IT WORKSHOP I

JavaScript

17-Jan-23

Topics

- Server-side vs. Client-side
- What is DHTML?
- What is JavaScript
- First JavaScript Program
- How to Add a Script to Pages
- JS Programming
- JS Functions
- HTML Events

2

Server-side vs. Client-side

- The web architecture has two vital parts,
- The browser is said to define the client-side of the web, the computer it is running on, and the user surfing the web being collectively referred to as the client.
- The web server is an application running on a computer.
- Like the client, the server application and the computer on which it runs define the server-side of the web, and are collectively referred to as the server.

3

Server-side vs. Client-side Applications

- Server-side applications are applications runs on the Web server
- Client-side applications are small applications which are embedded within the HTML code and executed by the browser.
- Server-Side Code
  - Languages include Python , PHP, C#, Servlets and JSP
  - Stores persistent data
  - Cannot be seen by the user
  - Can only respond to HTTP requests for a particular URL.
- Client-Side Code
  - Languages used include: HTML, CSS, and Java script.
  - Parsed by the user's browser
  - Reacts to user input

4

What is DHTML?

- Dynamic HyperText Markup Language (DHTML) is a combination of Web development technologies used to create dynamically changing websites.
- Web pages may include dynamic content, animation, dynamic menus and text effects.
- Makes possible a Web page to react and change in response to the user's actions
- combination of HTML, style sheets and client-side scripts (JavaScript, VBScript, or any other supported scripts)

5

What is JavaScript

- JavaScript is an interpreted, client-side, event-based, object oriented scripting language used to add dynamic interactivity to web pages.
- JavaScript is a scripting (lightweight programming) language designed primarily for adding interactivity to HTML pages
- It is an interpreted language (it executes without preliminary compilation)
- Usually embedded directly into HTML pages and most commonly used as a client side scripting language
- JavaScript defines dynamic behaviour
  - Programming logic for interaction with the user, to handle events

6

What Can JavaScript Do?

- validate form data
- performing complex calculations
- content loading and changing dynamically
- access / modify browser cookies
- detect the user’s browser and OS
- handle events
- read and write HTML elements
- changing the behaviour dynamically
- reacting to user actions
- reduce load at the server-end

History of JavaScript ?

- JavaScript was created by Brendan Eich in 1995 during his time at Netscape Communications.
- JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java
- JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.
- No relation to Java

JavaScript - Syntax

- JavaScript can be implemented using JavaScript statements that are placed within the <script>... </script> HTML tags in a web page
- You can place the <script> tags, containing your JavaScript code, anywhere within your web page, but it is normally recommended that you should keep it within the <head> tags.

```
<script language="javascript" type="text/javascript">
   JavaScript code
</script>
```

First JavaScript Program

```
<html>
   <body>
      <script language="javascript" type="text/javascript">
         <!--
            document.write("Hello World!")
         //-->
      </script>
   </body>
</html>
```

we call a function document.write which writes a string into our HTML document.

How to Add a Script to Pages

- In the <head> of a page: These scripts will be called when an event triggers them.
- In the <body> section: These scripts will run as the page loads.
- In an external file: write JavaScript in external documents that have the file extension .js. This is a particularly good option if your script is used by more than one page.

<script type=”text/javascript” src=”validation.js”/>

11

How to Add a Script to Pages

```
<html>
<head>
        <script type="text/javascript">  //<script src="jscript.js">
        function msg(){
        alert("Hello JavaScript");
        }
        </script>
</head>
<body>
<p>Welcome to JavaScript</p>
        <script language="javascript">
        alert("Hello JavaScript");
        </script>
</body>
</html>
```

12

JavaScript Comments

- JavaScript comments can be used to explain JavaScript code, and to make it more readable.
- Single Line Comments
  - Single line comments start with //.
  - Any text between // and the end of the line will be ignored by JavaScript (will not be executed).
- Multi-line Comments
  - Multi-line comments start with /* and end with */.
  - Any text between /* and */ will be ignored by JavaScript.
- semicolons are optional! However, semicolons are required if you want to put more than one statement on a single line.

