UNIT-5

JAVASCRIPT

Introduction to Javascript :

Javascript is a light weight, cross-platform and interpreted Scripting language. It is well-known for development of webpages many non-browser, environments also use it. Java Script can be used for client-side developments as well as server-side developments. Javascript Contains a Standard library of objects like Array, Date and Month, and also contains elements like operators control structures and statements.

If Javascript is client-Side :-

It supplies objects to control to control a browser and its Document object Model (DOM). Like if client-side extensions allow an application to place elements on. An HTML form and respond to user events such as mouse clicks, form input and page navigation. Useful libraries for the client-side are Angular Js, React JS, VueJs and so many others.

1. Server side : It supplies objects relevant to running Javascript on a server, Like if the server-side extensions allow an application to communicate with a database and produce Continuity of information from one invocation to another of the applications, perform file manipulations on a server, The useful framework which is the most famous these days in node.Js

Javascript can be added to your HTML file in 2 ways:

Internal JS :- We can add Javascript directly to our HTML file by writing the code inside the <Script1> tag. The

<script> tag can either be placed inside the <head> or the <body> tag according to the requirement.

External JS :- We can write Javascript code in other file having an extension .js and then link this file inside the <head> tag of the HTML file in which we want to add this code.

Syntax :
```
<Script>
  // Javascript code
</script>
```

Example :
```
HTML
<!DOCTYPE html>
<html lang ="en">
  <head>
  <title>Javascript Example </title>
 </head>
 <body>
 <script>
Console log ("welcome to CSE");
  </script>
  </body>
  </html>
```

Output :-

```
Welcome to CSE
```

History of Javascript :-

It was created in 1995 by Brendan Esch while he was an engineer at Netscape. It was originally going to be named LiveScript but was renamed. Unlike most programming languages, the Javascript language has no concept of input or output. It is designed to run as a scripting language in a host environment to provide mechanisms for communicating with the

Outside world. The most common host environment is the browser.

Features of Javascript :- According to a recent survey conducted by stack overflow. Javascript is the most popular language on earth.

With advances in browser technology and Javascript having moved into the server with the Node.js and other frameworks, Javascript is capable of so much more. Here are a few things but we can do with Javascript:

1) Earlier websites were mostly static, after Javascript was created dynamic websites were made.

2) Functions in JS are objects. They may have properties and methods just like another object. They can be passed as arguments, in other functions.

4) Can handle date and time.

5) Perform form validation although the forms are created using HTML.

6) No compiler is needed.

Applications of Javascript:

1. Web Development

We can add interactivity and behavior to static sites. By using Angular JS that can be achieved so easily.

2. Web Applications:-

By using this Javascript, we can create robust web applications. for Example, when we explore a map in Google

maps then we only need to click and drag the mouse.
All detailed view is just a click away, and this is possible only because of Javascript. It uses API (Application programming interface) that provides extra power to code.

3) Server Applications :-

With the help of Node.js, Java script made its way from client to server and node.js is the most powerful on server side.

4) Games :-

Not only in websites, but Javascript also helps in creating games. The combination of Javascript and HTML5 makes javascript popular in game development as well. It provides the Ease JS library which provides solutions for working with rich graphics.

5. Smart Watches

Javascript is being used in all possible devices and Applications. It provides a library pebble JS which is used in smart watch applications. This framework works for applications that require the internet for its functioning

6. Art :-

Artists and designers can create whatever they want using Java Script.

7. Machine Learning :-
This Javascript library can be used in web development by using machine learning.

Limitations of Java Script :-

1) Performance :

Java Script does not provide the same level of performance as offered by many traditional languages. So it is not Suitable to handle complex tasks.

2) Complexity :

To master a Scripting language, programmers must have a good knowledge of all the programming concepts, Core language objects, client and Server side objects. otherwise it would be difficult for them to write advanced Scripts using Java script.

3) Weak error handling and type checking facilities :-

It is weakly typed language as there is no need to specify the datatype of variable. So wrong type checking, is not performed by Compiler.

Java Script Variables :-

A Javascript variable is simply a name of storage location. There are 2 types of variables in Javascript.

1) Local variable

2) Global variable

There are Some rules while declaring a Javascript Variable

1) Name must start with a letter (a to z or A to z), underscore (_) or dollar sign ($)

2) After first letter, we can use digits (0 to 9), for example value 1.

