Overview of Java Script

- * Java Script is a client and Server Side object based Scripting language.
- * It is used to make interactive web Pages
- * It is a light weight Programming Language with less Complexity.
- * It reduce the load on the Server.
- * It is an interpreted Language
- * It can be written on the client Side & Server Side.
- * Client Side JS allows to validate the Programs that execute and Produce the result on the client machine.
- * Server Side JS allows to validate the Programs that execute on the Server.
- * another name "Live Script" or ECMA Script
- * It is developed by Netscape and Sun Microsystems

Features of Java Script

1. Imperative and Structured
   - JS Supports all the Syntaxes of the Structured programming language C. ~~So~~ Such as if, loop and switch Statements.

2. Dynamic text
   - JS Supports dynamic typing.
   - The type of variable is defined according to the value

Stored in it.

- for ex variable x, assign it number / String value

3. Functional
   - JS not Support classes.
   - But objects are created from the Constructor function.

4. Prototype - based
   - It is a Prototype-based Scripting Language.
   - It uses Prototypes instead of classes
   - Each Constructor function is associated with a prototype object.
   - ex of Constructor function are : Array (), Boolean (), Date (), math (), Number () ..

5. Platform - independent
   - JS supports Platform - independency or portability
   - write the Script once and run it anywhere at any time.

Using Java Script in an HTML Document

* SCRIPT element is used to insert JS code in an HTML document.
* use the SCRIPT element in a web page in the following ways.

1. In the HEAD element
2. In the BODY "
3. AS an external Script file

②

1. JS in the HEAD element

- place the SCRIPT element inside the HEAD element of an HTML document.

- the Script placed inside the HEAD element runs when you perform some actions like click a link, or submit button.

```
<head>
<Script type = "text / JavaScript ">
   Script Code here
</Script>
</head>
```

2. JS in the BODY element

- place the SCRIPT element inside the BODY element of an HTML document.

- The Scrit element placed inside the BODY elemet runs when a web page starts loading in a web browser.

```
<body>
<Script type = "text / JavaScript ">
   Script Code here
</Script> </body>
```

3. JS in an External File.

# when the JS Code created in an HTML document is very lengthy, it affects the readabili ty of HTML document

- Also use the Same JS Code in Several web pages.

- In this case, Store the JS code in an esternal file and Save it in .js extension.

- Src attribute of Script is used to link the esternal file with the HTML document.

```
<Head>
<Script Src="file.js">
</Script> </head>.
```

Lexical Structure of Java Script

+ Lexical Structure provide the set of rules to write programs.

+ It has.

1. ~~chart~~ Character Set
2. Case Sensitivity
3. White Spaces and line breaks
4. optional Semicolons
5. Comments
6. Literals
7. Identifiers
8. Reserved words

1. Character set

- It is a set of characters reserved ~~reserved~~ to write JS programs.

- ASCII, supports 128 different characters

- Unicode character set ~~[illegible]~~ which Contains over 1,00,000 characters.

- It starts with \u String followed by a four digit hexa decimal number.

Some Special Character in a character set

\t -- Represents tab

\f - " form feed (whitespace)

\n - " newline

\" - " double quote

\' - " Single quote.

\\ - " back slash.

2. Case Sensitivity

- JS is a Case Sensitive Language.
- keywords, variables, function names and identifiers should be typed with a Consistent casing of letters.
- eg function is a keyword that must be written in lowercase, if not JS code generates an error.

3. White Spaces and Line Breaks.

- JS interpreter ignores tabs, spaces and new lines that appear in a Program, except Strings and regular expressions

4. Optional Semicolon

- In JS, Statements are generally followed by ; which indicates the termination of the Statement.
- not use the ; at the end of the Statement.

5. Comments

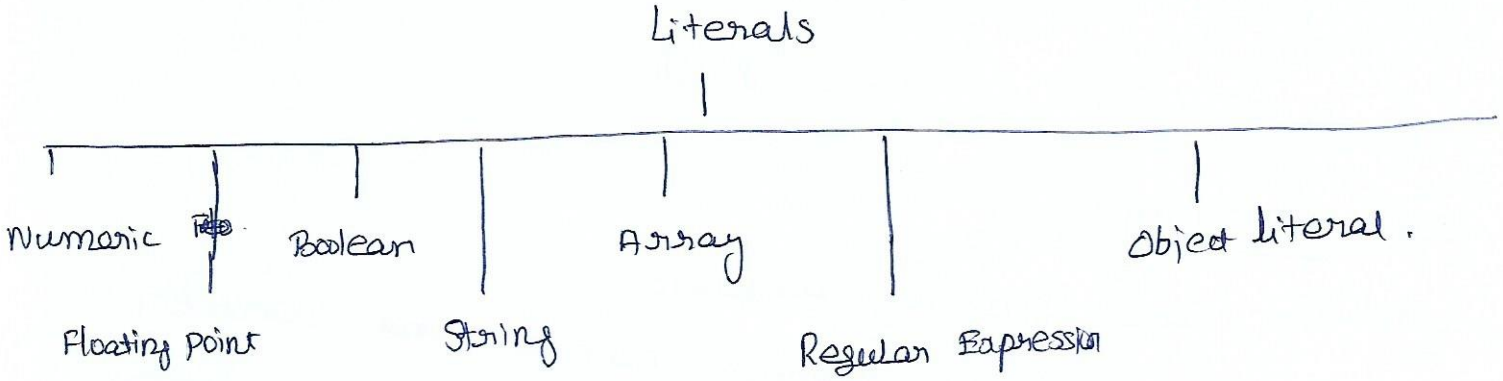
- Comments in a Program that is ignored at the time of executing the Program.

- It supports 2 types of comments

1. Single-line comments - starts with //
2. Multi-line " - Start with /* and end with */.

6. Literals.

- It represents fixed value in a program.



1. Numeric Literal
   - It represents decimal (base 10), hexadecimal (base 16), and octal (base 8) format.
   - ex 89, 021, 065, 0x5673, 0x9123.

2. Floating point Literal
   - Decimal Integer that can be +ve or -ve
   - It has fraction or exponent part
   - ex 4.89023, -123.235 +

3. Boolean literal
   - It is used to make a decision in a relational expression
   - It has only 2 boolean literal values like true/false

4. String Literal
   - It can be zero or more characters enclosed in double or single quotation marks.
   - ex "Hello" 'Hai' "1234"

5. Array Literal
   - Array elements are enclosed in square bracket []
   - ex var emp= ["Abc", "Bcd", "cde"];

④. Regular expression Literal.

- It is a Pattern used to match a character or String in Some text

- ex · Var myexp = /abc/;

⑤. Object Literal

- It is a Collection of name - value Pairs enclosed in curly braces.

- ex ·var games = {cricket : 11, chess : 2, carom : 4};

⑦. Identifiers.

- It refers to the names given to all the Componants Such as variables and methods.

- It Start with a letter, $, or underscore (_).

- It not Start with a number.

- It is case sensitive, not Contain reserved words &

⑧. Reserved words

- Another name keywords.

- It is already defined to the JS interpreter.

- It is not used as identifiers for variable and functions.

ex double, else, if, for.

Variables.

- A variable has a name, value and memory address.

- Data can be temporarily stored in Variables, which are the named locations in the memory.

- Syntax

```
Var variable_name;
```

- ex

```
Var a,b,c;
```

- var is a keyword and variable_name is the name of the variable.
- It allows to store multiple types of values in a variable.
- eg The same variable can store both string and number at different times.

Operators

- Every operator works on one or more operands
- Some operators work on a single operand.
- Some operators work on two or more operands.

1. Arithmetic operator.

+	- Add two numbers / Joins two string	45+20
-	- Subtract two numbers / Represent -ve no/-	40 - 20
*	- multiply 2 numbers	2*5
/	- Divides 2 numbers	6/2
%	- Return the Remainder	7%2
++	- Increment the value of a number by one	++i i++
--	- Decrement the value of a number by one	~~++~~ --i i--

2. Assignment operators.

=	- Assign value to the left side variable	a=10

+=	- Add right side operand to the left & assign result to the left operand	a+=1
		a=a+1

-=	- Subtract " " " " " "	a-=7
		a=a-7

*=	- multiplies " " " " "	a*=4 a=a*4
/=	- Divid " " " " "	a/=5

3. Comparison operators

== - Return true if both operands are equal otherwise, return false   5==5 return true
(equal to)

!= - Returne true if both operands are not equal otherise, return false   45 != 5. return true
(Not equal to)

> - Return true if left side is greater than Right otherwise, return false   35>2 return true
(greater than)

>= - " " " " " " or equal to "   35>=35 return troue
(~~less than~~ (greater than or equal to)

< - " " " " " less than Right   70<20 return false
(less than)

<= " " " " " less than or equal to Right   20<=20 , return true
(less than or equal to)

4. Logical operators

&& - Return true if both operands are true otherwise false   true&&false return false
(logical AND)

|| - Return true if either of the operands is true " " " " false " false "   true||false return true
(logical OR)

! - Negates the operand, return true if operand is false, & return false if operand is true   ! true return false
(logical NOT)

5. Conditional operators

?: - If the first operand is true, it returns the second operand. If the first operand is false it return the third operand   a=5, b=10 c=(a>b)? a:b return c=10

Control Flow Statements

* It is divided into the following categories.

1. Selection Statements
2. Looping "
3. Jump "

Selection Statements

- It allow the execution of a group of statements from multiple groups of statements.

- ~~It is used~~ It uses a condition to select or determine the statement that are to be executed.

- It has
  1. If Statement
  2. if.. else "
  3. Switch "

if Statement

- It is used to execute a group of one or more script statements only when a particular condition is met.

