Lab objectiveqs

* Understand DHTML and client side processing
* Understand how HTML elements can be accessed using the DOM
* Use javascript in a web page to dynamically change a web page

Instructions

Try to do as many of the exercises as possible, in the order listed. You may skip an exercise if it is impossible to accomplish the task described (for technical reasons). If you have trouble with an exercise, contact the instructor for help – do not just skip the exercise! You may work with a partner, but each of you must submit your individual work. Work should be done electronically on this document (cut and paste as needed but be careful of incorrect characters). Anything that involves code should be tried out in VS Code before submitting (you may cut and paste directly from VS Code). Submit work before the next class. If you cannot submit by this time, try to do so as soon as possible.

You may wish to open the [presentation slides](https://dport96.github.io/ITM352/morea/040.dynamic-web-pages/ITM352_client_side_processing.pptx) for convenience of viewing and copy-paste. When asked by the instructor do the lab exercises indicated. If you get done early, feel free to go on. If you are not done before the instructor indicates the next exercise, raise your hand and explain where you are at. Later parts often depend on completing previous parts so do not let yourself get behind and expect to catch up!

Note: Whenever you see something like and it's not obviously an HTML tag, it means replace this with you own information as indicated inside. e.g. replace with Joe (or whatever your name is).

For exercises that do not ask specific questions but have you perform tasks, copy any code you created and the output as your answer to the exercise. Some exercises ask for explanations. Make notes for your answer and complete these after class before submitting. Do not try to make complete answers if it keeps you from progressing!

Do all your code work in VS Code. Copy and paste from there. Please ask the instructor for help if you get stuck this is **NOT** a test.

Submission

When we are done, submit work before the next class. If you cannot submit by this time, try to do so as soon as possible after class. Copy this lab with your answers and paste your into the Laulima assignment submission box to [Laulima Assignment Lab 3: Client-side Dynamic Web Pages](https://laulima.hawaii.edu/). If you do not submit something to Laulima you will not get credit.

**COPY this entire page and paste into the Laulima assignment submission box for this assignment**

Exercise #1:

1. Open a blank webpage, inspect, then go to the Console and use document.write(new Date());. Explain how you could this to add the current date and time a web page using client side web page processing.

* The browser Console allows us to directly put in coding and the changes will reflect directly on the browser. This is client-side webpage processing.

1. Explain how you could add the current date and time a web page using server side web page processing.
   * We would use the getdate.html on VScode or whatever program is running the server. Then it would get the information from the server.

c. Explain why the date and time for (a) may be different than (b) and it would

* On the client-side, the date and time come from the machine running the browser. On the other hand, on server-side, the date and time come from the server’s clock.

Exercise #2:

How is a static web page different than a dynamic web page? Give an example of each (explain or show code).

* A static page HTML code only changes manually. A dynamic page’s html can change whether it’d be a client-side or server-side.

Exercise #3:

Make a directory Lab3 in your personal 352 repo. In this directory, create a new file hello.html and use an EMMET boilerplate to start and add the following inside the body:

<p id="demo">Hello World!</p>

Start your local server and request hello.html. Inspect the page and using the browser console type demo and see what you get. Then starting from document get the element by ID then the innerHTML property. Now change set the innerHTML to “Goodby World!”.

1. What happened in the browser? Explain this.

* Hello world changed to goodbye world because we changed it dynamically since we changed the html coding from hello to goodbye on the elements page of the web browser.

1. Now right-click in the page and select View Page Source. Looking at the HTML source, explain why the paragraph tag still has “Hello World!” and not “Goodby World!” in it. Hint: Think about the DOM what you are actually changing with the script code.

* We only modified the document object in memory which came from original source but not the actual file. This means that we only changed it from whatever code we originally got from the source.

1. Close the page source page and on the hello.html page right-click and select Inspect. Go to the Elements tab ann navigate to the paragraph element (or use the selector and click in the paragraph). Why is the value here “Goodby World!”? Hint: The Elements view is a DOM tree for the page.
   * The value is goodbye world because it was changed dynamically which means after being changed to goodbye world on the elements tab, it automatically displayed the changes to the code even if the source file was not affected.

Exercise #4: SmartPhoneProducts1\_1 “whirling dervish image game”

Using Javascript and the DOM, make a little image game that has the images rotate for an onmouseover event and stop on an onclick event.

Task 1:

Make a copy of SmartPhoneProducts1 and rename the directory SmartPhoneProducts1\_1. If you don’t have a working version of SmartPhoneProducts1 you can use get a copy [here](https://dport96.github.io/ITM352/morea/040.dynamic-web-pages/SmartPhoneProducts1.zip).

Task 2: Add the following to the stylesheet

**.rotate** img

{

animation: rotation .3s infinite linear;

}

**@keyframes** rotation {

from {

transform: rotate(0deg);

}

to {

transform: rotate(359deg);

}

}

Task 3:

Set the onmouseover event for the first <section> to change its className property to 'item rotate' (use the this reference to access the DOM object for **this** element). Save and refresh the page and test that the icon spins when the mouse moves over it. Refresh the page to stop the spin! Use find and replace to change the other <section> tags to match this one.

Task 4:

Now set the onclick event attribute for <section> elements to set the class to just 'item'. Refresh the page after saving the file and move the mouse over an image to get it spinning. When you click on the image it should stop. It may be a little tricky to click the mouse and get it to stop with moving the mouse which will start it spinnign again! The game is to get everything spinng and see how quickly you can get all of them to stop by clicking on the images.

Here’s an example of how SmartPhoneProducts1\_1 (with Extra Credit) should work when you’re done:

Extra Credit fun!:

Add a slider to change the rotation speed of the animation. There are many ways to do this and you are encouraged to do what you think is easiest. Here’s one way to do it by changing the internal (i.e. page scoped) stylesheet which will override the products-style.css stylesheet. It takes advantage of the onchange event when the slider value changes to update the animation-duration: style for .rotate img. Here are the steps for doing this:

* Add a slider form element (<input id='speedSlider' type='range' min=".1" max="5" step=".1" style="direction: rtl;">)
* Add an empty style element <style id="localstyle"></style> to the <head> element
* Add onchange="localStyle.innerHTML = '.rotate img {animation-duration: ' + speedSlider.value + 's;}';" to the slider element
* Refresh the page and try moving the slider :)