3) Javascript variables are case sensitive, for example x and X are different variables.

Var X=10; } Correct
Var-value = 'Sonoo"; } Javascript variable

Var 123= 30; } In Correct Javascript
Var *aa = 320; } variable.

Example of Javascript Variable :-

```
<html>
<body>
<Script>
var X=10;
var Y=20;
var z=x+y;
document.write (z);
</script>
</body>
</html>
```

Output = 30

Javascript Local Variable :-

A Javascript local variable is declared inside block or function. It is accessible within the function or block only for ex:-

```
1. <script>
function abc() {
var x=10; // Local variable
}
</script>
2 <script>
if (10<13) {
Var y=20; // Javascript local Variable
}
</script>
```

Javascript Global Variable

A Javascript global variable is accessible from any function. A variable i.e. declared outside the function or declared with, window object is known as global variable. for example:

```
<html>
<body>
<script>
var data=200; //global variable
 function a() {
 document.write ln(data);
 }
 function b() {
document.write ln(data);
}
 a(); // calling Javascript
  b();
  </script>
  </body>        output: 200 200
  </html>
```

3. Javascript Data Types:

Javascript provides different datatypes to hold different types of values. There are two types of data types in Javascript.

1) Primitive Type 2) Non-primitive (reference) data type.

Javascript is a dynamic type language, means you dont need to specify type of variable because it is dynamically used by Javascript engine, you need to use variable here to specify the data type. It can hold any type of values such as numbers, strings etc.. for example:

1. var a=40; // holding number

2. var b="Rohitha"; //holding string

Javascript primitive data types:-

There are 5 types of primitive data types in Javascript. They are as follows:

Data Type	Description
string	Represents sequence of characters eg:"hello"
Number	Represents numeric values eg: 100
Boolean	Represents boolean value either false or true
undefined	Represents undefined value
Null	Represents null i.e; no value at all.

Javascript non-primitive datatypes:-

The non-primitive data types are as follows:

Datatypes	Description
Object	Represents instance through which we can access members.
Array	Represents group of similar values
Reg Exp	Represents regular expression

Javascript conditional statement:

1) if statement
2) if else statement
3) if else if statement
4) switch statement

Javascript if Statement :-

It evaluates the Content only if expression is true

Syntax:-
```
if (expression) {
     // Content to be evaluated
}
```

Ex:-
```
<Script>
   Var a=20;
   if(a>10) {
 document.write ("value of a is greater than 10");
}
 </script>
```

Javascript if---else Statement :-

It evaluates the Content whether condition is true or false.

Syntax:-
```
   if (expression) {
    // Content to be evaluated if condition is true
   }
   else{
   // Content to be evaluated if condition is false
 }
```

Ex:-
```
      <script>
      Var a=20;
      if (a%2==0) {
     document.write ("a is even number");
    }
     else{
     document.write ("a is odd number");
    }
    </script>
```

Javascript if---else if statement :-

It evaluates the content only if expression is true from several expressions.

Syntax:
```
if (expression 1) {
// Content to be evaluated if expression 1 is true
}
else if (expression 2) {
// content to be evaluated if expression 2 is true
}
else if (expression 3) {
// Content to be evaluated if expression 3 is true
}
else {
// Content to be evaluated if no expression is true
}
```
Ex:-
```
<script>
var a=20;
if (a==10) {
document.write ("a is equal to 10");
   else if (a==15) {
document.write ("a is equal to 15");
}
else if (a==20) {
document.write ("a is equal to 20");
}
 else {
document.write ("a is not equal to 10,15 or 20");
}
 </script>
```
Javascript Switch :-

The Javascript switch statement is used to execute one code from multiple expressions.

Syntax :-

```
Switch (expression) {
```

```
Case value 1:
Code to be executed
  break;
  Case value 2:
  Code to be executed
   break;
   -----
   default:
  Code to be executed if above values are not matched;
}
```

Example of switch statement in Javascript:

```
 <Script>
 Var grade = 'B';
  Var result;
   Switch (grade){
     Case 'A':
      result = "A Grade";
       break;
      Case 'B';
     result = "B Grade";
       break;
      Case 'C';
     result = "C Grade";
       break;
      default:
    result = "No Grade";
    } document.write (result);
       </script>
```

Javascript Loops :

The Javascript loops are used to iterate the piece of code using for, while, do while or for-in-loops. It makes the code Compact. It is mostly used in array.