- Syntax

```
if (condition)
{
  Statements
}
```

- If the condition is true then the statements within the curly braces is evaluated. If the condition failed then the statements within the curly braces skipped and the statements immediately after the closing curly brace is executed.

ex

```
<html>
<body>
<Script type = "text/javascript">
 var n= 10;
 if (n% 2 ==0)
  {
   document. write ("It is even number");
  }
</script> </body> </html>.
```

O/P

It is even number.

if... else Statement

* In this statements if the Condition is true then the statements with in the if part is executed.

* If the Condition is failed then the statement with in the else part is evaluated.

* Syntax

```
if(condition)
{ Statement1
}
else
{ Statement 2
}.
```

* ex

```
<html>
<body>
<Script type = "text/ JavaScript">
 var n=5;
 if (n% 2 ==0)
  { document. write ("It is even number");
  }
 else { document. write ("It is odd number");
  }
</script> </body> </html>.
```

O/P

It is odd number.

Switch Statement

* It is used to select ~~as~~ a particular group of Statements to be executed among several other groups of Statements.

* Syntax

```
Switch (expression)
{
  case value 1:
            Statement 1;
            break;
  Case  value 2:
            Statement 2;
            break;
       :
  case  value n:
            Statement n;
            break;
  default:
            Statement ;
            break;
}
```

* ex

```
<html>
<body>
<Script type = "text/javascript">
 Var  letter = "I";
  switch (letter)
  { case "A": document.write ("Vowel-A");
                 break;
    case "E":
         document.write ("Vowel-E");
         break;
   Case "I":
        document.write ("Vowel-I");
        break;
   Case "O":
        document.write ("Vowel-O");
        break;
```

O/P

```
Vowel - I
```

```
        Case "u":
              document.write ("Vowel-U");
              break;
        default:
              document.write ("Consonant");
              break;
     }
   </Script> </body> </html>.
```

~~Control Flow Statements / Loop Statements~~

~~* It is divided into the following~~ →

Loop Statements

* loop or iteration statements are execute a particular group of statements repeatedly.

* JS support 3 loops.

1. while loop
2. do.. while loop
3. for loop.

while loop.

* In this loop, check the condition at the start of the loop.

* If the Condition is true in while loop, then the statements within the curly braces of while loop is executed.

* If the condition is failed then the loop is terminated

Syntax

```
while (condition)
{
   Statement 1;
}
```

ex

```
<html>
<body>
<Script Language= "JavaScript">
var i=1;
While (i<=10)
{
document.write (i);
i++;
}
            <body>.
</Script>, </html>
```

o/p

```
1
2
3
4
5
6
7
8
9
10
```

do.. while loop

* If the group of Statements to execute atleast once, even if the Condition is false. then ~~to~~ use this loop.

* Syntax

```
do
{
Statement1

}
While (Condition);
```

* ex

```
<html>
<body>
<Script Language= "JavaScript">
var i=1;
do
{
document.write (i);

i++;
}
While (i<=10);
</Script> </body> </html>.
```

o/p

```
1
2
3
4
5
6
7
8
9
10
```

for loop

* It allows to execute a group of Statements for a Pre determined number of times.

* The Statement do the loop is executed until the condition failed.

* Syntax

```
for (initialization; Condition ; increment/decrement)
  {
   Statement1
  }.
```

Ex

```
<html>
<body>
<Script language= "JavaScript">
var i;
for ( i=1 ; i<=n ; i++)
  {
   document.write (i);
  }
</Script> </body> </html>.
```

O/P

```
1
2
3
4
5
6
7
8
9
10
```

Jump Statement

* It allows to jump over or skip certain statements in a Script.. and execute some other statements.

* It has
  1. break Statement
  2. Continue "

break Statement

* It is used to break/exit a loop.
* It is used inside the loop, it stops executing the loop and the loop to be exited immediately.

Syntax   break;

eg

```
<html>
<body>
<script language="javascript">
var i;
for(i=1; i<=10; i++)
 {
   if(i==5)
      break;
   document.write(i);
 }
</script> </body> </html>
```

O/P

```
1
2
3
4
```

Continue Statement

* It does not exit the loop but stop the current iteration only. and start to execute the next iteration of the loop.

Syntax   Continue;

eg

```
<html>
<body>
<script language="javascript">
var i;
for(i=1; i<=10; i++)
 {
   if (i==5)
      Continue;
   document.write (i);
 }
</script> </body> </html>
```

O/P

```
1
2
3
4
6
7
8
9
10
```

⑨. Popup Boxes

* It is a window that displays a message along with ok button.
* Js Support 3 types of Popup Boxes
1. Alert Box
2. Confirm "
3. Prompt "

Alert Box

* It is used to display an alert message and also display error message.
* It Contains ok button,
* user clicks ok button than Continue with the execution of the code.

eg

```
<html>
<body>
<Script language = "javascript">
 var a=6;
 if(a%2==0)
    alert ("It is Even number");
 </Script> </body> </html>
```

Confirm Box

* It is used to display a message as well as return a true or false value.
* It displays a dialog box with 2 buttons like ok and cancel.
* click ok button, it returns a true value

* click Cancel button, it returns false value.

* ex

```
<html>
<body>
<Script language="javascript">
var qry = Confirm ("Are you delete a file");
if(qry)
  document.write ("File is Deleted");
else
  document.write ("File is not Deleted");
</script> </body> </html>.
```

Prompt Box

* It is used to input a value from a user.
* It Contains a text box and ok and Cancel buttons
* If the user clicks ok button, the input value is returned. otherwise the box returns null value

ex

```
<html>
<body>
<Script Language="java Script">
var a= prompt ("Enter a number");
if (a>=0)
  alert ("number is +ve");
else
  alert ("Number is -ve");
</Script> </Body> </html>.
```

Java Script Functions, Events, Image Maps and Animations

Functions.

* A function is a set of Statements used to perform a particular task.

* Function are executed when it called in a program.

* It is a Collection of Statements that is executed when it is called at some point in a program.

* In JS, functions are placed under the head and body sections of HTML.

* JS functions are divided into the following Categories

1. Function with Parameter.
2. Function without Parameters.

Built in Functions of JS

1. alert () - Display information in a message box.
2. Prompt() - Display a message box has ok and cancel button.
3. Confirm() - " " " "
4. eval () - Evaluates and executes a String & return result
5. is NaN() - Determine whether or not a value is an illegal numbers

Defining and Invoking a Function

* A function Contains Code that is executed ~~exp~~ when an event is triggered or the function is called.

* Call the function from anywhere within a Program.

Syntax function function-name (Parameter-name1, Parameter-name2 ...... Parameter-name n)

```
{.
   Block of Code
}.
```

- function is a keywork
- function-name is the name of the function.
- function can have any number of Paratmeters

Defining Function Argument

* Arguments are the values passed to the function. While calling it

* The number and order of arguments passed in the function call should match with define in the function.

* After the code is executed, the control of execution passes back to the calling statement.

Syntax function-name (arg1, arg2 ..... argn)

Return Statement

* It specifies the value that is returned from a function.

* Define a function that returns some value, then the function must have the return statement.

* It is used anywhere in the function.

* Syntax: return value;

(15). (1)

```
<html>
<head> <title> Function </title>
<Script language = "JavaScript">
function Simple()
  {
    alert ("Good Morning");
  }
</Script>
</head>
<body onload = "Simple()">
</body> </html>.
```

(2).

```
<html>
<body>
<h1> Passing Arguments to the Function </h1>
<Script Language = "JavaScript">
function calc( a, b, c)
  {   var sum = a+b+c;
      document.write ("Total is = "+ sum);
  }
  calc( 10, 20, 30);
</Script> </body> </html>
```

(3).

```
<html>
<body>
<h1> Using the Return Statement </h1>
<Script Language = "JavaScript">
function area (l, b)
  {
    var res = l * b;
    return res;
  }
var rect = area(4,6);
document.write ("Area of Rectangle
                          is : "+ rect);
</Script> </body> </html>
```

Functions with Timer

- Execute JS function after a specified period.

* JS Provides various methods.

1. setTimeout() - Execute code at a specified interval.

Syntax
setTimeout (function, delayTime)

- function specifies the method that the timer calls & delayTime specifies the number of milliseconds to wait before calling the method.

2. clearTimeout() - Deactivate / cancel the timer that is set using the setTimeout () method.

Syntax
clearTimeout (timer)

- timer is a variable that is created using the setTimeout()

3. setInterval() - Execute a function after a specified time interval

Syntax
setInterval (function, intervaltime)

- function specifies the method to be called, intervaltime specifies the time interval b/w the function calls

4. clearInterval() - Deactivate / ~~clear~~ cancel the timer that is set using the setInterval().

Syntax clearInterval (timer);

ex

```
<html> <head>
<script type= "text/javascript">
function f()
{
var t = setTimeOut ("alert ('5 ~~sec~~ seconds')", 5000);
}
</script> </head> <body> <form>
<input type= "button" value = "Timed alert box" ~~[illegible]~~ onClick= "f()">
</form> </body> </html>.
```

eg SetInterval()

- used to execute the code after every specified time intervals.

- eg

```
<html> <head>
<script type = "text/javascript">
function f()
{
var a = SetInterval ("alert ('1 seconds')", 1000);
}
</script> </head>
<body> <form>
<input type = "button" value = "Timed alert box"
    onClick = "f()">
</form> </body> </html>.
```

Exploring Events

- Events refers to actions that ~~s~~ are detected by a p.gmming Language when you perform a particular task.

+ Events are generally used in combination with functions.

- Event commonly occurs when ~~a~~ click the mouse button, web page is loaded, or form field is changed.

- Events are handled by a special function known as event handlers.

Java Script Events used within the HTML forms

1. onsubmit - Triggers on Submitting a form
2. onselect - " " Selecting an element
3. oninvalid - " when an element is invalid.

