The first week I felt so overwhelmed, I didn’t know what I was doing. I didn’t even get to my second class until Friday. Friday was so much better than Thursday.

**1/9/17-------------------------**

Today I felt pretty good about working on my school work. It then took 5 hours to finish my first program. It would have gone a lot better if Linux would have renamed my filename right off. I am just getting back to this class after 6:00pm.

CIT261Class

Stand19762

**1/10/17------------------------ *Loops and For Loops***

I am relearning about loops today using JavaScript. I use to know how to do this at Plexus but I and really rusty at it now. Both of these do the same thing. It takes a lot less space and easier to right especially if you have hundreds or thousands of entries. [Loops Example](http://www.w3schools.com/js/tryit.asp?filename=tryjs_loop_for)

|  |  |
| --- | --- |
| text += cars[0] + "<br>";  text += cars[1] + "<br>";  text += cars[2] + "<br>";  text += cars[3] + "<br>";  text += cars[4] + "<br>";  text += cars[5] + "<br>"; | for (i = 0; i < cars.length; i++) {      text += cars[i] + "<br>"; } |

I read to section 4 page 46 in Doing Stuff with WebThings. I learned how to save, edit and retrieve information that a user inputs. I’ve been using the Sublime Text program to write my code and I then test it in Google Chrome.

**1/11/17-------------------------- *Studying with Doing Stuff with WebThings***

Today, I am working on Doing Stuff with WebThings again. I had to troubleshoot some code that was in the textbook that wasn’t working. Using the Chrome Inspect option, I was able to figure out that the Input Box ID didn’t match what was in the function script.

I worked on my fluency letter which I think is coming along well. I got out group set up to use hangouts in out meeting. It is set for Friday at 2:00pm but that could change. Two have not accepted the time yet and one doesn’t appear to have hangouts.

**1/21/17 -------------------------- *YouTube Video Creation***

The 15th through the 21st was completely devoted to learning how to do videos on YouTube and trying to keep up in CS 124. These videos are taking about 1 full day to create. I am devoting every other day to CIT 261. Saturday, everyone showed for our group meeting. We had a good discussion. I attempted recording our meeting without being asked, but it did not record. I found we have to add people in YouTube not hangouts. During the group meeting we discussed we need to start doing it. I think it was Justin that said he would do it the next week.

**1/27/17 --------------------------**

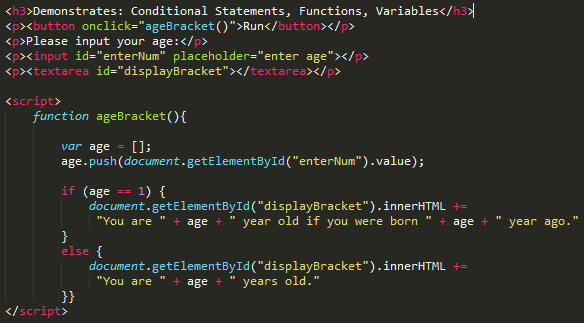
I finished my videos on Monday so I spent this entire week trying to catch up with CS 124. We have a Unit test that I am studying hard for and I am very worried about it. I have scheduled a meeting with a tutor and teacher. There are some very difficult things to understand. I have been keeping daily contact with my group. I really want to get back to coding in our class.

**1/30/17 -------------------------- *Using If Statements, Var, Arrays and Functions***

**Conditional IF Statement, Functions and Variables**

I was finally able to work on my age bracket code. I simplified it down and I was able to create it without any other help. I actually had some “document.get” statements that were sitting around unneeded. My code demonstrates the use of conditional if statements, variables, array and functions. I feel I am understanding if statements and functions, variables, if statements, arrays and parameters a lot better now. I am planning on requesting assessment tomorrow.

