Introduction to Javascript

Embedding JavaScript in HTML

- **The <SCRIPT> tag**

```
<SCRIPT>
   JavaScript statements …
</SCRIPT>
```

```
<html>
<head> </head>
<body>
<script type="text/javascript">
      document.write("<H1>Hello World!</H1>")
</script>
</body>
</html>
```

- **Where to Write JavaScript?**
  - Head Section
  - Body Section
  - External File

```
//common.js file contents
var msg
msg="<h1>in external file</h1>"
```

```
<head>
<script src="common.js">
       <!– no javascript statements →
   </ script>
</head>
<body>
   <script>
        document.write("display value of a variable"+msg)
   </script>
</body>
```

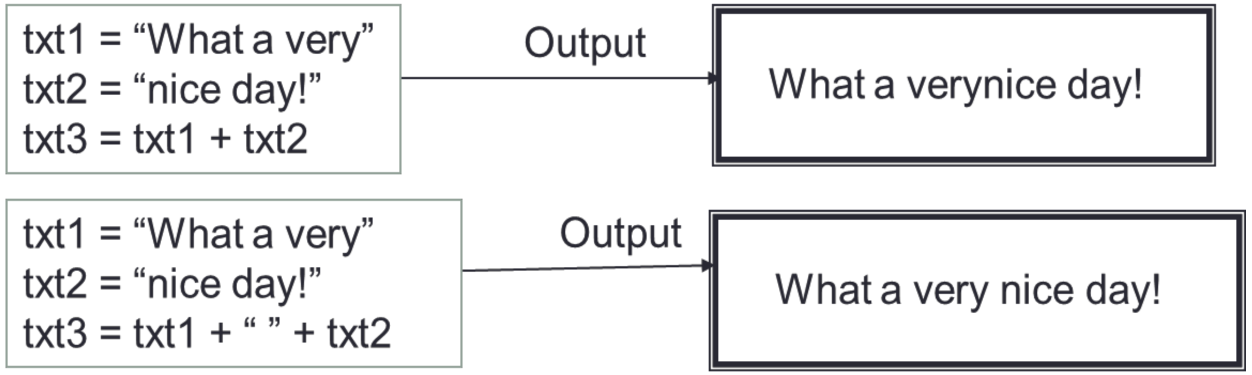
Data Types in JavaScript

- **JavaScript is a free-form language. Need not declare all variables, classes, and methods**
- **Variables in JavaScript can be of type:**
  - Number (4.156, 39)
  - String (“This is JavaScript”)
  - Boolean (true or false)
  - Null (null) → usually used to indicate the absence of a value

- **Defining variables. var variableName = value**
- **JavaScript variables are said to be un-typed or loosely typed**
  - letters of the alphabet, digits 0-9 and the underscore (_) character and is case-sensitive.
  - Cannot include spaces or any other punctuation characters.
  - First character of name must be either a letter or the underscore character.
  - No official limit on the length of a variable name, but must fit within a line.

Javascript operators:

- Arithmetic Operators (+ , -, * , / , %)
- Assignment Operators(=,+=,-=,*=,/=,%=)
- Comparison Operators (==,!=,<,<=,>,>=)
- Boolean Operators(&&,||,!)
- Bitwise Operators(&,|,!,^,<<,>>,>>>)
- String Operators(=,+,+=)



```
txt1 = “What a very”
txt2 = “nice day!”
txt3 = txt1 + txt2
```

```
What a verynice day!
```

```
txt1 = “What a very”
txt2 = “nice day!”
txt3 = txt1 + “ ” + txt2
```

```
What a very nice day!
```

Control Structures and Loops

- **JavaScript supports the usual control structures:**
  - the conditionals:
    - if,
    - if...else
    - If … else if … else
    - Switch

```
if(condition) {
      statement 1
} else {
       statement 2
}
```

```
if(a>10) {
document.write(“Greater than 10”)
} else {
document.write(“Less than 10”)
}
```

```
document.write( (a>10) ? “Greater than 10” : “Less than 10”);
```

```
switch (variable) {
      case outcome1 :{
              //stmts for outcome 1
               break; }
      case outcome2 :{
              //stmts outcome 2
                break; }
       default: {
               //No outcome chosen
           }
}
```