4. oninput -- Triggers when input is Provided for an element

5. onfocus - " " a window gets focus.

6. onchange - " " an element changes.

7. onblur - " " a window loses focus.

8. onclick - " on clicking the mouse button

9. ondblclick - " " double clicked the mouse button

10. ondrag - " " dragging an element

11. onmousedown - " " Pressing the mouse button.

12. onmousemove - " " moving the mouse Pointer.

13. onmouseout - " when the mouse pointer leaves an element

14. onmouseover - " when the mouse pointer moves over an element

15. onmouseup - " on releasing the mouse button

keyboard Events

1. onkeydown - Triggers on Pressing a key

2. onkeypress - " when a key is Pressed

3. onkeyup - " on releasing a key.

Events for the media Elements of HTML

1. onabort - Triggers on aborting a Process

2. oncanplay - " when an audio/video file starts

3. onended - " when the media file ends

4. onerror - " when an error occurs on loading an element

5. onloadeddata - " on loading the media file

6. ondurationchange - " on changing the length of audio/video file

7. onpause - Triggers on posing the media file

8. on play - " when a media file is going to play

9. on volume change - " on changing the volume

10. on seeking - " when the seeking process begins

Events of the Browser

1. onblur - Triggers when a window loses focus
2. onerror - " " an error occurs
3. onfocus - " " a window gets focus
4. onhaschange - " at the time of changing a document
5. onload - " at the time of loading a document
6. onoffline - " when a document gets offline
7. ononline - " " " " online
8. onpageshow - " " a window is visible
9. onreize - " at the time of ~~resing~~ resizing a window
10. onunload - " " " " leaving a document

Example of onclick Event

```
<html> <body>
<form name="form">
Name: <input type="text" id="field" value=" "> <br>
<Button onClick ="alert ('welcome '+ document.getElementById
        ('field').value)"> Click me </button>
</form>
</body> </html>
```

* working with onload Event

```
<htmb
 <body onload = "alert ( "Hai ! welcome ')">
 </body> </html>.
```

* working with mouse Event

```
<html>
<body>
 < button onmouse down = "document . images ['img']. src = 'car.jpg' "
onmouse up = "document. images ['img'] . src = 'go. png' ">
  <img name = "img">
  </button> </body> </html>.
```

* working with onSubmit Events

```
<html>
<body>
 < form onSubmit = "alert ('All the details are submitted')">
<h1><B> Application form </B> </H1>
  .<Br>
  Name : <input type = "text" name= "fname" value = " ">
  <br>
 Rollnumber: < input type = "text" name = "fnumber" value = "">
  <br>
 <input type = "Submit" value = "submit">
</form> < body> </Html>.
```

(13).

Image Maps

- An Image map is an image on a web page that Provides various links called hot Spots, to ~~nav~~ navigate to other web pages or other section of the same web page.
- use any shape like circle, rectangle, polygon.
- These shapes . are hotspot link in image maps.
- Add the usemap attribute the img element.
- Add events to the image map using Area element.
- Area element Support various events such as onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, on mouse out, onkeypress, onfocus, onblur.

~~eg flower.html:~~

~~<html>~~
~~<body>~~
~~<img src = "flower.jpg" width = "500" height = "500" alt = "flower"~~

eg

```
<html>
<head> <title> using JS in image map </title>
<Script type = "text/javaScript">
function f(txt)
{
document.getElementById("desc").Src = txt;
}
</Script>
</head>
<body>.
```

```
<img  src = "flower.jpg"  width= 300  height = "300"
alt = "flower"  usemap = "#flowermap"> </img>
<map  name = "flowermap">
    <area  shape = "rect"  coords = "2,7,22,50"
    onmouseover = "f('flower1.jpg')" />
    <area  shape = "circle"  coords = "90,60,10"
    onmouseover = "f('flower2.jpg')" />
</map> <br>
<img  id = "desc"  width = 300  height = "300" >
</body>  </html>
```

Animations

* Animation is a type of optical illusion where the rapid display of a sequence of images or frames of 2-D or 3-D artwork creates an illusion of movements.

* Animation has two categories

1. Sprite animation.
   - It occurs when you move an image called sprite over a background image.
   - It has the ability to move an image over another image.

2. Frame animation.
   - It allows to create fast slide shows.
   - create a set of slides/frames by making small changes in each frame.
   - The set of slides are displayed in quick succession.

* JS Provides the following timing functions to create animation.

1. set Timeout (function, duration) - Execute the code at the duration time
2. Set Interval (function, duration) - " " " after specified time interval
3. clearTimeout (SetTimeout -variable) - Clears the timer set by SetTimeout ().

* Example

```
<html> <head> <title> JS Animation </title>
 <Script Language = "Java Script">
    Var imgobj = null;
    Var animate;
   function init()
   {
    imgobj = document.getElementById('myimage");
    imgobj.style.position = 'relative';
    imgobj.style.left = '0px';
   }
  function moveright()
  {
 imgobj.style.left = parseInt(imgobj.style.left) +10 + 'px';
 animate = setTimeout (moveRight, 200);
   }.
  function Stop()
  {
   clearTimeout (animate);
   imgobj.style.left = '0px';
  }.
  window.onload = init;
 </Script> </head>
 <body>.
```

```
<form>
<img id= "myimage"   src= "car.jpg" />
 <p> click the start button to start animation </p>
<input type = "button" value= "start" onclick = "moveright()" /
<input type= "button" value= "Stop" onclick = Stop() " />
 </form> </body> </html>
```

Java Script Objects

- JS is an object based Scripting Language
- It is used to develop object oriented web applications
- object encapsulates various states and behaviors
- The various states of an object are defined as its Properties.
- behavior of an object is specified by method.
- objects are the instances of a class.
- In JS, an object is an entity that can contain other objects.
- Properties of one object cannot be inherited by any other objects.
- Categories of object in JS
  1. Language object - It specify the objects that are provided by JS Language.
  2. Browser " - It specify the objet that are provided by client browser

objects in JS

- Two ways to create an object
  1. ... Direct Instance
  2. Function template

(15)

Direct ~~Ins~~ Instance of an object

Syntax obj = new object();

- It ~~is~~ is created by using new ~~e~~ keyword
- add properties and methods to an object by using

Period (.)

eg obj. name = "cathy";

obj - object

obj. rollno = 30;

name, rollno - Properties

obj. getValue();

getValue () - method

Function Template of an object

- It ~~is~~ is created by using function keyword.
- add properties to the function template by using this keyword

eg
```
function bike (speed, engine, color)
{
    this.speed = speed;
    this.engine = engine;
    this.color = color;
}
```

Properties of an object

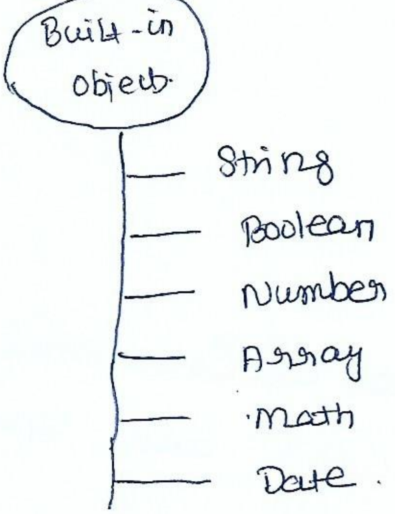
* Another name attribute lies the characteristics of an object.
* ~~o~~ vehicle is an object i. ~~Both~~ > Color and speed are its properties.
*
```
function myobject (Parameter)
{
    this.property1 = parameter
    this.property2 = "Hello world"
}
```

Methods of an object

- A method is a set of one or more statements that are executed by referring the name of the method.
- A method can be defined as an action performed by an object.
- vehicle is an object, Start(), Stop() are the methods
- It is more flexible, easy to maintain, easy to debug. and more readable, increase the reusability of code and many times calling in a Program.

```
function rect()
{
  Var area = this.l * this.b;
  return area;
}
```

The Standard / Built-in JS objects



String object

- String is a sequence of characters.
- In JS, all Strings are represented as instances of the String object.
- It is a wrapper class and a not of global objects.

Properties

Constructor - Return the function, which creates the prototype of a String object.

length - Return the length of the string

Prototype - Add properties and methods to an object

Methods

1. charAt() - Return the character in the specified Index
2. concat() - join 2 strings.
3. indexOf() - Return the position of the first occurance of the specified character in a string
4. LastindexOf() - " " " " Last occurance of a specified value in a string
5. replace() - Replaces the matched substring with a new substring
6 Search() - Search the string.
7 Split() - Split a string into substrings.
8. ~~tolowercase~~ substring() - Return characters in a string b/w two specified indices.
9. ~~too~~ toLowerCase() - Convert string value into lowercase letter.
10 toUpperCase() - Convert string into uppercase letter.

+ A String is created by using literal.
+ String literal is written as a sequence of characters enclosed within ' or " ".

Wrapper ~~both~~ methods that return a String object

1. big() - Displays a string with big font
2. blink() - Make string to blink.

3. bold() - Make String bold.

4. fontColor() - Display the String with specified color

5. font Size() - " " " " " Size

6. sub() - " " " " Subscript text

7. sup() - " " " " Superscript text

ex
```
<html>
<body>
<Script type="text/javaScript">
  var ~~mystring~~ S = "Hello Good morning";
document.write ("String is:" + S + "<Br>");
  "      "     ("Length is:"+ S.length + "<Br>");
  "      "     (" Bold "+ S.bold()+"<br>");
  "      "     ("Uppercase "+ S.toUpperCase()+"<br>");
  "      "     (" Big text " + S.big()+"<br>");
  "      "     (" Substring" after 7 character"+ S.substring(7)+
                                                   "<br>");
  "      "     ("Splitted text (Remove")"+ S.split('Good')+
                                                   "<br>");
</Script> </body> </html>.
```

Boolean object

* It is a wrapper class and a member of global objects.

