# Cookies & Sessions

Chapter 14

- HTTP is called a **stateless protocol**. It doesn't remember anything about the communication between server and client.
- Most websites allow some kind of "stateful" information to be shared between pages.
- You are able to "log in" to a website because of this.
- A website may remember you for future visits so it can customize the page for you.
- An IP address alone is not enough to "remember" a user.
- **Cookies** are small pieces of data that allow for more sophisticated interactions between you and a web site. They are sent from the server to the broser. On repeated visits, cookies are sent from the browser back to the server.

- Common uses for cookies include user logins, user tracking, site personalization, maintaining preferences, settings, shopping carts and advertising.
- Any type of information stored in a cookie.
- A cookie is a name/value pair stored as text strings.
- Cookies should never be used to save sensitive information.
- You can't rely on a browser to store the cookie and send it back.

## Cookie Myths

- Cookies are viruses or worms that can erase personal data.
- Cookies can see what I'm typing.
- Cookies are like spyware and can steal your personal information.
- Cookies generate pop-ups and spam.
- Cookies are only used for advertising.
- Cookies were seen on the grassy knoll.

## Cookie Facts

- Cookies only contain data (no code).
- Cookies cannot read any information from your computer.
- Cookies usually don't contain personal information.
- Cookies can be used by a particular site to track your viewing habits.
- Cookies are associated with a particular site domain name and are sent only on requests to that specific site. A cookie for one site will never be sent to a different site.
- Cookies may be stored in some encrypted format.

- **Tracking cookies** can help a site observe your browsing history on that site.
- It is only able to remember what pages you visited on that particular site.
- Advertisers can exploit this if 2 web sites use the same advertising company. An image from site A can view cookies for site A and the same image on site B can view cookies for site B.

- As with real cookies, there is an expiration date with cookies.
- **Session Cookie** - a cookie that is being used to store information about the user's current interaction with the site over a short period of time. Generally stored in RAM.
- **Persistent Cookie** - a cookie with an expiration date far into the future. Generally stored on the hard drive. Most persistent cookies last a few weeks.
- Websites often ask you if you are logging in from a public computer. Your answer determines the type of cookie.

- Cookies can be created and stored using server side programming or client side programming.
- **setcookie(name, value)** sets a cookie with a name/value pair. By default, the cookie is a session cookie.
- **setcookie(name, value, expire)** identical to the above function with an expiration date. 0 indicates a session cookie, a negative value will delete the cookie.

```php
//session cookie
setcookie("favoriteClass", "CS346");

//persistent cookie, expires in 1 hour
$expire = time() + 3600;
setcookie("favoriteProfessor", "Krohn", $expire);

//delete a cookie
setcookie("favoriteClass", "", -1);
```

- Due to HTTP limitations, setcookie must be called before any HTML output is generated.
- Once a cookie has been set, the browser will send it back on subsequent visits.
- PHP uses a superglobal associative array to access cookie values. It is called: $_COOKIE["name"];

```
<?php
//assume fname and lname were set already
if(isset($_COOKIE["fname"] && isset($_COOKIE["lname"])){
?>    <p> Welcome back <?= $_COOKIE["fname"] ?>
          <?= $_COOKIE["lname"] ?>! </p>
<?php
} else{
?>
      <p> Please log in... </p>
<?php
}
?>
```

- Expiration dates rely on using date and time.
- Below are some examples of date and time methods in PHP.
- There are many ways to display the date. Read all about them at http://us2.php.net/manual/en/function.date.php

```
//returns number of seconds since midnight
//on January 1, 1970.
$rightNow = time();

//assume it is 3:45 in the afternoon, this
//would print out 3:45 PM
echo date("g:i A", $rightNow);

//this would print 15:45:00
echo date("H:i:s", $rightNow);
```

- Cookies are usually used by the server.
- JavaScript cookies are generally used to remember something about the page state. For example, a checkbox being checked or text in a textbox.
- http://www.w3schools.com/js/js_cookies.asp provides some nice functions for dealing with cookies.

```
//session cookie
document.cookie = "favoriteClass=CS346";

//persistent cookie, expires at the end of the semester
document.cookie = "favoriteProfessor=Krohn; expires=Fri,
    12 Dec 2014 23:59:59 GMT";

//displays the following:
//favoriteClass=CS346; favoriteProfessor=Krohn
alert(document.cookie);
```

- Cookies are useful but have limitations.
- **Session** - set of related page requests from one client.
- Most complex systems use a combination of cookies and sessions.
- In many settings, the server sets 1 cookie containing 1 piece of information: **session ID number**.
- The server keeps track of a large associative array of sessions. The session ID is the key, the information (username, shopping cart items, etc) are the data.

## Basic Session Communication

- Client arrives at its first page.
- Server creates a unique session ID for that client.
- Server sends back a page response along with a cookie containing the session ID.
- On future page requests from the client, the client sends the request along with the session ID.
- The server uses this session ID to look up the user's information.
- When the client no longer requests pages from the server, the session ID cookie will time out. Default timeout for PHP is 24 minutes.

## Session Benefits

- Sessions can store more data. Cookies can only store a few kilobytes of data.
- As a corollary, sessions save bandwidth. Only the session ID is sent back and forth.
- Session data is more secure. Information is stored on the server and is not being sent back and forth between machines.

- **session_start** - function that must be called before any HTML is produced. Generally the first statement in the document.
  - Server looks at current cookies sent by the client to see if a sessions already exists.
  - If not, a new ID is created and sent to the client.
  - If there is a session, any data associated with the session is loaded.
- Once the session begins, data about the session is stored in a superglobal associative array called $_SESSION.
- A complete list on session functions can be found at http://php.net/manual/en/book.session.php

- Assuming you start the session at the top of each page, session variables will be accessible on every page of your website.
- Session variables will timeout after roughly 24 minutes. They are useful temporary variables.
- If you need data stored permanently, write it to a file or store in a database.

```
<?php
    session_start();
    //later in the code
    $_SESSION["someName"] = someValue; //store
   variable
    alert($_SESSION["someName"]); //access variable
    unset($_SESSION["someName"]); //delete variable
?>
```

- If you need to end a session and start a new one, the code below is how you would do it.

```php
<?php
    session_destroy();
    session_regenerate_id(TRUE);
    session_start();
?>
```

Example

Let's create a simple login page. We will have a few pages, a login page, a page to verify the login was successful, a page to log out and another random page to ensure our session works. Our login page shouldn't be accessible if the user is already logged in. Similarly, none of the other pages should be accessible if the user isn't logged in.