ActiveNET

JavaScript

3 Foundations of Web Techn ology

HTML, CSS, JS

Java Script Tools:

AJAX & RIA Framework:

Prototype http://www.prototypejs.org/
script.aculo.us http://script.aculo.us/
Yahoo! User Interface Library (YUI) http://developer.yahoo.com/yui/
Dojo http://dojotoolkit.org/
jQuery http://jquery.com/
MooTools http://mootools.net/
GWT (Google Web Toolkit) http://www.gwtproject.org/
ExtJS
Adobe Flex

JS SPA Frameworks/MVC based Frameworks:

AngularJS https://angularjs.org/
Angular (Typescript is a subset of JS) https://angular.io/
ReactJS https://reactjs.org/
Backbone.js https://backbonejs.org
Knockout.js https://knockoutjs.com/
Kendo
Sencha Touch
Ember.js
Meteor

Aurelia
Polymer
Vue.js
Mithril.js

Introduction

1) Introduction to JavaScript

JavaScript is invented by Brendan Eich in 1995 when working in Netscape. JavaScript is a high level, interpreted, cross platform, open source, procedural, object based scripting language used in millions of web pages and server applications across World Wide Web.

JavaScript is originally developed by Netscape, later standardized by ECMA (European Computer Manufacturing Association) called ECMAScript.

- HTML is meant for web page designing.
- CSS is for styling the web page
- JavaScript is meant for validations

JavaScript plays 5 roles on web page:

i) Client side/form validation
ii) AJAX communication with server for the SPA sake
iii) XML Parsing
iv) JSON Parsing emps [{eid:1,ename:"Sadhana",}, {}]
v) HTML DOM Manipulation (Dynamic HTML)

JavaScript Vs Java

JavaScript	Java
JavaScript is a Object based Language. 10 pre-defined classes are given, we have to write the application with those 10 pre-defined classes in user defined functions. Developer cannot write their own classes. When there is no class, there is no inheritance, no encapsulation, no abstraction, no polymorphism.	Java is a Object Oriented Language. In Java thousands of pre-defined classes are given. Developer can also write user defined classes.
JavaScript is loosely typed language. No datatypes exist in JS. Hence variables are declared with var keyword followed by	Java is Strictly typed language. Java contains both primitive (byte, short, int, long, float, double, char, boolean) and

var_name. For ex:
var name="MyName";
var age=18;
name=10000.00;
name=true; // No variable stick to any datatype.
	secondary datatypes (arrays and classes)

JavaScript is interpreted language. Interprets code line-line.	Java is both compiled and interpreted language. During compilation source code (*.java) file is converted byte code (.class) file. The byte code file is platform independent. How? In Java for every OS a seperate Java Interpreter is given (java.exe, java.sh, java.so), that interpreter reads neutral byte and converts into machine code on the client system.
JavaScript can be used to detect the visitors web browser using window.navigator object.	Java runs on Server programming language.
JavaScript can be used to create & manage cookies.	
JavaScript gives HTML designers a programming tool.	
JavaScript can put dynamic text into an HTML page called DHTML using HTML DOM API.	

HTML DOM API

window.document contains

- alinkColor
- all
- bgColor
- fgColor
- fullscreen
- height
- linkColor
- vlinkColor
- width
- documentElement
- forms
- head
- hidden
- images

links
plugins
createProcessingInstruction()
createDoctype()
createElement()
createAttribute()
createTextNode()
createComment()
createEntity()
createCDATASection()
getElementById()
getElementsByTagName()
getElementsByClassName()
getElementsByName()
write()
writeln()

Element/Node contains

parentNode
parentElement
childElementCount
addEventListener
firstChild
lastChild
insertBefore()
appendChild()
replaceChild()
removeChild()
childNodes
hasAttributes()
hasChildNodes()
getAttribute()
DOMNamedNodeMap getAttributes()
removeAttribute()
removeEventListener()
querySelector()
querySelectorAll()
textContent

innerHTML
innerText

window.document.querySelector() method returns elements matched with selector

Selector	Example	Example description
.class	.intro	Selects all elements with class="intro"
.class1.class2	.name1.name2	Selects all elements with both name1 and name2 set within its class attribute
.class1 .class2	.name1 .name2	Selects all elements with name2 that is a descendant of an element with name1
#id	#firstname	Selects the element with id="firstname"
*	*	Selects all elements
element	p	Selects all <p> elements
element.class	p.intro	Selects all <p> elements with class="intro"
element,element	div, p	Selects all elements and all <p> elements
element element	div p	Selects all <p> elements inside elements
element>element	div > p	Selects all <p> elements where the parent is a element
element+element	div + p	Selects the first <p> element that is placed immediately after elements
element1~element2	p ~ ul	Selects every element that is preceded by a <p> element
[attribute]	[target]	Selects all elements with a target attribute
[attribute=value]	[target=_blank]	Selects all elements with target="_blank"
[attribute~=value]	[title~=flower]	Selects all elements with a title attribute containing the word "flower"
[attribute|=value]	[lang|=en]	Selects all elements with a lang attribute value starting with "en"
[attribute^=value]	a[href^="https"]	Selects every <a> element whose href attribute value begins with "https"
[attribute$=value]	a[href$=".pdf"]	Selects every <a> element whose href attribute value ends with ".pdf"
[attribute*=value]	a[href*="ActiveNET"]	Selects every <a> element whose href attribute value contains the substring

		"ActiveNET"
:active	a:active	Selects the active link
::after	p::after	Insert something after the content of each <p> element
::before p::before Insert something before the content of each <p> element		
:checked input:checked Selects every checked <input> element		
:default input:default Selects the default <input> element		
:disabled input:disabled Selects every disabled <input> element		
:empty p:empty Selects every <p> element that has no children (including text nodes)		
:enabled input:enabled Selects every enabled <input> element		
:first-child p:first-child Selects every <p> element that is the first child of its parent		
::first-letter p::first-letter Selects the		

first letter of every <p> element		
::first-line p::first-line Selects the first line of every <p> element		
:first-of-type p:first-of-type Selects every <p> element that is the first <p> element of its parent		
:focus input:focus Selects the input element which has focus		
:fullscreen :fullscreen Selects the element that is in full-screen mode		
:hover a:hover Selects links on mouse over		
:in-range input:in-range Selects input elements with a value within a specified range		
:indeterminate input:indeterminat e Selects input elements that are in an indeterminate state		
:invalid input:invalid Selects all input elements with an invalid value		
:lang(language) p:lang(it) Selects every <p> element with a lang attribute equal to "it"		

(Italian)		
:last-child p:last-child Selects every <p> element that is the last child of its parent		
:last-of-type p:last-of-type Selects every <p> element that is the last <p> element of its parent		
:link a:link Selects all unvisited links		
::marker ::marker Selects the markers of list items		
:not(selector) :not(p) Selects every element that is not a <p> element		
:nth-child(n) p:nth-child(2) Selects every <p> element that is the second child of its parent		
:nth-last-child(n) p:nth-last-child(2) Selects every <p> element that is the second child of its parent, counting from the last child		
:nth-last-of-type(n) p:nth-last-of-type(2)Selects every <p> element that is the second <p> element of its parent, counting from the last child		
:nth-of-type(n) p:nth-of-type(2)		

Selects every <p> element that is the second <p> element of its parent		
:only-of-type p:only-of-type Selects every <p> element that is the only <p> element of its parent		
:only-child p:only-child Selects every <p> element that is the only child of its parent		
:optional input:optional Selects input elements with no "required" attribute		
:out-of-range input:out-of-range Selects input elements with a value outside a specified range		
::placeholder input::placeholder Selects input elements with the "placeholder" attribute specified		
:read-only input:read-only Selects input elements with the "readonly" attribute specified		
:read-write input:read-write Selects input elements with the "readonly" attribute NOT specified		
:required input:required		

Selects input elements with the "required" attribute specified		
:root :root Selects the document's root element		
::selection ::selection Selects the portion of an element that is selected by a user		
:target #news:target Selects the current active #news element (clicked on a URL containing that anchor name)		
:valid input:valid Selects all input elements with a valid value		
:visited a:visited Selects all visited links		

JavaScript consists of:

18 events:

i) onLoad

ii) onUnLoad

iii) OnAbort/OnError

iv) onChange

v) onSelect

vi) onFocus

vii) onBlur

viii) onClick

ix) onDblClick

x) onSubmit

xi) onKeyDown
xii) onKeyUp
xiii) onKeyPress

xiv) onMouseDown
xv) onMouseUp
xvi) onMouseOver
xvii) onMouseOut
xviii) onMouseMove

10 classes:

i) Window
- screen
- location
- history
- navigator
- document (HTML DOM Manipulations)
- cookie

ii) String
iii) Date
iv) Math
v) Number
vi) Boolean
vii) Array
viii) RegExpr
ix) ActiveXObject (load XML Parsers)
x) XMLHttpRequest (AJAX object)
json.js file is used to perform JSON parsing

Validations:

NotNull, Minlength, Maxlength, Datatype, Range, Date, Email, URL, Pattern validations

Actual 84 JavaScript Events:

Event	Description	Belongs To

1) abort The event occurs when the loading of a media is aborted
UiEvent, Event

2) afterprint The event occurs when a page has started printing, or if the print dialogue box has been closed
Event

3) animationend The event occurs when a CSS animation has completed
AnimationEvent

4) animationiteration The event occurs when a CSS animation is repeated
AnimationEvent

5) animationstart The event occurs when a CSS animation has started
AnimationEvent

6) beforeprint The event occurs when a page is about to be printed
Event

7) beforeunload The event occurs before the document is about to be unloaded
UiEvent, Event

8) blur The event occurs when an element loses focus
FocusEvent

9) canplay The event occurs when the browser can start playing the media (when it has buffered enough to begin)
Event

10) canplaythrough The event occurs when the browser can play through the media without stopping for buffering
Event

11) change The event occurs when the content of a form element, the selection, or the checked state have changed (for <input>, <select>, and <textarea>)
Event

12) click The event occurs when the user clicks on an element
MouseEvent

13) contextmenu The event occurs when the user right-clicks on an element to open a context menu
MouseEvent

14) copy The event occurs when the user copies the content of an element

ClipboardEvent

15) cut The event occurs when the user cuts the content of an element

ClipboardEvent

16) dblclick The event occurs when the user double-clicks on an element

MouseEvent

17) drag The event occurs when an element is being dragged

DragEvent

18) dragend The event occurs when the user has finished dragging an element

DragEvent

19) dragenter The event occurs when the dragged element enters the drop target

DragEvent

20) dragleave The event occurs when the dragged element leaves the drop target

DragEvent

21) dragover The event occurs when the dragged element is over the drop target

DragEvent

22) dragstart The event occurs when the user starts to drag an element

DragEvent

23) drop The event occurs when the dragged element is dropped on the drop target

DragEvent

24) durationchange The event occurs when the duration of the media is changed

Event

25) ended The event occurs when the media has reach the end (useful for messages like "thanks for listening")

Event

26) error The event occurs when an error occurs while loading an external file

ProgressEvent, UiEvent, Event

27) focus The event occurs when an element gets focus

FocusEvent

28) focusin The event occurs when an element is about to get focus

FocusEvent

29) focusout The event occurs when an element is about to lose focus

FocusEvent

30) fullscreenchange The event occurs when an element is displayed in fullscreen mode

Event

31) fullscreenerror The event occurs when an element can not be displayed in fullscreen mode

Event

32) hashchange The event occurs when there has been changes to the anchor part of a URL

HashChangeEvent

33) input The event occurs when an element gets user input

InputEvent, Event

34) invalid The event occurs when an element is invalid

Event

35) keydown The event occurs when the user is pressing a key

KeyboardEvent

36) keypress The event occurs when the user presses a key

KeyboardEvent

37) keyup The event occurs when the user releases a key

KeyboardEvent

38) load The event occurs when an object has loaded

UiEvent, Event

39) loadeddata The event occurs when media data is loaded

Event

40) loadedmetadata The event occurs when meta data (like dimensions and duration) are loaded

Event

41) loadstart The event occurs when the browser starts looking for the specified media

ProgressEvent

42) message The event occurs when a message is received through the event source

Event

43) mousedown The event occurs when the user presses a mouse button over an element

MouseEvent

44) mouseenter The event occurs when the pointer is moved onto an element

MouseEvent

45) mouseleave The event occurs when the pointer is moved out of an element

MouseEvent

46) mousemove The event occurs when the pointer is moving while it is over an element

MouseEvent

47) mouseover The event occurs when the pointer is moved onto an element, or onto one of its children

MouseEvent

48) mouseout The event occurs when a user moves the mouse pointer out of an element, or out of one of its children

MouseEvent

49) mouseup The event occurs when a user releases a mouse button over an element

MouseEvent

50) mousewheel Deprecated. Use the wheel event instead

WheelEvent

51) offline The event occurs when the browser starts to work offline

Event

52) online The event occurs when the browser starts to work online

Event

53) open The event occurs when a connection with the event source is opened

Event

54) pagehide The event occurs when the user navigates away from a webpage

PageTransitionEvent

55) pageshow The event occurs when the user navigates to a webpage

PageTransitionEvent

56) paste The event occurs when the user pastes some content in an element

ClipboardEvent

57) pause The event occurs when the media is paused either by the user or programmatically

Event

58) play The event occurs when the media has been started or is no longer paused

Event

59) playing The event occurs when the media is playing after having been paused or stopped for buffering

Event

60) popstate The event occurs when the window's history changes

PopStateEvent

61) progress The event occurs when the browser is in the process of getting the media data (downloading the media)

Event

62) ratechange The event occurs when the playing speed of the media is changed Event

resize The event occurs when the document view is resized

UiEvent, Event

63) reset The event occurs when a form is reset

Event

64) scroll The event occurs when an element's scrollbar is being scrolled

UiEvent, Event

65) search The event occurs when the user writes something in a search field (for <input type="search">)

Event

66) seeked The event occurs when the user is finished moving/skipping to a new position in the media

Event

67) seeking The event occurs when the user starts moving/skipping to a new position in the media

Event

68) select The event occurs after the user selects some text (for <input> and <textarea>)

UiEvent, Event

69) show The event occurs when a <menu> element is shown as a context menu

Event

70) stalled The event occurs when the browser is trying to get media data, but data is not available

Event

71) storage The event occurs when a Web Storage area is updated

StorageEvent

72) submit The event occurs when a form is submitted

Event

73) suspend The event occurs when the browser is intentionally not getting media data

Event

74) timeupdate The event occurs when the playing position has changed (like when the user fast forwards to a different point in the media)