* It has 2 values true and false.

* The Boolean object returns false when the object is passed with values. Such as 0, " ", false, null, undefined and NaN values.

(17) · It can be created in the following ways.

1. Using boolean literal notation

2. " the Boolean object as function.

3. " the testable expressions.

Boolean Literals

* · It uses two keywords true and false.

    Val bool = true;

* Automatically inherit the members of the boolean object.

* After converting the value of bool to a String by using the toString ().

    document.write (bool.toString () + " ");

Boolean object as Function

* Boolean objects are used to pass the desired initial value as an argument.

    Var bool = Boolean (false);

eg

    ·Var n = 10;

    Var bool = Boolean (n>5);

Testable Expressions

* It is used to evaluate the o/p in boolean values.

* · if(15>10)

    {

    3...

Properties

Constructor - Return the function, which has created the

of the boolean object.

Prototype - Allows to add Properties & methods to an object.

Method

to String() - Convert boolean value into string

value Of() - Return the primitive value of a boolean Object

eg
```
<html>
<body> <Script type="text/javascript">
~~var~~ var bool1 = new Boolean (0);
var bool2 = new Boolean (1);
var bool3 = new Boolean ("");
var bool4 = new Boolean (null);
var bool5 = new Boolean (NAN);
document . write ("0 is boolean" + bool1);
   "      "     ("1 is   "     " + bool2);
   "      "     ("empty string is boolean"+ bool3);
   "      "     ("null is boolean"+ bool4);
   "      "     ("NaN is boolean"+ bool5);
</script> </body> </html>.
```

o/p false
0 is boolean -
1 is boolean - true
empty string is boolean - false
null is boolean - false
NaN is boolean - false

Number object

* All no/- in JS are 64 bit(8 byte) floating point numbers.

* Syntax   var n = new Number (value);

Properties

1. Constructor - Holds the value of the Constructor function that has created the object

2. MAX . VALUE - Return the maximum numerical value.

(8) MIN VALUE - Return the minimum numerical value.

POSITIVE INFINITY - Represents the value of infinity.

NEGATIVE. INFINITY- " " , " negative "

Methods

to String () - Return the value String value for the number object.

value Of() - Return a Primitive value for the number object.

to Fixed (x) - Rounds up a number to x digits after the decimal.

to Precision (x) - " " " to a length of x digits.

ex

```
. <html>
 <body>
 < Script type= "text/javaScript">
 num = new Number ('15.603');
 document.write (num . toExponential () + "<br>");
   "        "     (num. toFixed () );
   "        "     (num. toPrecision (3));
   "        "     (num. toString ());

 </Script>
 </body></html>
```

o/p

```
us   1.5603e+1
     16
     15.6
     15.603
```

Array object

* It is used to Store multiple values in a Single variable.
* In JS, array can hold different types of ~~d~~ data types in a Single array Slot.
* An array can have a String, a number or an object.
* An Array object created in following ways.

1. using the Array Constructor.
2. " " Array Literal notation,

Array Constructor

* empty ~~area~~ array is created in case not know the exact number of elements to be inserted in an array.

```
Var arrayname = new array ();
Var arrayname = new array ( size);
```

ex

```
Var a = new array (5);
Var b= new array (" Sunday", "Monday");
```

Array Literal Notation

* It is comma- Separed list of items enclosed by Square bracket

Syntax

```
Var a = [ ];
Var b= [ 6, "hello", false, true ];
```

Properties

1. Constructor - Holds the value of the Constructor function
2. length - - Holds the nor of elements in an array
3. Prototype - Add properties and methods to the Array Object

Methods

1. Concat() - joins two or more arrays & return the joined array

2. join() - join all the elements of an array

3. Pop() - Remove the Last element of an array..

4. Push() - Add the new element of an array

5. reverse() - Reverse the order of list of elements

6. Sort() - Sort the elements of an array.

Eg

```
<html>
<body>
<Script language="javascript">
a= new Array ("Sunday", "monday", "Tuesday");
b= new Array (1,2,3);
document.write (a);
   "        "   (a. Concat (b));
   "        "   (a. join (b));
   "        "   (a. Pop());
   "        "   (a. reverse());
   "        "   (a. Sort());
</Script></body></html>
```

O/P

```
Sunday monday Tuesday
Sunday monday Tuesday 1.2 3
Sunday 1 2 3 monday 1 2 3
                Tuesday 1 2 3
Tuesday
monday Sunday Tuesday
```

Math object

- It is used to perform simple and complex arithmetic operations.

Properties

1. PI - numeric value of PI, approximate value 3.142
2. E - Holds Euler's number. " " 2.718
3. SQRT2 - Holds Square root of 2 " " 1.414
4. LOG2E - Holds base 2 logarithm of E " " 1.442

Methods

1. abs(x) - Return the absolute value of x
2. cos(x) - " cosine value of x
3. sin(x) - " sine " "
4. sqrt(x) - " Square root of x
5. tan(x) - " tangent value of x
6. log(x) - " the natural logarithmic value of x
7. pow(x,y) - " x to the power of y.

Example

```
<html><body>
<Script language="javascript">
document.write (math.floor(12.6));
  "       "   (math.pow(10,2));
  "       "   (math.sin(45));
  "       "   (math.sqrt(16));
  "       "   (math.tan(45));
  "       "

 </Script>
  </body> </html>
```

O/P

```
12
100
0.85090352453
4
1.61977519054
```

Date object

* It is used to display a date on a webpage or a timestamp in numerical or mathematical calculations.
* Date object is created with the help of constructors.

```
Var d = new Date();
 var d1 = new Date(ms);
 var d2 = new Date(yyyy, mm, dd [, hr, min, sec,
                             millisec]);
 Var d3 = new Date ("mm dd yyyy");
```

(20) Properties

Constructor - Holds the value of the constructor function.

Prototype - Adds properties and methods to the Date object

Methods

1. getDate () - Returns the day of the month.

2. getDay() - " the numerical equivalence of the day of a week range from 0 to 6

4. getFullYear() " " year in 4 digits.

5. getMinutes() - " Minutes ranges from 0 to 59.

6. getMonth () - " the month ranges from 0 to 11

7. getTime () - " " no. of milliseconds.

8. toDateString() - Convert date into String

9. toTimeString () - " time " "

10. SetDate () - Sets the day of a month. Range from 1 to 31

11. Set Hours () - " " hours range from 0 to 23

12. setMonth () - " " months " " 0 to 11

13. set Minutes() - " " minutes " " 0 to 59

14. Set MilliSeconds() - " " milliseconds. " " 0 to 999.

eg

```
<html>
<body>
<Script language="javascript">
var ~~o~~ mydate = new Date();
document.write (mydate.getDate() + "/" + (mydate.getMonth()+1 )+"/"
                                       + (mydate.getFullYear());
           ~~(mydate~~

document.write (mydate.getHours () + ":" + mydate.getMinutes()+ ":"
                    + mydate.getSeconds ());
</script>
```

o/p 08/02/2019

RegExp Object

* Regular Expression is an object that helps to validate the Pattern of characters.

* Defining Regular Expressions in JS is two ways.
  RegExp object's constructor function.

  1. RegExp
  2. Literal.

* Syntax of RegExp Constructor.

```
var RegularExpression = new RegExp("Pattern", "flag");
```

* Pattern is the text of the regular expression
* flag makes the regular expression search for a specific pattern.

ex Match all strings xxx.xxx.xxx.xxx IP address

```
var. regularExp= new RegExp("\\b\\d{1,3}.\\.\\d{1,3}
\\.\\d{1,3}\\.\\d{1,3}\\b", "g");
```

* Syntax of use the literal.

```
var regularexp = /pattern/flag;
```

ex

```
var regularexp= /\b\d{1,3}\.\d{1,3}\.\d{1,3}\.
                \d{1,3}\b/g;
```

- ~~Typ~~ Three types of flags in JS.

  g - Finds all the matches globally
  i - Ignores the case in searches
  m - Represents multiline matching.

Pattern used in the RegExp object

(Pattern) - retain the match

(?: Pattern) - does not retain the match.

?= Pattern - matches only if the given Condition is Satisfied.

?! pattern - " " " " " " is not satisfied.

\f - matches form-feed

\r - " Carriage return.

\n - " line feed.

\t - " horizontal tab.

\v - " vertical tab.

\0 - " NULL character.

\s - " whitespace

\w - " any alphanumerical character.

\d - " any digit

\D - " any non digit.

[characters] - " all the contained Characters.

[^characters] - " all but the " "

List the Qualifiers of the RegExp object

n+ - Specifies a String that Contains atleast one n

n* - " " " " , zero or more occurence of n

n? - " " " " , " " one " "

n$ - " " " " ends with n

^n - " " " " begins with n

?=n - " " " " is followed by a specific String n.

Properties

- global - Refers that the g modifier is set
- ignoreCase - " " " i modifier " "
- Last Index - " the index to start the next match.
- multiline - " " m modifier is set
- Source - " the text of the RegExp Pattern

Method

1. Compile() - Compile the RegExp object
2. exec() - Tests for a match in a String & return a result array
3. test() - " " " " & return true/false.

Browser Objects

* JS has 2 types of objects.
  1. Language objects
  2. Browser "
* when a HTML document is opened in a browser window, the browser interprets the document as a collection of hierarchical objects and displays the data contained in these objects
* browser parses the document and creates a collection of objects.
* Browser objects are of various types.

1. window 2. navigator 3. document 4. screen 5. history 6. location

(22) * window object is used to ~~Res~~ open a window in a browser.

* Navigator objects acts as a Storehouse of all the data and information about the browser.

* Document object is an HTML document that is loaded into a browser.

