Parameters

Do the two functions look similar?

```
<head>
<title>Our first javascript!</title>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
<script type="text/javascript">
   function makepicchange()
   {
          document.images["pic1"].src = "kittybelly.jpg";
   }
   function changepicback()
   {
          document.images["pic1"].src = "kittenasleep.jpg"
   }
</script>
</head>
 <body>
<p>
<img src = "kittenasleep.jpg" id = "pic1" width = "500" height = "375"
onmouseover = "makepicchange()"  onmouseout = "changepicback()" />
</p>
```

What if we added "onclick"?

```
<head>
   <meta http-equiv="Content-Type" content = "text/html;charset = utf-8" />
   <title>cats and kittens </title>
   <script type = "text/javascript">
   function makepicchange()
   {   document.images["pic1"].src = "kittybelly.jpg";
   }

   function changepicback()
   {   document.images["pic1"].src = "kittenasleep.jpg"
   }

   function changetonew()
   {   document.images["pic1"].src = "relaxed10.jpg";
   }
   </script>
</head>
<body>
<p>
<img src = "kittenasleep.jpg" id = "pic1"  width = "250" height = "187"
            onmouseover="makepicchange()"  onmouseout="changepicback()"
            onclick = "changetonew()"/>
</p>
```

We have 3 functions (each with its own name) that do pretty much exactly the same thing: change document.images["pic1"].src

```
            document.images["pic1"].src = "kittybelly.jpg";
            document.images["pic1"].src = "kittenasleep.jpg"
            document.images["pic1"].src = "relaxed10.jpg";
```

Is there a better way to do this (so that we don't have to write a new function every time we want the image's source to change to a different picture)? Yes! We can use parameters! Parameters are small pieces of memory that we name and assign values to. Think of a parameter as a box. There are many boxes, so distinguish between all the boxes, I'll give the boxes each their own name (I'll name my box boxvar). We put one thing into the box named boxvar at a time. So let's say you put a puppy in the box. I then tell you to bring boxvar over to me. If you hand boxvar to me, you're actually handing me the puppy (thank you very much – I love puppies!). Now, of course, I take the puppy out of the box (because he's squirmy and I want to cuddle him) and decide to put a potted plant in the box. I then tell you to place boxvar on the table over there. If you do this, you've actually placed the potted plant on the table. Thus if I refer to the name I've given the box, I'm actually talking about what is inside the box at the moment. Like the box, parameters get a name, and hold values. The value inside the parameter can change. But if I refer to the name of the parameter, I am referring to what is currently inside the parameter.

In the above javascript example, we can use a parameter to hold the name of the picture we want to change the src to. What I mean by this is that I can create a parameter (that I'll name *par1)*, and then, at different times, place the name of a picture inside that parameter par1 ("kittybelly.jpg", then "kittenasleep.jpg", then relaxed10.jpg") Now, instead of using the name of the picture, I can just use the parameter name. So the line would look like this:

```
document.images["pic1"].src = par1;
```

Note that we don't put quotes around the parameter's name. This is how we tell the computer to look inside the parameter instead of setting the src to be "par1" which is the name of a parameter, not the name of a picture. (What's inside par1 is the name of a picture!)

Parameter naming rules are like function naming rules:

- no special characters or spaces
- only letters, numbers, and the _
- the name must start with a letter

For now, each parameter only holds one thing at a time. So if you put a new thing in the parameter, the old thing gets ejected (overwritten).

The interesting part now is figuring out how to put the different pictures into the parameter par1 at the right time. We'll do by passing in the name of the picture into the function.