Event

75) toggle The event occurs when the user opens or closes the <details> element

Event

76) touchcancel The event occurs when the touch is interrupted

TouchEvent

77) touchend The event occurs when a finger is removed from a touch screen

TouchEvent

78) touchmove The event occurs when a finger is dragged across the screen

TouchEvent

79) touchstart The event occurs when a finger is placed on a touch screen

TouchEvent

80) transitionend The event occurs when a CSS transition has completed

TransitionEvent

81) unload The event occurs once a page has unloaded (for <body>)

UiEvent, Event

82) volumechange The event occurs when the volume of the media has changed (includes setting the volume to "mute")

Event

83) waiting The event occurs when the media has paused but is expected to resume (like when the media pauses to buffer more data)

WheelEvent, Event

84) wheel The event occurs when the mouse wheel rolls up or down over an element

JavaScript object, their properties and their methods:

window class properties & methods:

properties:

closed
console
defaultStatus
document
frameElement
frames
history
innerHeight
innerWidth
length
localStorage
location
name
navigator
opener
outerHeight
outerWidth
pageXOffset
pageYOffset
parent
screen
screenLeft
screenTop
screenX
screenY
sessionStorage
scrollX
scrollY
self

status
top

methods:

open(), close()

alert()
confirm()
prompt()

setInterval()
setTimeout()
clearInterval()
clearTimeout()
stop()

moveBy()
moveTo()

resizeBy()
resizeTo()

scroll()
scrollBy()
scrollTo()

atob()
btoa()

blur()

focus()
getComputedStyle()
getSelection()
matchMedia()
print()
requestAnimationFrame()

window.screen properties & methods:
window.screen contains information about user's screen.
properties:

width, height, availWidth, availHeight, colorDepth, pixelDepth

All modern computers use 24 bit or 32 bit hardware for color resolution:

24 bits = 16,777,216 different "True Colors"
32 bits = 4,294,967,296 different "Deep Colors"

Older computers use 16 bits: 65,536 different "High Colors" resolution.
Very old computers, and old cell phones used 8 bits: 256 different "VGA colors".

window.location properties & methods:
window.location object can be used to get the current page address (URL) and to redirect the browser to a new page.

properties:

href, hostname, pathname, protocol, port, assign

window.history properties & methods:
window.history contains browser's history.

methods:

back(), forward(), go()

window.navigator properties & methods:
window.navigator contains information about the visitor's web browser.

properties:

appName, appCodeName, appVersion, language, product, platform, cookieEnabled, userAgent

window.document.cookie:
Cookies lets developer store user information in browser.

Cookies are data, stored in a small text files on your computer.

When a web server has sent a web page to a browser, the connection is shut down, and the server forgets everything about the user.

Cookies were invented to solve the problem "how to remember information about the user":

When a user visits a web page, his/her name can be stored in a cookie.
Next time the user visits the page, the cookie "remembers" his/her name.
Cookies are saved in name-value pairs like:

username = Suryanarayana

Sample HTML page show how HTML form connects to JavaScript?

JavaScript_1.html

```
<html>
	<head>
		<script>
			function sayHello() {
				// var sname=window.document.enquiryForm.sname.value;
				var sname=window.document.getElementById("sname").value;
				window.alert("Hello "+sname);
			}
		</script>
	</head>
	<body>
		<form name="enquiryForm" method="get" action="./">
		Name <INPUT type="text" name="sname" id="sname" value="" placeholder="Aishwarya"/>
		<INPUT type="button" name="button" value="Click" onClick="sayHello()">
		</form>
	</body>
</html>
```

JS_0.html

```
<html>
	<head>
		<title>First JS Validation Example</title>
```

```
		<script>
			function validateTxt() {

				var
nameValue=window.document.getElementById("name").value;

				if(nameValue.length==0) {

	window.document.getElementById("errorspan").innerHTML="<font color='red'>Name
Field cannot be Empty</font>";
				} else {

	window.document.getElementById("errorspan").innerHTML="";


	window.document.getElementById("outputspan").innerText="Hello "+nameValue;

					window.document.getElementById("output").value="Hello
"+nameValue;
				}
			}
		</script>
	</head>
	<body>
		Name <input type="text" name="name" id="name" value=""
onBlur="validateTxt()">
		<span id="errorspan"></span>
		<br/><br/>

		<span id="outputspan"></span>
		<input type="text" name="output" id="output" value="">

		<br/><br/>
		<input type="button" name="button" id="button" value="Just Click">
	</body>

</html>
```

Variable declaration & initialization:

Syntax:

```
var $_var1_name
var _var1_name
var $var1_name
var $_my_age_is
var $_my_2_age_is
var myAgeIs
var my2AgeIs=18; // variable declaration and initialization
my2AgeIs=20; // variable reinitialization
```

Points to Remember about variables in JavaScript:

i) Variable stores a single data value that can be changed later.

ii) Variables can be defined using var keyword. Variables defined without var keyword become global variables.

iii) Variables must be initialized before using it.

iv) Multiple variables can be defined in a single line. e.g. var one = 1, two = 2, three = "three";

v) Variables in JavaScript are loosely-typed variables. It can store value of any data type through out it's life time.

```
<!-- JS_var_2.html -->
<html>
	<head>
		<script>
			var i=3;
			window.document.writeln(i+"<br/>");
			window.document.writeln("<hr/>");

			i="Three";
			window.document.writeln(i+"<br/>");
			window.document.writeln("<p align='right'>"+i+"</p>");
			window.document.writeln("<hr width='25%' align='right'/>");

			i=true;
			window.document.writeln(i+"<br/>");
```

```
                i=10.24;
                window.document.writeln(i+"<br/>");

                i=new Array(10,20,30,40,50);
                window.document.writeln(i+" len is "+i.length+"<br/>");

                i=new Array("ABC", "XYZ", "LMN");
                window.document.writeln(i+"<br/>");
        </script>
    </head>
    <body>
        Default text in HTML document
    </body>
</html>
```

Type Conversion / Global Functions in JavaScript:

- isNaN() NaN - Not a Number
- eval() - evaluates the expression written in it, though expression contains in quotes or not
- parseInt(str) - converts string to int, if string contains numeric value, other NaN will be written
- parseFloat(str) - converts string to float, if string contains numeric value, other NaN will be written
- typeof - will return what datatype value the give value is

isNan():

```
<!-- JS_NaN_3.html -->
<html>
    <head>
        <script>
            document.write(isNaN(123)+ "<br />");
            document.write(isNaN(-1.23)+ "<br />");
            document.write(isNaN(5-2)+ "<br />");
            document.write(isNaN(0)+ "<br />");
            document.write(isNaN("Hello")+ "<br />");
            document.write(isNaN("2005/12/12")+ "<br />");
            document.write(isNaN("123")+ "<br />");
            document.write(isNaN(" 123 ")+ "<br />");
```

```
                document.write(isNaN("Hello123")+ "<br />");
            </script>
        </head>
        <body>
        </body>
</html>
```

eval():

```
<!-- JS_eval_4.html -->
<html>
        <head>
            <script>
                eval("x=10;y=20;document.write(x*y)");
                document.write("<br />" + eval("2+2"));
                document.write("<br />" + eval(x+17));
            </script>
        </head>
        <body>
        </body>
</html>
```

parseInt():

```
<!-- JS_parseInt_5.html -->
<html>
        <head>
            <script>
                document.write(parseInt("10") + "<br />");
                document.write(parseInt("10.33") + "<br />");
                document.write(parseInt("34 45 66") + "<br />");
                document.write(parseInt(" 60 ") + "<br />");
                document.write(parseInt("40 years") + "<br />");
                document.write(parseInt("He was 40") + "<br />");

            /*
                var sage=window.prompt("Enter your age","5");
                window.alert(typeof(sage));
                var iage=parseInt(sage);
                if(iage>=18) {
```

```
                        window.alert("come and open account");
                } else {
                        window.alert("sorry you are minor");
                }
        */
        </script>
    </head>
    <body>
    </body>
</html>
```

parseFloat():

```
<!-- JS_parseFloat_6.html -->
<html>
    <head>
        <script>
            document.write(parseFloat("10") + "<br />");
            document.write(parseFloat("10.33") + "<br />");
            document.write(parseFloat("34 45 66") + "<br />");
            document.write(parseFloat(" 60 ") + "<br />");
            document.write(parseFloat("40 years") + "<br />");
            document.write(parseFloat("He was 40") + "<br />");
        </script>
    </head>
    <body>
    </body>
</html>
```

typeof:

```
<!-- JS_typeof_7.html -->
<html>
    <head>
        <script>
            var myvar=5;
            alert(typeof myvar);
            // alerts number | string | boolean | object | array | null | undefined
            var myName="Abc";
```

```
                alert(typeof myName);
                var isElected=true;
                alert(typeof isElected);
                var names=new Array(1,2,3,4,5);
                alert(typeof names);
                var blank;
                alert(typeof blank);
                alert(typeof xyz);
        </script>
    </head>
    <body>
    </body>
</html>
```

Operators in JavaScript

JavaScript operators are used to assign values, compare values, perform arithmetic operations, and more.

Arithmetic, Assignment, Increment/Decrement, Relational, Logical, Ternary Operators

JavaScript Arithmetic Operators:

Arithmetic operators are used to perform arithmetic between variables and/or values.

```
<!-- JS_Arithmetic_Operators_8.html -->
<html>
    <head>
        <script>
            /*
            var y = 5;
            function add() {
                var x = y + 2;
                document.getElementById("demo").innerHTML = x;
            }
            function sub() {
                var x = y - 2;
                document.getElementById("demo").innerHTML = x;

            }
```

```
                function mul() {
                        var x = y * 2;
                        document.getElementById("demo").innerHTML = x;

                }
                function div() {
                        var x = y / 2;
                        document.getElementById("demo").innerHTML = x;

                }
                function reminder() {
                        var x = y % 2;
                        document.getElementById("demo").innerHTML =
"<b>"+x+"</b>";
                }
                */
                function add() {
                        var
z=parseInt(window.document.getElementById("num1").value);
                        var
y=parseInt(window.document.getElementById("num2").value);
                        var x = y + z;
                        document.getElementById("demo").innerHTML = x;
                }
                function sub() {
                        var
z=parseInt(window.document.getElementById("num1").value);
                        var
y=parseInt(window.document.getElementById("num2").value);
                        var x = y - z;
                        document.getElementById("demo").innerHTML = x;

                }
                function mul() {
                        var
z=parseInt(window.document.getElementById("num1").value);
                        var
y=parseInt(window.document.getElementById("num2").value);
```

```
                        var x = y * z;
                        document.getElementById("demo").innerHTML = x;

                }
                function div() {
                        var
z=parseInt(window.document.getElementById("num1").value);
                        var
y=parseInt(window.document.getElementById("num2").value);
                        var x = y / z;
                        document.getElementById("demo").innerHTML = x;

                }
                function reminder() {
                        var
z=parseInt(window.document.getElementById("num1").value);
                        var
y=parseInt(window.document.getElementById("num2").value);
                        var x = y % z;
                        document.getElementById("demo").innerHTML =
"<b>"+x+"</b>";
                }
        </script>
    </head>
    <body>
        <p><input type="text" id="num1" value=""></p>
        <p><input type="text" id="num2" value=""></p>
        <p><button onclick="add()">Add</button></p>
        <p><button onclick="sub()">Sub</button></p>
        <p><button onclick="mul()">Mul</button></p>
        <p><button onclick="div()">Div</button></p>
        <p><button onclick="reminder()">Mod</button></p>
        <div id="demo"></div>
    </body>
</html>

<!-- JS_Incr_Decr_Operators_9.html -->
<html>
```

```
	<head>
		<script>
			var x = 5;
			function preIncr() {
				document.getElementById("demo").innerHTML = ++x;
			}
			function postIncr() {
				document.getElementById("demo").innerHTML = x++;
			}
			function preDecr() {
				document.getElementById("demo").innerHTML = --x;
			}
			function postDecr() {
				document.getElementById("demo").innerHTML = x--;
			}
		</script>
	</head>
	<body>
		<p><button onclick="preIncr()()">Pre Increment</button></p>
		<p><button onclick="postIncr()">Post Increment</button></p>
		<p><button onclick="preDecr()">Pre Decrement</button></p>
		<p><button onclick="postDecr()">Post Decrement</button></p>
		<div id="demo"></div>
	</body>
</html>

<!-- JS_Assignment_Operators_10.html -->
<html>
	<head>
		<script>
			function assign() {
				var x = 10;
				var y = 5;
				x = y;
				document.getElementById("demo").innerHTML = x;

			}
			function addAssign() {
```

```
                var x = 10;
                var y = 5;
                x += y;
                document.getElementById("demo").innerHTML = x;
            }
            function subAssign() {
                var x = 10;
                var y = 5;
                x -= y;
                document.getElementById("demo").innerHTML = x;

            }
            function mulAssign() {
                var x = 10;
                var y = 5;
                x *= y;
                document.getElementById("demo").innerHTML = x;

            }
            function divAssign() {
                var x = 10;
                var y = 5;
                x /= y;
                document.getElementById("demo").innerHTML = x;

            }
            function reminderAssign() {
                var x = 10;
                var y = 5;
                x %= y;
                document.getElementById("demo").innerHTML = x;

            }
        </script>
    </head>
    <body>
        <p><button onclick="assign()">Assignment</button></p>
        <p><button onclick="addAssign()">Addition Assignment</button></p>
```

```
            <p><button onclick="subAssign()">Subtract Assignment</button></p>
            <p><button onclick="mulAssign()">Multiplication Assignment</button></p>
            <p><button onclick="divAssign()">Division Assignment</button></p>
            <p><button onclick="reminderAssign()">Reminder Assignment</button></p>
            <div id="demo"></div>
        </body>
</html>
```

