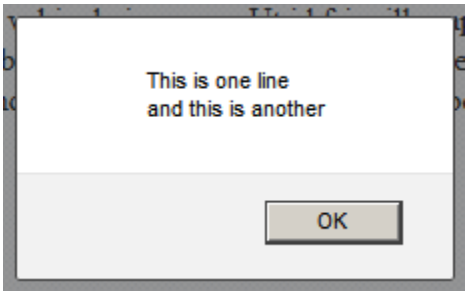
WEB-215 DOM Practice

Part 1

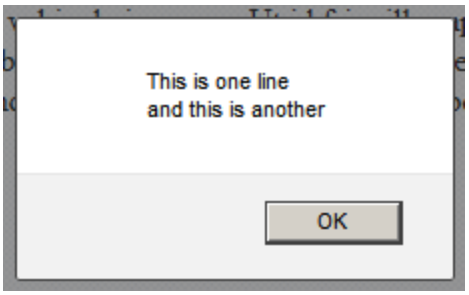
1. Use the provided **practice-dom.htm** document.
2. Link it to an external JavaScript file named **practice-dom.js** (substitute your first/last name – watch the camel-casing!) Be sure you place your SCRIPT tag in a location where the JavaScript will fire after the HTML page loads. Remember this is not the best practice, but the necessary way to do things now until we learn a better way.

Alerts with New Lines

1. Type this code in your JavaScript file:
```
alert('This is one line\nand this is another');
```
2. Save your JS file and run the HTML file in a browser to see the result.



3. Notice that "\n" creates a new line in the alert box. However, \n runs together with the rest of the code, making it hard to see. To be more efficient, create a variable to store the value "\n" and concatenate it to the alert message. (You will do this in the next step.)
4. Delete the alert() code you typed in Step 1 and type the following instead:
```
var br = '\n';
alert('This is one line' + br + 'and this is another');
```
5. Save your JS file and run the HTML file in a browser to see the result.

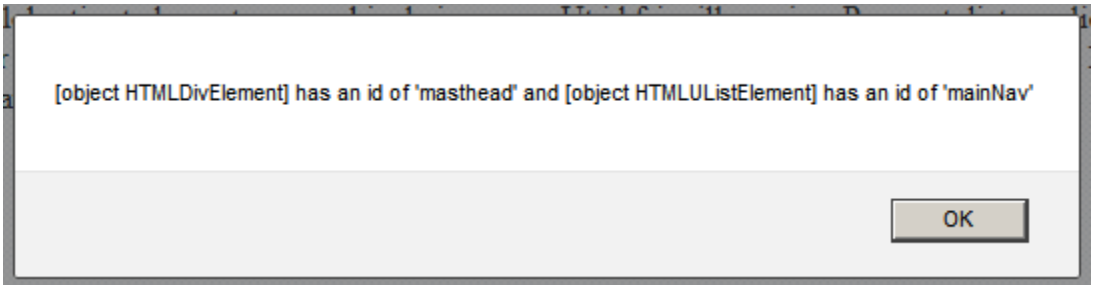


6. The alert box that pops up should be exactly the same as it was before, but now the JavaScript code is a little easier to read, and easier to see exactly where the line break will occur. Nothing magic about the variable name `br` – it must mimics the `<br />` tag name.
7. You will use this idea of creating new lines within an alert box in this assignment. Instead of alerting every section of this assignment, you will simply store variables and alert them all at once at the end – separated by line breaks to make them easy to read.
8. Delete all JavaScript code so you have a blank JavaScript file.

Finding elements by ID

1. Use getElementById

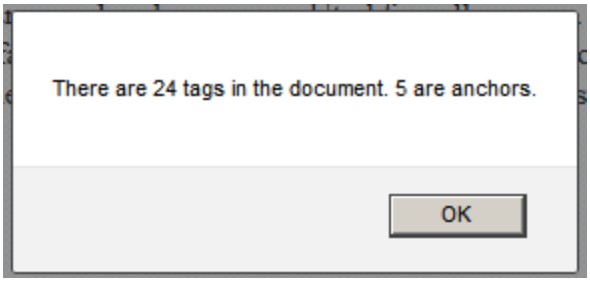
2. Create a variable that stores the element with the id of ***masthead***
3. Create a variable that stores the element with the id of ***mainNav***
4. Create a variable named `msg1` that stores the following string:
   STEP-2-VARIABLE has an id of 'masthead' and STEP-3-VARIABLE has as id of 'mainNav'
   for example:
   ```
   var msg1 = variable1 + " has an id of 'masthead' and " + variable2 + " has an id of 'mainNav'";
   ```
   a. Note that the words *masthead* and *mainNav* are enclosed in single quotes, which will make reading the alert message easier.
   b. Of course, your variable names will differ from `variable1` and `variable2`. You will display an alert of `msg1` at the end of this assignment along with all your other messages, separated by line breaks.
5. Create an `alert` that displays `msg1`.
6. Save your JS file and run the HTML file in a browser to see the result.