Figure 1: Conditional IF, Function, Variables



**2/1/17 --------------------------- *Saving and Loading data***

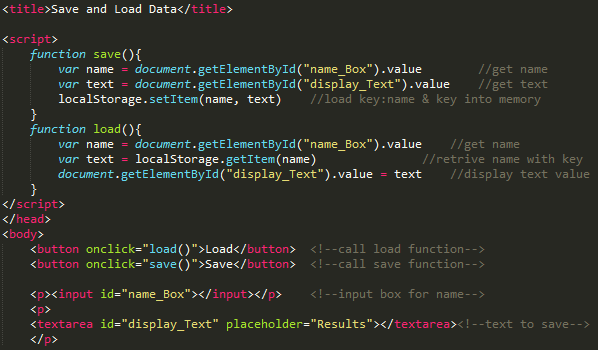
In learning about saving and loading data from memory and attempting to write it myself, I could not get the code to work I was writing. After checking one thing at a time, I discovered that the “d” was in caps in “getElementByI**D**”. It really counts to write things correctly the first time, because errors are very hard to find and takes time to find.

Note: I want to point out that it is important to remember that *name* is actually a key to find the text you will be saving and loading.

This program entitled “Save and Load Data” contains two functions which are initiated by two buttons “Load” and “Save”. The following is an explanation on the workings of both functions.

1. **Save Function:**
   1. It takes the value of the input *name\_Box* and adds it to the *name* variable.
   2. It then gets the value of textarea *display\_Text* and adds it to the *text* variable.
   3. localStorage.setItem then loads the key *name* and *text* value into memory.
2. **Load Function:**
   1. Takes the value of the input *name\_Box* and adds it to the *name* variable.
   2. localStorage.getItem uses the key *name* and put its associated text in variable *text.*
   3. It then displays the value of the *text* variable in textarea *display\_Text*.

Figure 2: localStorage



**2/8/17 ------------------------- *Created InsertBefore and Remove Child – Updated Add Child***

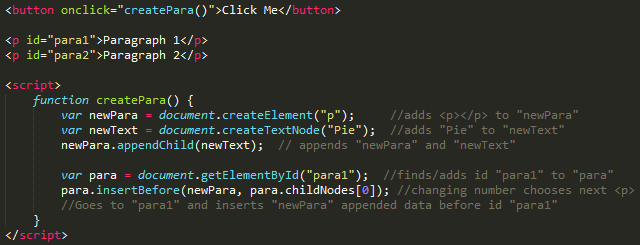
Today, I created and learned about InsertBefore and Remove Child and added them to GitHub and Webpage.

*Plan:* I am wanting to create a web Journal if I ever get enough time to do so. Creating these codes and understanding them takes a lot of time. I also want to play around with the code but not much time for that either.

**insertBefore**

In figure 3 below we first create a new <p> tag and add it to variable “newPara”. Next, we create text “Pie” and add that to the variable “newText”. We will then need to append or combine the data together into “newPara”. This data now consists of the <p> tag and the text. We then locate the ID “para1” and insert “newPara” before the (para.childnode) number 0. We can increment this number and so doing choose which <p> tag to enter the new tag before.

Figure 3: iInsertBefore

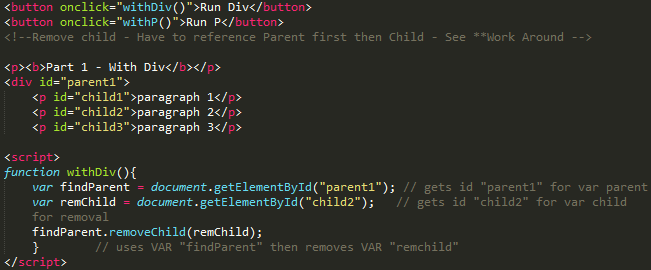


**Remove Child**

I have included two parts for the Remove Child code. In order to find the child <p> for removal it is pertinent to reference the parent <div> first. The 2nd part is a work around for this.