* History object stores URL visited by a user in the browser.

* Screen object stores the information of the visitor's screen.

* The location object stores the information of the currently visited URL.

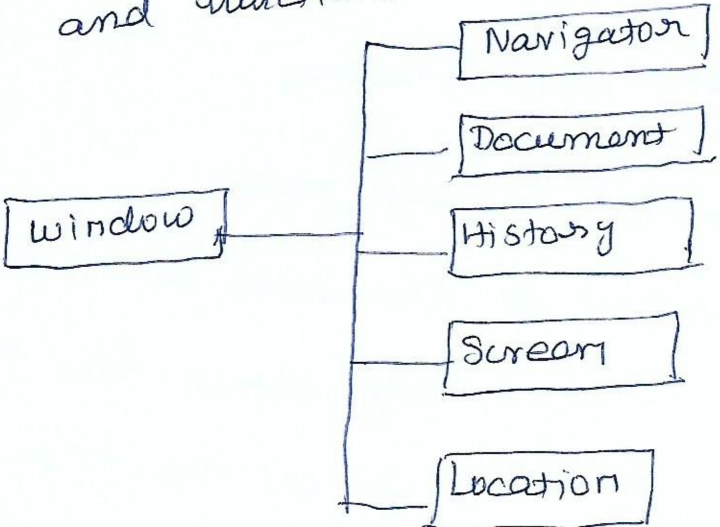
* The collection of browser objects is also known as the Browser Object Model (BOM).

window object

* It is used to open a window in a browser to display the web page.

* It provides global access to the associated variables and functions.

*



* The Properties of the window object are used to retrieve the information about the window.

* methods are used to perform Specific tasks such as opening, maximizing / minimizing the window.

* window object is the top-level object in the hierarchy of browser objects.

document. write ("Java Script")

* It has the following features.

1. window object Collection
2 " " Properties.
3. " " methods.

window Object Collection

* It is a set of all the window objets available in an HTML document.

* HTML document Contains FRAME or IFRAME elemnt.

window object Properties.

* Properties refers to the variables created inside the window object.

* Syntax window. property name.

1. closed - Return a boolean value.
2. document - specifies a Document object in the window
3. history - " a History object for the window
4. frames - " an array of all the frames in the current window

(23) history - Specifies a History object for the window.

innerHeight - " the inner height of a window's Content area

innerwidth - " " inner width " " ".

top - - " " reference of the top most browser window

length - " " no of frames contained in a window

Location - " " the location object for the window.

name - " " name of a window.

Methods

-) The methods associated with the window object Specify actions. Such as how it displays a message or gets I/P from the user.

Syntax window. methodname ();

alert() - Specifies a method that displays an alert box

blur() - " " " removes the focus from the window

close() - " " " Closes the current window

Confirm() - " " " that displays a dialog box

Prompt() - " " " Prompts for Input

Print() - " " " Print the Content of the current window

focus() - " " " Sets focus on the current window

open() - " " " opens a new browser window

resizeBy() - " " " resizes a window with Specified Pixels.

SetTimeout() - " " " evaluates an expression after a Specified no of milliseconds

Sol

```
<html>
<head> <title> window object </title>
  <Script language = "javaScript">
    var w;
   function openwindow(url)
    {
        var width = 400, height = 400;
  var winFeatures = " width =" + width + "heisht " + heisht
    w = window . open (url, "subwind", winFeatures);
    }
 function     show-alert()
    {
    alert ("Hi ! Good Buy");
    }
function   resize Window()
   {
    window . resize By (-100, -100);
   }
function   close window()
   {
      if (window. Confirm ( " Do you  Close the browser" ));
        window . close ();
    }
</Script> </head>
<body>
<input type = "button" . value = "open a new window"
 onclick = "openwindow ('winmethods. html')">
<input type = "button" value = "Alert" onclick = "show-alert
                                                     ()">
<input type = "button" onclick = resize window()" value = "Resre
                                                        window">.
<input type = "button" onclick = close Window()" value = "Close
                                                 the window">
</body> </html>
```

(24)

Navigator object

* It is used to display information about the version and type of the browser.

* It has the following features

1. Collections
2. Properties
3. methods

Collections

* Navigator object is automatically created by the JS runtime System.

* This object Contains all the information about the client browser.

* It Provides the following Collection objects.

1. plugIns[] - Returns a reference to all the embedded objects in the document

2. mimeTypes - Returns a collection of MIME. types Supported by client browser.

Properties

* The Properties of the Navigator object are the variables created inside the navigator object.

Syntax navigator. propertyname.

appcodename - Specifies the Code name of the browser.

appname - " " name of the browser.

appversion - " " version . " "

Cookieenabled - " Whether the Cookies are enabled or not

Platform - Containing a String indicate the machine type

Methods

-> It helpps to check whether or not a browser Support Java code.

JavaEnabled() - Test whether or not Java is enabled

taintEnabled() - Determines whether or not data taining is enabled.

ex

```
  <html>
   <body>
   <script   Language = "javaScript">
document.write (navigator.appName + "<Br>");
  "      "      (navigator.appversion" + "<Br>" );
  "      "      (navigator.appCodeName + "<Br>");
  "      "      (navigator.platform + "<Br>");
  "      "      (navigator. userAgent + "<Br>");

  </script>
   </body> </html>.
```

History object

- It Consists of an an array of URL.
- This object is returned by using the history Property of a window object.
- It has the following features.

1. Properties
2. Methods.

Properties

- It is a Variables Created inside the histay object

Syntax

.history - Property name.

length - specifies the not a element Contained in the object

Current - " " URL a the Current entry in the object

next - . " the URL a the next element in the History list.

Previous - " " " "' Previous " "' "

Methods

Methods - specifies the action related to the URL visited in the browser.

. back() - specifies a method that loads the Previous URL from the history list

forward() - " " " " " the next URL from the history list.

go() - . " " " " " a specific URL from the history list

ex

```
<html>
<head> <title> History object </title>
<Script Language = "javascript">
function f()
{
  var c = history.length;
  alert (" You visited " + c + " web pages");
3.
function fback()
{ . window-history. back();
3
function fbor()
{  window- history. forward();
3.
```

```
function fgo()
{
window. history. go(1);
3.
```

```
</Script>
<input type="button" name="activity" value="My activity"
        onClick = "f()">
<input type="button" value=" ~~Go to next line~~ Forward" onclick="ffor();">
<input type="button" value="Back" .onClick = "fback();">
<input type="button" value="Go to next line" onclick="fgo();">
```

</body> </html>.

Screen Object

→ It represents the current display screen in your browser.

→ The window ~~that~~ is currently displayed in your browser is referred as the screen object.

→ It provides the current information about the dimensions of a display screen such as height, width and color bits.

Syntax window.Screen;

Properties

availHeight - Specifies the height of the screen exclude Task bar

avail Width - " " width " " "

Color Dept - " " depth of the color palette

height - " the total height of the screen

width - " " " width " "

Pixel Depth - " Color resolution. ~~[illegible]~~

```
<html> <body>
<Script Language="javaScript">
document.writeln ( " Total height " + Screen.height + "<Br>");
document.writeln ("Total width" + screen.width + "<Br>");
"      "   ("Available width" + Screen.availWidth + "<Br>");
"      "   ("Available Height") + screen.availHeight + "<Br>");
</script> </body> </html>.
```

Location object

→ It is the child object of the window object.

→ It is used to store information of the current URL of the window object

→ It is used to automatically navigate to the another page

Properties

→ It is the variables created inside the location object

Syntax Location.propertyname;

href - Represents a String, Specify the URL.

protocol - Specifies the method of access of the URL
eg http or https.

host - Represents host name & Port strings.

hostname - " Servername, Subdomain & domain name of a URL

port - " a String specify the communication Port that the Server uses.

Pathname - " a String position of a URL, Specify how a Particular resource Can be accessed.

Methods

assign () - loads a new document in the browser.

reload () - Reload the current document

replace() - Replace the current document with the specified new one.

eg

```
<html>
  <head> <title> Location objekt </title>.
   <Script Language = "JavaScript">
    function fassign()
    {
  window.location.assign ("http://www.gmail.com");
    }
   function freplace()
   {
 window.location.replace ("http://www.google.com");
   }
 </Script> </head>
  <body>.
  <input type= "button" value= "gmail URL". onclick= "fassign()">
 <input type= "button" value= "~~re~~ google ~~cor~~ URL", onclick = "freplace()">
  </body> </html>.
```

Document object

- It Provides different Collection elements, such as anchor and links.
- It is used to access HTML elements of an HTML document.
- It is a Child object of the window object of a webpage.
- It ~~@~~ supports cookies.
- The document object Stores the elements of an HTML document ~~Set~~ such as HTML, HEAD and BODY as objects.

Syntax .window.document (or) document.

Document object Collections

- A collection is defined as an array of related objects.
- This array of objects provides a convenient way to refer to a related group of objects as ~~a~~ a single object

anchors - Refers to an array that returns all the anchor object of an HTML document

forms - Refers to an array that return all the form object of HTML

images - " " " all the image object of an HTML

Links - " " " that returns all the link objects of a document.

Properties

Syntax .object name. Property name;

Cookie - Returns a report that contains all the visible and unexpired cookies associated with the document

domain - Returns the domain name of the Server from which the document has originated.

lastModified - Returns the date on which the document was last modified.

readyState - Returns the loading status of the document.

title - Returns the name of the HTML document defined in b/w Title element

URL - Returns the full URL of the HTML

Methods

Syntax objectname . methodname (arguments);

open - open HTM document to display the o/p.

Syntax document. open (mime type, replace)

close - close an HTM document

Syntax document. close ( )

write - writes HTML expressions / JS code into an HTML

Syntax

document. write (exp1, exp2 -----)

getElementById - Returns the reference of an element with the ~~specified~~ name
Specified

Syntax document . getElementByName (~~name~~ id)

getElementsByName - Returns the reference of an element with the specified name.