13

Variables in JavaScript

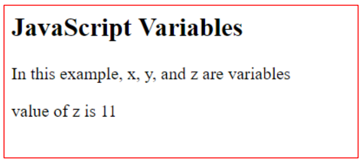
- Variables are used to hold data.
- A JavaScript identifier:
  - Starts with a letter or underscore, and
  - Is followed by letters, underscore or digits
  - JavaScript is a case-sensitive language
  - Variables are untyped (they can hold values of any type)
  - The word var is optional (but it's good style to use it)
- Syntax:
  - var varname;
  - var x = 5;
  - var y = 6;
  - var z = x + y;
    - In this example, x, y, and z, are variables (local and global)

Variables in JavaScript

```
<html>
<body>
<h2>JavaScript Variables</h2>
<p>In this example, x, y, and z are variables</p>
   <script>
   var x = 5;
   var y = 6;
   var z = x + y;
   document.write("value of z is" +z);
   </script>
</body>
</html>
```

document.write is a standard JavaScript command for writing output to a page.

- document is the object
- write is the method



JavaScript Variables

In this example, x, y, and z are variables

value of z is 11

JavaScript Popup/Dialog Boxes

Method	Description
alert()	• used to display a message/information to the user • displays the alert box containing message with ok button. Syntax: alert(“This is Alert Box”)
confirm()	• used to verify whether the user accept or cancel something • displays the confirm dialog box containing message with ok and cancel button. Syntax: confirm(“This is Confirm Box”)
prompt()	• displays a dialog box to get input from the user. Syntax abc=prompt("Enter a value")

JavaScript Popup/Dialog Boxes

```
<html>
<head>
</head>
<body>
<p>Welcome to JavaScript</p>
    <script language="javascript">
          alert("This is an Alert Message")
          confirm("This is a Confirm Message")
          abc=prompt("Enter a value")
          alert("The value entered is "+abc)
    </script>
</body>
</html>
```

JavaScript Data Types

- JavaScript variables can hold many data types
- Primitive Data Types
  - Numbers - A number can be either an integer or a decimal
  - Strings - A string is a sequence of letters or numbers enclosed in single or double quotes
  - Boolean - True or False
- Composite Data Types
  - Arrays
  - Objects

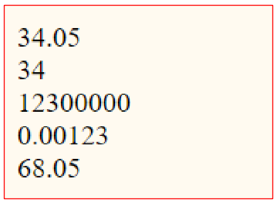
Variables & Data Types

- JavaScript is untyped; It does not have explicit data types
- For instance, there is no way to specify that a particular variable represents an integer, string, or real number
- The same variable can have different data types in different contexts
- Although JavaScript does not have explicit data types, it does have implicit data types
  - If you have an expression which combines two numbers, it will evaluate to a number
  - If you have an expression which combines a string and a number, it will evaluate to a string

JavaScript Numbers

- JavaScript has only one type of numbers.
- Numbers can be written with, or without decimals.

```
<html>
<body>
<script>
  var x1 = 34.05;
  var x2 = 34;
  var y = 123e5;
  var z = 123e-5;
  document.write( x1 + "<br>" + x2 + "<br>" + y + "<br>" + z);
  document.write("<br>" + (x1+x2))
</script>
</body>
</html>
```

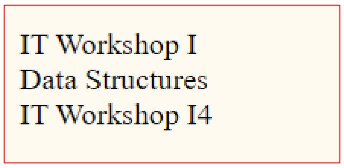


```
34.05
34
12300000
0.00123
68.05
```

