IST 263

# Lab 12

## This lab covers:

1. What is JavaScript?
2. How do you use it?
3. 1st Program
4. Functions
5. Event Handling
6. Preview Next Week

## SETUP

Create a folder in your Github repository called lab12. To walk you through JavaScript, I've placed a practice file in blackboard. Download it and save it in your lab12 folder. Examine the HTML and follow the directions below.

## overview

In this lab we are going to use JavaScript for the first time. We'll work through examples of the building blocks for writing JavaScript to create a shell game. If you are not familiar with the shell game, it is a game where three inverted cups or nutshells are moved about, and contestants must spot which is the one with a pea underneath.

## WHAT is Javascript and How do you use it?

JavaScript is a text file that is interpreted by the browser and allows you to add dynamic content. JavaScript can be added to the page embedded or externally just like CSS. Here is a sample of both ways.

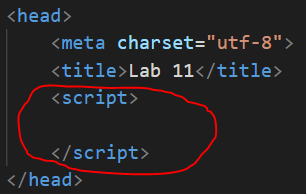
|  |  |
| --- | --- |
| Embedded |  |
|  |  |
| External |  |

**External JavaScript**

All the languages we've learned are created as text files with different file extensions. Your browser knows how to interpret the code, as HTML, CSS or JavaScript based on the file extension. For HTML files we use .html. For CSS files .css. For external JavaScript files we will use .js.

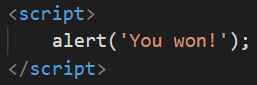
**Embedded JavaScript**

Embedded CSS allows you to write CSS code on your HTML page and embedded JavaScript allows you to do the same thing with JavaScript. One difference between JavaScript and CSS is that CSS must be added inside the head element and JavaScript can be added anywhere including inside the body element.

1. For this lab we are going to work with embedded JavaScript. In the head of your web page add the script element.   
   

## 1st Javascript Program

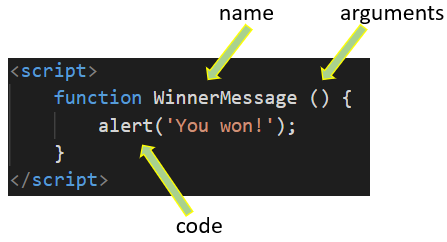
For our first program we are going to write one method called alert which brings up a dialog box. This dialog box we will tell the player if they picked the correct walnut.

1. Write the alert inside the script element in your browser. Load your page and you should see a dialog box with "You Won".  
   

Note that this JavaScript runs when you go to the web page. In the shell game the players get to pick a walnut and then they should get that message only if they picked correctly. We need to change our JavaScript to respond to what the user does. For that we need functions.

## JavaScript Functions

JavaScript functions are a way of naming different blocks of code and running them only when we specify. This will prevent our "You Won" message from happening every time a player goes to the page.

1. Replace your alert with the following function (named code). Refresh your page and no dialog box will appear. Now the code is there but waiting for us to use it. Pay special attention to the syntax.  
   
2. We also need a message in case the player doesn't pick the correct shell. Just under this function (after the closing curly brace } ) let's create that. Duplicate the function we just wrote and make some changes. The only difference between the function above and this second one is the name and alert message.
   1. name = TryAgain
   2. alert message = 'Wrong Shell - Try Again'

Note: You can put lots of functions inside one script element. There's no limit.

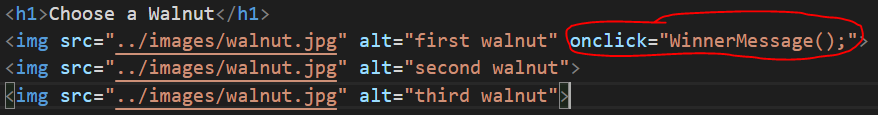
## Event Handling

If you've loaded our web page you see the page heading and three walnut shells. Nothing is happening. Perfect. We haven't asked the JavaScript code to run so it's sitting there waiting for us to use it. We are going to use event handling to do that.

Events can happen "onload" which means when an element in our HTML loads. They can happen "onclick" when a user clicks on something on the web page. Take a look at some of the other popular JavaScript events.

Table

Description automatically generated  
In our shell game we want something to happen when the user clicks a shell (onclick). For the purpose of our game let's make the first shell the one with the pea under it. The first shell is the winner.

1. Attributes that handle JavaScript events can be added to any HTML element. Our shell is in an img element. Add the attribute there. In the value of the attribute we are going to write JavaScript code to run the function that shows the winner message. To run a function in the attribute value just write the function's name with parenthesis and a semicolon at the end. 
2. The other two shells are losers and we want the user to get the try again message. Add the same onclick attribute to the second and third walnuts but have those run the TryAgain function instead of the WinnerMessage.

## Preview Next Week

1. In Visual Studio go to File -> Save As and save a second copy of your file as lab12\_preview.html.

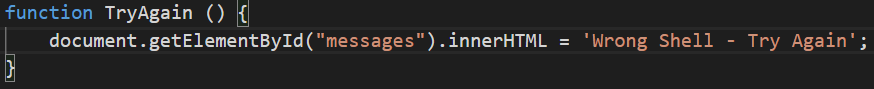
Alert boxes are a good JavaScript learning tool but they're kind of clunky. You will not see them used on production websites. Let's replace the alert boxes with a message that displays on the web page. First, we need a place to hold this dynamic text.