Example using parameters

```
...
<head>
   <meta http-equiv="Content-Type" content = "text/html;charset = utf-8" />
   <title>cats and kittens </title>
   <script type = "text/javascript">

      function changepic(par1)
      {
          document.images["pic1"].src = par1;
      }

    </script>
</head>
<body>
   <p>

   <img src = "kittenasleep.jpg" id = "pic1"  width = "250" height = "187"
      onmouseover="changepic('kittybelly.jpg')"
      onmouseout="changepic('kittenasleep.jpg')"
      onclick = "changepic('relaxed10.jpg')"/>

   </p>
...
```

In the code above, we now have a function with a parameter. It is

function changepic(par1)

That is how we set aside memory space (or, in essence, create a box), and give the box the name par1. The box is filled when the function is called, e.g.,

```
...
      onmouseover="changepic('kittybelly.jpg')"
      onmouseout="changepic('kittenasleep.jpg')"
      onclick = "changepic('relaxed10.jpg')"/>
```

In these examples, when we run the mouse over the image, the function changepic is called, and by putting inside the (' ') the name of the picture 'kittybelly.jpg', we are now putting 'kittybelly.jpg' into the box called par1.

When we run our mouse off the image, the function changpic is called again, and this time 'kittenasleep.jpg' is in between (' and '), so 'kittenasleep.jpg' is what goes into par1.

Now when the following code is executed :

```
...
<script type = "text/javascript">
      function changepic(par1)
      {
          document.images["pic1"].src = par1;
      }
    </script>
</head>
<body>
   <p>
   <img src = "kittenasleep.jpg" id = "pic1"  width = "250" height = "187"
      onmouseover="changepic('kittybelly.jpg')"
      onmouseout="changepic('kittenasleep.jpg')"
      onclick = "changepic('relaxed10.jpg')"/>
   </p>
...
```

what happens when we run our mouse over the image is the function changepic is called, and 'kittybelly.jpg' goes into the parameter par1. So inside the function, when the line:

```
document.images["pic1"].src = par1;
```

is executed, the src of the document's image id'd with 'pic1' is changed to what's INSIDE par1, which is kittybelly.jpg.

When we run our mouse off the image(onmouseout), the function changepic is called, only this time 'kittenasleep.jpg' goes into the parameter par1. So now inside the function, when the line:

```
document.images["pic1"].src = par1;
```

is executed, the src of the document's image id'd with 'pic1' is changed to kittenasleep.jpg.

Finally, when we click on the image(onclick), the function changepic is called, only this time 'relaxed10.jpg' goes into the parameter par1. So inside the function, when the line:

```
document.images["pic1"].src = par1;
```

is executed, the src of the document's image id'd with 'pic1' is changed to relaxed10.jpg.

Make sense?

Your turn:

Try a version of this, with your own pictures. Give the parameter and the function your own personal name that you choose. Does it work? Do you understand what is happening here?

What if we've got more than 1 picture on the page? Right now we'd have to write a different function for each picture:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns ="http://www.w3.org/1999/xhtml" xml:lang = "en" lang = "en">
   <head>
       <meta http-equiv="Content-Type" content = "text/html;charset = utf-8" />
       <title>cats and kittens </title>
       <script type = "text/javascript">

           function changepic(par1)
           {       document.images["pic1"].src = par1;
           }

           function changepic2(par1)
           {      document.images["pic2"].src = par1;
           }

           function changepic3(par1)
           {      document.images["pic3"].src = par1;
           }

           function changepic4(par1)
           {      document.images["pic4"].src = par1;
           }

      </script>
  </head>
  <body>
    <table><tr><td>
       <img src = "kittenasleep.jpg" width = "250" height = "187" id = "pic1"
        onmouseover = "changepic('kittybelly.jpg')"
        onmouseout = "changepic('kittenasleep.jpg')"/>
       </td><td>
       <img src="kittybreakfast.jpg" width="250" height="187" id="pic2"
        onmouseover = "changepic2('relaxed1a.jpg')”
        onmouseout = "changepic2('kittybreakfast.jpg')"/>
       </td><tr><tr><td>
       <img src="kittydove.jpg" width = "250" height = "187" id = "pic3"
        onmouseover = "changepic3('kittyno-regrets.jpg')"
        onmouseout = "changepic3('kittydove.jpg')"/>
       </td><td>
        <img src="kittyfight.jpg" width="250" height="187" id = "pic4"
        onmouseover = "changepic4('kittypanda.jpg')“
        onmouseout = "changepic4('kittyfight.jpg')"/>
       </td></tr>
     </table>
  </body>
</html>
```

These 4 functions look awfully similar!

