JavaScript

Java Script

- JavaScript is a scripting language that is used to build more functional and interactive websites
- Some of the uses of javascript
  - Alert messages
  - Pop up windows
  - Dynamic Dropdown menus
  - Form validation
  - Displaying Date/time
- Generally ,runs on client side (browser)
- It is an interpreted, object based scripting language

Java Script Syntax

- Scripts in HTML must be inserted between <script> and </script> tags.
- Scripts can be put in the <body> and in the <head> section of an HTML page.
  - <script type="text/javascript">
    alert("HelloVIT!");
    </script>
- Scripts can also be placed in external files(with a .js extension)
- To use the external javascript add 'src' attribute to <script> element
  - <script src="myScript.js"></script>

Variables

- Variables are named containers to hold data
  - var x=5;
  - var name="Smith";
- Variable names
  - must begin with a letter
  - can also begin with $ and _
  - are case sensitive
- Text values shd be enclosed in quotes
  - var answer='3.14'; //string
- Declaring a variable
  - var name; //declaration
  - name="John"; //assign a value
  - var name="John" //declare and assign a value

Variables

- Javascript is a loosely typed language – variables can hold data of any valid type.
- Variables are declared within a function using 'var' are local variables
  - Var x=10;
- Variables declared outside function with or without 'var' are global variables.
- Variables that are declared inside a function without using *var* keyword are considered as global variables.

Data Types

- String , Number ,Boolean ,Array, Object, null, undefined
- Strings
  - var name="John"
  - var name='John'
- Numbers
  - var x=10.5
  - var x=10
  - var x=123e5
- Boolean
  - var x=true
- Arrays
  - Collection of values in a single variable
  - var fruits = ["apple","orange","mango"] or var num=[1,2,3]

Data Types

- Object
  - Unordered collection of properties
  - Name:value pairs
  - var student={name:"Arun",regno:"10BIT001" ,cgpa:9.2};
  - a = student.regno (or) a= student["regno"]
- Undefined - no value
  - var x; //declared ,value not assigned
- Null - empty
  - var x=null;
- Dynamic data types – same variable can be used for different types
  - var x=5;
  - var x="Hai";

Control Structures

- Branching
  - If statement
  - If…else statement
  - If ….else if statement
  - Switch
- Looping
  - For
  - For In
  - While
  - Do while

Control Structures

- Branching
  - If statement
  - If…else statement
  - If ….else if statement
  - Switch
- Looping
  - For
  - For In
  - While
  - Do while

If ...

- If (condition)

  {

  Statements;

  }

```
if (time<20)
  {
  x="Good day";
  }
```

If ... Else...

- If (condition)

```
{
        Statements;
}
Else
{
        Statement;
}
```

```
if (time<20)
  {
  x="Good day";
  }
else
  {
  x="Good evening";
  }
```

If ... ElseIf...Else

- If (condition)

```
{
        Statements;
}
Else if(condition)
{
        Statement;
}
else
{
        Statement;
}
```

```
if (time<10)
  {
  x="Good morning";
  }
else if (time<20)
  {
  x="Good day";
  }
else
  {
  x="Good evening";
  }
```

Switch

- switch (n)
  {
  case label1:
    code to be executed if n=label1;
    break;
  case label2:
    code to be executed if n=label2;
    break;
  default:
    code to be executed
    if n is different from both label1
  and label2;
  }

```
 var num;
num=Prompt("Enter a number
(1-9)");
switch (num)
{
case 1:
  alert("Number is one!");
  break;
case 2:
  alert("Number is two!");
  break;
case 3:
  alert("Number is three!");
  break;
default:
  alert "Number is greater
than 3!";
}
```

While Loops

- While (condition)

  {

  Statements;

  }

```
while (i<5)
  {
  x="The number is " + i +"<br>";
  i++;
  }
```

Do While Loops

- Do

  {

  Statements;

  } While (condition);

```
do
 {
 x="The number is " + i + "<br>";
 i++;
 }
while (i<5);
```

for

- for (*initialization*; *condition*; *increment*)

  {

  code to be executed;

