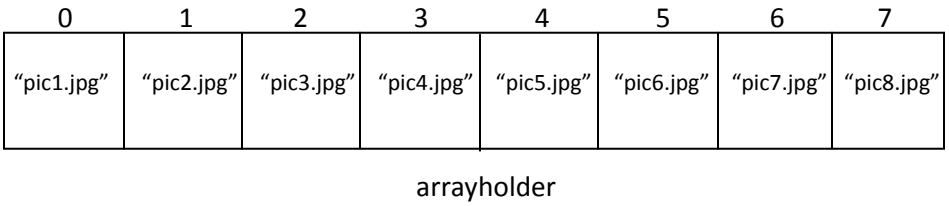
# Arrays/Branching Statements Tutorial:

In the last tutorial, you created a button that, when you clicked on it (the onclick event), changed another image on the page.

What if you have a series of pictures and you want to view the next picture in the series every time you click on the button? You need some structure that will hold on to all the names of each of the pictures you want to view, and then you need to be able to access each of the pictures in the structure. We can do this using Arrays.

Arrays are variables, or boxes, that hold more than one item. If you think of a variable as a box that holds something, and we use what's in the box by referring to the name of the box, then you can think of an array as one big long box with cubbyholes in it.




arrayholder

Now you can put things into each of the compartments in the Array. When you want to refer to one of the items in the array, you just refer to which cubbyhole of the Array it's in.

**WARNING:** Computer scientists always start counting at 0. (It has to do with memory space – if you want more of an explanation, ask me). So to refer to what is in the first cubbyhole in the Array arrayholder, I'd use arrayholder[0].

## Creating an Array:

To create an Array, you must first tell the browser that you're creating an Array. You can call my Array anything that starts with a letter and has no special characters (including spaces). In this case I'll call the Array I'm creating arrayholder:

```
arrayholder = new Array();  // we just created a brand new array, and we called
                            // it arrayholder.  Note the space between new and
                            // Array()!! People tend to forget that space.
```

Now you can put things into my Array. Most of the time we purposely put things into an Array in order, largely so the array doesn't have a bunch of empty boxes and so we can find items more easily. To put the names of different images into the array, you'd do the following:

```
arrayholder[0] = "pic1.jpg";
arrayholder[1] = "pic2.jpg";
arrayholder[2] = "pic3.jpg";
arrayholder[3] = "pic4.jpg";
arrayholder[4] = "pic5.jpg";
arrayholder[5] = "pic6.jpg";
arrayholder[6] = "pic7.jpg";
arrayholder[7] = "pic8.jpg";
```

Now to refer to one of the pictures in the array, you'd do this:

document.images[par1].src = arrayholder[2];

So to put this together in your code, you'd do the following:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  function ChangePic (par1)
  {
      document.images[par1].src = arrayHolder[3];
  }

</script>
</head>
<body>

   <img src = "mypic.jpg" id = "thispic" height = "164" width = "194">
   <img src = "button.gif" id = "buttonpic" height = "20" width = "50"
        onClick = "ChangePic ('thispic')" />
</body>
```

***1. Your Turn:*** Create an html page with a javascript that contains an array of images. Include in the body of your web page a picture and a button. When you click on the button (onClick), it should call the function you wrote to display a different image in the array (your choice of which image). In other words, make the above code work.

First thing you'll notice is that we put the array outside of the function, but inside the script. That's for efficiency and convenience, and conceptually it has to do with how your program works on your computer. Every time a function is called, the parameters and variables inside it are created in memory (e.g., your parameter box and variable box magically comes into existence in specific space inside your computer's memory space) and they leave memory (disappear) when the function is done running. An array is a whole bunch of cubbyholes. It is somewhat inefficient to make all those cubbyholes every time the function is run, and then they disappear so you have to remake them again the next time you run the function.

In our code, whenever you call a function, the code inside it runs. So every time we click on the button, we're calling the function ChangePic(). Now when we call it, we want the picture's src to change, so that code needs to be inside the function. However, once we've created the array, we don't necessarily need to recreate the array every time we call the function. We can create it once, and then whenever we call the function, the function can use the array, but doesn't need to recreate the array. If, however, we'd put the creation of the array inside the function, then every time the function was called, the array would be recreated. That would be a waste of computer time.

**Anything created outside of a function is available inside the function. The reverse is not true.** If we created something inside a function, it would not be available for use outside the function. So, for instance, if we had another function that needed to use the same array, it can use it because the array was created outside of all functions.

## *Index (number) Variable:*

At this moment, this function probably doesn't do exactly what you were hoping it would do. You probably wanted the function to display the next image in the array every time you click on the button, so it would cycle through the pictures in the array.

We can do this using a variable holding a single value.

We can change the value inside the variable num. We can increase the value inside the num every time we click on the button. Then we can see the next image in the array every time we click on the button.

