6G5Z2107: Web Design and Development

# Laboratory 05 – Cookies and Local Storage

John Darby

# Objectives

1. Recall how to serve up HTML using PHP
2. Send data to your server-side scripts using GET requests
3. Use that data to tailor the HTML you serve
4. Store user data in cookies for future re-use

The aim of today is to reinforce the concepts covered in this week’s lecture, and prepare for our use of cookies in 2CWK50.

# Warm up

Complete the following questions to check you can remember the key new ideas and syntax from the lecture. **Please type your answers into this Word document and save it to your H: drive so that your tutor can discuss your answers with you.**

|  |  |
| --- | --- |
| Question: If I edited the copy of basic.html on Mudfoot would everyone in this lab be able to see the changes?   |  | | --- | | Answer: yes because its widely accessible and not stored localy like xampp | |
| Question: When I call System.out.println("Hi!") in Java the output goes to the command line. When I call echo("Hi!") in PHP where does the output go?   |  | | --- | | Answer: website ie. Index.php and it will be visible as paragraph | |
| Question: Can people accessing our scripts on a server like Mudfoot see our PHP code?   |  | | --- | | Answer: no | |
| Question: What is an associative array in PHP?   |  | | --- | | Answer: its an virtual table that stored some data type | |
| Question: What is Unix time?   |  | | --- | | Answer: It’s a time describing point in time since 1970 till now. | |
| Question: Consider the following request taken from a browser address bar. How would you modify the request to pass a second "surname" parameter to myscript.php:  www.example.com/myscript.php?name=brian   |  | | --- | | Answer: ?name=brian&surname=surname | |
| Question: Consider the following request taken from a browser address bar. How could you access the “name” parameter from the PHP code inside myscript.php:  www.example.com/myscript.php?name=brian   |  | | --- | | Answer: echo $\_GET[“name”]; | |
| Question: The PHP function time() returns the current Unix and the code below returns the Unix time in one hour from now. How would you return the Unix time in one week from now?  $inOneHour = time() + 60\*60;   |  | | --- | | Answer: $inOneWeek = $inOneHour + 24\*7 | |
| Question: How can you store the PHP value $myValue on a client's machine in a cookie called “user”   |  | | --- | | Answer: setcookie(user,$myValue) | |
| Question: You have previously issued a cookie called “user” to a client. How can you access the value stored inside your cookie when the client next accesses your PHP script?   |  | | --- | | Answer: We can’t, we need to issue a new “user” cookie | |

# Exercise 1: One Cookie, Multiple Scripts

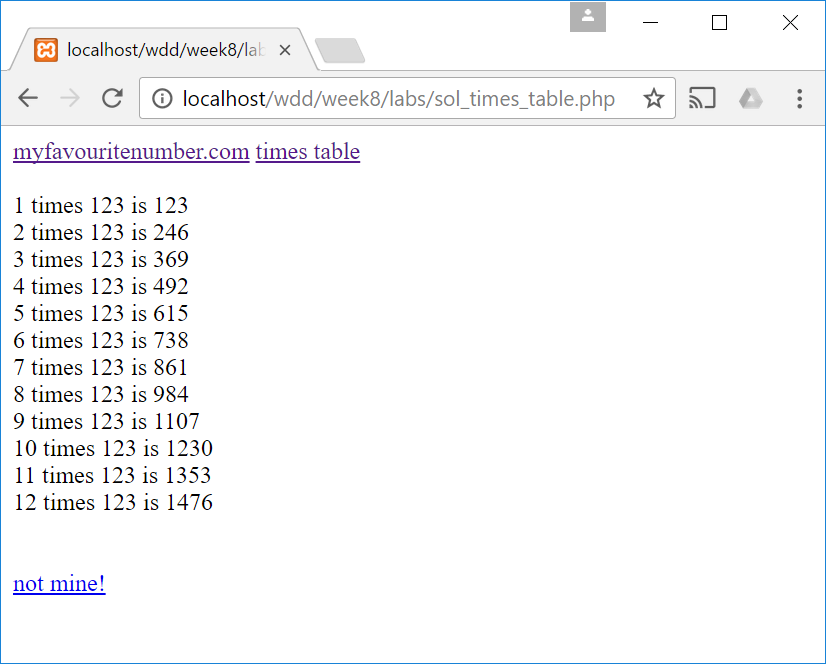
This week we created a cookie-driven version of the “myfavouritenumber.com” website. This is the script favourite\_cookies.php on Mudfoot:

* <http://mudfoot.doc.stu.mmu.ac.uk/students/john_student/PD1/>

Let’s start off by extending this example.

