# *Web Programming II (420-C20-HR)*

# *Lab 10 – Animation*

Date assigned: Monday, April 11, 2016

Date due: **Monday, April 11, 2016, 11:00 a.m.**

**Learning Objectives**

Upon successful completion of this lab exercise, the student will be able to:

* Perform simple animations with JavaScript

Lab Set-Up

1. Copy the folder C20\_L10 from Moodle and unzip it to your H:\420-C20\Labs folder. Rename the folder to **YourUserName\_C20\_L10** in your H:\420-C20\Labs folder.
2. All formatting and styling must be done using HTML and CSS based on HTML5. All CSS formatting characteristics must be kept in a separate file in the appropriate folder of your website.
3. For all the web pages you create, ensure that you give each page a title and provide an appropriate header and some formatting. Don’t spend a lot of time, but spend some time making the page look nice with background, font and other CSS formatting.

To do:

**Part A: Using setInterval**

1. In the images\concerts subfolder are 5 images numbered concert1.jpg through concert5.jpg.
   1. Create a web page called parta.html with an image of concert1.jpg displayed with the proper attributes for accessibility and the id theConcert.
   2. Add a JavaScript file called parta.js. Set a global variable called currPoster with the value 1. Then use setInterval to call a function called changeImage every 5 seconds (5000 milliseconds)
   3. In the function changeImage, check the value of currPoster. If it is 5, reset it to 1; otherwise, add one to currPoster.
   4. Update the current image to the image “concert” + currPoster + “.jpg”
2. In the end your file should cycle through the images concert1.jpg through concert5.jpg. Test it to make sure you are cycling through properly; common mistakes are to display the first image for 10 seconds (instead of 5) or miss the last image (only display 4).

**Part B: First Animation**

1. In the images\jackhammer subfolder of the folder there are 11 images numbered jackhammer0 through jackhammer11.
   1. Create a web page called partb.html with an image of jackhammer0 displayed with the proper attributes for accessibility and the id theImage
   2. Add a JavaScript file called partb.js. Create a function called setUp which uses a for loop to read the images into a global array of Image objects. The function also sets a global variable called currImg set to 0.
   3. At the end of the setUp function use setInterval to call the function startBouncing every 90 milliseconds.
   4. Create the function startBouncing which checks the global variable currImg. If currImg = 11 then it is reset to 0; otherwise, add one to currImg.
   5. After checking the currImg, change the image on the web page to be the image in the array at the offset currImg.
2. In the end you should see a man using a jackhammer moving on the screen.

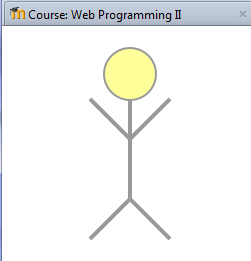
**Part C: Moving Animation**

1. The images\animatedPuppy folder contains an animated image of a puppy. Open the image in a browser to see its animation. Create a webpage called partc1.html that has the puppy run across the screen. To do this, add the animated image to the HTML file. In the <head> section of the HTML file add a style tag which sets the position of the image to relative.
   1. Then add a JavaScript file called partc1.js. In the JavaScript file, set an interval timer to call the function movePup. You may have to try different timings to get the motion of the dog to look okay; try 90 milliseconds to start. You’ll also need a global variable of the current position (currPos).
   2. You can use window.innerWidth to get the width of the browser; and document.docmentElement.style.overflowX = ‘hidden’ to prevent a horizontal scroll bar.
   3. Start the puppy at position -97 pixels (so he’s off the screen)
   4. In movePup, add 10 (you may need to adjust this) to the current position and use the .style.left property of the image element to move the image to that position.
   5. Continue updating currPos until the puppy runs off the right side of the screen (currPos greater than window.innerWidth).
2. Once you have this working the puppy should run across the screen from left to right and start over when he runs off the screen.
3. Now create an html file called partc2.html with a JavaScript file partc2.js. In this case you are going to use the 6 static images in the puppy folder to animate the running puppy. Set it up like you did the jackhammer in the previous part and make sure that the puppy runs properly on the spot. Then use the same algorithm as in step one to move the puppy across the screen. You can use either one or two setInterval timer(s) to both animate and move the puppy.

**Part D: Canvas**

1. Create an html file called partd.html with an attached JavaScript file called partd.js.
2. Add a canvas element with the id myCanvas and a width and height of 500. Add a paragraph between the begin and end tags of the element that says, “Unfortunately your browser does not support the canvas element”.
3. In the JavaScript file, create a function called draw that is called when the file is loaded.
4. In the function get the canvas element into a variable
5. Using the element variable, get the context of the canvas element to be 2d (getContext(‘2d’);) and assign it to a variable called ctx.
6. Set the following context parameters:
   1. fileStyle = “#ff9”
   2. strokeStyle = “#999”
   3. lineWidth = 4
7. Create the ‘head’ of the stick man using the arc method. Do the following (each of the following is a method of the context, so use context.blah to execute the method blah).
   1. Use **beginPath()** to start the path
   2. Create an **arc** with the following parameters: start at position 120, 40 (first two parameters; have a radius of 25 (third parameter); start at 0 radians and finish at 2\*Math.PI radians (fourth and fifth parameters) and go counter-clockwise (set sixth parameter to false)
   3. Use the **stroke** method to draw the circle
   4. Use the **fill** method to fill the circle
   5. Use the **closePath** method to complete the path.

1. At this point, save your html file and run it. There should be a yellowish circle with a grey border on your screen. Once you get this to work, move on to the next step.
2. Draw the body and arms and legs for the stick man. Once again, each method is part of the context method.
   1. Begin the path (beginPath);
   2. Move the pointer to position (120, 65) – use moveTo(120,65)
   3. Draw a line to position (120, 165) – use lineTo(120, 165)
   4. Draw a line to position (160,205)
   5. Move the pointer to position (120, 165)
   6. Draw a line to position (80, 205)
   7. Move the pointer to position (120, 105)
   8. Draw a line to position (80, 65)
   9. Move the pointer to position (120, 105)
   10. Draw a line to position (160, 65)
   11. Use the stroke method to draw the lines
   12. Use the closePath method to complete the path.
3. Note: You can use multiple paths to draw the lines if you want. Open the page in a browser. You should have drawn a stick man which looks like this:



**Part E: Bonus: Canvas Animation**

1. If you have time, try and animate the stick man as if he’s doing jumping jacks. To do this you need to use the clearRect function to “clear” (redraw) the entire canvas and then redraw the image in a new position. This is definitely non-trivial.

**To submit**

When you have completed the lab exercise, create a single zip file called YourUserName\_C20\_L10.zip. The zip file must contain all of the parts of the lab as listed below. Copy the file to the Moodle page for the course.