There are 4 types of loops in Javascript

1. for loop
2. while loop
3. do-while loop
4. for-in loop

1. Javascript for loop :-

The Javascript for loop iterates the elements for the fixed number of times. It should be used if number of iteration is known, The syntax of for loop is given below.

```
for (initialization ; Condition; Increment)
{
  Code to be executed
  }
```

```
Ex :- <script>
 for (i=1; i<=5; i++)
 {
  document write (i+ "<br>")
 }
  </script>
```

2. Javascript while loop :-

The Javascript while loop iterates the elements for the infinite number of times. It should be used if number of iteration is not known. The syntax of while loop is given below.

```
 while ( condition)
 {
  Code to be executed
 }
```

Ex:
```
<script>
 var i=11;
 while (i<=15)
 {
 document.write (i+"<br/>");
  i++;
 }
 </script>
```

3) Javascript do while loop :-

The Javascript do while loop iterates the elements for the infinite number of times like while loop. But, code is executed atleast once whether condition is true or false. The syntax of do while loop is given below.

```
    do
     {
      Code to be executed
     } while (condition);
```

Let's see the simple example of do while loop in javascript

```
    <script>
    var i=21;
    do{
    document.write (i+"<br/>");
    i++;
    }while (i<=25);
    </script>
```

4) Javascript for in loop :-

The Javascript for in loop is used to iterate the properties of an object.

Functions in Javascript :-

Javascript functions are used to perform operations, we can call Javascript function many times to reuse the code.

Advantage of Javascript function :-

There are mainly 2 advantages of Javascript functions.

1. Code reusability :- We can call a function several times so it saves coding.
2. Less Coding : It makes our program compact. we don't need to write many lines of code each time to perform a common task.

Javascript function Syntax :-

The syntax of declaring function is given below

```
function functionName([arg1, arg2, .... argN])
        {
          //Code to be executed
        }
```

Javascript functions can have 0 or more arguments.

Ex :-
```
<html>
<body>
<h2> Javascript functions </h2>
 <p id="demo"> </p>
  <script>
Var x = my Function (4,3);
document.getElement By Id ("demo"). inner HTML = x;
function myFunction (a,b)
  {
   return a*b;
  }
 </script>
```

→ when java script reaches return Statement it stops executing

o/p :
Javascript function
12

→ A Javascript function is defined by with the function keyword, followed by a name, followed by paranthoses

→ Function names can contain letters, digits, underscores and dollar signs.

→ The paranthoses may include parameter names separated by commas

Ex:- (parameter 1, parameter 2, ---)

→ The code to be executed by the function is placed inside curly brackets: { }

<u>Function invocation</u> :-

The code inside the function will execute when "Something" calls the function.

a) when an event occurs (when a user click a button)

b) when it is invoked (called) from javascript code.

c) Automatically (self invoked)

<u>Javascript Events</u> :-

HTML events are "things" that happen to HTML elements when Javascript is used in HTML pages.

Javascript can "react" on these events.

<u>HTML Events</u> :-

An HTML Event can be something the browser does or something a user does.

Here are some examples of HTML events

- An HTML webpage has finished loading
- An HTML input field was changed.
- An HTML button was clicked.

often, when events happen, you may want to do something. Javascript lets you execute code when events are detected. HTML allows event handler attributes, with Javascript code, to be added to HTML elements.

Ex:-

```
        <html>
         <head>
          <script type= "text/javascript
           <!..
             function Say Hello() {
                alert ("Hello world");
             }
            //-->
           </script>
           </head>
           <body>
            <p> click the following button </p>
             <form>
            <input type= "button" on click = "Say Hello()"
            value= "Say Hello"/>
              </form>
              </body>
               </html>
```

output:

click the following button

$\boxed{\text{Say Hello}} \longrightarrow$ Hello world.

Some of the events & their event handlers are:

Mouse Events

Event performed	Event handler	Description
click	onclick	When mouse click on an element.
mouse over	on mouse over	when the cursor of the mouse comes over the element
mouse out	on mouse out	when the cursor of the mouse leaves an element
mouse up	on mouse up	when the mouse button is released over the elements
mouse move	on mouse move	when the mouse movement takes place

Keyboard events :-

Event performed	Event Handler	Description
Key down and key up	on keydown & on key up	when the user press then release the key.