Comparison operators are used in logical statements to determine equality or difference between variables or values.

```
<!-- JS_Comparision_Operators_11.html -->
<html>
        <head>
                <script>
                        function equalTo() {
                                var x = 5;
                                document.getElementById("demo").innerHTML = (x == 8);
                                document.getElementById("demo").innerHTML = (x == 5);
                        }
                        function equalValueType() {
                                var x = 5;
                                document.getElementById("demo").innerHTML = (x === "5");
                                document.getElementById("demo").innerHTML = (x === 5);
                        }
                        function notEqual() {
                                var x = 5;
                                document.getElementById("demo").innerHTML = (x != 8);
                        }
                        function notEqualValueType() {
                                var x = 5;
                                document.getElementById("demo").innerHTML = (x !== "5");
                                document.getElementById("demo").innerHTML = (x !== 5);
                        }
                        function greaterThan() {
                                var x = 5;
                                document.getElementById("demo").innerHTML = (x > 8);
```

```
                }
                function lessThan() {
                        var x = 5;
                        document.getElementById("demo").innerHTML = (x < 8);
                }
                function greaterThanEqual() {
                        var x = 5;
                        document.getElementById("demo").innerHTML = (x >= 8);
                }
                function lessThanEqual() {
                        var x = 5;
                        document.getElementById("demo").innerHTML = (x <= 8);
                }
        </script>
    </head>
    <body>
        <button onclick="equalTo()">Equal to</button>
        <button onclick="equalValueType()">Equal value type</button>
        <button onclick="notEqual()">Not equal</button>
        <button onclick="notEqualValueType()">Not equal value type</button>
        <button onclick="greaterThan()">Greater Than</button>
        <button onclick="lessThan()">Less Than</button>
        <button onclick="greaterThanEqual()">Greater Than Equal</button>
        <button onclick="lessThanEqual()">Less Than Equal</button>
    </body>
</html>
```

Conditional/Ternary Operator:

variablename = (condition) ? value1:value2

```
<!-- JS_Ternary_Operator_12.html -->
<html>
    <head>
        <script>
            function myFunction() {
              var age, voteable;
              age = document.getElementById("age").value;
```

```
                    voteable = (age < 18) ? "Too young":"Old enough";
                    document.getElementById("demo").innerHTML = voteable + " to vote.";
                }
            </script>
        </head>
        <body>

        <p>Input your age and click the button:</p>

        <input id="age" value="18">

        <button onclick="myFunction()">Try it</button>

        <p id="demo"></p>

        </body>
</html>
```

Logical Operators:

Logical operators are used to determine the logic between variables or values.

```
<!-- JS_Logical_Operators_13.html -->
<html>
        <head>
                <script>
                        function andAND() {
                                var x = 6;
                                var y = 3;

                                document.getElementById("demo").innerHTML =
                                (x < 10 && y > 1) + "<br>" +
                                (x < 10 && y < 1);
                        }
                        function orOR() {
                                var x = 6;
                                var y = 3;
```

```
                document.getElementById("demo").innerHTML =
                (x === 5 || y === 5) + "<br>" +
                (x === 6 || y === 5) + "<br>" +
                (x === 6 || y === 3);
        }
        function not() {
                var x = 6;
                var y = 3;

                document.getElementById("demo").innerHTML =
                        !(x === y) + "<br>" +
                        !(x > y);
        }
    </script>
</head>
<body>
    <p>The AND operator (&&) returns true if both expressions are true, otherwise it returns false.</p>

    <p id="demo" onClick="andAND()">&amp;&amp; operator</p>
    <p id="demo" onClick="orOR()">|| operator</p>
    <p id="demo" onClick="not()">! operator</p>
</body>
</html>
```

Control Structures:

Control Structures actually controls the flow of execution of a program. Control Structures constitutes of conditions and loops.

- if, if-else, if-else if-else
- switch case
- for loop
- while loop
- do-while loop

if, if-else, if-else if-else:

i) Use if-else conditional statements to control the program flow.

ii) JavaScript includes three forms of if condition: if condition, if else condition and else if condition.
iii) The if condition must have conditional expression in brackets () followed by single statement or code block wrapped with { }.
iv) 'else if' statement must be placed after if condition. It can be used multiple times.
v) 'else' condition must be placed only once at the end. It must come after if or else if statement.

Validations:
notnull, datatype, length, minlength, maxlength, date, url, pattern validations using string, pattern validations using regexpr etc.

```
<!-- JS_SimpleIfElseCondition_14.html -->
<html>
	<head>
		<script>
			// var age = 20;
			var sage=window.prompt("Enter age to apply for driving", "0");
			var iage=parseInt(sage);
			if( iage >= 18 ){
				window.alert(iage+" age Qualified for driving");
				document.write(iage+"<b> age Qualified for driving</b>");
			} else {
				window.alert(" Not Qualified for driving");
				document.write("<b>Not Qualified for driving</b>");
			}
		</script>
	</head>
	<body>
	</body>
</html>

<!-- JS_MobileIf_14_1.html -->
<html>
	<head>
		<script>
			function validateMobile() {
				var smobile=window.document.getElementById("mobile").value;
```

```
                        if(isNaN(smobile)) { // if not a number
                                window.alert("Not a valida mobile number");
                                window.document.getElementById("mobile").value="";
                                window.document.getElementById("mobile").focus();
                        } else {
                                if(smobile.length!=10) {
                                        window.alert("Mobile number requires 10 digits");

        window.document.getElementById("mobile").value="";

        window.document.getElementById("mobile").focus();
                                }
                        }
                }
        </script>
        </head>
        <body>
                Enter Mobile <input type="text" id="mobile" name="mobile"/><br/>
                <input type="button" name="button" value="Button"
onFocus="validateMobile()"/>
        </body>
</html>


<!-- JS_SimpleIfElseIfElseCondition_15.html -->
<html>
        <head>
                <script>
                        var mySal = 500;
                        var yourSal = 1000;

                        if( mySal > yourSal) {
                                alert("My Salary is greater than your salary");
                        }
                        else if(mySal < yourSal) {
                                alert("My Salary is less than your salary");
                        }
                        else if(mySal == yourSal) {
```

```
                        alert("My Salary is equal to your salary");
                    } else {
                        alert("I don't know");
                    }
                </script>
        </head>
        <body>
        </body>
</html>
```

switch case:

The switch is a conditional statement like if statement. Switch is useful when you want to execute one of the multiple code blocks based on the return value of a specified expression.

i) The switch is a conditional statement like if statement.
ii) A switch statement includes literal value or is expression based
iii) A switch statement includes multiple cases that include code blocks to execute.
iv) A break keyword is used to stop the execution of case block.
v) A switch case can be combined to execute same code block for multiple cases.

Syntax:

```
switch(expression or literal value){
   case 1:
      //code to be executed
     break;
   case 2:
      //code to be executed
      break;
   case n:
      //code to be executed
      break;
   default:
      //default code to be executed
      //if none of the above case executed
}
```

<!-- JS_Switch_Condition_16.html -->

```
<html>
	<head>
		<script>
			var a = 3;

			switch (a) {
				case 1:
					alert('case 1 executed');
					break;
				case 2:
					alert("case 2 executed");
					break;
			  case 3:
					alert("case 3 executed");
					break;
				case 4:
					alert("case 4 executed");
					break;
				default:
					alert("default case executed");
			}

			switch (a/3) {
				case 1:
					alert("case 1 executed");
					break;
				case 2:
					alert("case 2 executed");
					break;
				case 3:
					alert("case 3 executed");
					break;
				case 4:
					alert("case 4 executed");
					break;
				default:
					alert("default case executed");
			}
```

```
                var str = "ActiveNET";

                switch (str) {
                        case "Surya":
                                alert("This is Surya");
                        case "narayana":
                                alert("This is narayana");
                                break;
                        case "ActiveNET":
                                alert("This is ActiveNET");
                                break;
                        default:
                                alert("Unknown Person");
                }

                // Combined switch cases
                var b = 2;

                switch (b) {
                        case 1:
                        case 2:
                        case 3:
                                alert("case 1, 2, 3 executed");
                                break;
                        case 4:
                                alert("case 4 executed");
                                break;
                        default:
                                alert("default case executed");
                }

        </script>
    </head>
    <body>
    </body>
</html>
```

for loop:

JavaScript includes for loop to execute code repeatedly.

syntax:

```
for(initializer; condition for termination; increment/decrement) {
   // Code to be executed
}
```

The for loop requires following three parts:

+ Initializer: Initialize a counter variable to start with
+ Condition: specify a condition that must evaluate to true for next iteration
+ Iteration: increase or decrease counter

```
<!-- JS_For_Loop_17.html -->
<html>
	<head>
		<script>
			for (var i = 0; i < 5; i++) {
				console.log(i);
			}

			// var arr = new Array(10, 11, 12, 13, 14);
			var arr = [10, 11, 12, 13, 14];

			for (var i = 0; i < arr.length; i++) {
				console.log(arr[i]);
			}

			var arr = [10, 11, 12, 13, 14];
			var i = 0;

			for (; ;) {
				if (i >= 5)
					break;
				console.log(arr[i]);
				i++;
```

```
                }
            </script>
        </head>
        <body>
        </body>
</html>
```

while loop:

JavaScript includes while loop to execute code repeatedly till it satisfies a specified condition. Unlike for loop, while loop only requires condition expression.

```
<!-- JS_While_Loop_18.html -->
<html>
        <head>
            <script>
                var i =0;
                while(i < 5) {
                    console.log(i);
                    i++;
                }
            </script>
        </head>
        <body>
        </body>
</html>
```

do-while loop:

do-while loop will execute a code block even if the condition turns out to be false in the first iteration.

```
<!-- JS_Do_While_Loop_19.html -->
<html>
        <head>
            <script>
                var i = 0;
```

```
                do {
                        alert(i);
                        i++;
                } while(i < 5)
        </script>
    </head>
    <body>
    </body>
</html>
```

Functions in JavaScript:

JavaScript provides functions similar to most of the scripting and programming languages.

In JavaScript, a function allows you to define a block of code, give it a name and then execute it as many times as you want.

A JavaScript function can be defined using function keyword.

Define and call a function:

JS_Define_Call_Function_19.html

```
<html>
    <head>
        <script>
            function ShowMessage() {
                alert("Hello World!");
            }
            ShowMessage();
        </script>
    </head>
    <body>
    </body>
</html>
```

Function with Parameters:

```
<!-- JS_Function_With_Parameters_20.html -->
<html>
    <head>
```

```
		<script>
			function ShowMessage(firstName, lastName) {
				// alert("Hello " + firstName + " " + lastName);
				// console.log("Hello " + firstName + " " + lastName);
				window.document.getElementById("output").innerHTML+="Hello
" + firstName + " " + lastName+"<br/>";
			}

			function displayMessages() {
				ShowMessage("Surya", "Active");
				ShowMessage("Active", "Surya");
				ShowMessage(100, 200);

				ShowMessage("Surya", "Active", "Mr."); // displays: Hello Surya
Active
				ShowMessage("Surya"); // displays: Hello Surya undefined
				ShowMessage(); // displays: Hello undefined undefined
			}
		</script>
	</head>
	<body onLoad="displayMessages()">
		<div id="output"></div>
	</body>
</html>
```

Function with Arguments object:

All the functions in JavaScript can use "arguments object" by default. An arguments object includes value of each parameter.

The arguments object is an array like object. You can access its values using index similar to array. However, it does not support array methods.

```
<!-- JS_Function_With_Arguments_Object_21.html -->
<html>
	<head>
		<script>
			function ShowMessage() {
				var len=arguments.length;
```

```
                        var output="Hello ";
                        for(var i=0;i<len;i++) {
                                // output+=arguments[i]+" ";
                                output=output+arguments[i]+" ";
                        }

                        //
window.document.getElementById("output").innerHTML+=output+"<br/>";


        window.document.getElementById("output").innerHTML=window.document.getEleme
ntById("output").innerHTML+output+"<br/>";

                }

                function dispMsgs() {
                        ShowMessage("Surya", "Active", "Sadhana");

                        ShowMessage("Active", "Surya", "Sadhana", "Revanth");

                        ShowMessage(100, 200);
                }
        </script>
    </head>
    <body onLoad="dispMsgs()">
        <div id="output"></div>
    </body>
</html>
```

Iterate all Arguments:

JS_Function_Iterate_Arguments_22.html

```
<html>
    <head>
        <script>
            function ShowMessage() {
                for(var i = 0; i < arguments.length; i++){
                    alert(arguments[i]);
```

```
                }
            }
            ShowMessage("Surya", "Active");
        </script>
    </head>
    <body>
    </body>
</html>
```

Function with return value:

JS_Function_with_return_value_23.html

```
<html>
    <head>
        <script>
            function Sum(val1, val2) {
                return val1 + val2;
            };

            var result = Sum(10,20); // returns 30
            console.log(result);

            function Multiply(val1, val2) {
                // return val1 * val2; // if we don't write return statement
                console.log( val1 * val2);
            };

            result = Multiply(10,20);
            console.log(result);
            // undefined, but return value is taken into a variable
        </script>
    </head>
    <body>
    </body>
</html>
```

Function returning a function:

```
<!-- JS_Function_returning_Function_24.html -->
<html>
	<head>
		<script>
			function multiple(x) {

				function fn(y)
				{
					return x * y;
				}

				// function returning a statement is routine
				// but function returning another function is a variety
				return fn;
			}

			// this triple is a pointer to inner function (fn())
			// here multiple function calling return fn() function
			var triple = multiple(3);

			var output=triple(2); // returns 6
			window.alert(output);

			output=triple(3); // returns 9
			window.alert(output);
		</script>
	</head>
	<body>
	</body>
</html>
```

Function Expression:

JavaScript allows us to assign a function to a variable and then use that variable as a function. It is called function expression.

```
<!-- JS_Function_Expression_25.html -->
<html>
```

```
	<head>
		<script>
			var add =  function sum(val1, val2) {
				return val1 + val2;
			};

			var result1 = add(10,20); // 30
			window.alert(result1);

			// var result2 = sum(10,20); // not valid
			// window.alert(result2);

		</script>
	</head>
	<body>
	</body>
</html>
```

Anonymous Function:

JavaScript allows us to define a function without any name. This unnamed function is called anonymous function. Anonymous function must be assigned to a variable.

```
<!-- JS_Anonymous_Function_26.html -->
<html>
	<head>
		<script>
			var showMessage = function (){
				alert("Hello World!");
			};

			showMessage();

			var sayHello = function (firstName) {
				alert("Hello " + firstName);
			};

			sayHello("Surya");
```

```
        </script>
    </head>
    <body>
    </body>
</html>
```

Nested Functions:

In JavaScript, a function can have one or more inner functions. These nested functions are in the scope of outer function. Inner function can access variables and parameters of outer function. However, outer function cannot access variables defined inside inner functions.

```
<!-- JS_Nested_Function_27.html -->
<html>
    <head>
        <script>
            function ShowMessage(firstName)
            {
                function SayHello() {
                    window.alert("Hello " + firstName);
                }

                return SayHello();
            }

            ShowMessage("Surya");
        </script>
    </head>
    <body>
    </body>
</html>
```

Array:

i) An array is a special type of variable that stores multiple values using a special syntax.

ii) An array can be created using array literal or Array constructor syntax.

iii) Array literal syntax: var stringArray = ["one", "two", "three"];

iv) Array constructor syntax:var numericArray = new Array(3);

v) A single array can store values of different data types.
vi) An array elements (values) can be accessed using zero based index (key). e.g. array[0].
vii) An array index must be numeric.
viii) Array includes length property and various methods to operate on array objects.