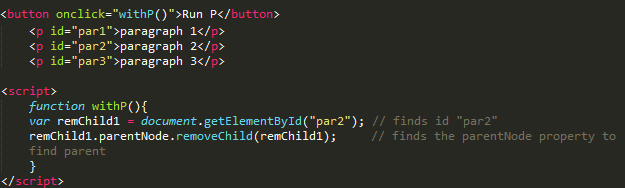
**Part 1 –** The code first of all locates the ID “parent1” which is a <div> tag and adds it to the variable “findParent”. Next, we find the ID “child2” which is a <p> tag and adds it to variable “remChild”. It then uses the findParent variable to locate the child within which is defined in the “remChild” variable. It then initiates the “removeChild” directive, removing “child2”.

Figure 4: Part1 - Remove Child



**Part 2 –** This code doesn’t need a <div> or parent to function. We still have to reference a parent though for it to work. We first need to locate “par2” and add it to a variable “remChild”. We then reference that same location with the “remChild1” variable, and reference the parentNode. We can then remove “par2” within the “remChild1” variable.

Figure 5: Part 2 – Remove Child \*Work Around



Other Tasks:

I also added some functions and buttons to several of my different codes to make them a little more interactive for the user.

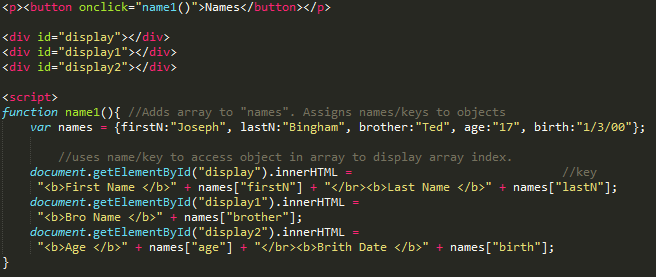
**2/8/17 ------------------------- Arrays & Objects and JSON**

**Arrays & Objects**

In the following code, we can use names to access each index instead of numbers.

The code reads as follows. We first have our Array “Name1”. It consists of 5 indexes in which each index has a name associated with it (e.g. firstN:”Joseph”). We can then access each index by name/key word. This is very similar to an associative array in which we are required to use the “document.write” element. Here we can use our “document.getElementById” to output our code to any tag of our choosing. We access an index by stating the array[“indexname”].

Figure 6 – Array & Object



**JSON**

JSON stringify is used to stringify or convert all data sent through it into a string. A string is just as it sounds. It is a string of numbers sent to storage for later access. When we do want to use the data again we will need to parse it. Parsing using JSON turns it back into the format before it was stringified.

Figure 7 - JSON



Referring to Figure 7 above, I am using my save and load code which uses the localStorage property. In our “function store”, we will use “JSON.stringify” to turn our array into a string and save it to our local storage in memory. In our “function load”, we will retrieve the data from storage and use “JSON.parse” to format the data back into a form we can use. We then display it to the screen.

**2/11/17 ------------------------- Group Meeting & Stylesheets**

Today, we had our group meeting. Everyone showed up which was nice. I re-learned something that I had forgotten when I worked for Plexus. We were talking about how to use a stylesheet located in a file. I worked a lot with these working with Plexus but it has been a long time. I instituted this into my webpage. Justin also showed us how to do it with PHP but it looks a bit foreign to me right now. The only thing I have done with PHP up to this point is submitting form data to the server.

**2/16/17 ------------------------- Object Creation containing properties and Methods**

This day, I watched Justin’s video about Object creation and assigning properties to that object using “this.” property keys. The Object itself is created with a “function name ()”. Within this same function () we create another function and create a variable to combine all of the properties.

To use this information, we will need to initialize home () by assigning it to a variable “johnsHome” We can then run the function by calling that function with the new variable “johnsHome” as demonstrated in Figure 8 below.