  - iterations:
    - for
    - while

```
for( [initial expression;][condition;][increment expression] )
 {    statements  }
```

```
for(var i=0;i<10;i++) {
         document.write(“Hello”);
}
```

```
while(condition) {
    statements
}
```

```
while(i<10) {
     document.write(“Hello”);
     i++;
}
```

JavaScript Functions

```
function myFunction (arg1, arg2, arg3) {
    statements ; [return]
}
```

Calling the function :

```
myFunction( “abc”, “xyz”, 4 )
myFunction()
```

```
function area(w1, w2, h) {
   var  area=(w1+w2)*h/2;
   alert(area+" sq ft");
}
area(2,3,7); //calling the function
```

```
function diameter(radius){
    return radius * 2;
}

var d=diameter(5); //calling the function
```

- **Function expressions - functions are assigned to variables**

```
var myFunction = function() {
    statements
}
```

```
var area = function (radius) {
      return Math.PI * radius * radius;
  };
alert(area(5));        // => 78.5
```

- **Global and Local Variables**

```
<script language=“Javascript”>
  var cName=“TechnoFlo”
  function f(){
    var empName=“Henry”
    document.write(“Welcome to ”+cName+ ”, “ +empName)
  }
</script>
```

Variables that exist only inside a function are called Local variables
Variables that exist throughout the script are called Global variables
Their values can be changed anytime in the code and even by other functions

Predefined Functions

- **isFinite: evaluates an argument to determine if it is a finite number.**

```
isFinite (number)      //where number is the number to evaluate
```

- **isNaN : Evaluates an argument to determine if it is “NaN” (not a number)**
  - isNaN (testValue), where testValue is the value you want to evaluate
- **Parseint and parsefloat**
  - Returns a numeric value for string argument.
  - parseInt (str)
  - parseFloat (str)

String Objects

- **Creating a string object:**
  - var myString = new String(“characters”)
  - var myString = “fred”
- **Properties of a string object:**
  - length: returns the number of characters in a string.

    - “Lincoln”.length // result = 7
    - “Four score”.length // result = 10
    - “One\ntwo”.length // result = 7
    - “”.length // result = 0

- **String functions**
  - charAt(index) : returns the character at a specified position.
    - Eg : var str = "Hello world!";
    - str.charAt(0); //returns H
    - str.charAt(str.length-1)); //returns !
  - concat() : joins two or more strings
    - stringObject.concat(stringX,stringX,...,stringX)
    - Eg: var str1="Hello ";
      var str2="world!";
      document.write(str1.concat(str2));

String function

- **indexOf () : returns the position of the first occurrence of a specified string value in a string.**
  - index values start their count with 0.
  - If no match occurs within the main string, the returned value is -1.
  - string.indexOf(searchString [, startIndex])

```
Eg : var str="Hello world, welcome";
str.indexOf("Hello"); //returns 0
str.indexOf("wor")); //returns 6
str.indexOf("e",5); //returns 14
```

- **split(“delimiterCharacter”[, limitInteger]) - Splits a string into array of strings**
  - string.split(“delimiterCharacter”[, limitInteger])

```
var str = "zero one two three four";
var arr = str.split(" ");
for(i = 0; i < str.length; i++){ document.write(“<br>” + arr[i]); }
```

```
Output :
     zero
      one
      two
      three
      four
```

```
var myString = “Anderson,Smith,Johnson,Washington”
var myArray = myString.split(“,”)
var itemCount = myArray.length // result: 4
```

String Objects

- **match(regExpression)**
  - Searches for a specified value in a string
  - string.match(regExpression)

```
var str="rain in SPAIN is mainly in plain";
var patt1=/ain/gi;
document.write(str.match(patt1));
```

- **replace(regExpression, replaceString)**
  - Replaces some characters with some other characters in a string.
  - string.replace(regExpression, replaceString)
  - Eg: var str="Hello World";
    document.write(str.replace("World", "Everyone"));

```
var str = “To be, or not to be”
var regexp = /be/
str.relace(regexp, “exist”)
```

