UNIT : 1

Introduction to JAVA Script

Reference :- HTML, JAVASCRIPT, DHTMAL AND PHP by IVAN BAYROSS
Chapter :- 8 : Introduction to Java Script

JAVA Script in web pages

- **Website**
  - Intelligent enough to **accept user input** and **dynamically structure web page content** as per user's requirement.
  - Content should be dynamic **based on what user wants to see**
  - Need for creating **interactive web pages**.
  - Web page will accept input from user, based on input customize web page is returned.
  - In absence of any user input website must be intelligent enough to return a default web page.
  - Environment requires coding techniques
    - **Capable of accepting client's request**
    - **Processing request**
    - **Result of processing passed back to client via HTML pages**

JAVA Script in web pages

- **Website**
  - Capturing user request is done via **Form.**
  - After capturing input form must have built in technique for **sending information captured back to the server for processing.**
  - This is done via **script (small programs)** that are based on server.

JAVA Script in web pages

- **Website**
  - Should provide facility for **validating user input.**
  - **Invalid input** will cause data to be sent back to browser from web server.
  - **Repeat visit** of the website for inputting valid values **is tedious**.
  - Need of environment which facilitates **coding that runs in a browser at a client side** for data.

JAVA Script in web pages

- **JAVA Script**
  - Have standard programming construct for:
    - Condition checking constructs
    - Case checking constructs
    - Super controlled loop constructs
    - Syntax to create and use memory variables, constants and functions
    - **Object Oriented Programming environment**.
    - Offers **event driven programming**
  - **Created by Netscape**
  - Netscape Client browser product is called – “Netscape Communicator”
  - Netscape Server product is called – “Netscape Commerce Server”

JAVA Script in web pages

- **JAVA Script**
  - Netscape product → **Live Wire**,
  - Permits **server side Java Script code to connect to RDBMS** (e.g. Oracle, MS SQL server, MySQL, mSQL)
  - Also supports non-relational database.

JAVA Script in web pages

- **Client side JAVA Script**
  - Embedded into standard HTML program.
  - **<SCRIPT>….</SCRIPT>** tag.
  - tag embedded **within <HEAD>…</HEAD> or <BODY>…</BODY>**
  - **Browser** with Javascript enabled will **interpret Java script code.**

JAVA Script in web pages

- **Capturing user input**
  - **<FORM>….</FORM>** used to create user Request form.
  - **<INPUT>….</INPUT>** used to instantiate HTML objects in HTML form for capturing user data.

- HTML itself is static. HTML allows a very minimum interaction with users by providing hyperlinks.

JAVA Script

- Object-oriented language
- Allows creation of interactive web pages.

Advantage of JAVA Script

- **Interpreted Language**
  - No compilation steps, syntax interpreted by browser.
- **Embedded within HTML**
  - Doesn’t require special editor, written in any text editor, script can be embedded within html file
- **Minimal syntax – Easy to learn**
  - Simple rules of syntax
- **Quick Development**
  - Doesn’t require time consuming compilation, scripts can be developed with short period of time
  - Many GUI elements like alert, prompt, confirm box.

Advantage of JAVA Script

- **Design for simple, small program**
  - Well suited for simple, small programs

- **Performance**
  - Script are fairly compact and quite small,
  - Minimizes storage requirements on web server and download time for client
  - Require few separate network access as embedded with HTML file.

- **Procedural Capabilities**
  - Condition checking, Looping, Branching etc.

- **Designed for programming user events**
  - Supports Object / Event based programming

Advantage of JAVA Script

- **Easy Debugging and Testing**
  - Script is tested line by line as it is interpreted language.
  - Errors are listed as they are encountered.
  - So appropriate error message along with line number is listed
  - So easy to locate errors, make changes and test it again.

- **Platform independence / Architecture Neutral**
  - Completely independent of hardware on which it works.
  - Understood by any Computer with Javascript enabled browser.
  - As browser is for specific platform , Javascript interpretation will be with respect to specific platform.
  - Browser will add platform specific information for Javascript.
  - Developed on Unix machine will work well for Windows machine.