We have learned that a variable can hold only one value, for example var i = 1, we can assign only one literal value to i. An array is a special type of variable, which can store multiple values using special syntax. Every value is associated with numeric index starting with 0.

Array Initialization:
An array in JavaScript can be defined and initialized in two ways, array literal and Array constructor syntax.

Array Literal:

Array literal syntax is simple. It takes a list of values separated by a comma and enclosed in square brackets.

```
var stringArray = ["one", "two", "three"];

var numericArray = [1, 2, 3, 4];

var decimalArray = [1.1, 1.2, 1.3];

var booleanArray = [true, false, false, true];

var mixedArray = [1, "two", "three", 4];
```

Array Constructor:

```
var stringArray = new Array();
stringArray[0] = "one";
stringArray[1] = "two";
stringArray[2] = "three";
stringArray[3] = "four";

var numericArray = new Array(3);
numericArray[0] = 1;
```

```
numericArray[1] = 2;
numericArray[2] = 3;

var mixedArray = new Array(1, "two", 3, "four");
```

Array Property:

Array is having only one property named "length" which returns number of elements in the array.

```
var stringArray = new Array("one", "two", "three", "four");

for (var i = 0; i < stringArray.length ; i++) {
   stringArray[i];
}
```

Events in JavaScript:

HTML events are that happens on HTML elements/tags when something browser does or user does.

19 events:
---
i) onLoad (body)
ii) onUnLoad (body)
iii) onAbort/onError (body)

iv) onChange (select, checkbox, radio)
v) onSelect (text, textarea)

vi) onFocus (on all form elements - text, textarea, password, file, button, submit, reset, select, checkbox, radio, image)
vii) onBlur (on all form elements)

viii) onClick (button, div, span, p, img, ul/ol->li)
ix) onDblClick (button, div, span, p, img, ul/ol->li)

x) onSubmit (form)

xi) onKeyDown (text, textarea)
xii) onKeyUp (text, textarea)
xiii) onKeyPress (text, textarea)

xiv) onMouseDown (any element)
xv) onMouseUp (any element)
xvi) onMouseClick (any element)
xvii) onMouseOver (any element)
xviii) onMouseOut (any element)
xix) onMouseMove (any element)

```
<!-- JS_Event_OnLoad_28.html -->
<html>
	<head>
		<script>
			function sayHello() {
				alert("Just now document is loaded");
			}
			function sayBye() {
				alert("Document is unloading...");
			}

		</script>
	</head>
	<body onLoad="sayHello()" onUnLoad="sayBye()">
	</body>
</html>
```

```
<!-- JS_Event_OnChange_29.html -->
<html>
	<head>
		<style>
			fieldset {
```

```
                width:50%;
                margin:0 auto;
            }
        </style>
        <script>
            function displayCountry() {
                alert(window.document.getElementById("country").value);
            }
            function displayGender() {
                alert(window.document.form1.gender.value);
            }
            function displayQual() {
                var quals=window.document.getElementsByName("qual");
                var len=quals.length;
                var qual="";
                for(var i=0;i<len;i++) {
                    if(quals[i].checked) {
                        qual+=quals[i].value+", ";
                    }
                }
                alert("Your Qualifications are "+qual);
            }
        </script>
    </head>
    <body>
        <form name="form1">
            <fieldset>
                <legend>Country:</legend>
                 <select id="country" onChange="displayCountry()">
                    <option value="India">India</option>
                    <option value="Srilanka">Srilanka</option>
                    <option value="Bhutan">Bhutan</option>
                    <option value="Nepal">Nepal</option>
                 </select>
            </fieldset>

            <fieldset>
                <legend>Gender:</legend>
```

```
                                Male: <INPUT type="radio" name="gender" value="male"
onChange="displayGender()">
                                Female: <INPUT type="radio" name="gender"
value="female" onChange="displayGender()">
                        </fieldset>

                        <fieldset>
                                <legend>Qualification:</legend>
                                SSC: <INPUT type="checkbox" name="qual" value="SSC"
onChange="displayQual()">
                                Inter: <INPUT type="checkbox" name="qual" value="Inter"
onChange="displayQual()">
                                Under Graduate/Degree: <INPUT type="checkbox"
name="qual" value="Degree" onChange="displayQual()">
                                Post Graduate: <INPUT type="checkbox" name="qual"
value="PG" onChange="displayQual()">
                        </fieldset>
                </form>
        </body>
</html>


<!-- JS_Event_OnSelect_30.html -->
<html>
        <head>
                <script>
                        function displaySelect() {
                                var sname=window.document.getElementById("sname");
                                var snameTxt=sname.value;
                                var selectedTxt=snameTxt.substring(sname.selectionStart,
sname.selectionEnd);
                                alert(selectedTxt);
                        }
                </script>
        </head>
        <body>
                Name <INPUT type="text" name="sname" id="sname" value=""
onSelect="displaySelect()">
```

```
	</body>
</html>


<!-- JS_Event_OnFocus_OnBlur_31.html -->
<html>
	<head>
		<script>
			function onBlr() {
				alert("onBlur event fired");
			}
			function onFcs() {
				alert("onFocs event fired");
			}
		</script>
	</head>
	<body>
		Name  <INPUT type="text" name="name" onBlur="onBlr()"><br/>
		Email <INPUT type="text" name="email"><br/>
		Mobile<INPUT type="text" name="mobile" onFocus="onFcs()"><br/>
	</body>
</html>


<!-- JS_Event_OnClick_OnDblClick_32.html -->
<html>
	<head>
		<script>
			function clickMe() {
				alert("onClick event fired");
			}
			function dblClickMe() {
				alert("Double click event fired");
			}
		</script>
	</head>
	<body>
		<INPUT type="button" name="button" value="Click" onClick="clickMe()"><br/>
```

```
		<INPUT type="button" name="button" value="ClikClick" onDblClick="dblClickMe()"><br/>
	</body>
</html>


<!-- JS_Event_OnSubmit_33.html -->
<html>
	<head>
		<script>
			function validateSuccess() {
				var err="";


	err+=(window.document.getElementById("sname").value.length==0)?"SName cannot be empty\n":"";

	err+=(window.document.getElementById("email").value.length==0)?"Email cannot be empty\n":"";

	err+=(window.document.getElementById("mobile").value.length==0)?"Mobile cannot be empty\n":"";

	err+=(window.document.getElementById("course").value.length==0)?"Course cannot be empty\n":"";

				if(err.length!=0) {
					alert(err);
					return false;
				} else {
					return true;
				}
			}
		</script>
	</head>
	<body>
		<form method="get" action="./login.php" onSubmit="return validateSuccess()">
			SName: <INPUT type="text" id="sname" name="sname" value=""><br/>
```

```
                    Email: <INPUT type="text" id="email" name="email" value=""><br/>
                    Mobile:<INPUT type="text" id="mobile" name="mobile" value=""><br/>
                    Course:<INPUT type="text" id="course" name="course" value=""><br/>
                    <INPUT type="submit" name="submit" value="Submit"><br/>
                    <!--
sname=sadhana&email=sadhana@gmail.com&mobile=9848012345&course=Full+Stack+PHP
                    -->
                    <!-- Whenever we submit a form the form content is converted into HTTP
encoding format (fieldname=value&fieldname=value) and will be submitted via HTTP protocol
request body (also called as payload) -->
                </form>
        </body>
</html>


<!-- JS_Event_OnKey_34.html -->
<html>
        <head>
                <script>
                        function chkMobile() {
                                var mobile=window.document.getElementById("mobile").value;
                                if(isNaN(mobile)) {
                                        alert("Mobile is not a number");
                                        window.document.getElementById("mobile").value="";
                                        window.document.getElementById("mobile").focus();
                                }
                        }
                </script>
        </head>
        <body>
                <form>
                        <!-- Mobile: <INPUT type="text" id="mobile" name="mobile" value=""
onKeyDown="chkMobile()"><br/> -->
                        <!-- Mobile: <INPUT type="text" id="mobile" name="mobile" value=""
onKeyPress="chkMobile()"><br/> -->
                        Mobile: <INPUT type="text" id="mobile" name="mobile" value=""
onKeyUp="chkMobile()"><br/>
```

```
                Email:  <INPUT type="text" name="email" value=""><br/>
            </form>
        </body>
</html>
```

JS_Event_OnMouseOverOut_35.html

```
<html>
    <head>
        <script>
            function mOver() {
                alert("mouse over on textfield");
            }
            function mOut() {
                alert("mouse out on textfield");
            }
            function oMMove() {
                alert("mouse moving on textfield");
            }
        </script>
    </head>
    <body>
        <form>
            Mobile: <INPUT type="text" id="mobile" name="mobile" value=""
onMouseOver="mOver()" onMouseOut="mOut()"><br/>

            Email: <INPUT type="text" name="email" value=""
onMouseMove="oMMove()"><br/>
        </form>
    </body>
</html>
```

JavaScript Objects, their properties and their methods

Browser handling:

i) Window
- screen
- location
- history
- navigator
- document (HTML DOM Manipulations)
  - cookie

Validations:
---
ii) String
iii) Date
iv) Math
v) Number
vi) Boolean
vii) Array
viii) RegExpr

AJAX & DHTML:
---
ix) ActiveXObject (load XML Parsers)
x) XMLHttpRequest (AJAX object)
json.js file is used to perform JSON parsing

Data Processing:

XML DOM Parser
JSON Parser
HTML DOM

Window Class:

window class properties & methods:
---
properties:
---

closed, console, defaultStatus, document, frameElement, frames, history, innerHeight, innerWidth, length, localStorage, location, name, navigator, opener, outerHeight, outerWidth pageXOffset, pageYOffset, parent, screen, screenLeft, screenTop, screenX, screenY, sessionStorage, scrollX, scrollY, self, status, top

methods:
---
open(), close()
alert(), confirm(), prompt()
setInterval(), setTimeout(), clearInterval(), clearTimeout(), stop()
moveBy(), moveTo()
resizeBy(), resizeTo(), scroll(), scrollBy(), scrollTo()
atob(), blur(), btoa()
focus(), getComputedStyle(), getSelection(), matchMedia(), print(), requestAnimationFrame()

Example in Window.open(), close(), closed:
---
window.open syntax:
---
var newWindow = Window.open("URL", "WindowName", "feature, feature, feature", replace);
features: width, height, innerWidth, innerHeight, outerWidth, outerHeight, toolbar, status, menubar, location, scrollbars, resizable, directories, copyHistory, left, top, alwasyLowered, alwaysRaised, z-lock

window.close syntax:
---
newWindow.close();

JS_Window_Open_36.html
```
<script>
	function myFunction() {
		window.open("https://www.activenetinformatics.com/index.html");
	}
</script>
<body>
	<button onClick="myFunction()">ActiveNET</button>
```

```
</body>
```

JS_Window_Open_37.html

```
<html>
	<head>
		<script>
		function myFunction() {
			var myWindow = window.open("", "", "width=200,height=100");
		}
		</script>
	</head>
	<body>
		<button onclick="myFunction()">Try it</button>
	</body>
</html>
```

JS_Window_Open_38.html

```
<html>
	<body>
		<p>Click the button to put the new window in the current window.</p>
		<button onclick="myFunction()">Try it</button>
		<script>
			function myFunction() {
			  var myWindow = window.open("", "_self");
			  myWindow.document.write("<p>I replaced the current window.</p>");
			}
		</script>
	</body>
</html>
```

JS_Window_Open_39.html

```
<html>
	<body>
		<p>Click the button to open a new window with some specifications.</p>
		<button onclick="myFunction()">Try it</button>
```

```
		<script>
			function myFunction() {
			window.open("http://www.activenetinformatics.com/index.html",
"_blank", "toolbar=yes,scrollbars=yes,resizable=yes, top=500,left=500,width=400,height=400");
			}
		</script>
	</body>
</html>
```

JS_Window_Open_Close_Closed_40.html

```
<html>
	<head>
		<script>
			var myWindow;

			function openWin() {
			  myWindow = window.open("", "myWindow", "width=400, height=200");
			}

			function closeWin() {
			  if (myWindow) {
				myWindow.close();
			  }
			}

			function checkWin() {
			  if (!myWindow) {
				document.getElementById("msg").innerHTML = "'myWindow' has
never been opened!";
			  } else {
				if (myWindow.closed) {
				  document.getElementById("msg").innerHTML = "'myWindow'
has been closed!";
				} else {
				  document.getElementById("msg").innerHTML = "'myWindow'
has not been closed!";
```

```
                        }
                    }
                }
            </script>
        </head>
        <body>

            <button onclick="openWin()">Open "myWindow"</button>
            <button onclick="closeWin()">Close "myWindow"</button>
            <br><br>
            <button onclick="checkWin()">Has "myWindow" been closed?</button>
            <br><br>
            <div id="msg"></div>

        </body>
</html>
```

JS_Window_Frames_41.html

```
<html>
        <body>

            <p>Click the button to change the location of the first iframe element (index 0).</p>

            <button onclick="myFunction()">Try it</button>
            <br><br>

            <iframe src="https://www.ActiveNET.com"></iframe>
            <iframe src="https://www.ActiveNET.com"></iframe>

            <script>
                function myFunction() {
                  window.frames[0].location = "http://www.activenetinformatics.com/Materials.html";
                }
            </script>
```

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

window.history

---

properties:

---

length

methods:

---

back(), forward(), go(number)

```
<!-- JS_Window_History_42.html -->
<html>
	<body>
		<button onclick="bck()">Back</button>
		<button onclick="fwd()">Forward</button>
		<INPUT type="text" id="pageno="" value="">
		<button onclick="go()">Go</button>

		<script>
			function bck() {
				window.history.back();
			}
			function fwd() {
				window.history.forward();
			}
			function go() {
				var
pageno=parseInt(window.document.getElementById('pageno').value);
				window.history.go(pageno);
			}
		</script>

	</body>
```

```
</html>
```

window.location

---

properties:

---

hash, host, hostname, href, origin, pathname, port, protocol, search

methods:

---

assign(), reload(), replace()

