Javascript Objects

1. OBJECTIVES

At the end of the session, the student should be able to:

a. relate Javascript and Object -oriented programming;
b. use the different built-in objects such as String, Date, Array, Boolean, and Math.

2. DISCUSSION

INTRODUCTION TO JAVASCRIPT OBJECTS

JavaScript is an Object Oriented Programming (OOP) language. An OOP language allows you to define your own objects and make your own variable types.

We will start by looking at the built-in JavaScript objects, and how they are used. The next pages will explain each built-in JavaScript object in detail.

Note that an object is just a special kind of data. An object has properties and methods.

Properties

- Properties are the values associated with an object.
- In the following example we are using the length property of the String object to return the number of characters in a string:

```
<script type="text/javascript">
var txt="Hello World!"
document.write(txt.length)          //output is 12
</script>
```

Methods

- Methods are the actions that can be performed on objects.
- In the following example we are using the toUpperCase() method of the String object to display a text in uppercase letters:

```
<script type="text/javascript">
var str="Hello world!"
document.write(str.toUpperCase())          //output is HELLO WORLD!
</script>
```

STRING OBJECT

The String object is used to manipulate a stored piece of text.

Examples of use:

The following example uses the length property of the String object to find the length of a string:

```
var txt="Hello world!"
document.write(txt.length)          //outputs 12
```

The following example uses the toUpperCase() method of the String object to convert a string to uppercase letters:

```
var txt="Hello world!"
document.write(txt.toUpperCase())          //outputs HELLO WORLD!
```

For the full reference about the String Object, http://www.w3schools.com/jsref/jsref_obj_string.asp

DATE OBJECT

Defining Dates

- The Date object is used to work with dates and times.
- We define a Date object with the new keyword. The following code line defines a Date object called myDate:

```
var myDate=new Date()
```

- Note: The Date object will automatically hold the current date and time as its initial value!

Manipulate Dates

- We can easily manipulate the date by using the methods available for the Date object.
- In the example below we set a Date object to a specific date (14th January 2010):

```
var myDate=new Date()
myDate.setFullYear(2010,0,14)
```

- And in the following example we set a Date object to be 5 days into the future:

```
var myDate=new Date()
myDate.setDate(myDate.getDate()+5)
```

- Note: If adding five days to a date shifts the month or year, the changes are handled automatically by the Date object itself!

Comparing Dates

- The Date object is also used to compare two dates.
- The following example compares today's date with the 14th January 2010:

```
var myDate=new Date()
myDate.setFullYear(2010,0,14)
var today = new Date()
if (myDate>today)
  alert("Today is before 14th January 2010")
else
  alert("Today is after 14th January 2010")
```

ARRAY OBJECT

Defining Arrays

- The Array object is used to store a set of values in a single variable name.
- We define an Array object with the new keyword. The following code line defines an Array object called myArray:

```
var myArray=new Array()
```

- There are two ways of adding values to an array (you can add as many values as you need to define as many variables you require).
- 1:

```
var mycars=new Array()
mycars[0]="Saab"
mycars[1]="Volvo"
mycars[2]="BMW"
```

- You could also pass an integer argument to control the array's size:

```
var mycars=new Array(3)
mycars[0]="Saab"
```

```
mycars[1]="Volvo"
mycars[2]="BMW"
```

- 2:

```
var mycars=new Array("Saab","Volvo","BMW")
```

- Note: If you specify numbers or true/false values inside the array then the type of variables will be numeric or Boolean instead of string.

Accessing Arrays

- You can refer to a particular element in an array by referring to the name of the array and the index number. The index number starts at 0.
- The following code line:

```
document.write(mycars[0])        // outputs Saab
```

Modify Values in Existing Arrays

- To modify a value in an existing array, just add a new value to the array with a specified index number:

```
mycars[0]="Opel"
```

- Now, the following code line:

```
document.write(mycars[0])        //outputs Opel
```

MATH OBJECT

- The Math object allows you to perform common mathematical tasks.
- The Math object includes several mathematical values and functions. You do not need to define the Math object before using it.

Mathematical Values

- JavaScript provides eight mathematical values (constants) that can be accessed from the Math object.
- These are: E, PI, square root of 2, square root of 1/2, natural log of 2, natural log of 10, base-2 log of E, and base-10 log of E.
- You may reference these values from your JavaScript like this:

```
Math.E
Math.PI
Math.SQRT2
Math.SQRT1_2
Math.LN2
Math.LN10
Math.LOG2E
Math.LOG10E
```

Mathematical Methods

- In addition to the mathematical values that can be accessed from the Math object there are also several functions (methods) available.
- Examples of functions (methods):
- The following example uses the round() method of the Math object to round a number to the nearest integer:

```
document.write(Math.round(4.7))          //outputs 5
```

- The following example uses the random() method of the Math object to return a random number between 0 and 1:

```
document.write(Math.random())            //outputs 0.6243711427134218
```

- The following example uses the floor() and random() methods of the Math object to return a random number between 0 and 10:

```
document.write(Math.floor(Math.random()*11))    //outputs 5
```

3. EXERCISE

Calculator