11.9 JavaScript: Using Cookies

- **Cookie**
  - Data stored on _____________ to maintain information about client during and between browser sessions
  - Can be accessed through `document.cookie` property
  - Set expiration date using `expires` keyword
  - Use `escape` function to convert non-alphanumeric characters to hexadecimal escape sequences
  - `unescape` function converts hexadecimal escape sequences back to English characters
Storing Cookies – Simple Version

document.writeln("<br/>Cookie is: " + document.cookie);

document.cookie = "name=" + escape("J Smith");
document.writeln("<br/>Cookie is: " + document.cookie);

document.cookie = "rank=" + escape("Captain");
document.writeln("<br/>Cookie is: " + document.cookie);

Cookie Example #1

// reset the document's cookie if wrong person
function wrongPerson() {
    // reset the cookie
    document.cookie = "name=null;" + " expires=Thu, 01-Jan-95 00:00:01 GMT";

    // after removing the cookie reload the page to get a new name
    location.reload();
}

// determine whether there is a cookie
if (document.cookie) {
    var myCookie = unescape(document.cookie);
    // split the cookie into tokens using = as delimiter
    var cookieTokens = myCookie.split("=");
    // set name to the part of the cookie that follows the = sign
    name = cookieTokens[1];
}
else {
    // if there was no cookie then ask the user to input a name
    name = window.prompt("Please enter your name", "Paul");
    document.cookie = "name=" + escape(name);
}

document.writeln("<h1>Hello, " + name + ". </h1>");
document.writeln("<a href='javascript:wrongPerson()'> " + "Click here if you are not " + name + "</a>");
Cookie Example #2

// reset the document's cookie if wrong person
function wrongPerson() {
    // reset the cookie
    document.cookie = "name=null;" + " expires=Thu, 01-Jan-95 00:00:01 GMT;";

    // after removing the cookie reload the page to get a new name
    location.reload();
}

// determine whether there is a cookie
if (document.cookie) {
    var cookie = document.cookie;
    var cookieTokens = cookie.split("=");

    // set name to the part of the cookie that follows the = sign
    name = cookieTokens[1];
    name = unescape(name);
} else {
    // if there was no cookie then ask the user to input a name
    name = window.prompt("Please enter your name", "Paul");
    document.cookie = "name=" + escape(name);
}

document.writeln("<h1>Hello, " + name + ". </h1>" + "Click here if you are not " + name + "</a>;

Storing Cookies – More Realistic

- By default, cookies expire when close browser
- Set “expires” attribute to make stick around longer

function createCookie(name,value,days) {
    if (days) {
        var date = new Date();
        date.setTime(date.getTime()+(days*24*60*60*1000));
        var expires = "; expires="+date.toGMTString();
    } else {
        var expires = ";
        document.cookie = name+"="+escape(value)+expires;
    }
}

function eraseCookie(name) {
    createCookie(name,"",-1);
}

(modified from http://www.quirksmode.org/js/cookies.html)
Parsing Cookies – More Realistic

// Return the 'value' of the cookie variable with name 'desiredVar'
// returns null if no match found.
function parseCookie(desiredVar) {
    // First split the pairs apart on ';'
    var pairs = document.cookie.split(';
);

    // Now split each pair on '='. Check if have a match
    for (var i=0; i < pairs.length; i++) {
        var aPair = pairs[i];

        // remove any leading spaces
        while (aPair.charAt(0) == ' ')
            aPair = aPair.substring(1, aPair.length);

        // split into desired parts and check for match
        var cookieTokens = aPair.split('="
');
        var name = cookieTokens[0];
        var value = cookieTokens[1];
        if (name == desiredVar) {
            // found desired variable -- return value
            return unescape(value);
        }
    }
    return null;  // no match;
}

Cookies – Java Script and Perl

• Cookies with JavaScript
  – Create cookie
    • document.cookie = “color=red”;
  – Read cookie (from JavaScript)
    • Read and parse document.cookie
    • Use parseCookie() function to help with this
  – Where are cookies stored??

• Cookies with Perl
  – Create cookie with print() BEFORE header
    • Sent to browser
  – Browser always send appropriate cookies back to server with request
  – Read cookie
    • Access $ENV{ "HTTP_COOKIE" } (book does this)
    • Or use cookie() function helper (easier!)
  – Where are cookies stored??

• Cookies created with Perl can read via JavaScript and vice versa
Creating cookie with Perl

(Assume this file invoked from a HTML form with fields name, height, and color)

```perl
use CGI qw(:standard);
use strict;

CGI->new;

my $name = param( "name" );
my $height = param( "height" );
my $color = param( "color" );

my $expires = gmtime( time() + 86400 );

print( "Set-Cookie: Name=$name; expires=$expires; \n" );
print( "Set-Cookie: Height=$height; expires=$expires; \n" );
print( "Set-Cookie: Color=$color; expires=$expires; \n" );

print( header() );
print( start_html( ) );

print h1("A cookie was stored!  Name:  $name");
print( end_html() );
```

Read Cookies With Perl

```perl
use CGI qw(:standard);
use strict;

print( header() );
print( start_html( ) );

my $name   = cookie( "Name" );
my $height = cookie( "Height" );
my $color  = cookie( "Color" );

if ($name || $height || $color) {
    print h1("A cookie was found!");
    print h2("Name: $name");
    print h2("Height: $height");
    print h2("Color: $color");
} else{
    print h1("Could not find cookies for name, Height, or Color");
}

print( end_html() );
```
Uses for Cookies

- Most common:
  - User logs in using secure page (https)
  - Server checks password. If good, creates cookie
    - E.g. “login=m078987&auth=356af12cd124552”
  - User redirected to other pages. These pages don’t ask for password – instead just check that have valid login cookie

- Why do we need the auth field?

Remember

- Relevant cookies always sent by browser to the server
- Can create with JavaScript and read with Perl
- Or create with Perl and read with JavaScript