JS_Window_Location_43.html

```
<html>
	<body>
		<div id="protocol" value=""></div>
		<div id="hostname" value=""></div>
		<div id="port" value=""></div>
		<div id="host" value=""></div>
		<div id="pathname" value=""></div>
		<div id="origin" value=""></div>
		<div id="href" value=""></div>

		<INPUT type="text" id="URL" value="">
		<button onclick="go()">Go</button>

		<script>
			displayLocation();
			function go() {
				var url=window.document.getElementById("URL").value;
				window.location=url;
				window.status=window.location;
				displayLocation();
			}
			function displayLocation() {
			window.document.getElementById("protocol").innerText="protocol:
"+window.location.protocol;
```

```
                window.document.getElementById("hostname").innerText="hostname:
"+window.location.hostname;
                window.document.getElementById("port").innerText="port:
"+window.location.port;
                window.document.getElementById("host").innerText="host:
"+window.location.host;
                window.document.getElementById("pathname").innerText="pathname:
"+window.location.pathname;
                window.document.getElementById("origin").innerText="origin:
"+window.location.origin;
                window.document.getElementById("href").innerText="href:
"+window.location.href;
                }

        </script>
    </body>

</html>
```

JS_Window_ScreenXY_44.html

```
<html>
    <head>
        <script>
            function myFunction() {
                var myWindow = window.open("", "myWin", "left=700, top=350,
width=200, height=100");
                myWindow.document.write("<p>This is 'myWin'");
                myWindow.document.write("<br>ScreenX: " +
myWindow.screenX);
                myWindow.document.write("<br>ScreenY: " +
myWindow.screenY + "</p>");
            }
        </script>
    </head>
    <body>
        <button onclick="myFunction()">Open "myWin"</button>
```

```
	</body>
</html>

<!-- JS_Window_ScreenXY_45.html -->
<html>
	<head>
		<script>
			function displayCoords() {
				window.document.write("ScreenX: " + window.screenX);
				window.document.write("<br>ScreenY: " + window.screenY + 
"</p>");
			}
		</script>
	</head>
	<body >
		<div onMouseUp="displayCoords()" 
style="position:relative;left:150px;top:150px;background-
color:red;width:200px;height:200px"></div>

	</body>
</html>
```

JS_Window_PageOffset_46.html

```
<html>
	<head>
		<style>
			body {
			  margin: 0;
			  font-size: 28px;
			}

			.header {
			  background-color: #f1f1f1;
			  padding: 30px;
			  text-align: center;
			}
```

```
#navbar {
  overflow: hidden;
  background-color: #333;
}

#navbar a {
  float: left;
  display: block;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 17px;
}

#navbar a:hover {
  background-color: #ddd;
  color: black;
}

#navbar a.active {
  background-color: #4CAF50;
  color: white;
}

.content {
  padding: 16px;
}

.sticky {
  position: fixed;
  top: 0;
  width: 100%
}

.sticky + .content {
  padding-top: 60px;
```

```
            }
        </style>
    </head>
    <body onscroll="myFunction()">

        <div class="header">
          <h2>Scroll Down</h2>
          <p>Scroll down to see the sticky effect.</p>
        </div>

        <div id="navbar">
          <a class="active" href="javascript:void(0)">Home</a>
          <a href="javascript:void(0)">News</a>
          <a href="javascript:void(0)">Contact</a>
        </div>

        <div class="content">
          <h3>Sticky Navigation Example</h3>
          <p>The navbar will stick to the top when you reach its scroll position.</p>
          <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
          <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
          <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
          <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>
          <p>Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae
```

gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>

```
		</div>

		<script>
			var navbar = document.getElementById("navbar");
			var sticky = navbar.offsetTop;

			function myFunction() {
				if (window.pageYOffset >= sticky) {
					navbar.classList.add("sticky")
				} else {
					navbar.classList.remove("sticky");
				}
			}
		</script>

		</body>
</html>
```

JS_Window_alert_47.html

```
<html>
	<body>

		<p>Click the button to display an alert box.</p>
		<button onclick="myFunction()">Try it</button>
		<script>
			function myFunction() {
			  alert("Hello! I am an alert box!");
			}
		</script>

	</body>
</html>
```

JS_Window_confirm_48.html

```
<html>
	<body>

	<p>Click the button to display a confirm box.</p>

	<button onclick="myFunction()">Try it</button>

	<script>
		function myFunction() {
		  confirm("Press a button!");
		}
	</script>

	</body>
</html>
```

JS_Window_prompt_49.html

```
<html>
	<body>

	<p>Click the button to demonstrate the prompt box.</p>

	<button onclick="myFunction()">Try it</button>

	<p id="demo"></p>

	<script>
		function myFunction() {
			var person = prompt("Please enter your name", "Harry Potter");
			if (person != null) {
				document.getElementById("demo").innerHTML =
				"Hello " + person + "! How are you today?";
			}
```

```
        }
    </script>

    </body>
</html>
```

window timeout

The setTimeout() method calls a function or evaluates an expression after a specified number of milliseconds.

Tip: 1000 ms = 1 second.

Tip: The function is only executed once. If you need to repeat execution, use the setInterval() method.

Tip: Use the clearTimeout() method to prevent the function from running.

JS_Window_timeout_50.html

```
<html>
    <body>

    <p>Click the button to wait 3 seconds, then alert "Hello".</p>

    <button onclick="myFunction()">Try it</button>

    <script>
        function myFunction() {
          // setTimeout(function(){ alert("Hello"); }, 3000);
          // setInterval(function(){ alert("Hello"); }, 3000);
        }
    </script>

    </body>
</html>
```

JS_Window_moveTo_51.html

```
<html>
	<body>

	<p>Open "myWindow" and move the new window to the top left corner of the screen:</p>

	<button onclick="openWin()">Open "myWindow"</button>
	<button onclick="moveWin()">Move "myWindow"</button>

	<script>
		var myWindow;

		function openWin() {
			myWindow=window.open("", "myWindow", "width=200, height=100");
			myWindow.document.write("<p>This is 'myWindow'</p>");
		}

		function moveWin() {
			myWindow.moveTo(500, 100);
			myWindow.focus();
		}
	</script>

	</body>
</html>
```

JS_Window_moveBy_52.html

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

```
	<p>Open "myWindow" and move the new window 250px relative to its current position:</p>

	<button onclick="openWin()">Open "myWindow"</button>
	<button onclick="moveWin()">Move "myWindow"</button>

	<script>
		var myWindow;

		function openWin() {
			myWindow = window.open("", "myWindow", "width=200, height=100");
			myWindow.document.write("<p>This is 'myWindow'</p>");
		}

		function moveWin() {
			myWindow.moveBy(250, 250);
			myWindow.focus();
		}
	</script>

	</body>
</html>
```

JS_Window_atob_btoa_53.html

```
<html>
	<body>

	<p>Click the button to decode a base-64 encoded string.</p>

	<button onclick="myFunction()">Try it</button>

	<p><strong>Note:</strong> The atob() method is not supported in IE9 and earlier.</p>

	<p id="demo"></p>

	<script>
```

```
		function myFunction() {
			var str = "Hello World!";
			var enc = window.btoa(str);
			var dec = window.atob(enc);

			var res = "Encoded String: " + enc + "<br>" + "Decoded String: " + dec;
			document.getElementById("demo").innerHTML = res;
		}
	</script>

	</body>
</html>
```

JS_Window_resizeTo_54.html

```
<html>
	<body>

	<p>Open a new window, and resize the width and height to 500px:</p>

	<button onclick="openWin()">Create window</button>
	<button onclick="resizeWin()">Resize window</button>

	<script>
		var myWindow;

		function openWin() {
		  myWindow = window.open("", "", "width=100, height=100");
		}

		function resizeWin() {
		  myWindow.resizeTo(250, 250);
		  myWindow.focus();
		}
	</script>
```

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

JS_Window_resizeBy_55.html

```
<html>
	<body>

	<p>Open a new window, and resize the width and height to 250px:</p>
	<p><b>Tip:</b> Press the "Resize window" multiple times (the window will increase 250px for each press).</p>

	<button onclick="openWin()">Create window</button>
	<button onclick="resizeWin()">Resize window</button>

	<script>
		var myWindow;

		function openWin() {
		  myWindow = window.open("", "", "width=100, height=100");
		}

		function resizeWin() {
		  myWindow.resizeBy(250, 250);
		  myWindow.focus();
		}
	</script>

	</body>
</html>
```

JS_Window_scrollBy_56.html

```
<html>
```

```
	<head>
	<style>
		body {
		  width: 5000px;
		}

		button {
		  position: fixed;
		}
	</style>
	</head>
	<body>

	<p>Click the button to scroll the document window by 100px horizontally.</p>

	<p>Look at the horizontal scrollbar to see the effect.</p>

	<button onclick="scrollWin()">Click me to scroll horizontally!</button><br><br>

	<script>
		function scrollWin() {
		  window.scrollBy(100, 0);
		}
	</script>

	</body>
</html>
```

JS_Window_scrollTo_57.html

```
<html>
	<head>
	<style>
		body {
		  width: 5000px;
		}
```

```
        </style>
        </head>
        <body>

        <p>Click the button to scroll the document window to 500 pixels horizontally.</p>

        <button onclick="scrollWin()">Click me to scroll horizontally!</button><br><br>

        <script>
                function scrollWin() {
                  window.scrollTo(500, 0);
                }
        </script>

        </body>
</html>
```

JS_Window_navigator_58.html

```
<html>
        <body>

        <h2>The Navigator Object</h2>
        <p id="demo"></p>
        <script>
                window.document.getElementById("demo").innerHTML="Hello";

                var msg="";

                msg+="navigator.cookieEnabled is " + window.navigator.cookieEnabled;
                msg+="<br/>navigator.appName is " + window.navigator.appName;
                msg+="<br/>navigator.appCodeName is " + window.navigator.appCodeName;
                msg+="<br/>navigator.product is " + window.navigator.product;
                msg+="<br/>navigator.appVersion is " + window.navigator.appVersion;
                msg+="<br/>navigator.userAgent is " + window.navigator.userAgent;
```

```
            msg+="<br/>navigator.platform is " + window.navigator.platform;
            msg+="<br/>navigator.onLine is " + window.navigator.onLine;
            msg+="<br/>javaEnabled is " + window.navigator.javaEnabled();

            window.document.getElementById("demo").innerHTML=msg;
        </script>

        </body>
</html>
```

String class

String is a series of characters. It converts any double quote placed value into String object.

var str=new String(string);

properties:
---
constructor, length, prototype

methods:
---
charAt(), charCodeAt(), concat(), indexOf(), lastIndexOf(), localeCompare(), match(), replace(), search(), slice(), split(), substr(), substring(), toLocalLowerCase(), toLocaleUppercase(), toLowerCase(), toUpperCase(), toString(), valueOf()

JS_String_59.html

```
<html>
        <body>
            <p id="out"></p>
        </body>
        <script>
            var str=new String("Activenet Informatics Pvt Ltd");

            var output="";

            output+="charAt() "+str.charAt(4);
```

```
        output+="<br/>charCodeAt() "+str.charCodeAt(4);
        output+="<br/>concat() "+str.concat(" Ltd");
        output+="<br/>indexOf() "+str.indexOf('n');
        output+="<br/>lastIndexOf() "+str.lastIndexOf('n');
        output+="<br/>replace() "+str.replace(/net/gi, "NET");

        output+="<br/>search() "+str.search("Info"); // The search() method searches a string for a specified value, and returns the position of the match.

        output+="<br/>slice() "+str.slice(6, 9); // The slice() method extracts parts of a string and returns the extracted parts in a new string.

        output+="<br/>split() "+str.split(" ", 3); // The split() method is used to split a string into an array of substrings, and returns the new array.

        output+="<br/>substr() "+str.substr(0, 3); // The substr() method extracts parts of a string, beginning at the character at the specified position, and returns the specified number of characters.

        output+="<br/>substring() "+str.substring(0, 4); // The substring() method extracts the characters from a string, between two specified indices, and returns the new sub string.

        output+="<br/>lowerCase() "+str.toLowerCase();
        output+="<br/>upperCase() "+str.toUpperCase();

        window.document.getElementById("out").innerHTML=output;
    </script>
</html>
```

Difference between substr() and substring():

substr(startIndex/offSet, length) second argument is length of the substring

substring(startIndex/offSet, endIndexExclusive) second argument is end index-1

Date class

The Date object is a datatype built into the JavaScript language. Date objects are created with the new Date( ) as shown below.

Once a Date object is created, a number of methods allow you to operate on it. Most methods simply allow you to get and set the year, month, day, hour, minute, second, and millisecond fields of the object, using either local time or UTC (universal, or GMT) time.

constructor:
---
new Date()
new Date(milliseconds)
new Date(dateString)
new Date(year, month, date[, hour, minute, second, millisecond])

methods:
---
getDate()
getDay()
getFullYear()
getHours()
getMilliSeconds()
getMinutes()
getMonth()
getSeconds()
getTime()
setDate()
setFullYear()
setHours()
setMilliSeconds()
setMinutes()
setMonth()
setSeconds()
setTime()
toString()

```
<!-- JS_Date_60.html -->
<html>
	<body>
		<p id="out"></p>
```

```
	</body>
	<head>
		<script>
			var d=new Date();

			var output="";
			output+="getDate() "+d.getDate();
			output+="<br/>getDay() "+d.getDay();
			output+="<br/>getMonth() "+d.getMonth();
			output+="<br/>getFullYear() "+d.getFullYear();

			output+="<br/>getHours() "+d.getHours();
			output+="<br/>getMinutes() "+d.getMinutes();
			output+="<br/>getSeconds() "+d.getSeconds();
			output+="<br/>getMilliSeconds() "+d.getMilliseconds();

			window.document.getElementById("out").innerHTML=output;
		</script>
	</head>
</html>
```

Math class

The math object provides you properties and methods for mathematical constants and functions. Unlike other global objects, Math is not a constructor. All the properties and methods of Math are static and can be called by using Math as an object without creating it.

