Week 7: Cookies in JavaScript

Agenda

- Overview
    - What's Due?
- Recap of Objects
- Cookies
    - How do we use Cookies?
    - Viewing Cookies in a Browser
    - Examples
- Cookies Challenge
- Next Week

Overview

Last week, we had our third bootcamp which covered JavaScript (JS). We briefly went over linking JS to our HTML, general syntax, and adding interactivity to our pages. This week, we will learn about Cookies and how they can be used on our web pages. By the end of this week, you should be able to add basic Cookies to a page and retrieve data from them.

What's Due

- Cookies Challenge

There are no discussion posts due this week. Please use any extra time to begin working on your Midterm. This will be due in a few weeks.

Recap of Objects

Last week, we learned a bit about Objects in HTML. Objects are entities that have their own properties (info about them), types, and functions/methods (things they can do). We use periods to link them together in our code which is sometimes called "dot notation."

A cat is an example of an object. Cats have different colors, sizes, fur, and do different things. Objects are the same in that they also have properties (attributes) and methods (things they do). The code below is an example of how a cat object might look in JS. "cat" would be the object, and "Meow();" would be a function we are referencing. On the other hand, "Color" is a property. We link them together with the period sign.

```
cat.Meow();
cat.Color;
```

One object we are familiar with is the document object. In JavaScript, we use the document object to access the DOM and read HTML. If we wanted to change the text of a paragraph tag, we could do it like this:

<u>HTML</u>

```
<p id="changeMe">This para will change.</p>
```

JS
```
document.getElementById("changeMe").innerHTML = "This para has been changed";
```

The outcome would be that the paragraph has the text highlighted in the JS above. The code is broken down below.

document is the object we use to refer to our page.

getElementById is a method of the Document object. Its purpose is to get an HTML element based on the provided id. The HTML element it grabs is an Element object.

innerHTML is a property of the Element object. When you use document.getElementById, you are using the Document object to get an Element object. innerHTML allows you to manipulate the HTML Element object to alter its text (stuff inside of quotation marks at the end).

How does this relate to Cookies? Well, the Document object has a Cookie property which we can use to set or get Cookies. Objects are also useful to understand for when we work with AJAX or PHP in future weeks.

In the next section, I'm going to go over the different parts of a cookie and how we can store them or access them later.

Cookies

While surfing the web, you have encountered cookies whether you have known it or not. Cookies are pieces of data that are stored on files inside of your computer from websites you visit. They are typically used to store information about a user to provide a more customized experience.

One example for business is the amount of "free articles" you may receive from a website. To track how many articles you've used, a website may put cookies on your machine. Each time you read an article, the number of article views will increment until you reach your maximum limit. The next time you try to read an article, a modal/popup will appear saying you need to pay/subscribe to see content (paywall). If you were to delete the cookies for that website on your computer (through the browser or in your file system), the counter for that would potentially reset and you could view more articles until the paywall appears again.

Normally, when you close your browser, the server you were connected to forgets all your information. Cookies give web developers a way to store information even when the browser/server are no longer connected. This week, we will learn about how we can implement Cookies on our own pages and use them to retrieve/store data. We'll also learn how to view cookies in our browser window.

Some important things to know are that cookies are limited in their size and scope. Cookies can only store up to 4KB of data and are private to the domain. This means a site can only read the cookies it created. There are also limits on the number of cookies that can be used, but that is dependent on the browser and how it is configured. Lastly, you can restrict how long a Cookie lasts and the domains on which it is used.

How do we use Cookies?

I will cover four parameters to consider for cookies and they are keys/names, values, expiration dates, and paths. When creating a cookie, the key/name and value properties are required, while all the other properties are optional. There are other parameters that can be used as well, but to keep things simple, we will only review these four.

You can get to all the cookies on a page with “document.cookie;”

We use the document object’s cookie property to return a string with all of the cookie information together. In the line below, I create a variable called “allCookies” to retrieve all the cookies present in the area of the website we are working in.
var allCookies = document.cookie;

allCookies would now be a name/value (name=value) string of **all** cookies present separated by semicolons. It would look something like this:

“name1=value1; name2=value2; name3=value3;”

This is one way to get all cookies, but what if we wanted to create a cookie or get one cookie by itself?