1. Add an empty div element below the walnut images. We will use this as a place to display the messages. Why is there an id attribute? JavaScript is going to use the id attribute value to locate this place on the page.  
   

JavaScript can change the contents of an HTML element based on an event.

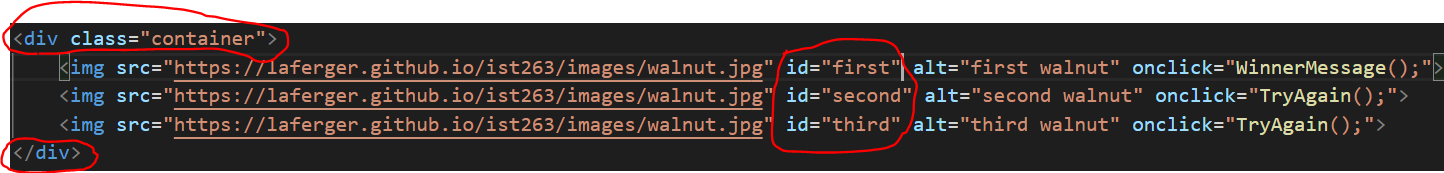
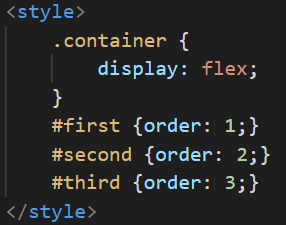
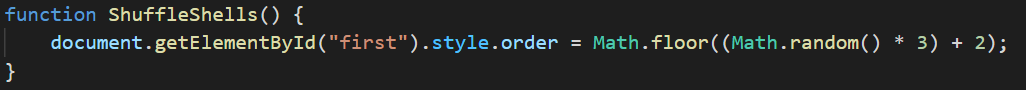
1. Replace the code inside your WinnerMessage JavaScript function with the following:  
   

document.getElementById() is a method that locates an element by it's id. Inside the parentheses you enter the value of the id attribute. The .innerHTML on the end says that we are going to change the text inside the element.

1. And we'll do the same change to the TryAgain function.  
   

Reload the page and we'll have a much nicer message coming up in the browser window.

This wouldn’t be much of a game if we didn’t change where the winning shell was located. To do this we are going to use flexbox to change the location of the shell.

1. Create a flex container around your images and give it a class with the value of container. Give each image and id of first, second and third respectively.  
   
2. Next, we’ll add embedded styles to create the flexbox. We are also going to add the flexbox order property. We’ll need the order established so we can change it later.  
     
   
3. The last thing we need to do is add JavaScript that dynamically makes a change to the order of the winning shell. Here’s the function to do that. It’s explained below. Add it inside your script tags.  
     
     
     
   **What are we doing above?**  
   document.getElementById("first").style.order tells JavaScript to look for an element with an id of first. The next part tells it to change the order property of the first element.   
     
   Math.floor((Math.random() \* 3) + 2) is a function that generates a random number. In this case we need to generate a 2, 3 or 4 randomly to change the order. JavaScript only generates random numbers between 0 and 1 so we have to do math to get the values we want.
4. The ShuffleShells function is not running right now. We’re going to add this function onload of the page so the flexbox order changes as the page is loading. In the body tag add:  
     
   

If you refresh the page, the winning shell should be in a different place. Play the game to see. Refresh again and it will change again. Now we’ve got a game!

## What will You hand IN?

Create a word document, pdf or use the "write submission" option in blackboard to provide the following:

1. Submit the url for the shell game with alert boxes.
   1. https://hrgordon-su.github.io/ist263/lab12/lab12practice.html
2. Submit the url for the shell game with messages displaying in browser.
   1. https://hrgordon-su.github.io/ist263/lab12/lab12\_preview.html
3. Answers to the following questions:
   1. Our shell game is very basic. What functionality do we need to add to make it more like a real game? List at least 3 improvements. I'm not looking for code here just your thoughts.
      1. One functionality that would make the game more realistic is animation. Allowing the player to following the moving shells
      2. Another functionality that would make the game realistic would be the game getting more challenging when you guess correctly
      3. Lastly, if you were to guess the correct shell there should be a function to lift up the shell and show the ball underneath the right one.
   2. In lecture and lab, we talked about the events: onclick, onblur and onload. Pick one of the other events in the list above and give me a scenario for how you could see it used on a web page.
      1. The event “onmousedown” could be utilized in a web page by an error message or options appearing when the mouse is clicked on anywhere on the page
   3. On this week's lecture slides, slide 7 lists example uses for JavaScript. What is another use for JavaScript? Some research will be needed.
      1. Another use for JavaScript is to add interactive behavior to web pages. For example, users would be able to Slide through a carousel of images on the homepage
4. Submit answers to the following:
   1. What questions did you have about the lab? What didn't you fully understand?
   2. What was the hardest part of the lab?
      1. I had a hard time troubleshooting when my images would not appear on my webpage. The original code I copied on to VS code did not have the proper image tags.
   3. Rate your comfort level with this week's topics.  
      1 ==> I can do this on my own and explain how to do it.