Download the exercise files for this week from Moodle, and copy the favourite\_cookies.php script into your xampp/htdocs/ folder. Test your new script and make sure it works as you expect. You should be able to find the “favourite” cookie that it creates on your computer through your browser settings (amongst the cookies belonging to “localhost”). **Take time to read and understand the code and how it’s working. Ask for help if you’re not sure about anything.**

Now create a **new script** which uses the favourite number stored in the cookie to print out a times table (see image).

Think about the following:

1. How can you check to see if there is already a “favourite” cookie? (Tip: you should be able to reuse code from the original favourite\_cookies.php script.)
2. How could you display a useful message to the user if the cookie doesn’t exist?
3. You should be able to use a simple loop to display the times table itself. The syntax for looping in PHP is almost identical to Java! A quick Internet search should help if you’re having trouble remembering.

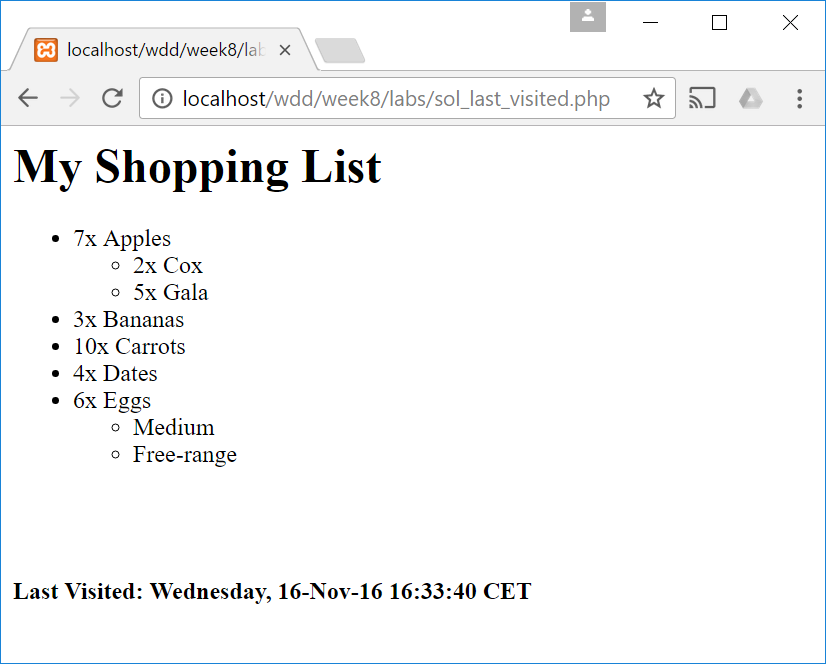
Follow-up question: If your second script was moved into a different directory on your server, how would you need to change the way you set the cookie in the favourite\_cookies.php script for everything still to work?

# Exercise 2: Last Visited

In this second exercise, you’re going to look at *updating* the contents of a cookie by extending some client-side code you have developed in a previous week. I’ve used the “shopping list” example from week 2 in the image below, but you can extend any code you like.

Move the client-side code you want to use into a new PHP script (you will probably want to make use of a heredoc). Ask for help if you get stuck. Your challenge now is to use a cookie to display information about when each user last visited your script (see the example in the image below). If it’s the first time they have visited, then you should display a welcome message instead. If a user doesn’t return to your script for over a year, then you should treat them as a new visitor.

Think about the following:

1. You will need to write a string containing a nicely formatted date into your cookie. The PHP date() function can help you here and you should look it up.
2. As in the last exercise, you’ll need to check to see if a cookie already exists on the user’s machine. If not, how can you display a “welcome” message instead?
3. This exercise requires you to *update* the information stored in user cookies. To do this you can simply set the cookie again, there’s no need to destroy it first.

