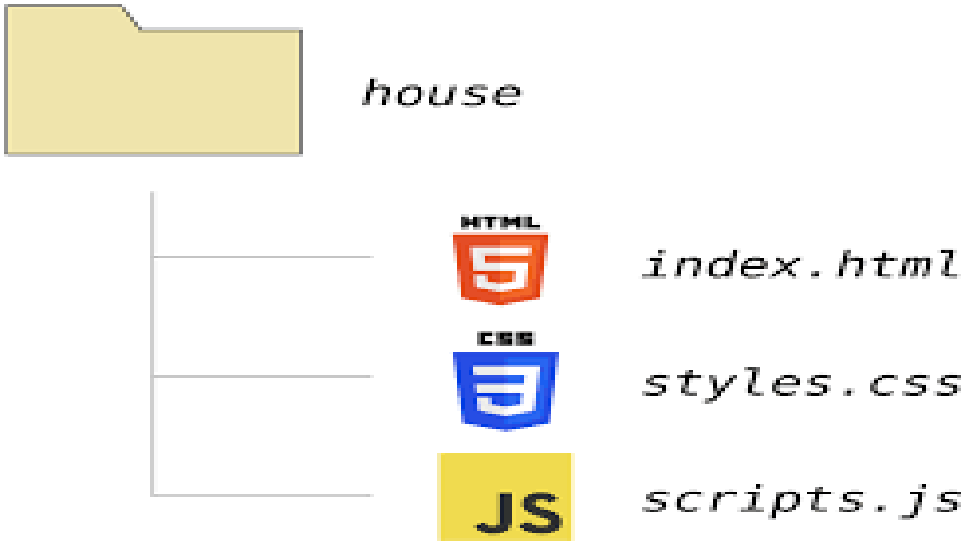
WEB TECHNOLOGIES 1

JavaScript

Lec9

Mohammed Sultan

HTML, CSS & JAVASCRIPT

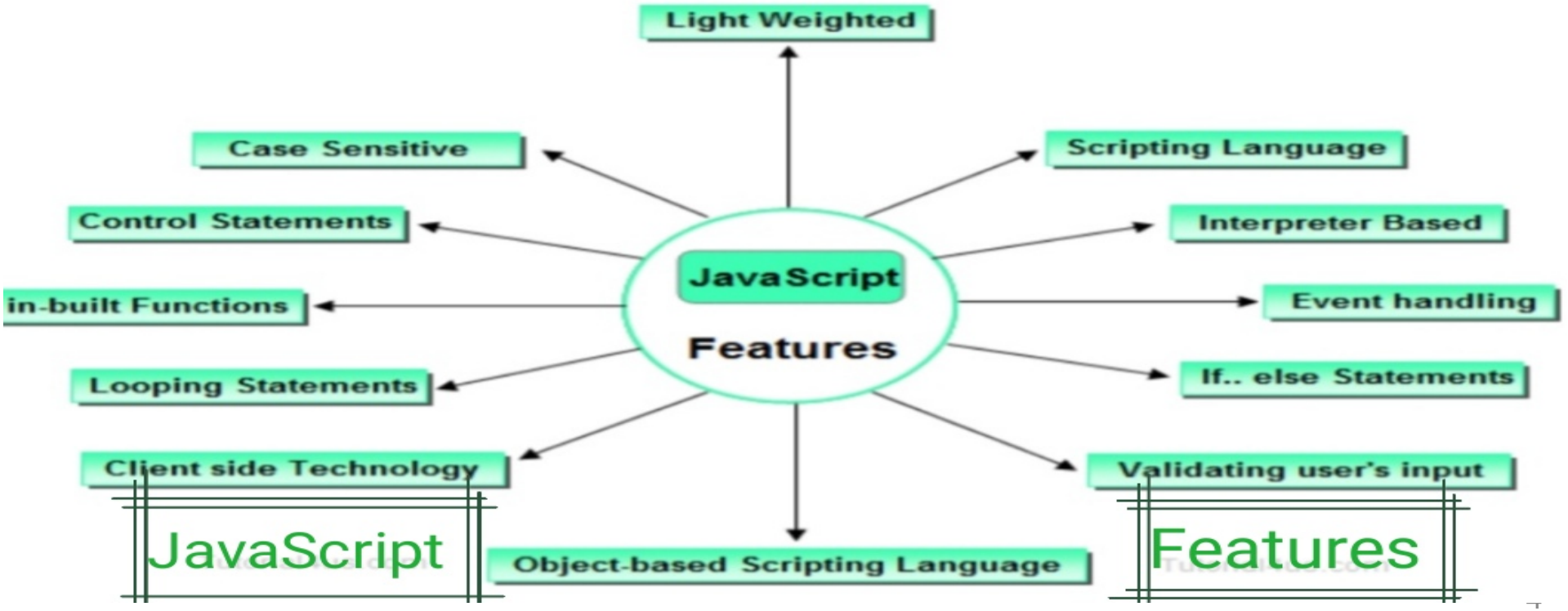


What is JavaScript?