Setting up Cookies

If you wanted to store your netID as a cookie, you could do it with the following code:

document.cookie = “netid=cv762525”;

Noticed that the name of the Cookie and its value are separated by an equal sign. This is all it takes to set up a very basic Cookie on a web page. However, this would only set the netid key/name and cv762525 value. The expiration and path would have default values (expire when browser closes and current document is the path).

Keys are the names you are giving your data (netid above). Values are the pieces of data associated with those names (cv762525). Expiration dates determine how long a cookie will be stored on a person’s computer. If you don’t provide a date, a cookie may disappear as soon as the browser is closed (known as a session cookie). Lastly, we have the path; this is used to determine where on the server a cookie will be active. If you don’t provide a path, it will default to the current document location.

Setting a cookie with all four parts requires a little work, but it’s very doable. Each parameter should be separated by a semicolon followed by a space. The first part is easy, you just need your key/value pairing. For our example, we’ll use “netid” for the key, and “cv762525” for the value.

After we have our key/value info, we should set an expiration date. One method for this is using the “expires” word. You would pair this with some fixed expiration date for a precise expiry date. The format for the date must be in a UTC/GMT format as follows:

<day-name>, <day> <month> <year><hour>:<minute>:<second> GMT.

This would translate to this potentially: “Monday, 7 March 2022:12:00:00 GMT.”

```
document.cookie = “netid=cv762525; expires=Monday, 7 March 2022:12:00:00 GMT;”;
```

That would create a cookie called netid with a value of cv762525 that expires on March 7th at midday (based on military time). Normally, you wouldn’t provide an exact date as I did above. You could create a date object with a date as shown below:

```
var myDate = new Date(2022, 3, 7);  // year, month, day
```

When you want to set the expires time, you could refer to myDate and add “.toUTCString()” to it. The .toUTCString() method is used on Date objects to convert them to a UTC format. The result would look like this:

```
var myDate = new Date(2022, 3, 7);
document.cookie = “netid=cv762525; expires=” + myDate.toUTCString()+ “;”;
```

Besides expires, you can use max-age to set an expiration date. The difference is that the expiration occurs a certain amount of time after the user receives it. Also, the format for the date used in max-age is in seconds.

The last parameter for setting a cookie that I will cover is path. The path determines the location for the cookie. If you don’t set a path, the current document becomes the default location. To set it, you can use “path=/” and that will mean use the current location. This is good for global cookies you want to use on any internal page on a website.

Piecing all four parameters together could be done with this code:

```
var myDate = new Date(2022, 3, 7);
document.cookie = “netid=cv762525; expires=” + myDate.toUTCString()+ “; path=/”;
```

A cookie named netid with a value of “cv762525” would be created. This cookie would expire on March 7th, 2022 and the path for it would be the current document.

<u>Updating or Deleting a Cookie</u>
You can update or delete a cookie in the same way that you create it.

If I wanted to **overwrite/update** the “netid” cookie to have a different value, I would just create the cookie again but change the value provided.

```
document.cookie = “netid=otherid; expires=” + myDate.toUTCString()+ “; path=/”;
```

My original value of “cv762525” would become “otherid” when the code is executed.

If we wanted to **delete** the cookie, you would use the same syntax. The major difference would be leaving the value blank and changing the date to an older date. In the example below, I refer to the myDate variable created earlier. I’m going to change it to March 6th (before this week) and when I use that date in my JS statement to create a cookie, the Browser will delete it since the date is considered expired. With more complicated cookies, it's important to specify a path as well. Omitting a correct path may result in the browser not deleting the cookie at all.

```
myDate = new Date(2022, 3, 6); // Day is now 6 instead of 7.
document.cookie = "netid=; expires=" + myDate.toUTCString()+ "; path=/";
```

No value was provided and the date is older than the current date, so this cookie will be deleted.