Form events :

Event performed	Event Handler	Description
focus	on focus	when the user focuses on an element
Submit	on submit	when the user submit the form
blur	on blur	when the focus is away from a form element
change	on change	when the user modifies or changes the value of a form element

Window / Document events :

Event performed	Event handler	Description
load	onload	when the browser finishes the loading of the page
unload	on unload	when the visitor leaves the current web page, the browser unloads it
resize	on resize	when the visitor resizes the window of the browser

Javascript form validation :-

It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is must to authenticate user.

Javascript provides facility to validate the form on the client-side so data processing will be faster than server-side validation. Most of the web developers prefer Javascript form validation.

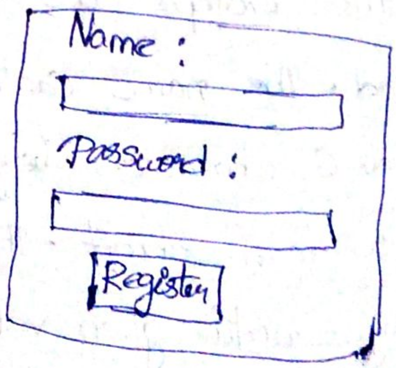
Through Javascript, we can validate name, password, email, date, mobile numbers & more fields.

Javascript form validation Example :-

In this example, we are going to validate the name and password. The name can't be empty and password can't be less than 6 characters ~~language~~ long. Here we are validating the form on form submit. The user will not be forwarded to the next page until given values are correct.

```
<html>
<body>
```

```
<script>
function validate form()
{
 Var name = document . myform . name. value ;
Var password = document . myform . password . value ;
 if (name == null || name == " ")
  {
   alert (" Name can't be blank");
    return false;
  }
   else
      if (password . length < 6)
      {
      alert(" password must be atleast 6 characters long");
      return false ;
      }
        }
      </script
 <form name= "myform" method = "post"
 action . "http://-----" onsubmit =" return validate form()>
 Name:<input type= "text" name ="name"> <br/>
 password :<input type = "password" name = " password"><br/>
  <input type = "Submit" value = "register">
      </form>
      </body>
      </html>
```



Javascript provides a way to validate form's data on the client's Computer before sending it to the web Server. Form validation generally perform two functions

Basic validation :-

First of all the form must be checked to make sure are the mandatory fields are filled in. It would require just a loop through each field in form and check for data.

Data format validation :-

Secondly, the data that is entered must be checked for correct form and value. your Code must include appropriate logic to test Correctness of data.

Document object Model (DOM)

The document object represents the whole html document. when html document is loaded in the browser, it becomes a document object. It is the root element that represents the html document. It has properties and methods. By the help of document object, we can add dynamic content to our webpage. As mentioned earlier, it is the object of window. So

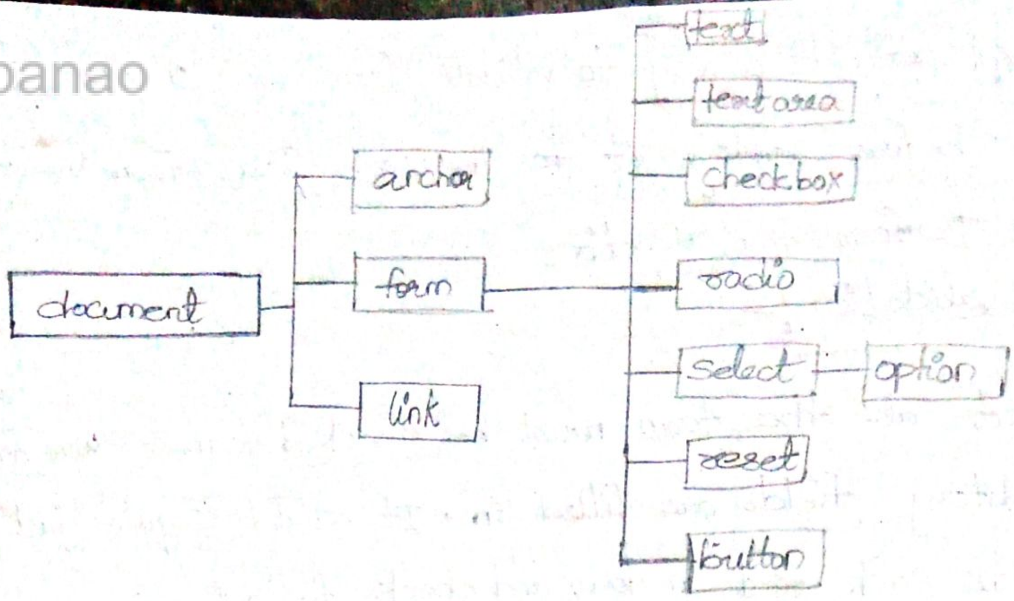
1. window.document

Is same as

1. document

Properties of document object :-

Let's see the properties of document object that can be accessed and modified by the document object.



