'.
b) Create an object assigned the variable sprite with three attributes: posX, posY and imagePos. The first two attribute values have already been described and the last one is the index of the image to show – start this at 0.
c) Your next step is to call a function when the sprite image has finished loading using the onload event. This fuction will:
    i) Call the clearRect() method with coordinates from the upper left corner of the canvas to the lower right corner (clear the entire canvas).
    ii) Draw the sprite for the current position. This will be done by calling the drawImage function of the context as using the following parameters:

        context.drawImage(sx, sy, sw, sh, dx, dy, dw, dh)

        where: s stands for source,
        d for draw,
        x & y for coordinates,
        w & h for width and height
    iii) Create an interval by calling setInterval(function, duration) where function is an anonymous function and duration is the number of milliseconds for each frame. The anonymous function will do the following if the animation is not paused:
        i. Draw the current sprite for the position (copy code from earlier)
        ii. Update the sprite position by subtracting the xVelocity from the sprite's xPos value and if that is less than -35 set it back to canvas.width+20. Also increment the imagePos until it exceeds 8 and then reset it back to 0.
    iv) Finally, create a mouseDown event attached to the canvas which toggles the paused variable (true to false or false to true), call the preventDefault function and the stopPropagation function.

Test the program by running example.html and clicking on the canvas. Hopefully you see the little guy walking across the screen.

Version #2: Now we would like to add a ball moving from left to right which the little man will walk forward on – but will end up moving backwards. To accomplish this:

a) Create a circle object under your sprite object. This object will have the following attributes:
   i. x: start at 40
   ii. y: start at canvas.height-31
   iii. radius: set to 30
   iv. colour: set to 'rgba(red, green, blue, 1.0) where red, green, and blue are random integers from 0-255.

b) Start the sprite object attribute xPos at 20. Also, change adjustment to xPos so that we increase xPos by xVelocity and when this value exceeds canvas.width+20, then set it to negative 35.

c) We want to add a background of horizontal lines. We will accomplish this by calling the following function:

```
function drawBackground(context, color, stepy) {
   context.strokeStyle = color;
   context.lineWidth = 0.5;
   i = context.canvas.height;

   while(i > stepy*4) {
      context.beginPath();

      context.moveTo(0, i);
      context.lineTo(context.canvas.width, i);
      context.stroke();

      i -= stepy;
   }
}
```

   Call the function with the colour of "lightgrey" and a step value of 12.

d) Make a copy of the current canvas using the context.getImageData() method. Then change the clearRect() method in the setInterval function parameter to use the putImageData() method to restore the background before drawing the circle and sprite.

e) Right after restoring the background, draw the circle as follows:

```
context.beginPath();
context.arc(circle.x, circle.y, circle.radius, 0, Math.PI*2,
      false);
context.fillStyle = circle.color;
context.fill();
```

f) Finally, adjust the position of the circle after drawing it by adding xVelocity to the circle.x property. Once this property is larger than canvas.width+40, then reset the value to negative 15.

Test this revised version and you should see the little sprite man walking forward but moving backwards on a ball.