JavaScript Strings

- A string (or a text string) is a series of characters .
- Strings are written with either single or double quotes.

```
<html>
<body>
<script>
    var courseName1 = “IT Workshop I”;
    var courseName2 = ‘Data Structures’;
    document.write(courseName1 + "<br>" + courseName2 );
    num1=4;
    document.write("<br>" + (courseName1 +num1))
</script>
</body>
</html>
```



```
IT Workshop I
Data Structures
IT Workshop I4
```

21

JavaScript Booleans

- Booleans can only have two values: true or false.
- Booleans are often used in conditional testing

```
<html>
<body>
<script>
    var x =true;
    var y =false;
    if(y==true)
    alert("True")
    else
    alert("False")
</script>
</body>
</html>
```

22

JavaScript Arrays

- JavaScript arrays are sequence of values written with square brackets.
- Array items are separated by commas.
- Array indexes are zero-based, which means the first item is [0], second is [1], and so on.

```
<html>
<body>
<script>
    var cars = ["Audi","Benz","BMW"];
    document.write(cars[1]);
</script></body>
</html>
```

```
Benz
```

JavaScript Operators

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- String operators

Arithmetic operators

- Arithmetic operators perform arithmetic operations upon operands.

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement

```
<script>
var a = 100;
var b = 50;
var x = a + b;
document.write(“x val is” +x);
</script>
```

25

Assignment Operators

- Assignment operators assign values to JavaScript variables

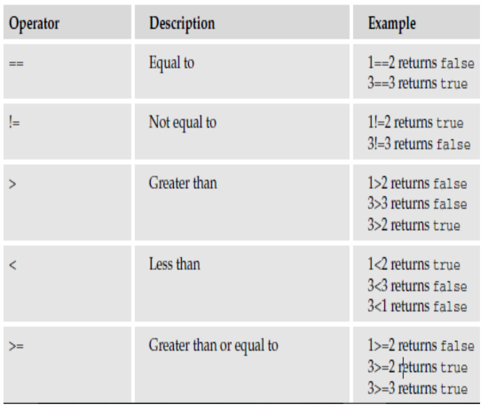
Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

```
<script>
var x = 10;
x += 5;
document.write(“x val is” +x);
</script>
```

26

Comparison Operators

- comparison operators compare two operands and then return either true or false based on whether the comparison is true or not.



Operator	Description	Example
==	Equal to	1==2 returns false 3==3 returns true
!=	Not equal to	1!=2 returns true 3!=3 returns false
>	Greater than	1>2 returns false 3>3 returns false 3>2 returns true
<	Less than	1<2 returns true 3<3 returns false 3<1 returns false
>=	Greater than or equal to	1>=2 returns false 3>=2 returns true 3>=3 returns true

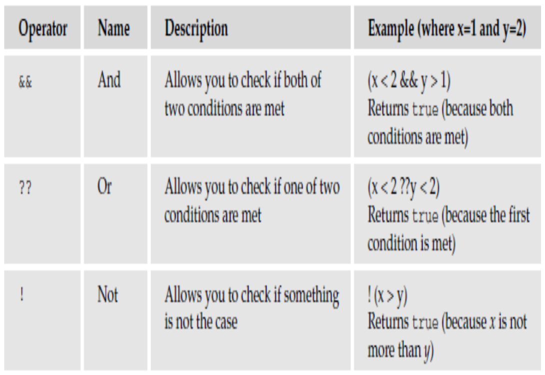
```
<script>
var a= 10,b=20;
if(a>b)
document.write(" a is big");
else
document.write(" b is big");
</script>
```

=== same as == ??

27

Logical Operators

- Logical operators return one of two values: true or false.
- allows to evaluate more than one expression at a time.