Methods of document object

We can access and change the contents of document by its methods, The important methods of document object are as follows:

Method	Description
write ("string")	writes the given string on the document
writeln ("string")	writes the given string on the document with newline character at the end
getElementById()	returns the element having the given id value
getElementsByName()	returns all the elements having the given name value
getElementsByTagName()	returns all the elements having the given tag name.
getElementsByClassName()	returns all the elements having the given class name.

Accessing field value by document object

In this example, we are going to get the value of input text by user. Here, we are using document.form1.name.value to get the name field.

Here the document is the root element that represents the html document.

form 1 is the name of the form

name 1 is the attribute name of the input text

value is the property, that returns the value of the input text.

Let's see the Simple example of document object that prints name with welcome message.

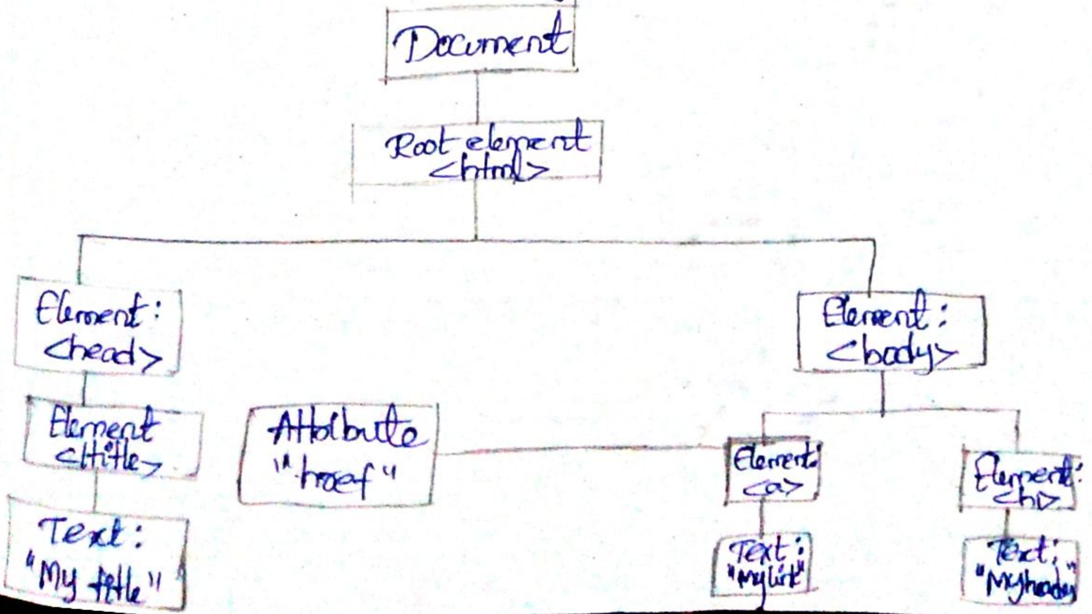
```
<script type="text/javascript">
  function print value() {
 var name = document.form1.name.value;
  alert ("welcome:"+name);
 }
  </script>
  <form name="form1">
Enter Name: <input type ="text" name = "name"/>
<input type = "button" on click ="print value ()" value = 'print name'>
  </form>
```

The HTML DOM (Document object Model):-

When a webpage is loaded, the browser creates a Document object Model of the page

The HTML DOM model is constructed as a tree of objects:

The HTML DOM Tree of objects.



With the object Model, Javascript gets all the power it needs to create dynamic HTML:

- Javascript can change all the HTML elements in the page
- Javascript can change all the HTML attributes in the page.
- Javascript can change all the css styles in the page
- Javascript can remove existing HTML elements and
- attributes
- Javascript can add new HTML elements and attributes
- Javascript can react to all existing HTML events in the page.
- Javascript can create new HTML events in the page