Thus, you refer to the constant pi as Math.PI and you call the sine function as Math.sin(x), where x is the method's argument.

properties:

Math.E, Math.LN2, Math.LN10, Math.LOG2E, Math.LOG10E, Math.PI, Math.SQRT1_2, Math.SQRT2

methods:

---

abs(), acos(), asin(), atan(), atan2(), ceil(), cos(), exp(), floor(), log(), max(), min(), pow(), random(), round(), sin(), sqrt(), tan(), toSource()

JS_Math_61.html

```
<html>
	<body>
	</body>
	<head>
		<script>
			window.document.write(Math.E);
			window.document.write("<br/>"+Math.LN2);
			window.document.write("<br/>"+Math.LN10);
			window.document.write("<br/>"+Math.LOG2E);
			window.document.write("<br/>"+Math.LOG10E);
			window.document.write("<br/>"+Math.PI);
			window.document.write("<br/>"+Math.SQRT1_2);
			window.document.write("<br/>"+Math.SQRT2);

			window.document.write("<hr/>");

			window.document.write("<br/> Math.abs(-1) "+Math.abs(-1));
			window.document.write("<br/> Math.abs(null) "+Math.abs(null));
			window.document.write("<br/> Math.abs(20) "+Math.abs(20));
			window.document.write("<hr/>");

			window.document.write("<br/> Math.acos(-1) "+Math.acos(-1));
			window.document.write("<br/> Math.acos(null) "+Math.acos(null));
			window.document.write("<br/> Math.acos(30) "+Math.acos(30));
			window.document.write("<hr/>");

			window.document.write("<br/> Math.asin(-1) "+Math.asin(-1));
			window.document.write("<br/> Math.asin(null) "+Math.asin(null));
			window.document.write("<br/> Math.asin(30) "+Math.asin(30));
			window.document.write("<hr/>");
```

```
window.document.write("<br/> Math.atan(-1) "+Math.atan(-1));
window.document.write("<br/> Math.atan(null) "+Math.atan(null));
window.document.write("<br/> Math.atan(30) "+Math.atan(30));
window.document.write("<hr/>");

window.document.write("<br/> Math.ceil(45.95) "+Math.ceil(45.95));
window.document.write("<br/> Math.ceil(45.20) "+Math.ceil(45.20));
window.document.write("<br/> Math.ceil(-45.95) "+Math.ceil(-45.95));
window.document.write("<br/> Math.ceil(-45.20) "+Math.ceil(-45.20));
window.document.write("<hr/>");

window.document.write("<br/> Math.cos(90) "+Math.cos(90));
window.document.write("<br/> Math.cos(30) "+Math.cos(30));
window.document.write("<br/> Math.cos(-1) "+Math.cos(-1));
window.document.write("<br/> Math.cos(2*Math.PI) "+Math.cos(2*Math.PI));
window.document.write("<hr/>");

window.document.write("<br/> Math.exp(1) "+Math.exp(1));
window.document.write("<br/> Math.exp(30) "+Math.exp(30));
window.document.write("<br/> Math.exp(-1) "+Math.exp(-1));
window.document.write("<br/> Math.exp(0.5) "+Math.exp(0.5));
window.document.write("<hr/>");

window.document.write("<br/> Math.floor(10.3) "+Math.floor(10.3));
window.document.write("<br/> Math.floor(30.9) "+Math.floor(30.9));
window.document.write("<br/> Math.floor(-2.9) "+Math.floor(-2.9));
window.document.write("<br/> Math.floor(-2.2) "+Math.floor(-2.2));
window.document.write("<hr/>");

window.document.write("<br/> Math.log(10) "+Math.log(10));
window.document.write("<br/> Math.log(0) "+Math.log(0));
window.document.write("<br/> Math.log(-1) "+Math.log(-1));
window.document.write("<br/> Math.log(100) "+Math.log(100));
window.document.write("<hr/>");

window.document.write("<br/> Math.max(10, 20, -1, 100) "+Math.max(10, 20, -1, 100));
```

```
                window.document.write("<br/> Math.max(-1, -3, -40) "+Math.max(-1, -3,
-40));
                window.document.write("<br/> Math.max(0, -1) "+Math.max(0, -1));
                window.document.write("<br/> Math.max(100) "+Math.max(100));
                window.document.write("<hr/>");

                window.document.write("<br/>      Math.min(10,      20,      -1,      100)
"+Math.min(10, 20, -1, 100));
                window.document.write("<br/> Math.min(-1, -3, -40) "+Math.min(-1, -3,
-40));
                window.document.write("<br/> Math.min(0, -1) "+Math.min(0, -1));
                window.document.write("<br/> Math.min(100) "+Math.min(100));
                window.document.write("<hr/>");

                window.document.write("<br/> Math.pow(7, 2) "+Math.pow(7, 2));
                window.document.write("<br/> Math.pow(8, 8) "+Math.pow(8, 8));
                window.document.write("<br/> Math.pow(-1, 2) "+Math.pow(-1, 2));
                window.document.write("<br/> Math.pow(0, 10) "+Math.pow(0, 10));
                window.document.write("<hr/>");

                window.document.write("<br/> Math.random() "+Math.random());
                window.document.write("<br/> Math.random() "+Math.random());
                window.document.write("<br/> Math.random() "+Math.random());
                window.document.write("<br/> Math.random() "+Math.random());
                window.document.write("<hr/>");

                window.document.write("<br/> Math.round(0.5) "+Math.round(0.5));
                window.document.write("<br/> Math.round(20.7) "+Math.round(20.7));
                window.document.write("<br/> Math.round(20.3) "+Math.round(20.3));
                window.document.write("<br/>      Math.round(-20.3)      "+Math.round(-
20.3));
                window.document.write("<hr/>");

                window.document.write("<br/> Math.sin(0.5) "+Math.sin(0.5));
                window.document.write("<br/> Math.sin(90) "+Math.round(90));
                window.document.write("<br/> Math.sin(1) "+Math.sin(1));
                window.document.write("<br/>                    Math.sin(Math.PI/2)
"+Math.sin(Math.PI/2));
```

```
                    window.document.write("<hr/>");

                    window.document.write("<br/> Math.sqrt(0.5) "+Math.sqrt(0.5));
                    window.document.write("<br/> Math.sqrt(81) "+Math.sqrt(81));
                    window.document.write("<br/> Math.sqrt(13) "+Math.sqrt(13));
                    window.document.write("<br/> Math.sqrt(-4) "+Math.sqrt(-4));
                    window.document.write("<hr/>");

                    window.document.write("<br/> Math.tan(-30) "+Math.tan(-30));
                    window.document.write("<br/> Math.tan(90) "+Math.tan(90));
                    window.document.write("<br/> Math.tan(45) "+Math.tan(45));
                    window.document.write("<br/>                    Math.tan(Math.PI/180)
"+Math.tan(Math.PI/180));
                    window.document.write("<hr/>");
                </script>
        </head>
</html>
```

Number class

The Number object represents numerical values integers or floating-point numbers. In general, you do not need to worry about Number objects because the browser automatically converts number literals to instances of the number class.

Syntax:
var val=new Number(number);

properties:

MAX_VALUE, MIN_VALUE, NaN, NEGATIVE_INFINITY, POSITIVE_INFINITY, prototype, constructor

methods:
---
toExponential(), toFixed(), toPrecision(), toString()

JS_Number_62.html

```
<html>
	<body>
	</body>
	<head>
		<script>
			window.document.write("Number.MAX_VALUE "+Number.MAX_VALUE);
			window.document.write("<br/>Number.MIN_VALUE
"+Number.MIN_VALUE);
			window.document.write("<br/>Number.NaN "+Number.NaN);
			window.document.write("<br/>NEGATIVE_INFINITY
"+Number.NEGATIVE_INFINITY);
			window.document.write("<br/>POSITIVE_INFINITY
"+Number.POSITIVE_INFINITY);
			window.document.write("<hr/>");

			var num=77.1234;
			window.document.write("<br/>                    num.toExponential()
"+num.toExponential());
			window.document.write("<br/>                    num.toExponential(4)
"+num.toExponential(4));
			window.document.write("<br/>                    num.toExponential(2)
"+num.toExponential(2));
			window.document.write("<hr/>");

			window.document.write("<br/> num.toFixed() "+num.toFixed());
			window.document.write("<br/> num.toFixed(6) "+num.toFixed(6));
			window.document.write("<br/> num.toFixed(1) "+num.toFixed(1));
			window.document.write("<br/>                    1.23e+20.toFixed(2)
"+1.23e+20.toFixed(2));
			window.document.write("<br/>      1.23e-10.toFixed(2)      "+1.23e-
10.toFixed(2));
			window.document.write("<hr/>");

			num=new Number(7.123456);
			window.document.write("<br/> num.toPrecision() "+num.toPrecision());
```

```
                    window.document.write("<br/>                    num.toPrecision(4)
"+num.toPrecision(4));
                    window.document.write("<br/>                    num.toPrecision(2)
"+num.toPrecision(2));
                    window.document.write("<br/>                    num.toPrecision(1)
"+num.toPrecision(1));
                    window.document.write("<hr/>");
                </script>
        </head>
</html>
```

Boolean class

var val=new Boolean(value);

properties:
---
constructor
prototype

methods:
---
toSource()
toString()
valueOf()

Array class

The Array object lets you store multiple values in a single variable. It stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Syntax:
---
creating array with size
var names=new Array(3);

```
names[0]="Sadhana";
names[1]='Revanth';
names[2]='Surya';
names[3]="Nivas";
window.document.writeln(names.length);
```

creating array with values initialized in the constructor

```
var fruits = new Array( "apple", "orange", "mango" );
```

creating array without using array object/array class constructor
note: anything written in sqaure brackets [ ] are by default assumed as array

```
var fruits = [ "apple", "orange", "mango" ];
```

creating array of JSON objects:

```
var attendees=
[
        {'id':1,'sname':'Manohar','course':'AJS'},
        { 'id':2,'sname':'Aasish','course':'JS'},
        {'id':3,'sname':'Vishwanath','course':'CJ'}
]
```

properties:

---

constructor
index
input
length
prototype

methods:

---

concat()
every()
filter()
forEach()
indexOf()
join()

lastIndexOf()
map()
pop()
push()
reduce()
reverse()
shift()
slice()
some()
sort()
splice()
unshift()

JS_Array_63.html

```
<html>
	<body>
	</body>
	<head>
		<script>
			var alpha = ["a", "b", "c"];
			var numeric = [1, 2, 3];

			var alphaNumeric = alpha.concat(numeric);

			document.write("alphaNumeric : " + alphaNumeric );
			document.write("<hr/>");

			var arr = new Array("First","Second","Third");

			var str = arr.join();
			document.write("str : " + str );

			var str = arr.join(", ");
			document.write("<br />str : " + str );

			var str = arr.join(" + ");
			document.write("<br />str : " + str );
```

```
document.write("<hr/>");

var index = [12, 5, 8, 130, 44].indexOf(8);
document.write("index is : " + index );

var index = [12, 5, 8, 130, 44].indexOf(13);
document.write("<br />index is : " + index );
document.write("<hr/>");

var index = [12, 5, 8, 130, 44].lastIndexOf(8);
document.write("index is : " + index );

var index = [12, 5, 8, 130, 44, 5].lastIndexOf(5);
document.write("<br />index is : " + index );
document.write("<hr/>");

var numbers = [1, 4, 9];
var roots = numbers.map(Math.sqrt);
document.write("roots is : " + roots );
document.write("<hr/>");

var numbers = [1, 4, 9];

var element = numbers.pop();
document.write("element is : " + element );

var element = numbers.pop();
document.write("<br />element is : " + element );
document.write("<hr/>");

var numbers = new Array(1, 4, 9);

var length = numbers.push(10);
document.write("new numbers is : " + numbers );

length = numbers.push(20);
document.write("<br />new numbers is : " + numbers );
document.write("<hr/>");
```

```
var total = [0, 1, 2, 3].reduce(function(a, b){ return a + b; });
document.write("total is : " + total );
document.write("<hr/>");

var arr = [0, 1, 2, 3].reverse();
document.write("Reversed array is : " + arr );
document.write("<hr/>");

var element = [105, 1, 2, 3].shift();
document.write("Removed element is : " + element );
document.write("<hr/>");

var arr = ["orange", "mango", "banana", "sugar", "tea"];
document.write("arr.slice( 1, 2) : " + arr.slice( 1, 2) );
document.write("<br />arr.slice( 1, 3) : " + arr.slice( 1, 3) );
document.write("<hr/>");

var arr = new Array("orange", "mango", "banana", "sugar");
var sorted = arr.sort();
document.write("Returned string is : " + sorted );
document.write("<hr/>");

var arr = ["orange", "mango", "banana", "sugar", "tea"];
var removed = arr.splice(2, 0, "water");
document.write("After adding 1: " + arr );
document.write("<hr />removed is: " + removed);

removed = arr.splice(3, 1);
document.write("<br />After adding 1: " + arr );
document.write("<br />removed is: " + removed);
document.write("<hr/>");

var arr = new Array("orange", "mango", "banana", "sugar");
var length = arr.unshift("water");
document.write("Returned array is : " + arr );
document.write("<br /> Length of the array is : " + length );
document.write("<hr/>");
```

```
        </script>
    </head>
</html>
```

RegExp class

Regular Expression is a powerful way of doing search and replace in strings.

A Regular Expression is an object that describes a pattern of characters.

The JavaScript RegExp class represents regular expressions, and both String and RegExp define methods that use regular expressions to perform powerful pattern-matching and search-and-replace functions on text.