Syntax document. getElementsByName (name)

(28)

example:

```
<html)
<head > <title> Document object </title>
   <script type = "text/javascript">
function f()
{
   newwin = window.open("")
   newin.document.open()
   Var abc = document.getElements By Name("nm");
   newwin.document.write ("name length is:" +
                          ab. length.);
   newwin.document.close();
}
</script>
<body>
<img src = "flower1.jpg" width="100" height="100">
<img src = "flower2.jpg" width=100" height="100">
Enter firstname <input name= "nm" type="text".
                                   size = "25">

Enter second Name < input ~~nam~~ name= "nm" type= "text"
                                   size="25" >.

<input type="button" onClick = "f()" value = "Namelength">
<Script Language= "javaScript">
document.~~object~~ write ( document. images. length );
document. write ( document. URL) ;
document. ~~title~~ write (documnt. title>

</Script> </body> </html>.
```

Cookies

* A Cookie Stores some Specific information about the user's Computer, which helps in accessing visited websites.

* A Cookie is a small ~~information~~ file containing information, which a server embeds on a user's Computer.

* The size of a Cookie depends on a browser, but not exceed 1k.

* It is mostly used ~~use~~ to Store user name and Password ~~and Passw~~

. Attributes

* name attribute - Represents a variable name & the relevent value stored in a cookie

* the expire atttributes - specifies the time when the Cookie is deleted.

* domain attribute - Repsents a domain name to which a Cookie is sent.

* path attribute - identifies sites among various paths in the same domain.

* secure attribute = Restricts a browser from sending Cookie information over unsecured Connections.

Create and storing cookies

* JS to create your own Cookies that stores users information.

J Query

* JQuery is a Concise JavaScript Library that enables to quickly and easily create webpages and web application.

* It is a lightweight, JavaScript and Asynchronous JavaScript and XML (AJAX) library.

* It Supports multiple browsers and emphasizes on interaction b/w JS and HTML.

* It enables to create web applications that can Contain animation, communicate with Server to send & requests as get response. and handle events.

* It also allows to select DOM (Document Object model) elements, traverse / navigate them and modify their Content.

* It also captures a wide variety of events. Such as click a button or press key on the keyboard.

* It also provides built in animation effects that can use in websites to make them more attractive and lively.

* It can be integrated with AJAX to create websites.

* It is a Framework that is created using JS capabilities

Advantages of jQuery

1. Allows to add animated effects such as fading in or out and sliding in or out, to HTML elements

2. Enables a webpage to make AJAX requests to a web server to add the data, without reloading the page

3. Allows to manipulate DOM by adding, removing, and reordering the content of a webpage.

4. Allows to create aimated image slide shows, aninzated multi-level dropdown menus and ~~drag-and-~~ [illegible] drag-and-drop interfaces.

5. Allows to create complex forms with validation that retrieve data from a server side database

* ~~[illegible]~~ jQuery supports all the concepts and functions that are available in JavaScript

* String

-) Refers to an immutable object that contains one / more, or none characters.

ex "Hello, Good Morning"

'Hello, Godd morning'

'This is "my" ~~[illegible]~~ Apple'

"This is 'my' Apple"

② → Number

* Represents double - precission 64 bit format values

* eg 5120, 34.52, 0.26

→ Boolean

* Refers to a value that can be either ~~true~~ true/ false

→ Object

* Refers to an instance of an entity.

```
var emp=
{
  name: "cathy";
  age : 12
};
```

→ Array

* Refers to a collection of variables having a single name.

eg
```
var x = [];
var y = [1, 2, 3, 4];
```

→ Function

* Represents a block of code that is defined using the function keyword.

* A function can be named and anonymous.

* A named function has a name.

* Anonymous function does not have name.

```
function named func()
{
  -----
}
var anonymousfunc= function()
{
  -----
```

→ arguments Variable

* Refers to a special variable that is available inside a JQuery function.

* It also has the length property but not the built-in methods of arrays.

```
function mymethod(x)
{
 Console.log( typeof x , arguments.length);
}                                              O/P
mymethod();                                    undefined 0
 mymethod(1);                                  number 1
 mymethod("1","2","3");                        String 3
```

* It has another property of the argument vbl is ~~var~~ callee, which provides the reference of the current function.

```
function myfun()
{
 return. arguments.callee;
}
myfun();
```

→ this keyword

* Refers to the current context of a function.

* The default context of the this keyword is the window object.

* It is used in a function, the current context depends on how the function is called.

(3).

```
$ (document ). ready ( function ()
{
  // this keyword provides the reference of window object
}
);
$ ("a"). click ( function ( )
{
  // this keyword provides the reference of anchor DOM element
});
```

→ <u>variable Scope</u>

* Refers to the region in a Program where a vbl can be accessed.

* A vbl can have either local Scope / global Scope.

→ <u>DOM</u>

* Refers to a tree Structure of HTML elements.

* JQuery provides various selectors to select DOM elements from a DOM document.

* Some Selectors are:

* : Contains () Selector Select all the elements that contain the given text.

* : file () selector Select all the elements that are of file type.

* Some methods are:

- get () - Return the DOM elements that matches with the specified jQuery object

* index () - Searches a specified element within a list of matched elements.

* Size() - Return the no/- of DOM elements that are matched
Specified jQuery object

. toArray() - Return the DOM elements that are matched with the specified jQuery object as array elements

Loading and using jQuery

Version : jQuery 1.5

Formal : regular (uncompresed) - ~~jquo~~ jquery-1.5.js.

Compact (minified) - jquery-1.5.min.js

Downloaded site : http://jquery.com/link.

* Use or load jQuery

```
<Script type = "text/javascript" Langu~~gege~~ = "javascript"
   Src = "jquery-1.5.js"> </Script>
                 (or)
<Script type = "text/javascript" Language = "javascript"
   Src = "jquery-1.5.min.js"> </Script>
```

* Load a jQuery library by providing a Content Distribution Network (CDN) source in the script.

* CDN is a group of computers that are placed at different location and connected with N/w.

* These computers contain copies of data files such as jQuery library and to easy access the data

* The following ways to use Microsoft CDN to load a jQuery file.

```
<Script type = "text/JavaScript" . Language = "javascript"
   src = "http://ajax.microsoft.com/ajax/jquery/
    jquery -1.5.min.js" > </Script>.
```

* The following ways to use Google .AJAX ~~API~~ API CDN to load a jQuery file

```
<Script type = "text/JavaScript" Language = "javascript"
Src = "https://ajax-googleapis.com/ajax/libs/jquery/
    1.5.0/jquery.min.js" > </script>.
```

jQuery Library File

* The following ways to include the jQuery Library file in a web page.

```
<body>
   <Script Src = "jquery- 1.5.js" > </Script>
</body>.
```

* Src attribute is used to specify the Path of the jQuery file.

* jQuery reads or manipulates a DOM document by using the ready event of the document.

* ready event always includes an event handler.

```
<body>
<a href = "www.xyz-com" > jQuery </a>
<Script Src = "jquery -1.5.js" > </Script>
<Script>
   $(document).ready(function () {
      $("a").click(function (event) {
        alert ("Hellow World");
    });
  });
</Script> </body>.
```

* ready event is used to handle the click event of the A element by displaying Hello world.

jQuery Selectors

* It is used to select and manipulate HTML elements as a group or as a single element.

* It support the CSS syntax to select HTML elements by element name, attribute ~~ma~~ name/contact.

ex $("p") - select all p element

$("p.summary") - select all p elements having the summary class

$("[href]") - Select all elements with an href attribute

$("[href='#']") - Select all elements with an href value equal to #

$("[href$='.jpg']") - select all elements with an href attribute that ends with jpg

* jQuery also provides the CSS selectors to change CSS properties of HTML elements.

ex $("p").csc("background-color", "green");

(5)

* - Return all elements - $("*")

#id - Return the element with the specified id a Lastname - $("#Lastname")

.class - Return all elements with class, intro - $(".intro")

element - Returns all p-elements - $("p")

:first - Return the first p element - $("p:first")

:last - " " Last " " - $("p:Last")

:even - " all tr elements at the even row - $("tr:even")

:odd - " " " " " " odd " - $("tr:oodd")

:not(Selector) - " all input elements that are not empty - $("input:not(:empty)")

:header - " all header elements - $(":header")

:animated - " all animated elements - $(":animated")

:Contains(text) - " all elements that Contain the speified text - $(":contains('abc')")

:hidden - " all hidden p elements - $("p:hidden")

[attribute] - " all elements with an href attribute - $("[href]")

:input - " all I/P elements - $(":input")

:text - " all I/P elements with type text - $(":text")

:radio - " " " " " " radio - $(":radio")

:submit - " " " " " " Submit - $(":submit")

:button - " " " " " " button - $(":button")

:image - " " " " " image - $(":image")

: disabled - Return all disabled input elements.
- $(" : disabled")

: selected - Return all Selected " "
- $(" : selected")

: checked - Return all checked input elements
- $(": checked")

Q1
```
<html> <head> <title>
    jquery selet a radio buttom </title>
<Script type= "text/javascript" src= "jquery - 1.5.js"> </Script>
  </head> <body>
  <h1> I think, theeretose I am </h1>
  <h2> select a radio button if you agree with the above statement
     </h2>
  <Script type= "text/javascript >
     $ (document ). ready (function () {
    alert ( $ ('input : radio [name = choice ]: checked').
                                        val ());
     });
   $ (" # selectyes "). click(function () {
                                   : nth (0)'). attr
   $ ('input : radio [name = choice]') ~~prop. checked~~

 ('checked ', true );

 // $ ('input : radio [name = choice ]') [0]. checked = true;
                   });
   $ (" # select NO "). click ( function () {
   $ ('input : radio [name = choice ] : nth (1)'). attr.

 ('checked ', true ) ;
// $ ('input : radio [name = Choice ] ') [1] . checked = true;
          });
  $ (" # select Confused "). click ( function () {
   $ ('input : radio [name = choice ] : nth (2)'). attr.
⑥. ('checked', true);
  // $('input : radio[name= choice]') [2] . checked = true;

      });
   $("#reset"). click (function () {
   $('input : radio[name= choice]'). attr ('checked", false);
  $('input : radio[name= choice]'

    });
});
   </Script>
<input type= "radio" . name= "choice" . value= "yes"> YES
                                                     </Input>
<input type = "radio" . name= "choice" value= "No" > NO </Input>
<input type= "radio" name = "choice" value= "confused">
                                  Confused </input>.

   <Br> <br>
<input type= "button" value= "Display Selected" id='isselect'>
<input type = "button"  value = "yes"  id= 'select yes'>
<    "      "       "  = "No'   id = 'select NO'>
<    "      "       "  = 'confused'  id= 'select Confused'>
<    "      "       "  = 'Reset'   id= 'reset'>

  <body> </html>
```

jQuery Methods to Access HTML attributes

- jQuery enables to read, add, modify or remove an attribute from an HTML element.
- It provides various methods to add, modify or remove the attributes of elements.