- **search(regExpression)**
  - Searches a string for a specified value
  - Eg : var str="Hello World";
    str.search("World") //returns 6

```
var text = "testing: 1, 2, 3"; // Sample text
var pattern = /\d+/g // Matches all instances of one or more digits
text.search(pattern) // => 9: position of first match
text.match(pattern) // => ["1", "2", "3"]: array of all matches
text.replace(pattern, "#"); // => "testing: #, #, #"
```

String functions

- **toLowerCase() / toUpperCase()**

```
Eg: var str="Hello World!";
str.toLowerCase() //returns hello world
str.toUpperCase() //returns HELLO WORLD
```

- **slice(startIndex [, endIndex])**
  - Extracts a part of a string and returns the extracted part in a new string

```
Eg : var str=“Hello World";
      str.slice(6) //returns World
      str.slice(0,1) //returns H
```

Date

- **Date object allows the handling of date and time information.**
  - All dates are in milliseconds from January 1, 1970, 00:00:00.
  - Dates before 1970 are invalid dates.
- **There are different ways to define a new instance of the date object:**

```
var d = new Date()        //Current date
var d = new Date(milliseconds)
var d = new Date(dateString)
var d = new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

```
<script>
    var d=new Date();
    document.write(d);
</script>
```

```
var d = new Date(86400000);
var d = new Date(99,5,24,11,33,30,0);
```

Date Object - Methods

getDate( )	Date of the month (1 - 31)
getDay( )	Day of the week (0 - 6, 0-Sunday)
getMonth( )	The month (0 - 11, 0 - Jan.)
getFullYear( )	The year (4 digits)
getHours( )	Hour of the day (0 - 23)
getMinutes( )	Minutes (0 - 59)
getSeconds( )	Seconds (0 - 59)
getTime( )	Milliseconds since 1/1/1970
getTimezoneOffset( )	Offset between local time and GMT
setDate(dayValue)	1-31
setHours(hoursValue)	0-23
setMinutes(minutesValue)	0-59
setMonth(monthValue)	0-11
setSeconds(secondsValue)	0-59
setTime(timeValue)	>=0
setYear(yearValue)	>=1970

Array

- **An array is data structure for storing and manipulating ordered collections of data.**
- **An array can be created in several ways.**
  - Eg1: Regular: --→
  - Eg 2: Condensed:
    - var cars=new Array("Spark","Volvo","BMW");
  - Eg 3: Literal:
    - var cars=["Spark","Volvo","BMW"];
  - Eg 4: var matrix = [[1,2,3], [4,5,6], [7,8,9]];
  - Eg 5 : var sparseArray = [1,,,,5];

```
var cars=new Array();
cars[0]="Spark";
cars[1]="Volvo";
cars[2]="BMW";
```

- **Deleting an array element eliminates the index from the list of accessible index values**
  - delete is a unary operator that attempts to delete the **object property** or **array element** specified
  - This does not reduce array's length

```
myArray.length// result: 5
delete myArray[2]
myArray.length// result: 5
myArray[2] // result: undefined
```

Array Object Methods

- arrayObject.reverse()
- arrayObject.slice(startIndex, [endIndex])
- arrayObject.join(separatorString) : array contents will be joined and placed into arrayText by using the comma separator“
- arrayObject.push(): add one or more values to the end of an array

```
arrayObject.slice(startIndex [, endIndex])          //Returns: Array
var solarSys = new Array (“Mercury”,”Venus”,”Earth”,”Mars”,”Jupiter”,”Saturn”)
var nearby = solarSys.slice(1,4)
// result: new array of “Venus”, “Earth”, “Mars”
```

```
arrayObject.concat(array2)
var a1 = new Array(1,2,3)
var a2 = new Array(“a”,”b”,”c”)
var a3 = a1.concat(a2)
// result: array with values 1,2,3,”a”,”b”,”c”
```

```
var arrayText = myArray.join(“,”)
```

```
a = []; // Start with an empty array
a.push("zero") // Add a value at the end. a = ["zero"]
a.push("one", "two") // Add two more values. a = ["zero", "one", "two"]
```

Creating New Objects

1. **Using Object Initializers**
   - Syntax : objName = {property1:value1, property2:value2, … }
   - person = { "name ":"amit", "age":23};
   - myHonda = {color:“red”, wheels:4, engine:{cylinders:4, size:2}}

2. **Using Constructors**
   - Define the object type by writing a constructor function.
   - Create an instance of the object with new.