Figure 8 – Object Creation, Properties and Methods

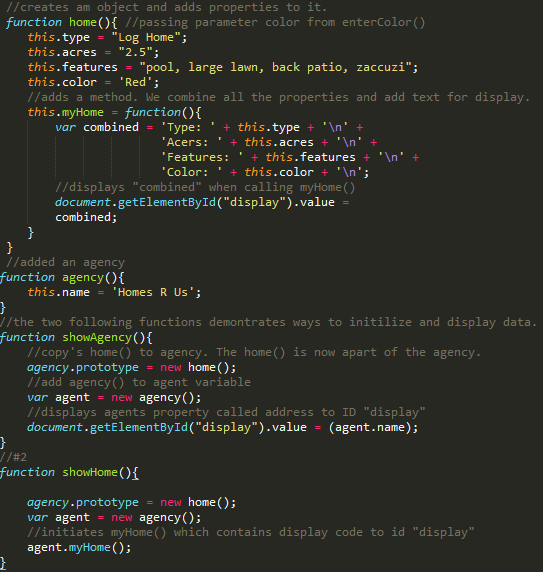


**2/16/17 ------------------------- Object Creation – Inheriting Objects and Properties.**

I again watched another one of Justin’s video’s, this time learning about Object inheritance. Everything thing is the same as in the previous code except we are adding another object for an Agency.

We will demonstrate this by creating a function called agency () and give it properties just like we did for the home (). Let’s say that this Agency acquired the home. It is now part of the agency’s assets. In order to copy/use the data from the first function home (). We will use a prototype. The agency can then access the data of the home ().

Figure 9 – Object Inheritance



**2/24/17 ------------------------- Group Meeting**

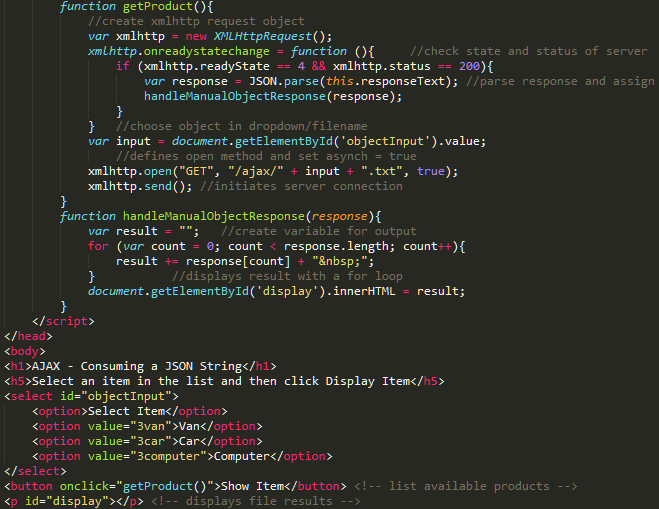
I was really sick this weekend and could not talk at all in our group meeting. I instead typed to everyone.

**3/1/17 ------------------------- AJAX – Consuming a JSON with AJAX**

AJAX makes it possible to communicate with the server and interact with files.

In figure 9 below, we open a connection with AJAX by first checking on the state of the server and define what will happen to the data. We then assign it to a variable. We then define the open method and initiate the send procedure. To display the data from file. We first create a variable to contain the result and with a for loop we capture one character at a time and send it to our display paragraph tag.

Figure 9 – AJAX Consuming a JSON



**3/04/17 ------------------------- Group Meeting**

I was feeling a lot better this week. We had some good discussions with code. I got help with AJAX. I and others have not been able to get the XML Response part of AJAX to work. We discussed it and tried to get it working but failed. Kailey put a lot of effort into it as well but she couldn’t get it to work on her servers either. We think the server may not be setup for that part. At least we all have the first AJAX working.

**3/08/17 ------------------------- AJAX – XML Response**

As stated in the group meeting, I could never get the second part of AJAX working. I spent days on it with no fruition. I have decided to move onto more pressing code and matters. It is becoming increasingly difficult to find time to work on code. Between my wife starting a new business and the projects they entail and my other class, it has been very difficult.

We only have 3 weeks left and I am determined to put forth all the effort I can.

I don’t fully understand what is happening with the code for the XML Response in Figure 10. This is because a lot of it is in PHP. I will have to spend more time on it after this semester, where I will have more time.