Methods

attr (properties) - Sets a key/value object as a Property for all matched elements.

attr (key, fn) - sets a single property to a Computed value for all matched elements

removeAttr(name) - Remove attributes with the specified name from all matched elements.

addClass ('classes) - sets a Style Sheet to all the matched elements based on classes.

removeClass (class) - Remove all / the specified classes from all matched elements

html ( ) - Gets the HTML content of the first matched element.

text (val) - sets the text of all matched elements.

eg

```
<html>
<head>
 <Style>
    img { Padding : 10px; }
    div { Color: red; font-Size : 24px; }.
 </Style>
 <Script Src= "jquery-1.5.js" > </Script>
 </head>
<body>
  <img />
  <img />
  <img />
<Div><B> Not yet defined </B></div>.
<Script>
```

(7)

```
$("img").attr({
        Src : "Laptop.jpg",
        title : "Laptop",
        alt : "Laptop Image"
    });
$("div").text( $("img").attr("alt"));
</Script> </body> </html>
```

jQuery Events

- An event is an action that occurs at a Specified time. due to the completion of some task.
- An event is an action that is performed outside the Scope of a program and is handled by a piece of code inside the program.
- Event handler to handle an event.

eg Clicking of mouse, loading web page, Taking mouse over an element, Submitting HTML form, press key on keyboard.

Event types Supported by jQuery

blur - Specifies an event that occurs when an element loses focus.

change - " " " " " " " an element change

click - " " " " " " " mouse button is clicked

error - " " " " " " " an error in the loading web Page.

focus - " " " " " " " an element gets focus

keypress - " " " " " " a key is Pressed & released

" " " " " " when a key is pressed.

load - Specifies an event that occurs when a document is loaded.

submit - " " " " " " form is submitted.

mouse down - " " " " " " mouse button is pressed.

mouse enter - " " " " " " mouse enters in an element region.

mouse leave - " " " " " " mouse leaves an element region.

mousemove - " " " " " mouse pointer moves.

mouse out - " " " " " when a mouse pointer moves out of an element.

Methods

bind() - Adds one or more event handlers to elements.

blur() - Binds a function to the blur event of selected element.

change() - " " " " change " "

click() - " " " " click " "

error() - " " " " error " "

focus() - " " " " focus " "

focusin() - " " " " focusin " "

focus out() - " " " focusout " "

keydown() - " " " keydown " "

keypress() - " " " keypress " "

keyup() - " " " keyup " "

load() - " " " load " "

submit() - " " " submit " "

mouse down() - " " " mousedown " "

mouse enter() - " " " mouse enter " "

mouse leave() - " " " mouseleave " "

mouse move() - " " " mouse move " "

mouse out() - " " " mouse out " "

⑧·

Sol

```
<htm> <head>
< Script type= "text/javaScript" src = "jquery-1.5.js">
                                                    </Script>
< Script type= " text/javaScript " >
   $ (init);
  function (init)
  {
   $ ('#myButton') . bind ('click', animate para);
  }
 function  ' animate para()
  {
     $ (my Para). ~~fa~~ fadeOut();
     $ (my Para) . fade In ()";
  }
 </Script> </head> <body>
 <Div> . <Input type = "button" id= "myButton' value= "click me
     to Animate the Paragraph"> </Div>.

 < Div>
 <P id = "my Para" Style = "background : Lightgreen; ">
 demo text demo text demo text demo text demo
 demo text demo text demo text demo text demo
 text demo text demo text demo text demo text demo
 </P> </Div>
</body> <htm>
```

Introduction to AJAX

* Asynchronous Java Script and XML (AJAX) is a collection of web Technologies to develop more interactive webapplications /web sites..

* It helps to create interactive web application by retrieving a small amout of data from a webserver.

* need not refresh the entire web application / web Page while retrieving the data.

* AJAX ~~ex~~ exchanges data with a web server and update parts of a webpage without reloding the whole page.

* It is not a Programming language

* It is a way to use existing standards. Such as JS & XML

Exploring different web Technologies

-) Common technologies ~~to~~ to create a webapplication,

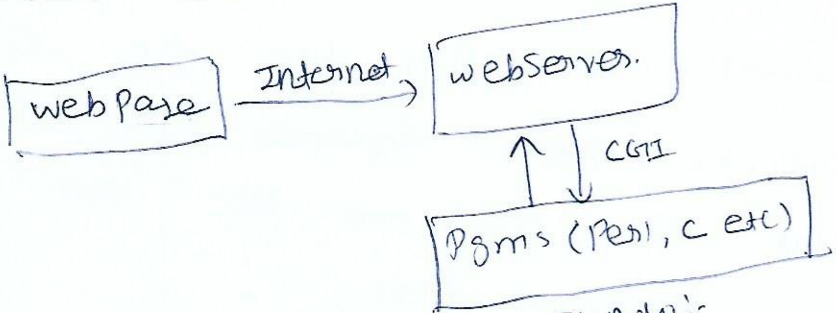
1. Common Gateway Interface (CGI) -

- It refers to a Standard Protocol that is used by web applications to interact with a webserver.

- It is not a programming language, but a Protocol that defines a set of rules on how the web server Communicate with the program.

- CGI is a Specification used to transfer information b/w the web server & CGI Program.

⑨. * CGI pgm is written in any Pgmming Language such as C, Perl & java.



Adv :

→ It is Simple

→ It is quick to develop.

→ It has more libraries.

DisAdv :

→ It is slow.

→ Not preferred fos scripting

→ not provide state persistency

→ create & reload an entire resource

2. Applet

→ It refers to a pgm written in Java Language.

→ It can be included in the HTML Page & run within a web browser.

→ It is used to provide dynamic user interface and a number of graphical effects to the webpages.

3. Java Script

→ It refers to an object-based scripting Language

→ used to create Interactive web pages.

→ JS with HTML code to add dynamic content to the websites.

→ JS is an interpreted language. It runs on browser only

Functions | Task of JS

* create small set of code into HTML Page,

* " dynamic text in HTML Page.

* Read & change the content of HTML controls

* Perform field validations at the client side.
* Create Cookies, which can be used to either store / retrieve relevant information from the client's computer.
* load a specific page. depends upon client's request
* write functions that can use the DOM objects to access ~~to~~ the elements of HTML.

4. Servlet

* Refers to a Java Programming Language class that extends the capabilities of web servers through a request - response Pomming model.
* It is used to Store / Process data that is Submitted by an HTML form.
* Servelets are persistent to manage state information.
* It runs on Java enabled web server.
* the main purpose is to provide HTML pages to a client through HTTP protocol.

5. JSP (Java Server Pages)

* Refers to a technique that generates web pages dynamically.
* It is an enhancement of Servlet
* It Simplifies the Process ~~of~~ creating / developing web Based applications
* It allows HTML to be combined with Java on the same page.
* Dynamically generates HTML documents.

⑥ Active Server Pages (ASP)

→ It refers to a Microsoft · Server side ·Scripting engine used to dynamically generate HTML documents./ web pages.

→ VBScript is the default Scripting Language.

→ ASP runs inside Internet Information Services.

→. ASP page Contains text, XML. & Scripts.

→ It also access the data from a database.

⑦. Hypertext PreProcessor (PHP)

→ Refers to an HTML embedded Scripting Language used to create dynamic web pages

→ It is an Interpreted Language that is executed on Server Side.

→ It is enclosed within <? and ?> .

→ It Provides data base Connection with various types of database management Systems.

→ It Can Contain text, HTML tags and Scripts.

⑧. DHTML (Dynamic Hyper Text Markup Language)

- Refers to a Combination of technologies ~~used~~ used to create dynamic and interactive websites and webapplication.

- DHTML is the art of making HTML pages dynamically

→ It uses various techniques Such as · DOM, CSS, HTML, Java Script · to develop interactive web applications

→ DOM allows to change a part of web page using DHTML.

→ CSS to Control the appearance & feel of a web page / HTML document.

9. XML (Extensible Markup Language)

- Refers to a markup language that is used to exchange information b/w the applications / organizations

- It allows the designers to create their own user defined tags and enable the transmission, Validation, and interpretion of data b/w applications.

- XML uses DOM model to convert an XML document into a tree.

- XML uses XML Stylesheet Language for Transmission (XSLT) Languser to ~~Random [illegible]~~ transform XML document into Extensible HTML (XHTML). or other XML ~~document~~ documents.