  }

```
for (var i=0; i<5; i++)
  {
  x="The number is " + i + "<br>";
  }
```

Arrays

- Collection of multiple values in a single variable

```
var colors=[“blue”,”yellow”,”green”];

Colors[0]  is blue
Colors [2] is green
Length of this array is 3 (colors.length)
```

- Values can be added dynamically

```
Colors[5]=“pink”;

Colors[3] and colors[4] will be undefined
Length of this array is 6
```

For In Loop

- Used for looping through properties of object
- For (*x* in *object)*

{

code to be executed;

}

```
var student={name:”Arun",regno:“10BIT001",age:19};

for (x in student)
  {
  txt=student[x];
  }
```

Popup boxes – Alert

- An alert box is often used if you want to make sure information comes through to the user.
- When an alert box pops up, the user will have to click "OK" to proceed.

```
<script>
alert(“Good Morning!")
</script>
```

Popup boxes – Prompt

- A prompt box is often used if you want the user to input a value before entering a page.
- The user will have to click either "OK" or "Cancel“ after entering an input value.
- "OK“ - returns the input value.
- "Cancel“ - returns null.

```
<script>
x=prompt (“Enter ur name”, “ ”)
document.write(“Good Morning  “+x)
</script>
```

Popup boxes – Confirm

- A confirm box is often used if you want the user to verify or accept something.
- The user will have to click either "OK" or "Cancel“.
- If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

```
<script>
x=confirm (“Are you sure you want to delete ?”)
</script>
```

Functions

```
function func_name(parameter1,parameter2,.... ,parameterN)
{
        statement1
}

func_name(argument1,argument2,.....,argumentN)

return value; // used to return info from functions.
```

```
function add(a,b,c)
{
        var sum=a+b+c;
        return sum;
}

 var  total=add(10,20,30) ;
Document.write("Total is:"+total);
```

Functions

- Can also have a named function like

```
var sum = function add(a,b,c)      //both “sum” and “add” refer to same function
{
        var total=a+b+c;
        return total;
}

var  sum1=add(10,20,30);
var  sum2=sum(1,2,3);
```

Function constructors

- JS functions are objects called function objects

```
var add= new Function (“x”,”y”,”return x+y”);
var  sum=add(10,20);

Var greeting= new Function(“return ‘Hello World!’”);
Greeting();
```

JavaScript Objects

- Everything is an object in javascript
  - Built-in objects - String,number,array ,function,..
  - User defined objects
- An object has
  - Properties – Values associated with an object

```
Objectname.propertyname
```

  - Methods – Actions that can be performed on objects.

```
objectname.methodname()
```

```
var str="vit university";
document.write(str.length);   //outputs 14 - property
document.write(str.toUpperCase());//outputs VIT UNIVERSITY
                                  //method
```

JavaScript Objects

- User defined objects
  - Creating a direct instance
  - Using an object constructor
  - Using a constructor function
- Creating a direct instance
  - Student ={name:"Arun",regno:"10BIT001" ,cgpa:9.2};
- Using an object constructor
  - Student =new Object();
  - Student.name="Arun";
  - Student.regno="10BIT001";
  - Student.program="B.tech";

JavaScript Objects

- Using a Constructor function

```
function student(name,regno,program)
{
   this.name=name;
   this.regno=regno;
   this.program=program;
}
var std1=new student(“Ajay”,”10bit002”,”b.tech”);
```

- Adding new properties to an existing object

```
   std1.cgpa=9;
```

- Adding methods
  - Defining methods to an object is done inside the constructor function

JavaScript Objects

```
function book(title,author)
{
   this.title=title;
   this.author=author;
   this.addprice=addprice;      //method
}
function addprice(amount)
{
      this.price=amount;      //new property is added
}

var mybook=new book("WT","Jeffrey Jackson");
mybook.addprice(500);
```