- HTML and CSS concentrate on a static rendering of a page; things do not change on the page over time, or because of events.
- To do these things, we use scripting languages, which allow content to change dynamically.
- Not only this, but it is possible to interact with the user beyond what is possible with HTML.
- Scripts are programs just like any other programming language; they can execute on the client side or the server.

What is JavaScript?

- it is mainly used for gives client side validation, but it have lot of features which are given below;



JavaScript

- In HTML, JavaScript code is inserted between <script> and </script> tags.

- The script is containing 2 attributes :
  - Language attribute
    - <script language= "JavaScrip" >
  - Type attribute :
    - It indicates MIME (multi purpose internet mail extension) type of scripting code. It sets to an alpha-numeric MIME type of code.
      - <script type="text/javascript ">

Benefits

- JavaScript Can Change HTML Content and css
  - One of many JavaScript HTML methods is getElementById().
    - document.getElementById("demo").innerHTML = " Hello JavaScript";
    - document.getElementById("demo").style.fontSize = " 35px";
- Hide HTML Elements
  - document.getElementById("demo").style.display = "none";

JavaScript and Java are completely different languages, both in concept and design.

Advantages of client side scripting

- The web browser uses its own resources, and eases the burden on the server.
- It has fewer features than server side scripting.
- It saves network bandwidth

Disadvantages of client side scripting

- Code is usually visible.
- Code is probably modifiable.
- Local files and databases cannot be accessed.
- User is able to disable client side scripting.

Methods of using JS

- Embedded JavaScript
  - JavaScript can be embedded in an HTML document.
  - To embed it in HTML you must write:
    - <script type=”text/javascript”></script>
  - The JavaScript can be placed in the head section of your HTML or the body.
- Placing scripts at the bottom of the <body> element improves the display speed, because script interpretation slows down the display.

Methods of using JavaScript

- External JavaScript
  - If you want to use the same script on several pages it could be a good idea to place the code in a separate file, rather than writing it on each.
  - That way if you want to update the code, or change it, you only need to do it once.
  - Simply take the code you want in a separate file out of your program and save it with the extension .js.

External JavaScript

```
<html>
<body>
    <script src="myScript.js"></script>
</body>
</html>
```

- It makes HTML and JavaScript easier to read and maintain
- Cached JavaScript files can speed up page loads

JavaScript Variables

- 4 Ways to Declare a JavaScript Variable:
  - Using var
  - Using let
  - Using const
  - Using nothing

  - Examples :
  - Var x = 5; /1/for older versions of JS to 2015 /2/global /3/ can redeclare variables
  - let y = 6; /1/the declaration after 2015 /2/local /3/can't not redeclare variables
  - const body_temp=37; // for not change values /2/can't not redeclare variables
  - x = x + 5

the equal sign (=) is an "assignment" operator, not an "equal to" operator

JavaScript naming and Comments

- A JavaScript name must begin with:
  - A letter (A-Z or a-z)
  - A dollar sign ($)
  - Or an underscore (_)
  - No space in name or spacial characters or reserved word
  - Using camelCase for naming variables

- Comments :
  - Code after double slashes // or between /* and */ is treated as a comment.

  - Comments are ignored, and will not be executed
    - let x = 5; // I will be executed

- JavaScript is Case Sensitive
  - The variables lastName and lastname, are two different variables

Data type

- 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

Data type

- Primitive Data Types
  - Numbers
  - Strings
  - Boolean (True, False)
- Composite Data Types
  - Arrays
  - Objects

String

- A string is a sequence of letters or numbers enclosed in single or double quotes
- To create a string — just wrap any sequence of characters in quotes.
- Quotes can be single, double or backtick.
  - let hello = 'Hello, World!';
  - let name = "Hello ";
  - let s = `some string`;
  - let hello = 'Hello, "World!" ';//let name = "He'll'o ";
  - \ for printing " or '
  - \n for new line

Numbers

- A number can be either an integer or a decimal
- Number can be both integers and floating point.
  - let a = 1;
  - let b = 0.5;
  - let c = -15.25;
  - let year = 2021;