Syntax:

```
var pattern = new RegExp(pattern, attributes);
                    or simply
var pattern = /pattern/attributes;
```

attributes: g- global match on all instances rather than just the first

i- Perform matching that is not case-sensitive

m- Perform multi-line matcher rather than stopping on the first line LF/CR character (Line Feed \f + Carraige Return \r => \n)

Example:

```
    var myStr = "Teach Yourself jQuery & JavaScript in 24 Lessons";
    var re = /yourself/i;
    var newStr = myStr.replace(re, "Your Friends");
```

Result: Teach Your Friends jQuery & JavaScript in 24 Lessons

Example:

```
<!-- JS_RegExp_64.html -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
	<body>
		<script language="JavaScript" type="text/JavaScript">
			var myString = "Paul, Paula, Pauline, paul, Paul";

			// var myRegExp = /Paul/;
			// var myRegExp = /Paul/g;
			var myRegExp = /Paul/gi;

			myString = myString.replace(myRegExp, "Ringo");
			alert(myString);
		</script>
	</body>
</html>
```

Output: Ringo, Ringoa, Ringoine, paul, Ringo

Sample Regular Expressions:

/[^a-z\d]/i	any character that is not a letter, a digit, or a space
/[a-z\d]/i	any character that is a letter, a digit, or a space

Brackets:

[abcABC] Any one of the character between abcABC
[abcABC] excatly only one
[abcABC]? 0 or 1
[abcABC]* 0 or more
[abcABC]+ 1 or more

[a-zA-Z]+	Any character between abcABC
[^abcABC]	Any character other than between abcABC
[0-9]	Any one the digit between 0-9
[a-z]	Any one of the character between abc
[A-Z]	Any one of the character between ABC
[a-Z]	Any one of the character between aZ (small letter a-z to A-Z)

Repetition of Characters/Quantifiers:

---

p+ Given pattern for One or many occurances
p* Given pattern for None or many occurance
p? Given pattern for None or one occurance
p{N} Given pattern for n number of occurances
p{2,3} Given pattern for minimum 2 to maximum 3 number of occurances
p{2,} Given pattern for minimum 2 to maximum any number of occurances
p$ The pattern 'p' must be at the end of the String or if it is multi line String, then at the end of the line
^p The pattern 'p' must be at the start of the String

Metacharacters:

. Any single character other than the new line character (\n). Ex: a or @ or 4
\s Any white space character (tab, carraige return, form feed and vertical tab) \t \r \f \v
\S Any non-white space character
\d Any digit from 0 to 9
\D Any character that is not a digit
\w Any word character (A-Z,a-z,0-9 and _)
\W Any non-word character
[\b] Used to find a match at the begining or end of a word.

```
var str= "HELLO LOOK AT YOU";
var patt=/\bLO/
```

[aeiou] Any character within 'aeiou'
[^aeiou] Any character other than 'aeiou'
(foo|bar|baz) Any word foo or bar or baz

JS RegExp class properties:

---

constructor
global
ignoreCase
lastIndex
multiline
source

Ex constructor:

```
var re = new RegExp( "string" );
document.write("re.constructor is:" + re.constructor);
```

Ex global:

---

global is a read-only boolean property.

```
var re = new RegExp( "string" );

if ( re.global ) {
        document.write("Test1 - Global property is set");
} else {
        document.write("Test1 - Global property is not set");
}

re = new RegExp( "string", "g" );

if ( re.global ) {
        document.write("<br />Test2 - Global property is set");
} else {
        document.write("<br />Test2 - Global property is not set");
}
```

Ex ignoreCase:

---

ignoreCase is a read-only boolean property.

```
var re = new RegExp( "string" );

if ( re.ignoreCase ) {
        document.write("Test1-ignoreCase property is set");
} else {
        document.write("Test1-ignoreCase property is not set");
}
```

```
	re = new RegExp( "string", "i" );

	if ( re.ignoreCase ) {
		document.write("<br/>Test2-ignoreCase property is set");
	} else {
		document.write("<br/>Test2-ignoreCase property is not set");
	}
```

Ex lastIndex:

```
	var str = "Javascript is an interesting scripting language";

	var re = new RegExp( "script", "g" );

   re.test(str);
   document.write("Test 1 - Current Index: " +  re.lastIndex);

   re.test(str);
   document.write("<br />Test 2 - Current Index: " + re.lastIndex);
```

JS RegExp class methods:

---

exec()
test()
toSource()
toString()

Ex exec():

[] exec()
The exec method searches string for text that matches regexp. If it finds a match, it returns an array of results; otherwise, it returns null.

```
	var str = "Javascript is an interesting scripting language";
   var re = new RegExp( "script", "g" );

   var result = re.exec(str);
```

```
document.write("Test 1 - returned value : " +  result);

re = new RegExp( "pushing", "g" );

var result = re.exec(str);
document.write("<br />Test 2 - returned value : " +  result);
```

Ex test():

---

boolean test()

The test method searches string for text that matches regexp. If it finds a match, it returns true; otherwise, it returns false.

```
        var str = "Javascript is an interesting scripting language";
var re = new RegExp( "script", "g" );

var result = re.test(str);
document.write("Test 1 - returned value : " +  result);

re = new RegExp( "pushing", "g" );

var result = re.test(str);
document.write("<br />Test 2 - returned value : " +  result);
```

Email Address RegExpr:

"^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"

URL RegExpr:

/[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[a-zA-Z0-9()]{1,6}\b([-a-zA-Z0-9()@:%_\+.~#?&//=]*)?/gi;

XMLHttpRequest (AJAX)

AJAX Asynchronous JavaScript and XML a term that was coined by Jesse James Garrett at Adaptive Path in Feb 2005. It describes a methodology of developing web applications that is different from traditional one.

Traditional web apps work synchronously. - every time you click on a link or submit a form, the browser sends request to server. Server responds and the whole page gets loaded.

AJAX applications work aysnchronously, which means that you send data back and forth between browser (userAgent) and the server without reloading the whole page. You replace only the parts of the page that needs to be changed. When request is under process with server we can parallelly access other parts of the HTML document without server response waiting.

Imp Note: All JavaScript applications can work with out any server side coding, but AJAX cannot work without server side code. A Servlet/JSP/ASP/PHP/Python/Node.js application is required to run on server.

We can create XMLHttpRequest (XHR) object:

XMLHttpRequest object in non-IE browsers.
new ActiveXObject("Microsoft.XMLHTTP"); (or) new ActiveXObject("Msxml2.XMLHTTP"); in IE browsers.

```
<script>
	var xhr;
	function createXHR() {
		if(ActiveXObject) {
			xhr=new ActiveXObject("Microsoft.XMLHTTP");
			if(xhr==null) {
				xhr=new ActiveXObject("Msxml2.XMLHTTP");
			}
			log("Your's is IE browser");
		} else if(XMLHttpRequest) {
			xhr=new XMLHttpRequest();
			log("Your's is non-IE browser");
		} else {
			log("Your browser doesn't support XHR");
		}
	}
</script>
```

XHR object is having following properties and methods:

---

properties:

---

i) onreadystatechange: Register a function to XHR, called automatically every time when readyState property changes

ii) readyState: Holds the state of XHR. The value changes from 0 to 4

0: request not initialized. As soon as XHR object is created
1: server connection established. After calling xhr.open() function
2: request sent. After calling xhr.send() function
3: Processing request/Receiving response
4: Response received

iii) status: Returns HTTP response status 200 OK, 404 Not Found, 500 Internal Server Error

iv) statusText: "Not Found", "OK" etc

v) responseText: Returns response in a form of String

vi) responseXML: Returns response in a form of XML

methods:

---

i) open(method, url_with_queryString_in_case_of_GET, asynch, uname, pwd)
ii) abort()
iii) setRequestHeader()
iv) send(payload_in_case_ofPOST)
v) getAllResponseHeaders()
vi) getResponseHeader()

Example on AJAX:

Write a AJAX application on fetching user mobile bill details from server.
i) User enters mobile number in the text field and press tab
ii) The AJAX request must be submitted to server
iii) Server must fetch user mobile bill details
iv) the bill details including Subscriber Name, Bill Amount, Last Date, Type of Connection (Postpaid/Pre-paid)

v) and send that information back to client in a form of XML or JSON
vi) Client should disyplay that information in a seperate textfield

Window String Date Math Number Array Boolean
RegExp XMLHttpRequest ActiveXObject

JS_AJAX_65.html

```
<!-- JS_AJAX_65.html -->
<!DOCTYPE html>
<html>
	<head>
		<script>

			var xhr;

			function createXHR() {
				xhr=new XMLHttpRequest();
				// console.log("Your's is non-IE browser");

				/*
				if(ActiveXObject) {
					xhr=new ActiveXObject("Microsoft.XMLHTTP");
					if(xhr==null) {
						xhr=new ActiveXObject("Msxml2.XMLHTTP");
					}
					log("Your's is IE browser");
				} else if(XMLHttpRequest) {
					xhr=new XMLHttpRequest();
					console.log("Your's is non-IE browser");
				} else {
					console.log("Your browser doesn't support XHR");
				}
				*/
			}

			function fetchBillDetails() {
				window.alert("Hello");
				var mobile=window.document.getElementById("mobile").value;
```

```
                window.alert(mobile);
                createXHR();
                window.alert("xhr created");
                xhr.onreadystatechange=callback;
                xhr.open("GET", "./FetchBill.jsp?mobile="+mobile, false);
                window.alert("xhr opened");
                xhr.send();

                window.alert("xhr sent");
        }

        function callback() {

                displayState(xhr.readyState);

                if(xhr.readyState==4 && xhr.status==200) {

                        var xml=xhr.responseXML;
                        window.alert(xml);
                        var
nameTxt=xml.getElementsByTagName("name")[0].textContent;
                        var
billAmtTxt=xml.getElementsByTagName("billAmt")[0].textContent;
                        var
lastDateTxt=xml.getElementsByTagName("lastDate")[0].textContent;
                        var
typeTxt=xml.getElementsByTagName("type")[0].textContent;

                        alert(nameTxt+"  "+billAmtTxt+"  "+lastDateTxt+"
"+typeTxt);


        window.document.getElementById("name").value=nameTxt;

        window.document.getElementById("billAmt").value=billAmtTxt;

        window.document.getElementById("lastDate").value=lastDateTxt;
                        window.document.getElementById("type").value=typeTxt;
```

```
                        }
                }

                function displayState(num) {
                        if(num==0) {
                                alert("Initialized");
                        } else if(num==1) {
                                alert("Opened");
                        } else if(num==2) {
                                alert("Sent");
                        } else if(num==3) {
                                alert("Receiving");
                        } else {
                                alert("Recevied");
                        }
                }
        </script>
</head>
<body>
        Mobile No   <INPUT type="text" id="mobile" value=""
onBlur="fetchBillDetails()"><br/>
        Name        <INPUT type="text" id="name" value=""><br/>
        Bill Amt    <INPUT type="text" id="billAmt" value=""><br/>
        Last Date   <INPUT type="text" id="lastDate" value=""><br/>
        Type        <INPUT type="text" id="type" value=""><br/>
</body>
</html>
```

response/output goes like this:

```
<bill>
        <mobile>9848111288</mobile>
        <name>Suryanarayana</name>
        <billAmt>1200.00</billAmt>
        <lastDate>10/9/2019</lastDate>
        <type>Postpaid</type>
</bill>
```

FetchBill.jsp

```
<%-- FetchBill.jsp --%>
<%
	response.setContentType("text/xml");
	long mobile=Long.parseLong(request.getParameter("mobile").trim());
	System.out.println(mobile);
	if(mobile==9848111288L) {
		out.println("<bill>");
				out.println("<mobile>9848111288</mobile>");
				out.println("<name>Suryanarayana</name>");
				out.println("<billAmt>1200.00</billAmt>");
				out.println("<lastDate>10/9/2019</lastDate>");
				out.println("<type>Postpaid</type>");
		out.println("</bill>");
	} else if(mobile==6302151056L) {
		out.println("<bill>");
				out.println("<mobile>6302151056</mobile>");
				out.println("<name>Suryanarayana</name>");
				out.println("<billAmt>399.00</billAmt>");
				out.println("<lastDate>23/9/2019</lastDate>");
				out.println("<type>Prepaid</type>");
		out.println("</bill>");
	} else {
		out.println("<bill>");
				out.println("<mobile>9848012345</mobile>");
				out.println("<name>Idea Owner</name>");
				out.println("<billAmt>1500.00</billAmt>");
				out.println("<lastDate>12/9/2019</lastDate>");
				out.println("<type>Postpaid</type>");
		out.println("</bill>");
	}
%>
```

Copy the above files into D:\Program Files (x86)\Apache Software Foundation\Tomcat 8.5\webapps\ROOT directory

Goto bin directory and click on startup.bat to start tomcat

Open different browsers (IE, Chrome, Firefox and Opera)

open browser and type URLS as
http://localhost:8081/FetchBill.jsp?mobile=9848111288
first of all we must request JSP from browser. Then JSP internally runs on server.

http://localhost:8081/JS_AJAX_65.html

Then you open HTML file, enter mobile number and press TAB so that AJAX request goes to JSP and gets the bill details.

HTML DOM

(Document Object Model)

HTML DOM

Attributes, Document, Element, Events,
Event Objects, HTML Collection, Location,
Navigator, Screen, Style, Window

Web APIs

Console, Geolocation, History, Storage

Full form of DOM

Document Object Model

Purpose of DOM:

DOM API is used to create various HTML objects such as Processing Instruction, Doctype declaration, Comment, Element, Attribute, Text, Entity, CDATASection etc and to append, insert, remove, replace, retrieve these objects dynamically to and from HTML body.

Building HTML document dynamically from JavaScript using HTML DOM API is called as DHTML (Dynamic HTML).

Below is an example on directly writing <b> tag in html document.

```
<html>
	<body>
		<b>Hi iam bold text</b>
	</body>
</html>
```

Hence to dynamically create Element, Attribute, Text -
methods are given in window.document object:

createElement(elementName)
createAttribute(attributeName)
attr.value=val
createTextNode(textValue)

Following are the methods avaiable in DOMElement:

appendChild(node)
replaceChild(oldNode, newNode)
removeChild(node)
insertBefore(beforeNode, newNode)
node.setAttributeNode(attribute)
getElementById(id)
getElementsByTagName(tagName)
getElementsByClassName(className)
getAttribute(attrName)

1) how to create <b> bold tag with text enclosed in it
JS_HTMLDOM_66.html

```
<!-- JS_HTMLDOM_66.html -->
<!--
	Currently HTML body is empty
	The task is to create a <b> tag, append some text content to it
		<b>Hi iam Bold Text</b>, <i>Hi iam Chitti, iam a Robo</i>
	and append that content back to div whose id is output
-->
<html>
	<head>
```

```
		<script>
			function dynaText() {
				var bEle=window.document.createElement("b");
				var bTxt=window.document.createTextNode("Hi, iam Bold Text ");
				bEle.appendChild(bTxt);

				var iEle=window.document.createElement("i");
				var iTxt=window.document.createTextNode("Hi, iam Chitt, iam a Robo");
				iEle.appendChild(iTxt);

				var outputDiv=window.document.getElementById("output");
				outputDiv.appendChild(bEle);
				outputDiv.appendChild(iEle);
				// outputDiv.innerHTML="<b>Hi iam bold text</b>";
			}
		</script>
	</head>
	<body >
		<div id="output">
		</div>
		<input type="button" name="button1" value="Dyna Text" onClick="dynaText()"/>
	</body>
</html>

<!-- HTMLDOM_66_0.html -->
<!--
	insert one <b>Hi iam Revanth,</b><i> Im  Fine</i>
-->
<!doctype html>
<html lang="en">
 <head>
  <title>Sadahana First HTML DOM</title>
  <script>
	function updtMsg() {
		// window.document contains createElement() and createTextNode() functions
```

```
            // create <b> tag
            var bEle=window.document.createElement("b");

            // create text
            var bTxt=window.document.createTextNode("Hi iam Revanth,");

            // Append text to <B> tag
            bEle.appendChild(bTxt);

            // create <i> tag
            var iEle=window.document.createElement("i");

            // create text
            var iTxt=window.document.createTextNode(" Im Fine");

            // append text to <i> tag
            iEle.appendChild(iTxt);

            // append <b> and <i> tags to output
            // get the output id element
            var output=window.document.getElementById("output");
            output.appendChild(bEle);
            output.appendChild(iEle);
        }
 </script>
 </head>

 <body>
  <DIV id="output">
  </DIV>
  <input type="button" name="button" value="Wish" onClick="updtMsg()"/>
 </body>
</html>

<!-- HTMLDOM_66_1.html -->
<!--
        insert one <b>Hi iam Revanth</b> when user presses button1
        when button2 is pressed the content in the output must be removed and
```

```
        new content <i> Im  Fine</i> must be appended
-->
<!doctype html>
<html lang="en">
 <head>
  <title>Sadahana Second HTML DOM</title>
  <script>

        function dispMsg1() {
                // create <b> tag
                var bEle=window.document.createElement("b");

                // create text
                var bTxt=window.document.createTextNode("Hi iam Revanth,");

                // Append text to <B> tag
                bEle.appendChild(bTxt);

                // append <b> tag to output
                // get the output id element
                var output=window.document.getElementById("output");
                output.innerHTML="";
                output.appendChild(bEle);
        }

        function dispMsg2() {
                var output=window.document.getElementById("output");
                output.innerHTML="";

                // create <i> tag
                var iEle=window.document.createElement("i");

                // create text
                var iTxt=window.document.createTextNode(" Im Fine");

                // append text to <i> tag
                iEle.appendChild(iTxt);
```

```
                // append <i> tag to output
                output.appendChild(iEle);
        }
 </script>
</head>

<body>
 <DIV id="output">
 </DIV>
 <input type="button" name="button" value="Button1" onClick="dispMsg1()"/>
 <input type="button" name="button" value="Button2" onClick="dispMsg2()"/>
</body>
</html>
```