```
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;      // a variable, or box we named num, set to hold 0 to start with…

  function ChangePic(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;     // adding 1 to the current value inside the num variable
                         // (box) This means that num starts at 0. When the
                         // function is called, it adds 1 to num, so now
                         // the value inside num is 1.  The next time the function
                         // is called, it will add 1 to the value inside num so
                         // num will now hold 2.  Next  time the function is
                         // called, it adds 1 to the value in num, so the value
                         // becomes 3, etc.
  }

</script>
</head>
<body>

   <img src = "mypic.jpg" id = "thispic" height = "164" width = "194">
   <img src = "button.gif" id = "buttonpic" height = "20" width = "50"
        onClick = "ChangePic ('thispic')" />
</body>
```

***2. Your Turn:*** Add a counting variable to your code. Click the button on your web page multiple times to see what happens. Does it do what you think it should do? Do you understand why?

What if you want a particular image to show up, like the first or the last image? You can do that by setting the value in num to either 0 (for the first image), or to a number representing the last picture in the array. To find the last image in the array, you could just set num to 3. But what if you add or delete an image from the array, then the array will be a

different length and the last picture will not be in the box labeled 3 in the array. Instead, we can use a function that javaScript gives us called length.

So, for instance, if I said:

```
num = arrayHolder.length;
```

num would get set to 4 (NOT 3!) because there are 4 elements in the array. Look at the array below. Count the images. There are 4.

However, notice that the last index value is 3. This is because we started counting at 0 (we have to – it's a computer science thing). So if I want to look at the last image in the array, I'd set num to:

num = arrayHolder.length -1;

This sets num to hold 3, or one less than the number of elements in the array, which is what we want.

So below we have two functions, one sets the image's src to be the first image in the array, and one that sets the last image in the array:

```
<script type = "text/javascript">
  arrayHolder = new Array();
  arrayHolder[0] = "Images/start.jpg";
  arrayHolder[1] = "Images/relaxed11.jpg";
  arrayHolder[2] = "Images/relaxed4.jpg";
  arrayHolder[3] = "Images/TheEnd.jpg";
  num = 3;
  function ChangePicFirst(par1)
  {   num = 0;
      document.images[par1].src = arrayHolder[num];
  }
  function ChangePicLast(par1)
  {   num = arrayHolder.length – 1;
      document.images[par1].src = arrayHolder[num];
  }
</script>
</head>
<body>
   <img src = "Images/relaxed12.jpg" id = "thispic" height = "164"   width = "194"/>
   <img src = "Images/button.jpg" id = "buttonpic" height = "60"    width = "53"
                          onClick = "ChangePicFirst('thispic')" />
   <img src = "Images/button.jpg" id = "buttonpic" height = "60" width = "53"
                        onClick = "ChangePicLast('thispic')" />
```

***3. Your turn:*** Write a function that sets an image's src to the last image in an array.

In function ChangePic (the exercise before), each time you clicked on a button, it changed the image's src to the next picture in the array. When you clicked on the button, the first image in the array appeared. Then when you clicked on the next button, the next image appeared. This is because num starts out holding 0. Then when we click on the button, ChangePic () is called, the document.images[par1].src is changed to arrayHolder[0] (because num holds 0). And then we add 1 to num, so now num holds 1.

Next time through document.images[par1].src is changed to arrayHolder[1] (because num holds 1). And then we add 1 to num, so now num holds 2. This continues until we've seen all the images in the arrayHolder array. However, when you get to the last image in the array, no matter how many times you click, the last image stays.

This is because once the num holds a number that is greater than the index of the last image in the array, the array has no more images to display. So every time you click on the button, the num value goes up by one, which continues to be a num greater than the index of the last image in the array, so no new image shows up.

We can change this.

## If (Branching) Condition:

Maybe what you'd want is, when we hit the end of the array, to start back over at the beginning. In English, you might say something like,

> If the index is at the last box in the array or past it, reset the index to 0.

We say something similar in JavaScript:

