Week 4

Week 3 Review

- What are the various techniques to associate an event handler with an event trigger and why would you use each?
- You have a function defined as follows:
  `calc = (a, b) => a % b;`
  Call the function such that the returned value is 2
- Given the code:
  `a = new Array(3,4,5); a[3]=0; a.push(17);`
  What is displayed: `console.log(a.join('-'))`

Callback Functions

- A callback function is used frequently for asynchronous operations
- The operation – which could take some time – executes the callback function when it is complete.
- Example- setTimeout: setTimeout takes a callback as a parameter to indicate what happens when the timer goes off.

```
setTimeout(hey, 2000)
function hey()
{
     alert('hey')
}
```

Recipe: Create a "slide show"

- Create a page with a div called slider that contains a background-image
  - Size the div at 700 x 500px
  - Position the background to cover the div
- Get three images that will work in that space. Store the files in the same folder.
  - Create an array with the names of the three images
- Create a window.onload event to set a global variable called currentSlide to zero (0). It should also call a function that will use load the current slide as the div background. It should then increment the currentSlide (wrap around if it gets past the number of images) and call setTimeout to set a 3 second timer. The function itself will be the callback for the timer.
- Add a button to stop the slide show. Hint: use clearTimeout()

A few mind-blowing concepts!

Interesting things we can do with arrays!

- forEach
- map
- destructuring
- spread operator
- arrays as return values
- associative arrays

Arrays and forEach

```
//sum all items
sum = 0;
numbers = [2,4,6,8,10];
for (i=0; i<numbers.length;i++)
   sum+= numbers[i];
```

```
//sum all items using forEach
numbers = [2,4,6,8,10];
sum = 0;
numbers.forEach (function(item) {
   sum+= item;
})
```

- How could you rewrite this passing an arrow function to forEach?

Event Example revisited

HTML

```
<div class="theButton btn1" >Press Me</div>
<div class="theButton btn2" >Press Me</div>
<div class="theButton btn3" >Press Me</div>
<div id= "result"></div>
```

JS

```
window.onload= function() {
        document.querySelectorAll(".theButton").forEach(
                        btn => {btn.onclick= ()=> {showNumber(btn)}} )
} // end window onload

function showNumber(btn)
{
        result = document.getElementById("result");
        theClass = btn.className;
        index = theClass.indexOf("btn")+3;
        number = parseInt(theClass.substring(index));
        result.innerHTML = btn.className + "<br />" + number * 2;
}
```

Try it: Count the Vowels

- Create a string that contains a sentence
- Use split() to create an array consisting of each letter in the string
- Create an array or string of the 5 vowels and create a variable called count. Set count to zero (0)
- Use forEach to iterate through the array and use the vowels array to determine if a letter is a vowel – if so, add one to the count.
- Display the count at the end.

The Array map method

- map() produces a new array from an existing array - items in the new array are typically based on the original array and modified using a function.
- Given:

```
x = [33,5,44,63];

//Create a new array with each item doubled
a= x.map(item =>  2*item);          // why aren't curly brackets needed?

// Turn a set of items into a list
a = x.map(item=>"<li>"+item+"</li>").join("");
```

Try it: Count the Vowels and Change the Array

- Create a string that contains a sentence
- Use split() to create an array consisting of each letter in the string
- Create an array of the 5 vowels and create a variable called count. Set count to zero (0)
- Iterate through the array and use the vowels array to determine if a letter is a vowel – if so, add one to the count *and* change the item to a * How can you do this?
- Display the string with the * in place (use join) as well as the count.

Try it

- Create a set of 4 divs that each have a unique id and an X or an O as their content.
- Create an array that contains the id's only
- Using a map function, create a new array of only the X or O data in each div
- Display that array as a string

Associative Arrays

- The index to an array can be a key

```
flowers = [];
flowers["daisy"] = 12
flowers["rose"] = 15
flowers["carnation"] = 8
document.write(flowers["rose"])  //displays 15

for (key in flowers)
   document.write(key + " $" + flowers[key] + " - ")
//Output:  daisy $12 - rose $15 - carnation $8
```

Try it

- Create an associative array of (5) cities and the state they are in
  The cities should be the keys.
- Display cities only
- Create a function to locate a city and return its state using the array as a parameter to the function

Simple Objects

- An object can be a set of name - value pairs:

```
var flowers = {
      daisy: 12,
      rose: 15,
      carnation: 8
}
document.write (flowers['rose’])  //displays 15
document.write (Object.keys(flowers))
// displays: daisy,rose,carnation
```

Or is it an associative array?

```
flowers["tulip"] = 7;
document.write (Object.keys(flowers))
```

Try it

- Create a simple object using the same city / state data
- Display cities only using Object.keys()
- Rewrite your function to locate a city and return its state using the new object as a parameter to the function.