Getting a Cookie
As I mentioned before, you can use "document.cookie" to retrieve all of the cookies currently being used in a given context.

```
var allCookies = document.cookie;
```

The problem is that this returns **all** cookies and not any specific one. Additionally, the cookie information is stored in a way where the string needs to be manipulated to extra a single cookie. The data is returned in a format similar to what is depicted below.

```
"name1=value1; name2=value2; name3=value3;"
```

If you want to get a specific name/value pairing, you'll need to manipulate that string so that you can access them. You'll need a function which takes in the string and splits it up into usable pieces. A popular way to do this is using the .split() method. If you use this on a string, it will use a character provided to split up that string into an array.

```
var allCookiesSplit = allCookies.split(";");
```

Splitting the string by a semicolon would give you an array that looks like the following:
allCookiesSplit[0] – "name1=value1"
allCookiesSplit[1] – " name2=value2"
allCookiesSplit[2] – " name3=value3"

There are 3 elements in total, and you'll notice that some of them have an extra space at the front. This is because when we split by semicolons, all characters left over are saved to each index in the array. Cookies typically have a space directly after each semicolon which can affect how you get their data.

To get more accurate values, we could use any number of methods to clean up the array. One that I like to use is the trim() method.

```
allCookiesSplit[1].trim(); // This would change " name2=value2" to "name2=value2"
```

After we've cleaned up our cookie name/value, we can then use various methods to extract either piece of information and use it on our page or somewhere else.

Rather than explain that code in this document, I've created two examples for you to review. Each example will demonstrate how we can set cookies on our page. The "get-cookies-example.html" page will show you how to set cookies, but also ways to retrieve specific cookies and their values on the page. Make sure to review the JavaScript on the page to gain a better understanding of how we can use cookies.

Example Links

- https://www.albany.edu/~cv762525/cinf362/examples/set-cookies-example.html
- https://www.albany.edu/~cv762525/cinf362/examples/get-cookies-example.html

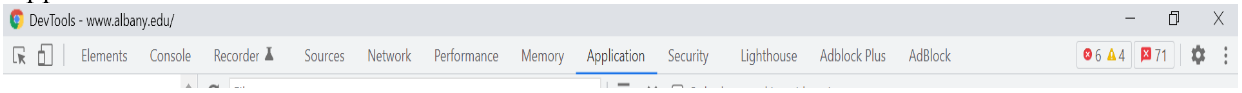
Viewing Cookies in a Browser

To view cookies in a browser, you need to use a browser's web tools. I've listed the steps below with some screenshots so you can find the area using Albany.edu as my example.

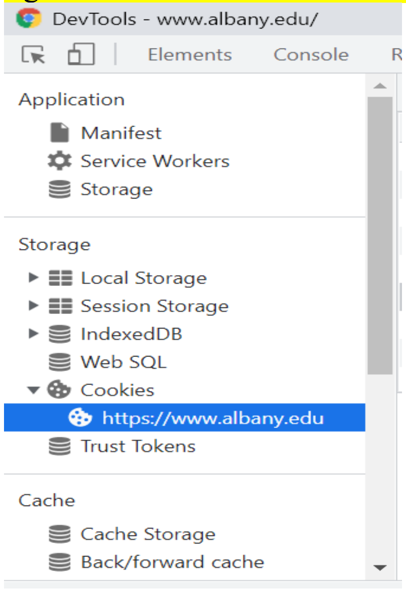
- Right-click a page and inspect it or click F12 if you are on a PC.
- Navigate to the "Applications" toolbar.
- Under Storage on the left-hand side, click the arrow next to "Cookies"
- Click on of the links for Cookies
- Review Cookie information in the panel

Please note that you cannot use Cookies in a local file (file://). To use Cookies, your file must be uploaded to a web server (Albany or 000webhost) first, and then your Cookies should appear inside of your developer tools area. Depending on what you use, it might take some time for your page updates to appear in the browser. Sometimes, Albany servers cache an old version of your page for a few minutes before new updates are reflected. Just be patient and refresh your page once in a while if your code doesn't appear to be updating.