```
if (num >= arrayHolder.length)
{
      num = 0;
}
```

Now this says that if the value in num is equal to the length of the array (length is the number of elements in the array), then reset the value in num back to 0.

Altogether, your code will now look like:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;

  function ChangePic3(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
</script>
</head>
<body>

   <img src = "mypic.jpg" id = "thispic" height = "164" width = "194">
   <img src = "button.gif" id = "buttonpic" height = "20" width = "50"
```

```
                onClick = "ChangePic3('thispic')" />
</body>
```

Okay, so what if you want to add a button that allows you to go backwards through the pictures as well? Well, that would almost be the same function, so let's start with the same function:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;

  function ChangePic3(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
  function ChangePic4(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }

</script>
</head>
<body>

   <img src = "mypic.jpg" id = "thispic" height = "164" width = "194"><br />
   <img src = "backbutton.gif" id = "bbutton" height = "20" width = "50"
                onClick = "ChangePic4('thispic')" />

   <img src = "forwardbutton.gif" id = "fbutton" height = "20" width = "50"
                onClick = "ChangePic3('thispic')" />
</body>
```

Okay, that's a start, but both buttons take you forward. So let's make the back button take you backwards through the images:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
```

```
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;

  function ChangePic3(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
  function ChangePic4(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num - 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }

</script>
</head>
<body>
```

Try this. What happens when you get to the end? Does it loop around? So this still isn't quite what we want. What we want to happen is for ChangePic4 to show the 3rd pic, then the 2nd, then the 1st, then the 0th, and then back to the last picture. So if num == 0, reset it to the end. So modify the code as follows:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;

  function ChangePic3(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num + 1;
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
```

```
  function ChangePic4(par1)
  {
      document.images[par1].src = arrayHolder[num];
      num = num - 1;
      if (num <= 0)
      {
         num = arrayHolder.length-1;
      }
  }

</script>
</head>
<body>
```

Almost: but try reversing directions. Does it go in the wrong direction for one picture before it reverses? This is because num holds the value it had when it left the last function. So we need to reorder things a bit:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = 0;

  function ChangePic3(par1)
  {
      num = num + 1;
      document.images[par1].src = arrayHolder[num];
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
  function ChangePic4(par1)
  {
      num = num - 1;
      if (num <= 0)
      {
         num = arrayHolder.length-1;
      }
      document.images[par1].src = arrayHolder[num];

  }

</script>
</head>
<body>
```

Oh, we're so close. But what happens the first time you click on the forward button? Does it show the very first image in the array, or does it show the second one? To fix this, we'll start num at -1:

```
<head>
<script type = "text/javascript">

  arrayHolder = new Array();
  arrayHolder[0] = "pic1.jpg";
  arrayHolder[1] = "pic2.jpg";
  arrayHolder[2] = "pic3.jpg";
  arrayHolder[3] = "pic4.jpg";
  arrayHolder[4] = "pic5.jpg";

  num = -1;

  function ChangePic3(par1)
  {
      num = num + 1;
      document.images[par1].src = arrayHolder[num];
      if (num >= arrayHolder.length)
      {
         num = 0;
      }
  }
  function ChangePic4(par1)
  {
      num = num - 1;
      if (num <= 0)
      {
         num = arrayHolder.length-1;
      }
      document.images[par1].src = arrayHolder[num];

  }

</script>
</head>
<body>
```

***4. Your Turn:*** Get functions working so that you have two buttons, one forward and one backwards, and you can go both forwards and backwards through the array.

## Tools You Have in the language so far:

1. Functions – code sitting there waiting for you to call it.
2. Parameters – boxes in functions in which you can put numbers or strings, or anything else you want
3. Using the document's images objects, a way of changing an images:
    1. src
    2. width
    3. height
    4. alt
4. Arrays – boxes with more than one value in them
    1. myarray.length: a way of getting the number of things in the array
    2. myarray[2]: a way of getting at the value in myarray at location 2

5. If control statements – ways of controlling the conditions under which code is executed.

## Your Turn:

5. Add 2 more images to the end of your arrayHolder array. Before you run it, what do you think will happen now? Will those two images show up? Why or why not?

6. Add a "reset" button. Make it call a function that "resets" the array, so that the very first image in the array shows up.

7. In the example I showed you in class, I added another image that showed a picture of the number currently in indexnum. Do that. (You will need to add another array for the pictures of numbers).

## *To turn in:*

Your Turns 1-7