Date Object

- The Date object is used to work with dates and times.
- Date objects are created with the Date() constructor.

```
new Date() // current date and time
new Date(milliseconds) //milliseconds since 1970/01/01
new Date(yy,mm,dd) //with specified date
new Date(yy,mm,dd,hh,mm,ss) //with specified date & time
new Date("Month dd,yyyy") //datestring in the given format
new Date("Month dd,yyyy hh:mm:ss") //datestring in the
                                   //given format
```

```
var today = new Date()
var d1 = new Date("October 13, 1975 11:13:00")
var d2 = new Date(13,5,24)
var d3 = new Date(13,5,24,11,33,0)
```

Date Object-Methods

- getFullYear() //4-digit year
- getMonth() //(0-11 as Jan=0,feb=1…)
- getDate() //(1-31)
- getDay() //(0-6 as Sunday=0)
- getHours //(0-23)
- getMinutes //(0-59)
- getSeconds //(0-59)
- getMilliseconds //(0-999)
- getTimezoneOffset //time diff between local PC and GMT

Current date and time

```
<script>
      var currentDate = new Date()
      var day = currentDate.getDate();
      var month = currentDate.getMonth() + 1;
      var year = currentDate.getFullYear();
      var my_date = day+"-"+month+"-"+year;
      document.write("Todays date is : "+my_date);

      var hours = currentDate.getHours()
      var minutes = currentDate.getMinutes()
      if (minutes < 10){
          minutes = "0" + minutes
      }
      document.write(hours + ":" + minutes + " ")
      if(hours > 11){
          document.write("PM")
      } else {
          document.write("AM")
      }
</script>
```

String Object & Methods

- Can be created using a string literal or constructor

```
name = “VIT university”;
 school=new String (“SITE”);
```

- Methods
  - charAt(index)
    - Returns char at specified index
  - indexOf(searchstring [, index]);
    - Returns index of first char,else returns -1
  - lastIndexOf(searchstring [, index]); //starting from index
    - Searches backwards for last occurence

String Object & Methods

- Methods
  - replace(exp [, replacement text]);

```
name = “Hello World”;
message=name.replace(/World/,”VIT”);       // Hello VIT
```

  - Search(expression)
  - Split(delimiter[,count]); / / an array of substring

```
name = “Hello VIT”;
message=name.split(“ ”);       // Hello, VIT
```

  - Substr(start[,length]) ;
  - toUpperCase()
  - toLowerCase();

```
name = “Hello VIT”;
message=name.toLowerCase();
```

Array Object & Methods

Methods	Description
concat()	Joins two or more arrays, and returns a copy of the joined arrays
indexOf()	Search the array for an element and returns its position
join()	Joins all elements of an array into a string
lastIndexOf()	Search the array for an element, starting at the end, and returns its position
pop()	Removes the last element of an array, and returns that element
push()	Adds new elements to the end of an array, and returns the new length
reverse()	Reverses the order of the elements in an array
shift()	Removes the first element of an array, and returns that element
slice()	Selects a part of an array, and returns the new array
sort()	Sorts the elements of an array
splice()	Adds/Removes elements from an array
toString()	Converts an array to a string, and returns the result
unshift()	Adds new elements to the beginning of an array, and returns the new length
valueOf()	Returns the primitive value of an array

Math Object & Methods

Property	Description
E	Returns Euler's number (approx. 2.718)
LN2	Returns the natural logarithm of 2 (approx. 0.693)
LN10	Returns the natural logarithm of 10 (approx. 2.302)
LOG2E	Returns the base-2 logarithm of E (approx. 1.442)
LOG10E	Returns the base-10 logarithm of E (approx. 0.434)
PI	Returns PI (approx. 3.14)
SQRT1_2	Returns the square root of 1/2 (approx. 0.707)
SQRT2	Returns the square root of 2 (approx. 1.414)

Math Object & Methods

Methods	Description
abs(x)	Returns the absolute value of x
ceil(x)	Returns x, rounded upwards to the nearest integer
cos(x)	Returns the cosine of x (x is in radians)
exp(x)	Returns the value of E^x
floor(x)	Returns x, rounded downwards to the nearest integer
log(x)	Returns the natural logarithm (base E) of x
max(x,y,z,…,n)	Returns the number with the highest value
min(x,y,z,…,n)	Returns the number with the lowest value
pow(x,y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1
round(x)	Rounds x to the nearest integer
sin(x)	Returns the sine of x (x is in radians)
sqrt(x)	Returns the square root of x
tan(x)	Returns the tangent of an angle

