Sessions and Cookies

So far, we have only passed information between pages via a form post. When a form's submit button is hit, it can return to the same page, or use the **action** attribute (within the form element) to forward the $_POST data to another webpage.

However, for many websites you need to move data between many pages. For example, you don't want to lose shopping cart data once you navigate away from the “Add Product” page.

The Web is stateless – Unless we store information via sessions, cookies, files, or databases, a state is never passed between one page and another. This requires some creative coding to create productive websites. This isn't a challenge that most software faces.

Sessions

Using Sessions is a short-term way to store data. Sessions are usually cleared when the web-browser is closed or when the user logs out. This would be good for storing a user's shopping cart for this visit to the web-page, or making sure the user stays logged-in after they fill out the login form the first time.

Storing data in the session is simple, the session variable, **$_SESSION** is an associative array (aka, a key-value array) similar to $_POST and $_GET. You can freely assign items to this array:

```
$_SESSION["username"] = $_POST["username"];
```

However, before you start storing information in the session array, you must call **session_start();** function before any HTML code:

```
<? session_start(); ?>
<!DOCTYPE html>
<html>
      <head>
            <title>Session Page 1</title>
      </head>

      <body>
            …
      </body>
```

```
</html>
```

* Note that **session_start();** needs to be called at the beginning of any page using sessions. *

Session information can then be accessed across various pages.

When you're done with the session information and want to explicitly destroy the session, you can use the **session_destroy();** function.

```
<?
// Destroy session
session_destroy();
?>
```

That's about all there is for sessions! You could combine file I/O to load in information into the session, or eventual database queries!

Cookies

Cookies are longer-term storage (but still temporary) that is stored on a user's local machine. A user can clear their cookies and history at any time, so cookies are not a good choice for long-term storage. They can be useful for improving the user's experience, however, saving data from their last visit for easier use in subsequent uses.

Using cookies is a little different than using sessions. You can store a new cookie with the **setcookie** function, which takes three arguments: cookie name, cookie value, and the expiration date.

```
if ( isset( $_POST["save-data"] ) )
{
	// Name, Value, Expiration
	// Current time, plus a month, which we have to convert into seconds.
	$expirationDate = time() + ( 60 * 60 * 24 * 30 );
	setcookie( "mysitecookie_username", $_POST["username"], $expirationDate );
	setcookie( "mysitecookie_email", $_POST["email"], $expirationDate );
}
```

Then, to access a cookie, you just need the **$_COOKIE** array:

```
<pre>
	<? print_r( $_COOKIE ); ?>
</pre>
```

Which will result in:

```
                        Array
(
    [mysitecookie_username] => asdf
    [mysitecookie_email] => fdsa
    [PHPSESSID] => c39468761b414eabbffbcdb6901b11b5
)
```

Other Resources

More resources to help you understand Sessions and Cookies:

- http://www.tizag.com/phpT/phpsessions.php
- http://www.tizag.com/phpT/phpcookies.php
- http://www.w3schools.com/php/php_sessions.asp
- http://www.w3schools.com/php/php_cookies.asp