- Read, update, Create and manipulate XML documents with the help of an XML Parser.

AJAX

- It is a new approach to create fast and dynamic web applications.

- Developed by Jesse James Garrett.

- It uses JavaScript and XML ~~too~~ for developing web applications.

- AJAX makes the request - response cycle of client and server fast by update the parts of a web page without reload the whole page.

eg. Google maps, Gmail, Youtube & Facebook.

(11) AJAx is based on Some Internet Standard.

1. XML HTTP Request - Refers to an object that is used to exchange data asynchronously with a server.

2. JavaScript and DOM - Displays and interacts with the information

3. CSS - Provides styles to display the data.

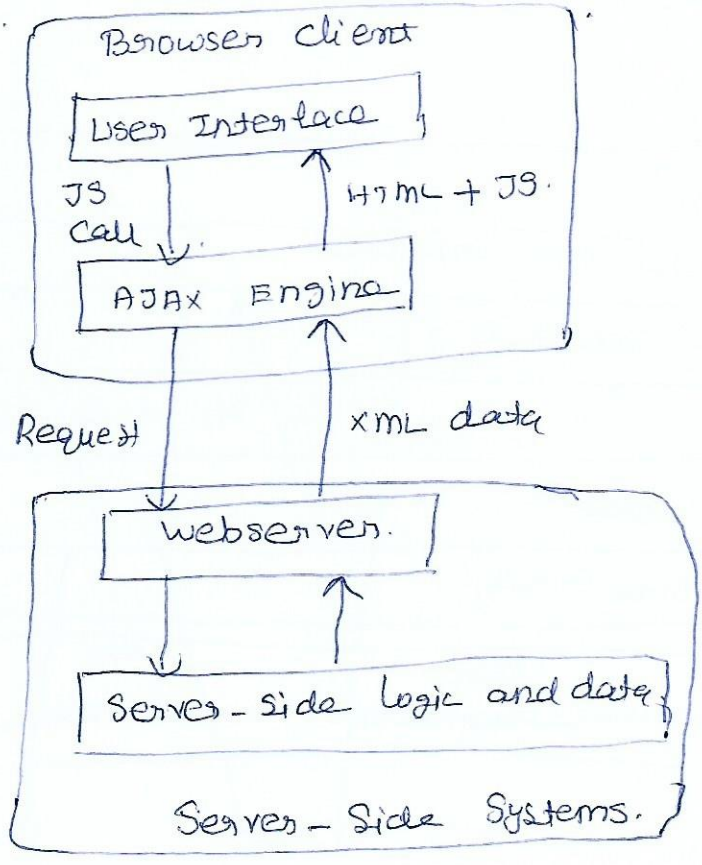
4 - XML - Provides format to transfer the data from a server to client

AJAX web Application model.

* It does not follow the Start-Stop - Start - Stop approach.

* It does not load a webpage during the beginning of a session of client/Server Communication.

* It loads an ASP engine which written in JS.



* established Communication b/w user interface & AJAX engine on the client-side.

* web Page sends its request using JS function.

* ~~@~~ JS function makes a request to the server.

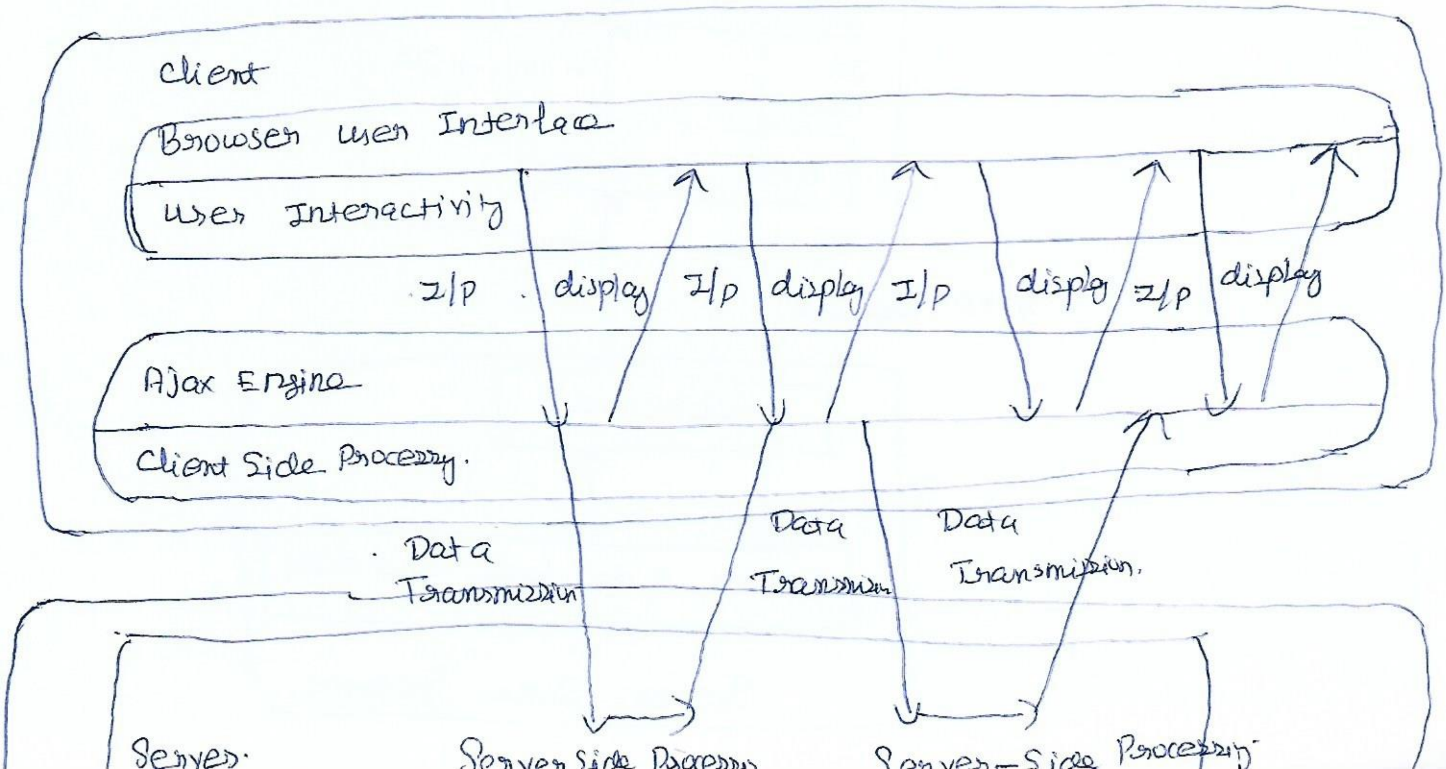
* the server response comprises data like the data required by the page and the Style / presentation is implemented on that data.

* JS dynamically updates the webpage & without reloading the entire web page.

* ~~The~~ The response is made by the server, only a part of the page is updates, so not wait for loading of an entire web page.

* AJAX prevents the user from waiting for the server to complete its processing.

* AJAX provides a partial screen update & asynchronous communication.



(12) * Every user action generates an HTTP request that takes the form a JS to call the AJAX engine

* AJAX engine handled the user request (ie) data validation, data navigation & data manipulation are handled by AJAX engine.

* AJAX engine needs some data/ information from the server, then the engine makes the asynchronous interaction with the server. using JS & XML HttpRequest object.

* AJAX engine's interaction with the server does not interrupt the user's interaction with the application.

How AJAX works

* AJAX works by creating an instance of the XMLHttpRequest object and sending an HttpRequest.

* This request passes over the Internet to a web server.

* The server process the Http Request, request, creates a response and sends the data back to the web browser over the Internet.

1. Creating an instance of the XML HttpRequest object to send an HttpRequest from a client to server.

Syntax Variable = new XMLHttpRequest();

2. Create a request to a server by using open() of the XMLHttpRequest object:

Synta .open(method, url, async).

method - Specify the type of request (ie) GET / POST

Url - Specify the location of a file on the server.

async - " whether the request is handled / not.

eg
```
xmlhttp.open("GET", "file.asp", true);
xmlhttp.send();
```

3. Send a request to a Server by using send() of xmlHttpRequest object

eg
```
xmlhttp.open(" POST", "file.asp", true)
xmlhttp.setRequestHeader("Content-type", "application/
    x-www-form-urlencoded");
xmlhttp.send("fname = charu & lname = verma");
```

(or)
```
xmlhttp.open("GET", "file2.asp? fname = charu & lname =
                verma", true);
xmlhttp.send().
```

eg
```
<html>
<head> <title> Simple AJAX </title>
<script language= "javaScript">
 var ReqObj = false;
 if (window.xmlHttpRequest)
 {
 ReqObj = new XMLHttpRequest();
 }
else if (window.ActiveXObject)
{ ~~xmlHttpRequest of~~ ReqObj = new ActiveXObject("Microsoft.
                                              XMLHTTP");
}
function load (dataSource, divID)
{
    if (~~xmlh~~  ReqObj)
    {
```

13

```
        var obj = document.getElementById ( divID );
        Reqobj.open ("GET", dataSource);
        Reqobj.onreadystatechange = function ()
        {
          if ( Reqobj.readyState = = 4 && Reqobj.status==200)
            {
                obj.innerHTML = Reqobj.responseText;
            }
        }
              Reqobj.Send (null);
     }
  }
</script> </head>                  <body>.
<DIV id="myDiv"> <H2> AJAX change this text </H2></DIV>
                                          Load
<Input type = "button"  onclick = "~~loadXMLDoc~~ ('msg.xml",
            'myDiv')" > change Content </button>
        </body> </html>.
```

msg.xml

```
        <?xml version ="1.0"?>
        <Msgs>
          <msg> Hi ! This is first AJAX application
                                                  </msg>
        </msgs>
```