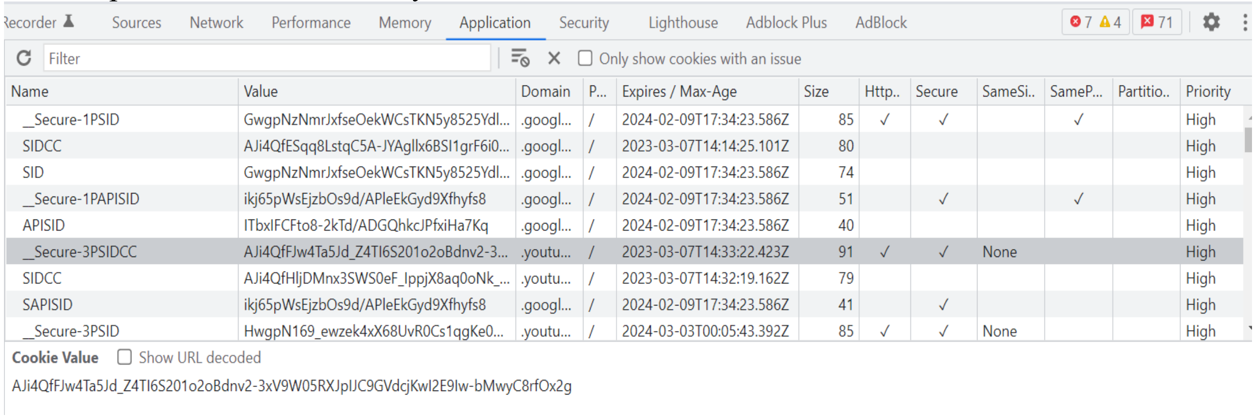
Application Toolbar:



Storage area where data is displayed. The Cookies dropdown menu contains all relevant domains/sources where cookies are being used. If you want to clear cookies for a website, simply right-click the link under "Cookies" and select "Clear."



Cookie data is displayed in the panel. You can click any of the Names and see information about it in the panel below where it says, “Cookie Value.”



Name	Value	Domain	P...	Expires / Max-Age	Size	Http..	Secure	SameSi..	SameP...	Partitio..	Priority
__Secure-1PSID	GwgpNzNmrJxfseOekWCsTKN5y8525Ydl...	.googl...	/	2024-02-09T17:34:23.586Z	85	✓	✓		✓		High
SIDCC	AJi4QfESqq8LstqC5A-JYAgllx6BSI1grF6i0...	.googl...	/	2023-03-07T14:14:25.101Z	80						High
SID	GwgpNzNmrJxfseOekWCsTKN5y8525Ydl...	.googl...	/	2024-02-09T17:34:23.586Z	74						High
__Secure-1PAPISID	ikj65pWsEjzbOs9d/APleEkGyd9Xfhyfs8	.googl...	/	2024-02-09T17:34:23.586Z	51		✓		✓		High
APISID	ITbxIFCFto8-2kTd/ADGQhkcJPfxiHa7Kq	.googl...	/	2024-02-09T17:34:23.586Z	40						High
__Secure-3PSIDCC	AJi4QfFJw4Ta5Jd_Z4TI6S201o2oBdnv2-3...	.youtu...	/	2023-03-07T14:33:22.423Z	91	✓	✓	None			High
SIDCC	AJi4QfHljDMnx3SWS0eF_IppjX8aq0oNk_...	.youtu...	/	2023-03-07T14:32:19.162Z	79						High
SAPISID	ikj65pWsEjzbOs9d/APleEkGyd9Xfhyfs8	.googl...	/	2024-02-09T17:34:23.586Z	41		✓				High
__Secure-3PSID	HwgpN169_ewzek4xX68UvR0Cs1qgKe0...	.youtu...	/	2024-03-03T00:05:43.392Z	85	✓	✓	None			High

Cookie Value ☐ Show URL decoded

AJi4QfFJw4Ta5Jd_Z4TI6S201o2oBdnv2-3xV9W05RXJpIJC9GVdcjKwI2E9Iw-bMwyC8rfOx2g

Examples
New JavaScript examples have been added to the examples area. They are listed below:
- https://www.albany.edu/~cv762525/cinf362/examples/set-cookies-example.html
- https://www.albany.edu/~cv762525/cinf362/examples/get-cookies-example.html

Here’s a link to all course examples:
https://www.albany.edu/~cv762525/cinf362/examples/ (coded examples)
https://www.albany.edu/~cv762525/cinf362/videos/ (videos)

There is also content on Blackboard to help you with understanding/debugging JavaScript code. These items are located on Blackboard → Resources → JS and will help further explain JavaScript and demonstrate some best practices you should adopt moving forward.

Online Resources (If you don’t like my content, here are some alternatives)
- https://developer.chrome.com/docs/devtools/storage/cookies/?utm_source=devtools (Viewing Cookies)
- https://www.tutorialspoint.com/javascript/javascript_cookies.htm
- https://www.w3schools.com/js/js_cookies.asp

Before completing the assignment(s) for this week, please read the “Viewing Your Web Pages.docx” file on Blackboard. You will not be able to submit anything for the assignment without completing that portion first. It is located directly in the Lecture Notes folder.