JavaScript Operators

```
let x = 5;
let y = 2;
```

- JavaScript Arithmetic Operators:
  - + Addition // let z = x + y;
  - - Subtraction
  - * Multiplication //let z = x * y;
  - ** Exponentiation (ES2016)
  - / Division
  - % Modulus (Division Remainder)
  - ++ Increment
  - -- Decrement

Aggregate Assignments

- Aggregate assignments provide a shortcut by combining the assignment operator with some other operation
- The += operator performs addition and assignment
- The expression x = x + 7 is equivalent to the expression x += 7

Increment and Decrement

Both the increment (++) and decrement (- -) operator come in two forms: prefix and postfix

These two forms yield different results

```
x = 10;   x = 10;
y = ++ x;   z  = x ++;
 y = 11
 z = 10
 x = 11 in both cases
```

CONTROL STRUCTURES

Control Structures

- There are three basic types of control structures in JavaScript: the if statement, the while loop, and the for loop

- Each control structure manipulates a block of JavaScript expressions beginning with { and ending with }

The If Statement

The if statement allows JavaScript programmers to a make decision
Use an if statement whenever you come to a “fork” in the program

```
If ( x  = =  10)
{    y  =  x*x;
}
else
{    x  =  0;
}
```

Repeat Loops

- A repeat loop is a group of statements that is repeated until a specified condition is met

- Repeat loops are very powerful programming tools; They allow for more efficient program design and are ideally suited for working with arrays

JavaScript Loops

- JavaScript supports different kinds of loops:
  - for - loops through a block of code a number of times
  - for/in- loops through the properties of an object
  - while - loops through a block of code while a specified condition is true
  - do/while - also loops through a block of code while a specified condition is true

- The JavaScript for/in statement loops through the properties of an object.
  - var person={fname:"mohammed",lname:"ahmed",age:25};
  - for (x in person)
  - {
  - txt=txt + person[x];
  - }

The While Loop

The while loop is used to execute a block of code while a certain condition is true

```
count = 0;
while (count <= 10) {
  document.write(count)
;
  count++;
}
```

The For Loop

- The for loop is used when there is a need to have a counter of some kind

- The counter is initialized before the loop starts, tested after each iteration to see if it is below a target value, and finally updated at the end of the loop

The For Loop

```
<SCRIPT
LANGUAGE=
     "JavaScript">
document.write("1");
document.write("2");
document.write("3");
document.write("4");
document.write("5");
</SCRIPT>
```

```
<SCRIPT
LANGUAGE="JavaScript">
for (i=1; i<=5; i++)
  document.write(i);
</SCRIPT>
```

ARRAYS

JavaScript Array

- An array is a special variable, which can hold more than one value.

- Each numbered datum is called an element of the array and the number assigned to it is called an index.
- The elements of an array may be of any type. A single array can even store elements of different type.

- Syntax
  - const array_name = [item1, item2, ...];
  - By using new Array()
    - const cars = new Array("Volvo", "BMW");

```
const cars = [
  "Volvo",
  "BMW"
];
```

Accessing Array Elements

Array elements are accessed using the [ ] operator

- Example:
  - var colors = [“red”, “green”, “blue”];
  - colors[0] => red
  - colors[1] => green

- Adding Elements
- To add a new element to an array, simply assign a value to it
  - Example:
  - var a = new Array(10);
  - a[5] = 17;

Array Length

- All arrays created in JavaScript have a special length property that specifies how many elements the array contains
  - Example:
    - var colors = [“red”, “green”, “blue”];
    - colors.length => 3

JavaScript Array Methods

- toString()
  - converts an array to a string of (comma separated) array values.
  - document.getElementById("demo").innerHTML = array.toString();

- pop() : delete from the end
- push() : add at the end
- shift() :removeing the first element
- concat() :merge two arrays
  - arr1.concat(arr2, arr3);

FUNCTIONS

JavaScript Functions

- in javascript functions are created with the keyword function as show below:

```
function funname( )
{
Your code here.....
}
```

- Types of functions :
  - built-in functions
  - user-defined functions

- There are two ways to call the function.
  - direct call function
  - Events handlers to call the function dynamically

JavaScript Functions

```
<script>
    function myFunction(g1, g2) {
      return g1 / g2;
    }
    document.getElementById("demo").innerHTML =
myFunction(12, 3);
</script>
```