```
function changepic(par1)
{       document.images["pic1"].src = par1;
}
function changepic2(par1)
{      document.images["pic2"].src = par1;
}

function changepic3(par1)
{      document.images["pic3"].src = par1;
}

function changepic4(par1)
{      document.images["pic4"].src = par1;
}
```

What is the difference between these functions? It is just the id of the image! (Go look on the previous page. We're using the different ids of the 4 images on the page, and that's how we tell which image is being changed in each function.) So if we used a parameter to hold the id of the image, we could then use the parameter in place of the actual id, and then we could have just one function that works for all the images!

Add another parameter

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns ="http://www.w3.org/1999/xhtml" xml:lang = "en" lang = "en">
<head>
   <meta http-equiv="Content-Type" content = "text/html;charset = utf-8" />
   <title>cats and kittens </title>
   <script type = "text/javascript">

   function changepic(par2, par1)
   {       document.images[par2].src = par1;
   }

   </script>
</head>
<body>
<table><tr><td>
      <img src = "kittenasleep.jpg" width = "250" height = "187" id = "pic1"
            onmouseover = "changepic('pic1','kittybelly.jpg')"
            onmouseout = "changepic('pic1', 'kittenasleep.jpg')"/>
</td><td>
      <img src="kittybreakfast.jpg" width = "250" height = "187" id = "pic2"
            onmouseover = "changepic('pic2','relaxed1a.jpg')"
            onmouseout = "changepic('pic2','kittybreakfast.jpg')"/>
</td><tr><tr><td>
     <img src="kittydove.jpg"  width = "250" height = "187" id = "pic3"
            onmouseover = "changepic('pic3','kittyno-regrets.jpg')"
            onmouseout = "changepic('pic3','kittydove.jpg')"/>
</td><td>
     <img src = "kittyfight.jpg"  width = "250" height = "187" id = "pic4"
            onmouseover = "changepic('pic4','kittypanda.jpg')"
            onmouseout = "changepic('pic4','kittyfight.jpg')"/>
</td></tr></table>
</body>
</html>
```

Now the function is:

```
function changepic(par2, par1)
{       document.images[par2].src = par1;
}
```

and it has 2 parameters, par2 and par1 (note: I made both those names up. You could have called them anything you wanted , as long as they follow the naming rules).

So now when the function is called, e.g.,

```
onmouseover = "changepic('pic1','kittybelly.jpg')"
```

we find the function changepic, and put 'pic1' into par2, and we put 'kittybelly.jpg' into par1.

How do we know which parameter gets which value? It is based on the order. So in the call to

```
changepic('pic1','kittybelly.jpg')
```

'pic1' is first, and 'kittybelly.jpg' is second. In the function definition (the code at the top of the page),

function changepic(par2, par1)

par2 is the first parameter, and par1 is the second parameter. So 'pic1' goes into par2 and 'kittybelly.jpg' goes into par1. Now when the computer executes the line,

```
document.images[par2].src = par1;
```

what's happening is imgholder holds 'pic1' and par1 holds 'kittybelly.jpg', so the src of 'pic1' is changed to 'kittybelly.jpg', or in essence the computer is executing the line,

```
document.images['pic1'].src = 'kittybelly.jpg';
```

Your turn:

1. Get this code to work. Use your own images, and your own parameter names.
2. Explain what happened when we run our mouse over and off the following code in terms of what parameters go where and what is executed:

```
<img src="kittydove.jpg"  width = "250" height = "187" id = "pic3"
      onmouseover = "changepic('pic3','kittyno-regrets.jpg')"
      onmouseout = "changepic('pic3','kittydove.jpg')"/>
```

Other things you can change:

- src - The URL of the image to be displayed. (we've changed this)
- border - The width of the border around the image in pixels.
- height - The read only height of the image in pixels.
- width - The read only width of the image in pixels.