JAVA Script

- What can JavaScript do
  1) JavaScript can change HTML content
  2) JavaScript can change HTML attributes
  3) JavaScript can change CSS style
  4) JavaScript can validate input

<SCRIPT> tag

- Marks beginning of snippet of scripting code.
- Paired tag
- Attribute → Language
- Purpose : Indicates the scripting language used for writing the snipped scripting code.
- Default is : Javascript for Netscape communicator
- Default is : VB script for Internet Explorer.
- E.g.
  - **<SCRIPT Languge="JavaScript">**

```
                ---------
</SCRIPT>
```

<SCRIPT> tag

```
<html>
<head>
   <title> JAVASCRIPT </title>
</head>
<body>
   <script language="JavaScript">
         document.write("Welcome to JAVA Script");
   </script>
</body>
</html>
```

Variables and Constants

- **<HEAD>…</HEAD>** is ideal place → Create Javascript variables and constants.
- As head of HTML document is always processed before body.
- Attempt to use any variable before it is defined will give error.
- Variables →used to store values, have a name associated with them via which they can be referenced later.

Variables and Constants

```
<html>
<head>
        <script>
        var name=prompt("enter your name");
        document.write("Welcome "+name + " to java script");
        </script>
</head>
<body>
</body>
</html>
```

Variables and Constants

- **Variables**
  - Begin with upper case letter , lower case letter, underscore character, dollar sign character.
  - Remaining characters can consist of letter, underscore, dollar sign or digits.
  - Variable names are case sensitive.
  - If two words used then start first letter of first word in lower case and first letter of second word in upper case
    - E.g. variableName.
  - Doesn’t allow data type of variable to be declared when variable is create.
  - Same variable may be used to hold different types of data at different times when javascript code runs.

Data types and Literals

- Supports four primitive types , complex types such as arrays and objects.
- Literals are fixed values, provides value in a program.
- Number
  - Consists of integer and floating point numbers and special NaN (Not a Number) value.
  - Integer literal can be represented in : decimal, hexadecimal, Octal form.
  - Floating point literal : used to represent very large or very small numbers,
    - E.g. 12.10, 2E3, 0X5F (12.1, 2000,95)

Data types and Literals

- **Boolean**
  - Consist of logical value **true and false.**
  - Supports pure Boolean type consist of two values.
  - Logical operators can be used in Boolean expressions.
  - Automatically converts the boolean values true and false into 1 and 0 when used in numerical expressions.
  - Example : var d=10+true;
  - Here d will hold value 11.

Data types and Literals

- **String**
  - Consist of string value enclosed in single or double quotes.
  - Sequence of zero or more characters.
  - E.g. “24, abc nagar, Banglore” Valid
    - “abc’ invalid.
  - To include quote character in string it must be preceded by the backslash (\) escape character.

Data types and Literals

- **Null**
  - Identifies null, empty or nonexistent reference.
  - Used to set variable to initial value.
  - Prevents from error which is caused by use of un-initialized variable.
  - Automatically converted to default value of other type when used in expression.

Data types and Literals

- **Type casting**
  - Variables are loosely cast.
  - Type of variable is implicitly defined based on literal value assigned to it.
  - E.g. “Total amount is “ with literal 1000 results to string
  - 10.5 - “10” results in floating point literals 0.5.

Creating Variables

- Variable can be created to hold any type of data.
- Syntax :
  - var <variable name> = value ;
- Example:
  - var first_name;
  - var last_name="sanghvi";
  - var phone = 123456123;
  - Example

Javascript Array

- Capable of storing sequence of values.
- Values are stored in indexed location within the array.
- Length of array is number elements that an array contains.
- Individual elements of array are accessed by name of array followed by index value of array element enclosed in square brackets.

Javascript Array

- Array must be declared before it is used.

- Syntax:
- **var arrayname=[“item1”,”item2”,….];**

  var arrayName = new Array(Array length)

  var arrayName = new Array()

- Example:

  cust_Orders = new Array();

  cust_Orders[50] = “test”;

  cust_Orders[100] = “test1”;

Javascript Array

- Encounter reference to order[50], will extend the size of array to 51 and initializes order[50].

- Even if array is initially created of fixed length it still be extended by referencing elements that are outside the current size of the array.

- This is done same manner as with zero-length arrays.