Cookies Challenge
Due Sunday, March 13th at midnight
Download the “Cookies-Challenge.zip” folder from Blackboard under this week’s Lecture Notes or the Assignment folder. Inside of the zip folder will be an HTML file to use for this assignment.

The JS and HTML are already provided/linked for you. I've placed an internal script tag inside of the HTML to keep things simple. If you want to place your JS code externally, that is fine as well. Just remember to submit all relevant files to Blackboard.

For this challenge, you will use cookies to store a user's name, preferred language, and the number of times they have visited the page (for any period, expiration doesn't matter). You will also use one of those cookies to create output on the page each time a user visits it.

I have provided a form, an output area, and JS functions for you to use. Your task is to add more JS to the script tag so that the content appears on the page. I've left comments throughout the document where code was initially. You will place JS under those comments based on what the comment says. The output should be as depicted below.

First Visit
"This is your first visit. Please fill out the form below."

Third Visit:
"Welcome back! This is visit #3 for you."

Example Output When Form is Submitted (Chris/English as inputs)
"Thank you for submitting your information, Chris! Your preferred language has been set to English."

For this challenge, the 3 cookies you should be creating are "firstName", "preferredLang", and "numVisits." Two of the cookies will receive and set their values from the form (first name and preferred language). The last cookie will be based on how many times the user has visited the page.

On the first visit, the page should say, "This is your first visit. Please fill out the form below." In the output area. When the user fills out the form, their First Name and Preferred Language should be saved as cookies and a message should appear as depicted above under "Example Output…"

For this challenge, you will need to check to see if the user has a "numVisits" cookie set on their computer. If that cookie is not set, you should add that cookie and then create output on the page telling the user that it is their first visit (as shown above).

If the cookie is present, then what you'll do is get the number of visits from the cookie, increment it by one, and then display the output letting the user know how many times they have visited.

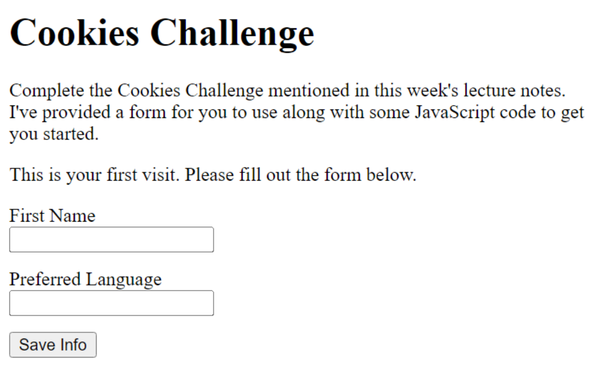
For the form part, you should be taking in user input using the .value property. After you have the user's values, you should check to see if they are empty or not (empty quotations marks –""). If the user provides any empty inputs, you should produce a message telling them to provide valid information. If the information is present when they submit the form, the values should be used to create output as shown under "Example Output…"

All output will go to the same place on the page. <p id="infoOutput"></p> is the container you will use for the information.

I've left comments throughout the JS to describe what you should be adding. If you see a comment with no code directly beneath it, that's an area where you should add JS. Feel free to start from scratch if you'd like. However, I would recommend following along with my comments and viewing the class examples as they have the code necessary to complete this assignment.

Here are some screenshots to help guide you as far as the expected output. I decreased the width of my page to make the screenshots smaller.

First Visit:



Cookies Challenge

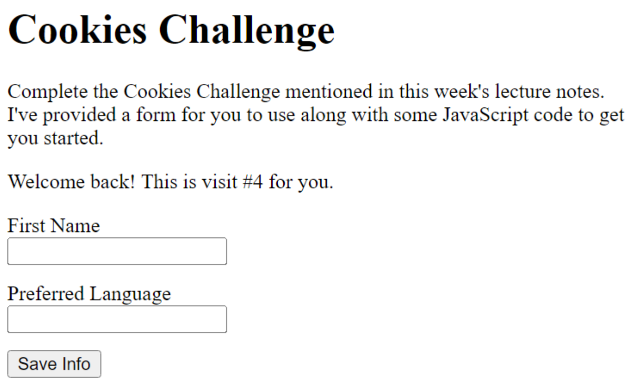
Complete the Cookies Challenge mentioned in this week's lecture notes. I've provided a form for you to use along with some JavaScript code to get you started.