Example:

```
<head>
   <meta http-equiv="Content-Type" content = "text/html;charset = utf-8" />
   <title>cats and kittens </title>
   <script type = "text/javascript">
   function changepic(par2, par1)
   {     document.images[par2].src = par1;
          document.images[par2].border = "10px";
   }
   function changepicback(par2, par1)
   {      document.images[par2].src = par1;
          document.images[par2].border = "0px" ;
   }
   </script>
</head>
<body>
<table><tr><td>
      <img src = "kittenasleep.jpg"  width = "250" height = "187" id = "pic1"
            onmouseover = "changepic('pic1','kittybelly.jpg')"
            onmouseout = "changepicback('pic1', 'kittenasleep.jpg')"/>
   </td><td>
      <img src="kittybreakfast.jpg" width = "250" height = "187" id = "pic2"
            onmouseover = "changepic('pic2','relaxed1a.jpg')"
            onmouseout = "changepicback('pic2','kittybreakfast.jpg')"/>
   </td><tr><tr><td>
      <img src = "kittydove.jpg"  width = "250" height = "187" id = "pic3"
            onmouseover = "changepic('pic3','kittyno-regrets.jpg')"
            onmouseout = "changepicback('pic3','kittydove.jpg')"/>
   </td><td>
      <img src = "kittyfight.jpg"  width = "250" height = "187" id = "pic4"
            onmouseover = "changepic('pic4','kittypanda.jpg')"
            onmouseout = "changepicback('pic4','kittyfight.jpg')"/>
   </td></tr></table>
</body>
```

Troubleshooting (Debugging):

If you run into problems, it means that there are bugs in your code. Yep, they're called bugs, and they're just problems with your code. They can be typos, invalid html or javaScript code, or logical errors (we're probably not at the point of having logical errors). Regardless of the type of error, you need to develop strategies for figuring out what to do when your code doesn't work. Here are a few techniques.

1. Does the html page look the way you think it should look?

- If so, then the problem is probably either in your javascript or your call to the javascript function (onmouseover = "changepic()", etc., is considered a call to your javascript function).
- If not, then look more closely at the html code itself.

2. If the problem is with the javascript, first check very carefully for typos.

- Make sure the word "script" is spelled correctly (trust me – I've seen an awful lot of "scipt"s and "scirpt"s that didn't work).
- Make sure the function call is inside double-quotes and the parameters inside the function call (aka parameters) are in single quotes (e.g., onmouseover = "changepic('pic1')").
- If you copied and pasted any code, make sure the quotes are retyped in plain text (" " are not the same as "". You want the plain ones (the second ones))
- Make sure the name of the function is EXACTLY the same in the call and in the script (ChangePic() and changepic() are not considered the same).
- Make sure you have an opening and closing script tag (<script type = "text/javascript")
- Make sure the function has an opening and closing { and }
- Finally, if you've double and triple-checked for typos, try getting rid of everything inside the function. Change the content of the function to something basic like,
    - document.write("Hello");
- Run your code. If it doesn't work, that means there's a problem in the call to the function or the function name or the script tag. If it does work, that means there's a problem with the code inside the function.

Your Turn (To Turn In: Hwk1):

1. Create an html page with more than one image in it. Write a function that modifies an image's size when you click on the image.
2. Create an html page with more than one image in it. Write a function that doubles the image's size when you click on the image. This will involve passing the size of the image into the function (using parameters) and then multiplying the size by 2. Note: to multiply by two you'd use the following notation:

    widthparameter * 2

    (where widthparameter is a parameter that holds the width of the image)
3. Create an html page with two images on it. Write a function that changes an image's src to a new picture using parameters. Now make it so that when you run your mouse over the first image, it's the second image that changes (it's a matter of what parameters you pass into the function).

4. Create an html page with at least 3 images on it. The first image should be fairly large – it doesn't need to have a src to start with, just a width and a height. The other images should be smaller and below it. Write code such that when you click on one of the smaller images, a larger version of that image's picture shows up in the first image's src. Think of it like this: You've got a bunch of small snapshots of photos and a place where a larger version of each of those snapshots can show up one at a time (the place is made with the fairly large first image). Now when you click on one of the snapshots, a larger version of that particular photo shows up in the large place.