Operator	Name	Description	Example (where x=1 and y=2)
&&	And	Allows you to check if both of two conditions are met	(x < 2 && y > 1) Returns true (because both conditions are met)
??	Or	Allows you to check if one of two conditions are met	(x < 2 ??y < 2) Returns true (because the first condition is met)
!	Not	Allows you to check if something is not the case	! (x > y) Returns true (because x is not more than y)

```
<script>
var a = true;
var b = false;
result = (a && b);
document.write(result);
</script>
```

28

String Operator

- You can also add text to strings using the + operator. For example, here the + operator is being used to add two variables that are strings together.

```
firstName = “Bob”;
lastName = “Stewart”;
name = firstName + lastName;
```

- The value of the name variable would now be
  - Bob Stewart
- The process of adding two strings together is known as *concatenation*.

Conditional Statements

- Conditional statements allow you to take different actions depending upon different statements.
- There are three types of conditional statement
  - if statements
  - if...else statements
  - switch statements

if Statements

- if statements allow a part of the code to be executed when the condition specified is true, else another part is executes.
- The syntax is as follows:

```
if (condition)
{
code to be executed if condition is true
}
else
{
code to be executed if condition is false
}
```

if Statements

```
<html>
<body>
<script>
    time=prompt("Enter the current time")
    if(time<12)
       document.write("<h1>Good Morning");
    else
       document.write("<h1>Good Evening");
</script>
</body>
</html>
```

Switch Statement

- The switch statement is used to select one of many blocks of code to be executed.
- The switch expression is evaluated once.
- The value of the expression is compared with the values of each case.
- If there is a match, the associated block of code is executed.

```
switch (expression) {
case condition 1: statement(s)
break;
case condition 2: statement(s)
break;
...
case condition n: statement(s)
break;
default: statement(s)
}
```

33

Switch Statements

```
<html><body>
<script>
 day=prompt("Enter the day 0 - 7")
switch (d) {
   case 0:
      day = "Sunday";
      break;
   case 1:
      day = "Monday";
      break;
   case 2:
      day = "Tuesday";
      break;
   case 3:
      day = "Wednesday";
```

```
      break;
    case 4:
      day = "Thursday";
      break;
    case 5:
      day = "Friday";
      break;
    case  6:
      day = "Saturday";
}
document.write("Today is " +
day);
</script>
</body></html>
```

34

JavaScript Loops

- Looping statements are used to execute the same block of code a specified number of times.
  - while
  - do…while
  - for

JavaScript Loops

```
<script language="javascript">
var i;
var input=prompt("Enter the number of inputs");

for(i=1;i<=input;i++)
{
var str= "Enter the number" + i;
var val=prompt(str);
alert(val);
}
</script>
```

Functions

- A function is some code that is executed when an event fires or a call to that function is made.
- Functions are either written in the <head> element and can be reused in several places within the page, or in an external file that is linked from inside the <head> element
- How to Define a Function - There are three parts to creating or defining a function:
  - Define a name for it.
  - Indicate any values that might be required as arguments.
  - Add statements

37

Functions - Syntax

```
function  function-name(parameter1, parameter2...)
{
   code to be executed
}
```

We will define the function in head section and call the function from the body

```
function calculateArea(width, height)
{
area = width * height
return area
}
```

38

Functions - example

```
<html><head><script>
function add(){
var a,b,c;
a=Number(document.getElementById("first").value);
b=Number(document.getElementById("second").value);
c= a + b;
return c;
}
</script></head>
<body>
First Value:<input type="text" id="first"><br>
Second Value:<input type="text" id="second"><br>
Result:<input id="answer"><br>
<button onclick="alert(add())">Add</button>
</body></html>
```

Functions with return- example

```
<html><head><script>
function add(){
var a,b,c;
a=Number(document.getElementById("first").value);
b=Number(document.getElementById("second").value);
c= a + b;
document.getElementById("answer").value= c;
}
</script></head>
<body>
First Value:<input type="text" id="first"><br>
Second Value:<input type="text" id="second"><br>
Result:<input id="answer"><br>
<button onclick="add()">Add</button>
</body></html>
```

HTML Events

Event	Description
onclick	The user clicks an HTML element
onsubmit	occurs when form is submitted.
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page
onchange	An HTML element has been changed

End