Output 4

Calling functions example

```
<HTML>
<HEAD>
<TITLE> Function direct call</TITLE>
<script language="JavaScript">
function add(x,y)
{
z=x+y;
return z;
}
</script>
</HEAD>
<BODY>
<script>
var r=add(30,60);
document.write("addition is :"+r);
</script>
</BODY>
</HTML>
```

```
<HTML>
<HEAD>
<TITLE> Function dynamically</TITLE>
<script language="JavaScript">
function add( )
{
x=20;
y=30;
z=x+y
document.write("addition is :"+z);
}
</script>
</HEAD>
<BODY> to call function:
<input type="button" value="click hare"
onclick="add( )">
</script>
</BODY>
</HTML>
```

Events handling in JavaScript

- Event handlers are attributes that force an element to "listen" for a specific event to occur.
- Event handlers all begin with the letters "on".
- Events are not case sensitive.
- There are two types of events in Javascript
  - Interactive i.g. onClick
  - Non-interactive i.g. onLoad

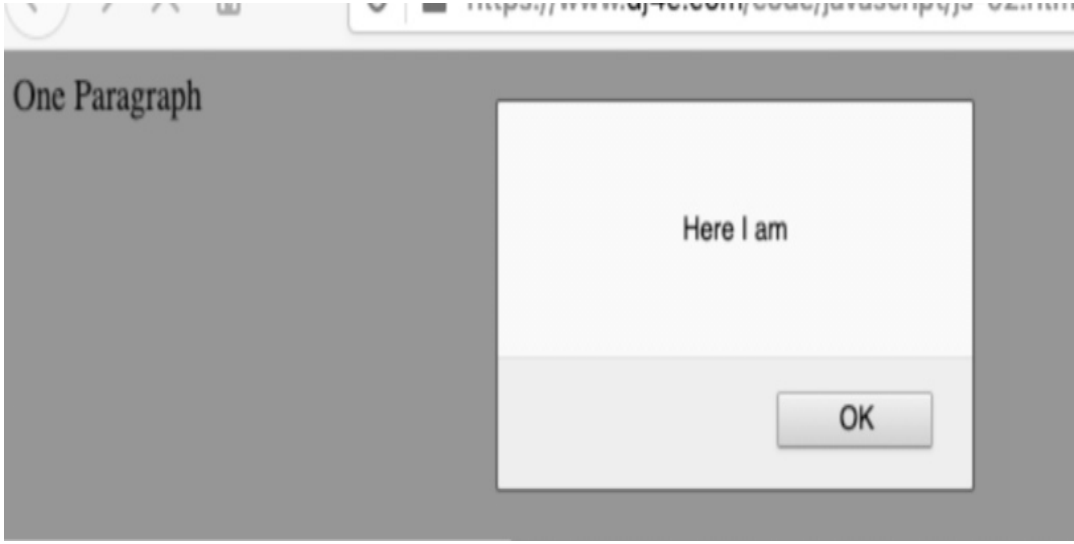
Events

Attribute	The event occurs when…
onclick	mouse click an object
ondblclick.	mouse double clicks
onmouseover	a mouse cursor on touch here
onmousedown	a mouse button ispressed
onmousemove	the mouse is moved
on mouse out	the mouse is moved out anelement
onmouseup	a mouse button isreleased
onkeydown	a keyboard key ispressed
onkeypress	a keyboard key is pressed or held down
onkeyup	a keyboard key isreleased
onfocus	an elements getfocus
onchange	the content of a fieldchange
onsubmit	the submit button isclicked

Example

```
<HTML>
<HEAD>
<script language="JavaScript">
function myf( )
{
document.write("Hai Mohammed")
}
</script>
</HEAD>
<BODY>
to execute script code:
<input type="button" value="click me" onclick="myf( )">
To execute script code:
<input type="button" value="touch me" onmouseover="myf( )">
</BODY>
</HTML>
```

```
html>
<head>
<title>Hello World</title>
</head>
<body>
<p>One Paragraph</p>
<script type="text/javascript">
  alert("Here I am");
  document.write("<p>Hello World</p>")
</script>
<noscript>
Your browser doesn't support or has disabled
JavaScript.
</noscript>
<p>Second Paragraph</p>
</body>
```



```
<html>
<head>
<title>Hello World</title>
</head>
<body>
<p>One Paragraph</p>
<p><a href="js-01.htm"
 onclick="alert('Hi'); return false:">Click
Me</a></p>
<p>Third Paragraph</p>
</body>
</html>
```


CELEBRATING
25 years of
JavaScript
1,444,231 libraries
and counting...

Any Questions?