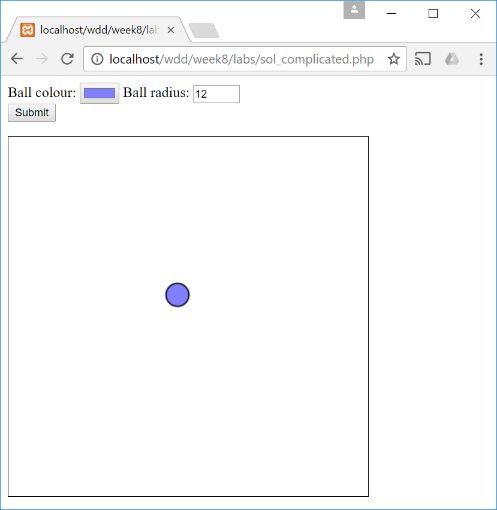
# Extension Exercise 3: Bouncing Balls

Cookies are fine for storing single plain text values, but what if you want to store several different values? This extension exercise encourages you to think about this problem, and to investigate some of the other types of data that HTML forms can help you send to the server.

Your challenge is to extend the “bouncing ball” code (see complicated.php) so that it allows the user to change both of the following parameters:

1. The radius of the ball
2. The colour of the ball

Here are some pointers to help guide you:

1. HTML5 defines form elements for selecting both numbers and colours. Have a look at the type="number" and type="color" options on the page in reference [1].
2. A string and a number will sit happily alongside one another in a PHP array. So, one nice approach to this problem would be to store all the ball’s parameters in an associative array with usefully named elements (e.g., parameters["radius"]) and write this array to a cookie.
3. A problem with this approach is that cookies can only store plain text, and setcookie() will only accept a simple string or number as its second argument. You can use the PHP function serialize() to convert your array into a plain text representation before storing it in a cookie, and unserialize() to turn the data you read back from your cookie into an array!
4. If you’re interested in alternative approaches, see the use of “array cookies” in reference [2]. This special syntax stores a separate cookie for each element in your array, but reads them back into $\_COOKIE as a single associative array.

[1] <http://www.w3schools.com/html/html_form_input_types.asp>   
[2] <http://php.net/manual/en/function.setcookie.php>

# Help! I don’t know what to do!

Don’t panic!

There are plenty of resources available, including the following, which are all great starting points:

* The lecture slides (available on Moodle) which recap what we covered in the lecture
* w3school’s PHP tutorial: <http://www.w3schools.com/php/>
* w3school’s cookies tutorial: <http://www.w3schools.com/php/php_cookies.asp>
* Lynda.com – PHP tutorials: <https://www.lynda.com/PHP-training-tutorials/282-0.html>
* Lynda.com – a cookies tutorial: <https://www.lynda.com/MySQL-tutorials/Working-cookies/119003/136994-4.html>

If the PHP topics we’ve covered this week feel unfamiliar then now might also be a good time to look back at your notes from last year’s “Introduction to Web Design and Development” unit.

Exercise 1 extends the example from the lecture, and getting comfortable with this example and how it works is an important first step. Don’t worry if you need to spend a lot of time reading and understanding the original script before trying to add your new script. This is time well spent. Make sure you ask for help if you’re struggling to understand how it works.

Exercise 2 feels trickier because you’re being asked to write a script from scratch rather than modifying existing code. But actually there is lots of overlap with exercise 1 and potential to reuse ideas and code from your exercise 1 solution.

You shouldn’t attempt the extension exercise until you are completely comfortable with the first two exercises. If you are having difficulties with them then please ask your tutor for some help.

# I fancy a challenge, what do you recommend?

Two main options here. The first is to look at another option for local storage of data on the client: HTML5 local storage. But keep in mind that, as we said in the lecture, this storage is very different from cookie storage: it’s not accessible from the server-side. The following book chapter is very interesting and covers some of the historical context: <http://diveinto.html5doctor.com/storage.html>. The second option is to start reading up on PHP sessions. Sessions take the other approach we touched on in the lecture and store user data on the *server*. However, they use cookies in order to do this! We’ll be looking at sessions next week, but if you feel confident about how cookies work you’re in a position to have a look at sessions in advance if you wish! W3Shools is a good place to start: <http://www.w3schools.com/php/php_sessions.asp>