Figure 10 – AJAX XML Response



**3/11/17 ------------------------- Group Meeting**

Today in our group meeting we have talked about our group project. We have decided to go with Makrams’ project. I think there is a lot less to do with his and things will be a lot simpler for the 2 – 3 weeks we have.

I will be working hard to finish my code topics by mid next week. There is just so much of other things going on. I never give up though and will push through to the end.

Probably the biggest obstacle I am having is learning two different programs at the same time. It is hard to keep them separate. It’s like having to change modes. I get into one program and really start getting it, just to switch gears.

Anyways, here is the project that I put forward. There are many sites where you can search for lyrics in a song or song title. There isn’t any where you can hum a tune and it then finds that song. It is probably a bit in-depth in this class which is why I chose Makrams’. There are some aspects I thought a lot about, such as, simplifying the hum and the music being searched so to lower incorrect results.

I am requesting assessment today. I have created 4 or 5 new code topics today in which I will be adding while assessment is happening.

**3/11/17 ------------------------- CSS Manipulation, CSS Transitions and Animations**

## CSS Transition

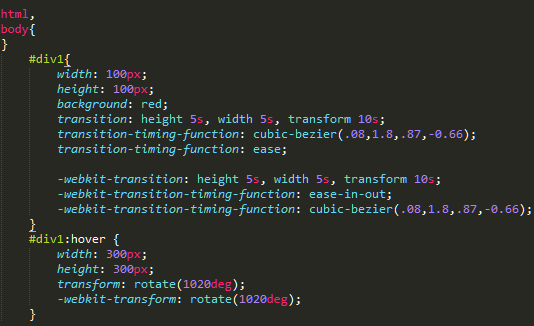
For the CSS Transition code, we are using two different files. In figure 10 below, the div is located within an HTML file I called 2cssTrans.html. In figure 10.5, I placed in a CSS file called instyleTran.css. We can refer to #div1 in our CSS file by using this; <link rel="stylesheet" type="text/css" href="/css/instyleTran.css" media="screen">.

Using transitions, we can create a shape, such as a square. We can then use a hover tag (#div1:hover) if we want to do things like enlarge and rotate the shape. In the #div1, we can change transition effects for timing and display characteristics.

Figure 10 - CSS Transition: HTML File

**C:\Users\johnb\AppData\Local\Microsoft\Windows\INetCache\Content.Word\temp1.png**

Figure 10.5 – CSS Transition: CSS File

****

## CSS Animation

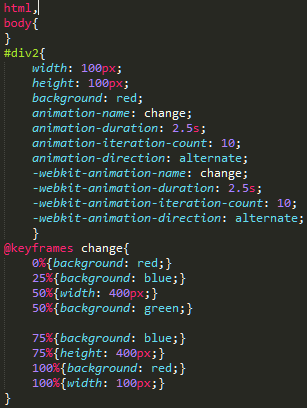
This code uses the animation function. We can define a shape and then by using the different animation effects. We can change aspects such as duration, size, motion and other neat features. We are also starting with a HTML file I called 2cssAnim.html just as we did with the CSS transition. We also put the style information in a CSS file called instyleAnim.css. Also, we need to link to the CSS file by; <link rel="stylesheet" type="text/css" href="/css/instyleAnim.css" media="screen">.

In figure 11.5. I tried to use a hover as in transition but it didn't work. I will have to look into that later. Instead in the CSS file we need to use @keyframes and then the name of the effect.

Figure 11 – CSS Animation: HTML File

## C:\Users\johnb\AppData\Local\Microsoft\Windows\INetCache\Content.Word\temp1.png

Figure 11.5 – CSS Animation: CSS File



**3/15/17 ------------------------- Updating all code**

Today I am updating all of the code and buttons on my webpage so my instructor, brother Klein, can go through my code a lot easier. I am basically matching the buttons and titles to the code topics of the week. I am also adding explanation to some of the code where it was lacking. Also, adding some functionality to some of the topics as well.