Javascript Array

- Dense array
  - Created with each of its elements being assigned a specific value.
  - E.g. arrayName = new Array(value0,value1,………,valuen)
  - Elements starts with 0 index
- Join()
  - return all elements of the array joined together as single string.
  - Takes one argument → a string to be used as separator between each element in the final string.
  - Default is comma-space
- Reverse()
  - Reverses the order of the elements in the array
  - Example

Javascript Array

- Element of Array
  - No restriction on the values
  - Values can be of different types or can refer other array object
  - Example
- Length property
  - Arrays are implemented as objects
  - Objects are name collection of data that have properties and methods.
  - Property returns a value → state of an object
  - Method use to read / modify data contained in object’s property.
  - Length is property of array.
  - To access property → objectname.propertyname.

Operators & Expressions

- Operator
  - Used to transform one or more values into a single resultant value.
  - Value to which operator is applied is operand

- Expression
  - Are evaluated to determine the value of the expression.

Arithmetic Operator

Operator	Description
+	Additon
-	Subtraction / Unary Negation
*	Multiplication
/	Division
%	Modulus
--	Decrement by 1
++	Increment by 1

Example

Logical Operator

Operator	Description	Example (Given that x=6 and y=3)
&&	Logical AND	(x < 10 && y > 1) is true
||	Logical OR	(x == 5 || y == 5) is false
!	Logical NOT	!(x == y) is true

Comparison Operator

Operator	Description	Comparing (Assuming x=5)	Result
==	Equal	x == 8, x == 5,x==“5”	False, true , true
===	Strictly Equal Example	x === "5“, x === 5	False,true
!=	Not equal	x != 8	true
!==	Strictly not equal	x !== "5“, x !== 5	True,false
<	Less than	x < 8	true
>	Grater than	x > 8	false
<=	Less than or equal to	x <= 8	true
>=	Grater than or equal to	x >= 8	False

Assignment Operator

Operator	Description
=	Sets the variable on left of the = operator to the value of the expression on its right
+=	Increments the variable on L.H.S. By the value on R.H.S. In case of string value is appended
-=	Decrements the variable on L.H.S. By the value on R.H.S.
*=	Multiplies the variable on L.H.S. By the value on R.H.S.
/=	Divides the variable on L.H.S. By the value on R.H.S.
%=	Takes modulus of variable on L.H.S. using the value of the expression on R.H.S.

String & Conditional Expression operator

- String
  - Used to perform operations on string.
  - Javascript supports **+** string concatenation operator.
  - Used to join two strings.

- Ternary operator
  - Condition ? Value1 : value2
  - Must return value true or false.
  - Example :

```
var age=14;
var voteable = (age < 18) ? "Too young” : "Old enough";
document.write(voteable);
```

Special Operator

- delete operator
  - Used to delete property of an object or an element at an array index.
  - E.g. delete stud[5] will delete sixth element of array stud.

- new operator
  - Used to create an instance of an object type.

- void operator
  - The void operator is used to evaluate a JavaScript expression without returning a value.
  - Example :

```
<a href="javascript:void(alert('Thank you.'))">
Click here to see a message </a>
```

Javascript Programming Construct

Construct / Statement	Purpose	Example
Assignment	Assign the value of an expression to a variable	x = y + z
Data declaration	Declares a variable and optionally assigns a value to it	var myVar = 10

if	Program execution depends on the value of return by the condition if true program executes else does not	`if (x>y)`
		`{`
		`    z = x;`
		`}`

Switch	Selects from a number of alternatives
	Example
		`Switch(val)`
		`{`
		`    case 1 :`
		`        break;`
		`    case 2 :`
		`        break;`
		`    default :`
		`}`

Javascript Programming Construct

Construct / Statement	Purpose	Example

while	Repeatedly executes set of statements until a condition becomes false	`while (x!=7)`
		`{`
		`        a++;`
		`}`

do while	Repeatedly executes set of statements while a condition is true	`do`
		`{`
		`        stmt1;`
		`} while(x!=7);`

For	Repeatedly executes set of statements until a condition becomes false	`for (i=0;i<7;i++)`
		`{`
		`    document.write(x[i]);`
		`}`

Label	Associates a label with a statement Example	`LabelName:`
		`        stmt;`

Javascript Programming Construct

Construct / Statement	Purpose	Example

break	Immediately terminates a do while or for loop	`if (x>y)`
		`        break;`

continue	Immediately terminates the current iteration of a do, while or for loop	`if (x>y)`
		`      continue;`

function call	Invokes a function	x = abs(y);
return	Returns a value from function	return x*y

with	Identifies the default object
	Example
		`with(Math)`
		`{`
		`        d = PI  * 2;`
		`}`

delete	Deletes an object property or an array element	delete a[5]
Method invocation	Invokes a method of an object	document.write(“Hello”);

Functions

- Blocks of JavaScript code designed to do specific task and often return value.