2) how to create <a> anchor tag with href attribute pointing to

```
<a href="http://www.activenetinformatics.com/index.html">ActiveNET</a>
```

JS_HTMLDOM_67.html

```
<html>
        <head>
                <script type="text/javascript" lang="javascript">
                        function init() {
                                var aEle=window.document.createElement("a");
                                var hrefAttr=window.document.createAttribute("href");

        hrefAttr.value="http://www.activenetinformatics.com/index.html";
                                aEle.setAttributeNode(hrefAttr);
                                var aTxt=window.document.createTextNode("ActiveNET");
                                aEle.appendChild(aTxt);
                                var outputEle=window.document.getElementById("output");
                                outputEle.appendChild(aEle);
                        }
                </script>
        </head>
        <body onLoad="init()">
                <div id="output">
                </div>
```

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

3) how to create countries drop down list box containing India, Nepal, Bhutan, Srilanka, Bangladesh

```
<select name="countries">
        <option value="India">India</option>
        <option value="Nepal">Nepal</option>
        <option value="Bhutan">Bhutan</option>
        <option value="Srilanka">Srilanka</option>
        <option value="Bangladesh">Bangladesh</option>
</select>
```

JS_HTMLDOM_68.html

```
<html>
<head>
  <script type="text/javascript" lang="javascript">
  function init() {

    var sEle=window.document.createElement("select");

                var nameAttr=window.document.createAttribute("name");
                nameAttr.value="countries";
                sEle.setAttributeNode(nameAttr);

    var oEle=window.document.createElement("option");
                var valueAttr=window.document.createAttribute("value");
                valueAttr.value="India";
                oEle.setAttributeNode(valueAttr);
    var txt=window.document.createTextNode("India");
    oEle.appendChild(txt);
                sEle.appendChild(oEle);

    oEle=window.document.createElement("option");
                valueAttr=window.document.createAttribute("value");
                valueAttr.value="Nepal";
                oEle.setAttributeNode(valueAttr);
```

```
    txt=window.document.createTextNode("Nepal");
    oEle.appendChild(txt);
            sEle.appendChild(oEle);

    oEle=window.document.createElement("option");
            valueAttr=window.document.createAttribute("value");
            valueAttr.value="Bhutan";
            oEle.setAttributeNode(valueAttr);
    txt=window.document.createTextNode("Bhutan");
    oEle.appendChild(txt);
            sEle.appendChild(oEle);

    oEle=window.document.createElement("option");
            valueAttr=window.document.createAttribute("value");
            valueAttr.value="Srilanka";
            oEle.setAttributeNode(valueAttr);
    txt=window.document.createTextNode("Srilanka");
    oEle.appendChild(txt);
            sEle.appendChild(oEle);

    oEle=window.document.createElement("option");
            valueAttr=window.document.createAttribute("value");
            valueAttr.value="Bangladesh";
            oEle.setAttributeNode(valueAttr);
    txt=window.document.createTextNode("Bangladesh");
    oEle.appendChild(txt);
            sEle.appendChild(oEle);

    var outputEle=window.document.getElementById("output");
    outputEle.appendChild(sEle);
  }
  </script>
</head>
<body onLoad="init()">
  <div id="output">
  </div>
</body>
</html>
```

4) how to find count of <a> tags in HTML document

```
<!-- JS_HTMLDOM_69.html -->
<html>
<head>
   <script type="text/javascript" lang="javascript">
   function init() {
                    var outputTxt="";
                    var aEles=window.document.getElementsByTagName("a");
                    var len=aEles.length;
                    window.alert(len);
                    for(var i=0;i<len;i++) {
                              // outputTxt+=aEles[i].tagName;
                              outputTxt+=aEles[i]+"<br/>";
                              // window.alert(aEles[i].tagName);
                              window.alert(aEles[i]);
                    }
                    window.document.getElementById("output").innerHTML=outputTxt;
                    window.alert(outputTxt);
   }
   </script>
</head>
<body onLoad="init()">
          <a href="http://www.activenetinformatics.com">ActiveNET</a>
          <a href="http://www.google.com">Google</a>
          <a href="http://www.msn.com">MSN</a>
          <a href="http://www.gmail.com">GMail</a><br/><br/>
   <div id="output">
   </div>
</body>
</html>
```

Student Assignments:

1) how to create <b> bold tag with text enclosed in it

2) how to create <a> anchor tag with href attribute pointing to
ActiveNET

3) how to create countries drop down list box containing India, Nepal, Bhutan, Srilanka, Bangladesh

```
<select name="countries">
        <option value="India">India</option>
        <option value="Nepal">Nepal</option>
        <option value="Bhutan">Bhutan</option>
        <option value="Srilanka">Srilanka</option>
        <option value="Bangladesh">Bangladesh</option>
</select>
```

4) how to find count of <a> tags in HTML document

5) how to find count of tags in HTML document

6) how to find the text placed in a tag whose id="one"

7) how to find the text placed in a tags whose class="oneness"

8) how to remove text from a particular div tag with id="two"

9) how to remove attribute style from a div tag with id="three"

10) how to add attribute style to a div tag with id="four"

window.document/DOMDocument Object Properties and Methods:

---

activeElement Returns the currently focused element in the document

addEventListener Attaches an event handler to the document

anchors Returns a collection of all <a> elements in the document that have a name attribute

applets	Returns a collection of all <applet> elements in the document
baseURI	Returns the absolute base URI of a document
body	Sets or returns the document's body (the <body> element)
cookie	Returns all name/value pairs of cookies in the document
createAttribute()	Creates an attribute node
createComment()	Creates a Comment node with the specified text
createElement()	Creates an Element node
createTextNode()	Creates a Text node
doctype	Returns the Document Type Declaration associated with the document
documentElement	Returns the Document Type Declaration associated with the document
documentURI	Sets or returns the location of the document
domain	Returns the domain name of the server that loaded the document
forms	Returns a collection of all <form> elements in the document
getElementById()	Returns the element that has the ID attribute with the specified value
getElementsByClassName()	Returns a NodeList containing all elements with the specified class name
getElementsByName()	Returns a NodeList containing all elements with a specified name
getElementsByTagName()	

Returns a NodeList containing all elements with a specified name

hasFocus() Returns a Boolean value indicating whether the document has focus

head Returns the <head> element of the document

images Returns a collection of all <img> elements in the document

links Returns a collection of all <a> and <area> elements in the document that have a href attribute

normalize() Returns a collection of all <a> and <area> elements in the document that have a href attribute

querySelector() Returns a collection of all <a> and <area> elements in the document that have a href attribute

querySelectorAll() Returns a static NodeList containing all elements that matches a specified CSS selector(s) in the document

readyState Returns the (loading) status of the document

removeEventListener() Removes an event handler from the document (that has been attached with the addEventListener() method)

scripts Returns a collection of <script> elements in the document

title Sets or returns the title of the document

URL Returns the full URL of the HTML document

write() Writes HTML expressions or JavaScript code to a document

writeln() Same as write(), but adds a newline character after each statement

Element/Node DOMElement contains:

---

addEventListener() Attaches an event handler to the specified element

appendChild() Adds a new child node, to an element, as the last child node

attributes Returns a NamedNodeMap of an element's attributes

blur() Removes focus from an element

childElementCount Returns the number of child elements an element has

childNodes Returns a collection of an element's child nodes (including text and comment nodes)

children Returns a collection of an element's child element (excluding text and comment nodes)

classList Returns the class name(s) of an element

className Sets or returns the value of the class attribute of an element

click() Simulates a mouse-click on an element

contains() Returns true if a node is a descendant of a node, otherwise false

contentEditable Sets or returns whether the content of an element is editable or not

dir Sets or returns the value of the dir attribute of an element

firstChild	Returns the first child node of an element
firstElementChild	Returns the first child element of an element
focus()	Gives focus to an element
getAttribute()	Returns the specified attribute value of an element node
getAttributeNode()	Returns the specified attribute node
getElementsByClassName()	Returns a collection of all child elements with the specified class name
getElementsByTagName()	Returns a collection of all child elements with the specified tag name
hasAttribute()	Returns true if an element has the specified attribute, otherwise false
hasAttributes()	Returns true if an element has any attributes, otherwise false
hasChildNodes()	Returns true if an element has any child nodes, otherwise false
id	Sets or returns the value of the id attribute of an element
innerHTML	Sets or returns the content of an element
innerText	Sets or returns the text content of a node and its descendants
insertBefore()	Inserts a new child node before a specified, existing, child node
isContentEditable	Returns true if the content of an element is editable, otherwise

	false
isEqualNode()	Checks if two elements are equal
isSameNode()	Checks if two elements are the same node
lastChild	Returns the last child node of an element
lastChildElement	Returns the last child element of an element
nextSibling	Returns the next node at the same node tree level
nextElementSibling	Returns the next element at the same node tree level
nodeName	Returns the name of a node
nodeType	Returns the node type of a node
nodeValue	Sets or returns the value of a node
normalize()	Joins adjacent text nodes and removes empty text nodes in an element
parentNode	Returns the parent node of an element
parentElement	Returns the parent element node of an element
previousSibling	Returns the previous node at the same node tree level
previousElementSibling	Returns the previous element at the same node tree level
querySelector()	Returns the first child element that matches a specified CSS selector(s) of an element
querySelectorAll()	Returns all child elements that matches a specified CSS selector(s) of an element

removeAttribute()	Removes a specified attribute from an element
removeAttributeNode()	Removes a specified attribute node, and returns the removed node
removeChild()	Removes a child node from an element
removeEventListener()	Removes an event handler that has been attached with the addEventListener() method
replaceChild()	Replaces a child node in an element
scrollLeft	Sets or returns the number of pixels an element's content is scrolled horizontally
scrollTop	Sets or returns the number of pixels an element's content is scrolled vertically
scrollHeight	Returns the entire height of an element, including padding
scrollWidth	Returns the entire width of an element, including padding
setAttribute()	Sets or changes the specified attribute, to the specified value
setAttributeNode()	Sets or changes the specified attribute node
style	Sets or returns the value of the style attribute of an element
tabIndex	Sets or returns the value of the tabindex attribute of an element
tagName	Returns the tag name of an element
textContent	Sets or returns the textual content of a node and its descendants

title Sets or returns the value of the title attribute of an element

Creating User defined JavaScript Objects:

sid, sname, email, mobile, qual, stream, college, year_of_passing, techs_known
30 nos

students.xml

```
<students>
	<student>
		<sid>1</sid>
		<sname>ABC</sname>
		<email>abc@gmail.com</email>
		<mobile>9848012345</mobile>
		<qual>BTech</qual>
		<stream>CSE</stream>
		<college>JNTUCH</college>
		<year_of_passing>2020</year_of_passing>
		<techs_known>C, C++, PHP</techs_known>
	</student>

	<student>
		<sid>2</sid>
		<sname>XYZ</sname>
		<email>xyz@gmail.com</email>
		<mobile>9848012345</mobile>
		<qual>BTech</qual>
		<stream>CSE</stream>
		<college>JNTUC</college>
		<year_of_passing>2020</year_of_passing>
		<techs_known>C, C++, PHP</techs_known>
	</student>
</students>
```

students.json

```
students = [
        {"sid":"1", "sname":"Sadhana", "email":"sadhana@hmail.com",...},
        {"sid":"2", "sname":"Revanth", "email":"revanth@hmail.com",...},
]
```

student.json

```
student{"sid":"1", "sname":"Sadhana", "email":"sadhana@gmail.com",...}
```

Way I:

This is how we create class-cum-object in JavaScript

```
var user = new Object();
user.first="Surya";
user.last="narayana";
```

Way II:

This is how we create class-cum-object in JavaScript

```
var user = {'first':'Surya','last':'narayana'};
```

Way III:

This is how we create class in JavaScript

```
function User(first, last){
        this.first = first;
        this.last = last;
}
```

This is how we create object in JavaScript for above class

```
var user = new User("Surya", "narayana");
```

This is how we print object

```
document.write(user.first + " " + user.last);
```

Adding Methods to JavaScript objects:

```
var user = new Object();
user.first="Surya";
```

```
user.last="narayana";
user.getFullName = makeFullName; // assigning makeFullName() function to getFullName() function

var user2 = {'first':'Surya', 'last':'narayana', 'getFullName':makeFullName};

// defining function
function makeFullName(){
	return this.first + " " + this.last;
}

document.write(user.getFullName());
```

We can also write methods in class like below:

```
function User(first, last){
	this.first = first;
	this.last = last;
	this.getFullName = function(){
		return this.first + " " + this.last;
	};
}

This is how we create object in JavaScript for above class
var user = new User("Surya", "narayana");

This is how we print object
document.write(user.getFullName());
```

Object class material:

---

javascript Beginner to Expert with javascript Programming Ch15
JavaScript & jQuery Interactive Front-End Development
JavaScript 209/177P
JavaScript Pocket Reference, 3rd Edition Ch5 75P