Number Object & Methods

Property	Description
MAX_VALUE	Returns the largest number possible in JavaScript
MIN_VALUE	Returns the smallest number possible in JavaScript
NEGATIVE_INFINITY	Represents negative infinity (returned on overflow)
NaN	Represents a "Not-a-Number" value
POSITIVE_INFINITY	Represents infinity (returned on overflow)

Number Object & Methods

Methods	Description
toExponential(x)	Converts a number into an exponential notation
toFixed(x)	Formats a number with x numbers of digits after the decimal point
toPrecision(x)	Formats a number to x length
toString()	Converts a Number object to a string

Global Properties and methods

Global properties and functions can be used with all the built-in JavaScript objects.

Property	Description
Infinity	A numeric value that represents positive/negative infinity
NaN	"Not-a-Number" value
undefined	Indicates that a variable has not been assigned a value

Global Methods

Methods	Description
decodeURI()	Decodes a URI
decodeURIComponent()	Decodes a URI component
encodeURI()	Encodes a URI
encodeURIComponent()	Encodes a URI component
escape()	Deprecated in version 1.5. Use encodeURI()or encodeURIComponent()instead
eval()	Evaluates a string and executes it as if it was script code
isFinite()	Determines whether a value is a finite, legal number
isNaN()	Determines whether a value is an illegal number
Number()	Converts an object's value to a number
parseFloat()	Parses a string and returns a floating point number
parseInt()	Parses a string and returns an integer
String()	Converts an object's value to a string
unescape()	Deprecated in version 1.5. Use decodeURI()or decodeURIComponent() instead

Window Object

- It represents the browser ‘s window
- All javascript objects ,functions and variables are members of window object
- *Document* object is property of window object

Properties	Description
innerHeight	Inner height of the browser window
innerWidth	Inner width of the browser window
defaultStatus	Default message displayed in browser status bar
Document	Current document displayed in the window
Frames	Array of frames of the current window
Length	Number of frames in the current window
Name	Name of the window
Opener	Reference to the window that opened this window
Closed	Indicates whether window is closed or not(true or false)
Self	Reference to cuurent window
tus	Specifies a temp message to display in status bar

Window Object

Methods	Description
Open()	Open a new window
Close()	Close the current window
moveTo(left,top)	Move the current window
resizeTo()	Resize the current window
Alert	Pop ups alert dialog box
Confirm	Pop ups confirm box
Prompt	Pop ups prompt box
moveBy(x,y)	Changes position of window by specified pixels
Print()	Prints the contents of the window
setInterval,clear Interval	
setTimeout, clearTimeout	
Stop	Stops the loading of current page

Screen Object

- It represents the user's screen

Usage : screen.availWidth - returns in pixels

Properties	Description
availHeight	Available screen height
availWidth	Available screen width

Location Object

- It represents the current page address

Properties	Description
Hostname	Domain name of the web host
Pathname	Path and filename of the current page
Port	Port of the web page
Protocol	Returns the web protocol used
Href	Returns the URL of the current page
Methods	
Replace(URL)	Replaces the current URL
Reload()	Reload the current URL

History Object

- It represents the browser ‘s history

Properties	Description
Length	Number of entries in history object

Methods	Description
Back()	Previous URL in the history list
Forward()	Next URL in the history list
Go(relPos|String)	Loads specific URL in the list specified by relative position from the current position or by the URL

Navigator Object

- It contains the browser information

Properties	Description
appcodeName	Code name of the browser
appName	Name of the browser
cpuClass	Type of CPU
Platform	OS
Plugins	Array of plugins installed in the browser
appVersion	Version of the browser
userAgent	Detailed info about the browser
cookieEnabled	Whether cookie is enabled or not(true or false)

Form Validation

- Client side form validation usually done with javascript.
- Form data that typically are checked by a JavaScript could be:
  - Required fields
  - Valid user name
  - Valid password
  - Valid email address
  - Valid phone number