This is your first visit. Please fill out the form below.

First Name

Preferred Language

Fourth Visit (after refreshing three times)



Cookies Challenge

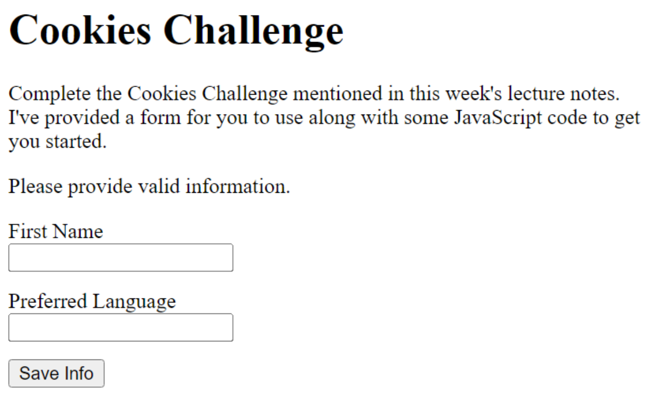
Complete the Cookies Challenge mentioned in this week's lecture notes. I've provided a form for you to use along with some JavaScript code to get you started.

Welcome back! This is visit #4 for you.

First Name

Preferred Language

After Filling out and Submitting the Form with Invalid Information:



Cookies Challenge

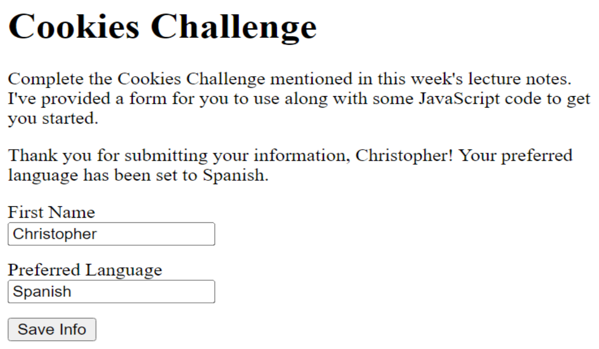
Complete the Cookies Challenge mentioned in this week's lecture notes. I've provided a form for you to use along with some JavaScript code to get you started.

Please provide valid information.

First Name

Preferred Language

After Filling out and Submitting the Form with Valid Information:



Cookies Challenge

Complete the Cookies Challenge mentioned in this week's lecture notes. I've provided a form for you to use along with some JavaScript code to get you started.

Thank you for submitting your information, Christopher! Your preferred language has been set to Spanish.

First Name: Christopher

Preferred Language: Spanish

You'll know you have completed the assignment if the output being produced is as depicted in the screenshots above. You can also check your cookies in the Applications area of the Developer Tools as demonstrated in the "Viewing Cookies in a Browser" section.

If your JavaScript isn't working at all, it's probably because the src value you've provided in the script tag is incorrect. If some JavaScript works, but other parts don't, look at your JavaScript in the console. Inspect the page and then go to the console tab. Any lines of code with an error will appear there. The error is usually a misspelling or incorrect reference to something on the page.

Your webpage is **due on Sunday, March 13th at midnight.** To submit the work for this exercise, visit Blackboard → Course Materials → Lectures Notes for this week's class. You can also go into the "Assignments" folder and the submission area will be titled "Cookies Challenge." You should be submitting a live link to me; it can be through the UAlbany server or 000webhost. The work submitted will be evaluated based on the rubric explained below.

Cookies Challenge Rubric – 10pts

- Live link submitted – 1pt
- JS is organized/easy to read – 1pt
- Cookies are set correctly – 4pts
- Output is generated correctly – 4pts
  - For # of visits
  - For user name/preferred language
  - For empty inputs being submitted

Next Week

Next week we will start learning about server-side technologies such as PHP which allow us to add more features to our websites. The best resource for PHP is php.net but other websites have good tutorials as well. PHP, like JS, is a programming language but it's useful for manipulating HTML and working with forms and databases.

- https://www.php.net/manual/en/intro-whatis.php
- https://www.php.net/manual/en/intro-whatcando.php
- https://www.w3schools.com/php/ (more of a reference)