## CSS Manipulation – Direct Tag Change

There are two different ways I want to demonstrate CSS Manipulation. We can change tag styles directly or we can create a style object in the header and append an element with it using JavaScript.

Figure 12 shows how this is done changing the tag styles directly.

Figure 12 – CSS Manipulation: HTML - Part1

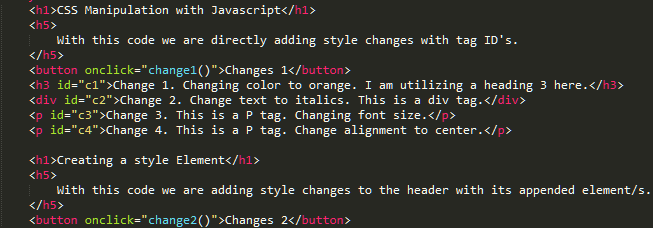
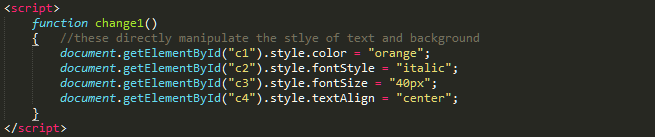


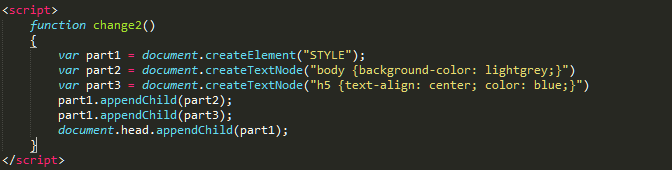
Figure 12.4 – CSS Manipulation: JavaScript - Part2



## CSS Manipulation – Change by Header Style

Using the same html code in figure 12 above we are changing the format of tag h5 and the background of the entire page. We are centering and turning blue all <h5> tags and turning the page background lightgrey.

Figure 13.8 – CSS Manipulation: Style Change by Header - Part3



**3/13/17 - 3/15/17 ------------------------- Group up with Makram**

Since the 13th, Makram and I became a group. He came up with a Pizza menu app. I have been helping him create pages for it and code. We were pretty much booted from our other group, because I think they thought we didn’t have enough time to work on the project. They actually chose Makrams app idea in our group weekly meeting.

After the meeting, Justin was thinking that Kailey’s idea might be easier. I said that is fine with me but we should vote on it. I didn’t hear anything else about it. They decided to go on without me because I just requested my last assessment on Saturday and I had a couple more pieces of code to right.

The next thing I knew, Makram asked if I wanted to group with him. I was surprised he wasn’t with the group still. I didn’t ask questions. We are now working on our Pizza app together. I am excited to work with Makram on the app. I wish things worked out better with the others in our group. I am always open to learning new things and I think we will be able to learn some new things together.

**3/16/17 ------------------------- Event Triggering**

There are several different ways to trigger when an event or action is started on a webpage. They are all pretty explanatory. I will be pasting them all here.

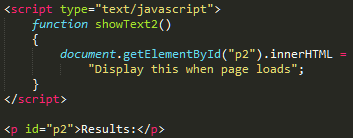
## onload Trigger

The first is the onload trigger. It will run whatever code when the page loads, in this case the “showText2()” function. You add the onload trigger to the body tag.

Figure 14 – onload trigger

C:\Users\johnb\AppData\Local\Microsoft\Windows\INetCache\Content.Word\temp1.png

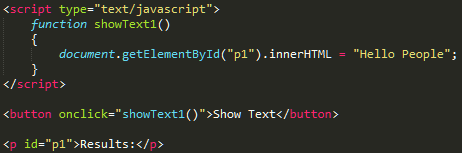
Figure 14.5 – onload trigger: function



## onclick Trigger

This function will run when the button is clicked on. This is very common with a button.