```
function car(make, model, year) {
  this.make = make
  this.model = model
  this.year = year
}
…..
mycar = new car( “Ford” , “Mustang” , 2013)
```

```
function person(name, age) {
   this.name = name
   this.age = age
}
ken = new person( “Ken” , 33 )
```

```
function car(make, year, owner) {
  this.make = make
  this.year = year
  this.owner = owner
}
car1 = new car( “Mazda”, 1990, ken )
```

Creating New Objects(Contd.)

- **Accessing properties**

```
car2.owner.name
```

```
car1.make = “corvette”
```

- **Defining methods**

```
obj.methodName = function_name

obj.methodName(params)
```

```
function car(make, model, year, owner) {
                this.make = make;
                this.model = model;
                this.year = year;
                this.displayCar = displayCar;
}
function displayCar() {
                document.writeln( “A beautiful” + this.year
               + “ ” + this.make + “ ” + this.model
 }
….

car1.displayCar();   car2.displayCar()
```

Examples : Using Object Initializers

```
// Example 1
var myFirstObject = {};
myFirstObject.firstName = "Andrew";
myFirstObject.lastName = "Grant";
console.log(myFirstObject.firstName);
```

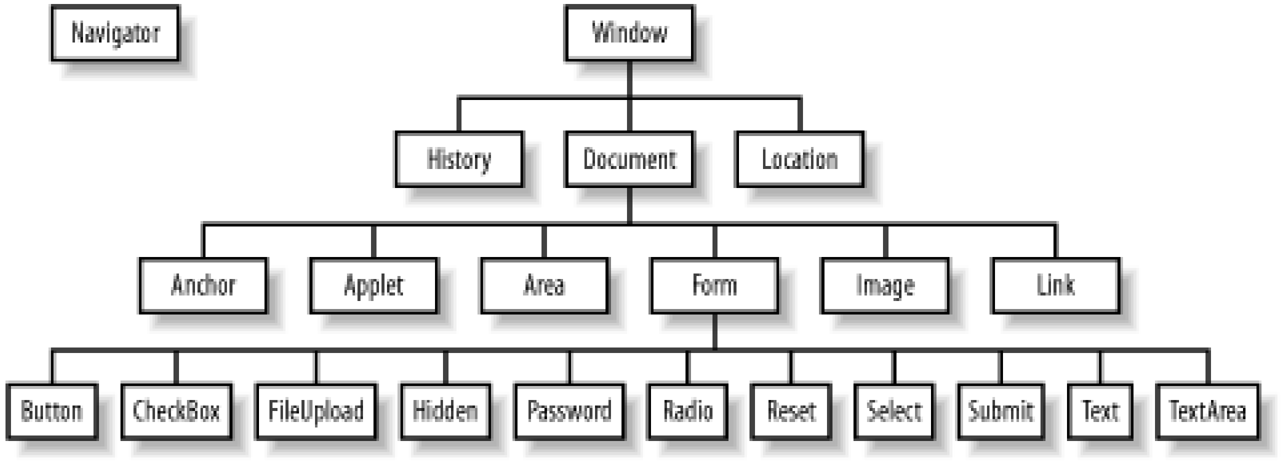
```
// Example 2
var mySecondObject = {
    firstName: "Andrew",
    lastName: "Grant"
};
console.log(mySecondObject.firstName);
```

```
// Example 3
var myThirdObject = new Object();
myThirdObject.firstName = "Andrew";
myThirdObject.lastName = "Grant";
console.log(myThirdObject.firstName);
```

```
var myFirstObject = {};
myFirstObject.firstName = "Andrew";
console.log(myFirstObject.firstName);
myFirstObject.firstName = "Monica";
console.log(myFirstObject.firstName);
myFirstObject["firstName"] = "Catie";
console.log(myFirstObject["firstName"]);
```