7. Comment out the `alert` but leave the other code as is.

Finding elements by tag name

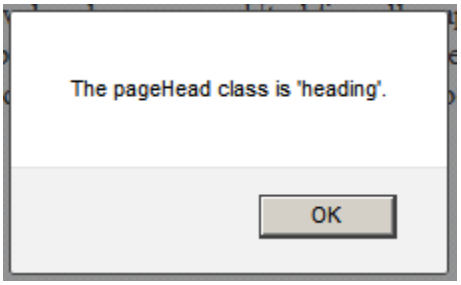
1. Create a variable that stores the total number of tags in the document.
2. Create a variable that stores the total number of anchor tags in the document.
3. Create a variable named `msg2` that stores the following string:
   There are XXX tags in the document. YYY are anchors.
   for example:
   ```
   var msg2 = "There are " + variable3 + " tags in the document. " + variable4 + " are anchors.";
   ```
4. It is assumed you "get the gist" now. From this point on, you will continue to store values in variables where appropriate for each exercise and then store a final string using those variables in a "msgX" variable.
5. Create an `alert` that displays `msg2`.
6. Save your JS file and run the HTML file in a browser to see the result.



7. Comment out the `alert` but leave the other code as is.

Get Attributes

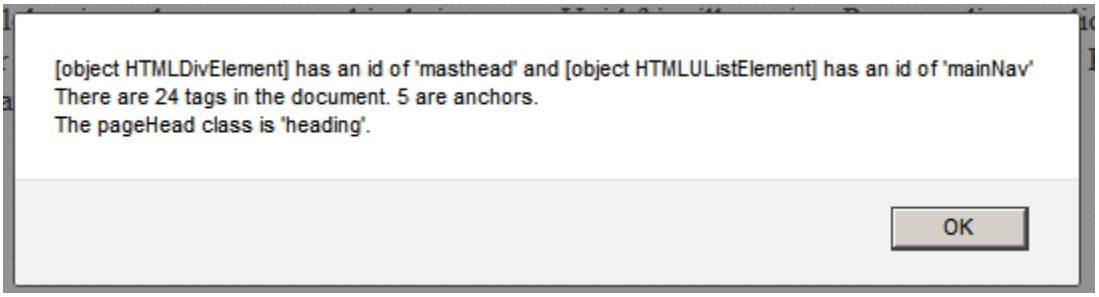
1. Create a variable that stores the value of the class attribute of the element with the id of *pageHead*.
2. Create a `msg3` variable that stores the string *The pageHead class is 'XXX'*. (Substitute XXX with the appropriate variable and be sure to includes the surrounding single quotes.)
3. Create an `alert` that displays `msg3`.
4. Save your JS file and run the HTML file in a browser to see the result.



5. Comment out the `alert` but leave the other code as is.

Display Your Alerts

1. Create a final ***single*** alert that strings together your `msg1`, `msg2`, and `msg3` variables and displays them each on a separate line. Note that `msg1` may actually wrap to a second line because it is so long.
2. Save your JS file and run the HTML file in a browser to see the result.



3. Comment out the `alert` but leave the other code as is.

Loop Through the DOM

1. Test each of the five links on the page and notice they are all broken (because they link to pages that do not exist).
2. Set each link within the **main navigation section** to go *http://www.google.com* when clicked. At the same time, create a `target` attribute for each, setting its value to *_blank* so the links open in a new tab/window.
3. Set each link in the **sidebar area** to go to *http://www.abtech.edu* when clicked. At the same time, create a `target` attribute for each, setting its value to *_blank* so the links open in a new tab/window.
4. Don't know what I mean by "the main navigation section" or "the sidebar area"? Look at the HTML code.
5. You do not need to create a `msgX` variable for this exercise.
6. Test each link. The Home, Products, and Contact links should all open Google in a new window/tab. The two 'read more' links should open the A-B Tech web site in a new window/tab.

Part 2 – Something a little more useful

In this practice, you will have JavaScript mark the navbar link for the current page. I provide the entire script. Your job is to duplicate and understand it.