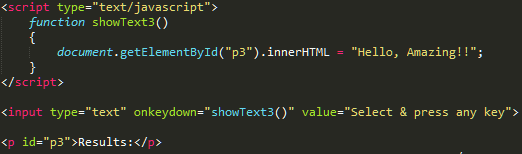
Figure 15 – onclick trigger



## onkeydown Trigger

This function will run when a key is pressed. You will need to select the input box and then press anykey.

Figure 16 – onkeydown trigger



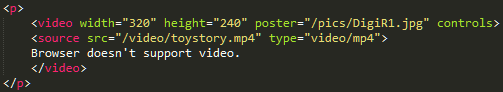
**3/16/17 ------------------------- HTML5 Video, Audio & Canvas**

Video is placed with just two tags. The video tag is where we define the image size. There are few options we can enter. We can list it as autorun, which causes it to run when the page opens and you can add a poster if wanted. A poster is an image that appears when your waiting for the video to load or before you press play.

## HTML5 Video Tag

The second is the source tag. This is where you put in the filename and location. You also need to specify the video format. Here I am just declaring the mp4 format but there are others such as .ogg and others. Mp4 is most widely format used among browsers. This process just uses HTML when JavaScript is not needed.

Figure 17 – HTML5 Video - HTML

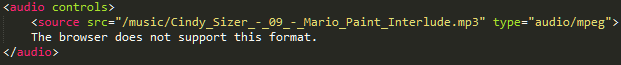
****

## HTML5 Audio Tag

Audio also uses 2 tags to play audio or music. With the audio tag, you can also choose to show controls or have it automatically load when page loads. To initiate by autoplay just replace “controls” with “autoplay”.

The second tag is the same as with video. This is where you give the source of your file and format.

Figure 18 – Audio



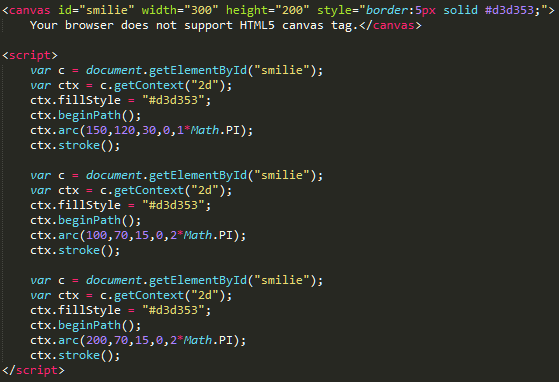
# **Canvas**

With the canvas utility/method. It places a customizable rectangle or square on the page. In this area, you can draw arcs, circles, lines, gradients, display drawings and many other items. I looked up 3d canvas and there are some very cool things you can do with 3 points. You can also add motion and other aspects. We will only be going over 2d context in these examples.

## Canvas: Circles/Arcs

In this canvas, it uses the .arc method. We can draw a circle or arc with any dimension. This draws a smiley face or two circles and an arc. I attempted to turn the arc and circles yellow as well, but it is not working.

Figure 19 – Canvas Circles



## Canvas: Gradient

This canvas again creates an area we can work in. Here I created a gradient with the colors of blue and red 300 pixels wide. Each of the numbers in purple control a different aspect. I added a note of create that controls the different parts of the gradient color position. The “fillRect” controls the size of the gradient.

Figure 20 – Canvas: Gradient

## C:\Users\johnb\AppData\Local\Microsoft\Windows\INetCache\Content.Word\temp1.png

## Canvas: Text

We can also output text within a canvas area. Just as in all the others we can choose the size of the canvas. Here we can choose the size and font of the text and how the text is shown.

Figure 21: Canvas: Text

## C:\Users\johnb\AppData\Local\Microsoft\Windows\INetCache\Content.Word\temp1.png

## Canvas: Draw

I was thinking we should be able to draw with Canvas draw but it is used to display an image in the canvas. I will look into this more. You would think it would be called Canvas Image.

We first source the image, and then we put it into the variable img. When we click the button “Draw” it will display the 2d image on the screen above the button.

Figure 22 – Canvas: Draw



End -----------------------------------