- May take zero or more parameters

Built – in Functions

- Type conversion functions
  - **eval()**
    - Used to convert string expression to numeric value
    - E.g. var a = eval(“10*10+5”);
  - **parseInt()**
    - Used to convert a string value to an integer.
    - Return first integer contained in the string
    - Return 0 if string doesn’t begin with an integer.
    - E.g. var a = parseInt(“123xyz”); → Result a will contain 123
    - var a = parseInt(“xyz”); → Result a will contain NaN.

Built – in Functions

- Type conversion functions
  - **parseFloat()**
    - Return first float contained in the string
    - Return 0 if string doesn’t begin with an integer.
    - E.g. var a = parseFloat(“1.23xyz”); → Result a will contain 1.23
    - var a = parseFloat(“xyz”); → Result a will contain NaN.

User Defined Functions

- Declaring functions
  - Declared and created using function keyword.
  - Contains
    - Name of a function
    - List of parameters
    - Block of javascript code that defines what the function does
  - Syntax :

```
function function_name(parameter1,parameter2….)
{
    block of code.
}
```

Case sensitive;
Can include underscore, has to start with a letter

User Defined Functions

- Place of Declaration
  - Can be declared anywhere within HTML file
  - Preferably IN <HEAD> … </HEAD> → ensures all functions will be parsed before they are invoked.
  - If called before it is declared / parsed will lead to error.

- Passing Parameters
  - Values are listed in parentheses separated by comma.
  - During declaration function need to be informed about the no. of values that will be passed.

  Example

User Defined Functions

- **Variable scope**
  - Parameter are local to the function.
  - Come into existence when function is called and cease to exist when function ends.
  - Any variable declared within function will have scope within it.
  - If declared outside body of function then available to all stmt. of script.
  - If global and local variable have same name then if used within function then local will get priority over global variable.
  - Example

User Defined Functions

- Return value
  - return statement is used to return value.
  - Any valid expression that evaluates to single value can be returned.
  - Example :

```
function cube (number)
{
    return number * number * number;
}
```

  - Example

User Defined Functions

- Recursive function
  - Function calls itself.
  - If-else construct can prevents infinite recursion.
  - Example:

```
function  factorial(number)
{
        if (number>1)
        {
                return number * factorial(nuber-1);
        }
        else
                return number;
}
```

Dialog Boxes

- Provides ability to pick up user input or display small amount of text
- Appears as a separate window.
- Three types of dialog box:
  - Alert Dialog Box
  - Prompt Dialog Box
  - Confirm Dialog Box

Alert Dialog Box

- Purpose : To display a cautionary message or display some information.
- Takes single string argument.
- Displays string passed
- Have “OK” button
- Will not continue processing until OK is clicked.
- **Example:**
- <script>
- alert("Thank You...")
- document.write("Welcome to java script");
- </script>

Prompt Dialog Box

- Purpose : To get input from user which allows user interaction.
- Prompt Dialog box
  - Displays predefined message
  - Displays textbox and accepts user input
    - Can pass the text back to Javascript
  - Displays “OK” and “Cancel” button.
  - Program execution gets halt until user clicks OK or Cancel button.
- Prompt() method has two parameters
  - A message to be displayed as a prompt to the user.
  - Any message to be displayed in textbox(optional)

  Syntax :

```
prompt(“<msg>”,”<default value>”);
```

Confirm Dialog Box

- Purpose : Serves as a technique for confirming user action.
- Confirm Dialog box
  - Displays predefined message
  - Displays “OK” and “Cancel” button.
  - Program execution gets halt until user clicks OK or Cancel button.
  - “OK” causes TRUE to be passed to program and
  - “Cancel” causes FALSE to be passed to the program

Syntax :

```
confirm(“<message>”);
```