1. Load **page1.htm** from the **Current Link folder** in your browser. Click through pages 2 and 3 and notice the main nav bar does not indicate which page you are on.
2. Open the CSS file in a text editor and notice the last rule defines a style named #currentlink, which should style the link of the current page. But – this style is not being applied right now. To apply it, you would have to hard-code id="currentlink" in the correct <a> tag on each page, which prevents you from using an include file as your navbar. Instead, you will use JavaScript to dynamically apply the #currentlink ID to the correct navbar link.
3. Here's a simple logic map:
    a. Get the current URL in the browser address bar
    b. From that full URL, extract just the name of the page
    c. Loop through every mainnav link in the HTML document and examine its HREF
    d. Within each HREF, extract just the page name
    e. Compare the link's HREF's page name to the page name in the browser
        i. If they match, that link needs to be flagged as the "current link" or the "You Are Here page marker"
4. Here's the same logic map with additional notes:
    a. Get the current URL in the browser address bar
        i. `window.location.pathname` will return the entire string shown in the browser's address bar
    b. From that full URL, extract just the name of the page
        i. `substring()` will return a string of text that is part of a longer string. For example, "birth" is a substring of "Happy birthday".
        ii. `lastIndexOf()` returns the position of a character(s) you specify. For example, `lastIndexOf('irt')` would return the number 7 in 'Happy birthday' because 'irt' begins at the 7th position (remember to start counting at 0 and include spaces).
        iii. `lastIndexOf()+1` would return 8 instead of 7 in the previous example.
        iv. Combining `substring()` and `lastIndexOf()` gives you a useful string:
        `var msg = 'Happy birthday';`
        `msg.substring(msg.lastIndexOf('irt'))` returns the remainder of the string, rather than the numerical starting point. For example, this would return 'irthday'.
        `msg.substring(msg.lastIndexOf('irt')+1` would return 'rthday'.
    c. Loop through every mainnav link in the HTML document and examine its HREF
        i. `document.getElementsByTagName('a')` can be used
    d. Within each HREF, extract just the page name
        i. Again, `substring()` and `lastIndexOf()+1` will help
    e. Compare the link's HREF's page name to the page name in the browser
        i. If they match, that must be the link for the page you are on. So, apply the #currentlink CSS to that link.
    f. You need to delay your JavaScript from running until the HTML page finishes loading. `window.onload` will do that.

5. Here's the complete script, a line at a time, with comments. Type this in the blank JavaScript file (in the js folder). Make sure you understand each line.

```
/*
window.onload delays the script until the HTML is loaded
here we are setting it equal to the name of a function we will create later
this really means 'wait until the HTML finishes loading, then call the setCurrentLinks function
*/
window.onload = setCurrentLinks;

// create the function
function setCurrentLinks(){

// grab all the links in the mainnav section
    var mainnavLinks = document.getElementById('mainnav').getElementsByTagName('a');

//get the full address bar string
    var currentPath = window.location.pathname;

/*
from that long string, extract just the last part, the page name, which starts after the final slash
*/
    var currentPage = currentPath.substring(currentPath.lastIndexOf('/')+1);

// set up a loop to iterate through all the links you collected earlier
    for(i = 0; i < mainnavLinks.length; i++){

//get the current link's HREF
        var theFullHref = mainnavLinks[i].href;

/*
from that long string, extract just the last part, the page name, which starts after the final slash
*/
        var thePageHref = theFullHref.substring(theFullHref.lastIndexOf('/')+1);

//compare the current link's HREF's page name to the page name in the address bar
            if(thePageHref == currentPage){

//if they match, set the ID of the current link to match the CSS rule's ID
            mainnavLinks[i].setAttribute('id', 'currentlink');

//close the IF
            }

//close the FOR
        }

//close the function
}
```

Here is the entire script without comments:

```
window.onload = setCurrentLinks;

function setCurrentLinks(){
    var mainnavLinks = document.getElementById('mainnav').getElementsByTagName('a');
    var currentPath = window.location.pathname;
    var currentPage = currentPath.substring(currentPath.lastIndexOf('/')+1);
        for(i = 0; i < mainnavLinks.length; i++){
        var theFullHref = mainnavLinks[i].href;
        var thePageHref = theFullHref.substring(theFullHref.lastIndexOf('/')+1);
            if(thePageHref == currentPage){
            mainnavLinks[i].setAttribute('id', 'currentlink');
            }
        }
}
```