

IT350 Web and Internet Programming

Fall 2006

SlideSet #11: JavaScript Arrays, Objects, & Cookies

(from Chapter 11/12 of the text)

Everything you ever wanted to know about arrays...

```
function initializeArrays()
{
    var n1 = new Array( 5 );      // allocate 5-element Array
    var n2 = new Array();         // allocate empty Array

    for ( var i = 0; i < n1.length; ++i )
        n1[ i ] = i;

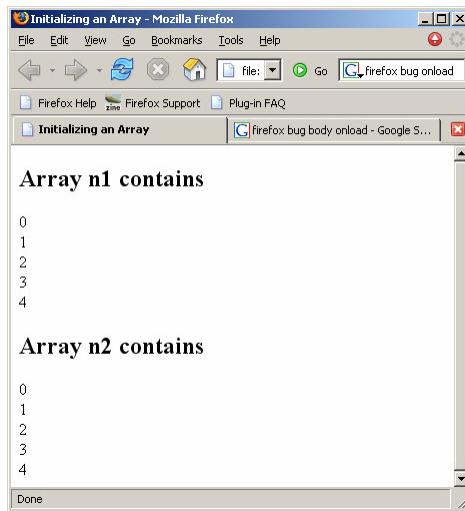
    for ( i = 0; i < 5; ++i )
        n2[ i ] = i;

    outputArray( "Array n1 contains", n1 );
    outputArray( "Array n2 contains", n2 );
}

function outputArray( header, theArray )
{
    document.writeln( "<h2>" + header + "</h2>" );
    for ( var ii in theArray ) {
        document.write( theArray[ ii ] + "<br/>" );
    }
}

initializeArrays();
```

...but were afraid to ask.



Scope – Revisited

```
function mystery( x, y )
{
    for ( var ii = 0; ii < x.length; ++ii )
        x[ii] = x[ii] * y;

    y = 7;

    document.writeln("<br/> x: ",x);
    document.writeln("<br/> y: ",y);
}

var myArray = [3, 4, 5];
var factor = 2;

document.writeln ("<br/> myArray: ", myArray);
mystery(myArray, factor);

document.writeln ("<br/> myArray: ", myArray);
document.writeln ("<br/> factor : ", factor);
```

Arguments are passed _____,
so original argument values in caller are _____
BUT array/object arguments are a “reference”, so contents may be _____

Exercise #1

- a.) Write a function “sumArray” as follows:
Input: an array
Output: the sum of that array
- b.) Write test code to create an array and call “sumArray” on it.

Exercise #2 – What's the output?

```
function printme( z ) {
    document.writeln("<br/> z is ",z);
}

var array1 = [17, 21, 42];
var array2 = [14, 19];
var x = 1;

printme (array1);
printme (array2[1]);
printme (x);

array1[x] = 57;
printme (array1);
```

Exercise #3 – What's the output?
(Hint: assume JavaScript ignores any errors it finds)

```
function changeMe1( z ) {
    z[0] = 75;
}
function changeMe2( a, b) {
    a = b;
}
var array1 = [17, 21, 42];
var array2 = [14, 19];
var array3 = [7, 8, 9];
var x = 63;
changeMe1 (array1);
document.writeln("<br/> array1: ", array1);
changeMe1 (array2[1]);
document.writeln("<br/> array2: ", array2);
changeMe1 (x);
document.writeln("<br/> x: ", x);
array1 = array2;
document.writeln("<br/> array1: ", array1);
changeMe2 (array1, array3);
document.writeln("<br/> array1: ", array1);
```

Exercise #4

- Write a function perfect(N) that returns an array of size N containing the first N perfect squares. So perfect(4) would return [0, 1, 4, 9].

Exercise #5

- a.) Write a function `dotProduct(x, y)` that takes two arrays of size n and returns the sum:
$$x[0]*y[0] + x[1]*y[1] + \dots + x[n-1]*y[n-1]$$
- b.) Look ahead to “Cookie Example #1” (but don’t peek at #2!). Can you find the bug?

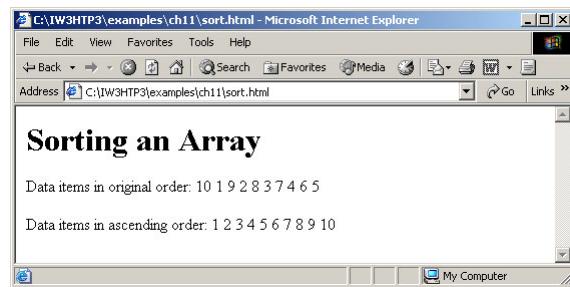
Functions as Arguments

```
function start()
{
    var a = [ 10, 1, 9, 2, 8, 3, 7, 4, 6, 5 ];

    document.writeln( "<h1>Sorting an Array</h1>" );
    document.writeln( "Data items in original order: ", a );
    a.sort( compareIntegers ); // sort the array
    document.writeln( "Data items in ascending order: ", a );
}

// comparison function for use with sort
function compareIntegers( value1, value2 )
{
    return parseInt( value1 ) - parseInt( value2 );
}
```

Sorting Output



12.7 document Object

Method or Property	Description
<code>write(string)</code>	Writes the string to the XHTML document as XHTML code.
<code>writeln(string)</code>	Writes the string to the XHTML document as XHTML code and adds a newline character at the end.
<code>document.cookie</code>	This property is a string containing the values of all the cookies stored on the user's computer for the current document. See Section 12.9, Using Cookies.
<code>document.lastModified</code>	This property is the date and time that this document was last modified.

Fig. 12.12 Important `document` object methods and properties.

12.8 window Object

Method or Property	Description
<code>open(url, name, options)</code>	Creates a new window with the URL of the window set to <code>url</code> , the name set to <code>name</code> , and the visible features set by the string passed in as <code>option</code> .
<code>prompt(prompt, default)</code>	Displays a dialog box asking the user for input. The text of the dialog is <code>prompt</code> , and the default value is set to <code>default</code> .
<code>close()</code>	Closes the current window and deletes its object from memory.
<code>window.focus()</code>	This method gives focus to the window (i.e., puts the window in the foreground, on top of any other open browser windows).
<code>window.document</code>	This property contains the <code>document</code> object representing the document currently inside the window.
<code>window.closed</code>	This property contains a boolean value that is set to true if the window is closed, and false if it is not.
<code>window.opener</code>	This property contains the <code>window</code> object of the window that opened the current window, if such a window exists.

Fig. 12.14 Important `window` object methods and properties.

12.9 Using Cookies

- **Cookie**

- Data stored on user's computer to maintain information about client during and between browser sessions
- Can be accessed through `cookie` property
- Set expiration date through `expires` property
- 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", "GalAnt" );  
    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", "GalAnt" );
    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>" );
```

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;
}
```