```
//Adding Methods to Objects
var person= {
  name: "Andrew",
  age: 21,
  info: function () {
    console.log("Name" + this.name );
    console.log("Age" + this.age );
   }
};
person.info();
for (var prop in person) {
     console.log(person[prop]);
}
```

DOM Model



Window Object Methods

- **alert(message)**
  - window.alert(“message”)
- **confirm(message)**
  - window.confirm(“Exit Application ?”)
- **prompt(message,[defaultReply])**
  - var input=window.prompt(“Enter value of X”)
- **window.open(*URL,name,specs*)**
  - URL : Specifies the URL of the page to open. If no URL is specified, a new window with about:blank is opened
  - Name : Specifies the target attribute or the name of the window.
  - Specs : comma-separated list of items.

```
myWindow=window.open('','','width=200,height=100');
myWindow.document.write("<p>This is 'myWindow'</p>");
myWindow.focus();
```

example opens an about:blank page in a new browser window:

setInterval and setTimeout methods

```
<body>
<input type="text" id="clock" size="35" />
<script language=javascript>
var int=self.setInterval("clock()",50)
function clock() {
   var ctime=new Date()
   document.getElementById("clock").value=ctime
  }
</script>
<button onclick="int=window.clearInterval(int)">Stop interval</button>
</body>
```

```
<head> <script type="text/javascript">
function timedMsg()   {
    var t=setTimeout("alert('5 seconds!')",5000)
 }
</script> </head>
<body> <p>Click on the button. An alert box will be displayed after 5 seconds.</p>
<form>
<input type="button" value="Display timed alertbox!“ onClick="timedMsg()">
</form>
</body>
```

Document Object

- When an HTML document is loaded into a web browser, it becomes a document object; root node of the HTML document and owns all other nodes

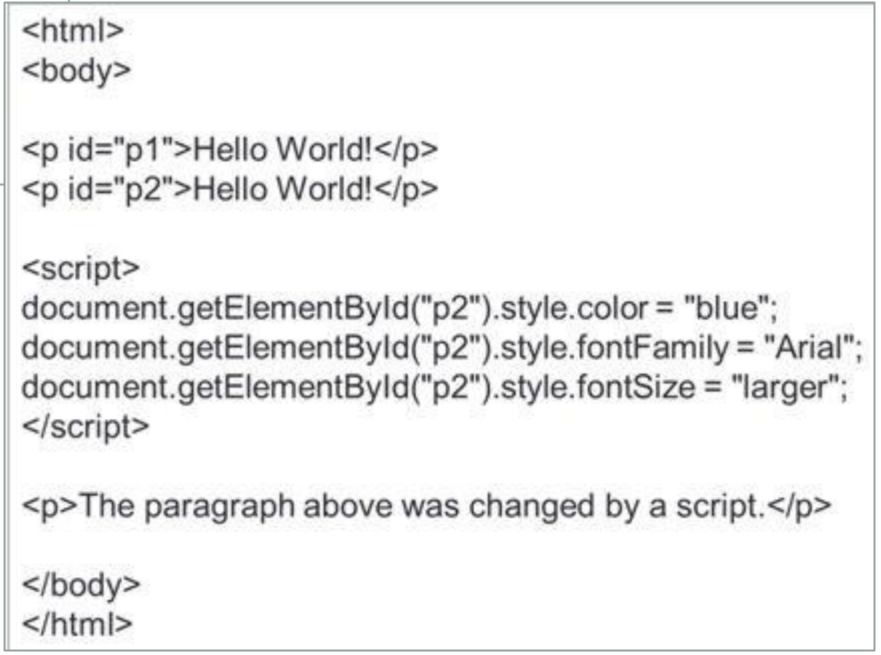
document.anchors	Returns a collection of all the anchors in the document
document.baseURI	Returns the absolute base URI of a document
document.cookie	Returns all name/value pairs of cookies in the document
document.forms	Returns a collection of all the forms in the document
document.getElementById()	Returns the element that has the ID attribute with the specified value
document.getElementsByName()	Accesses all elements with a specified name
document.getElementsByTagName()	Returns a NodeList containing all elements with the specified tagname
document.images	Returns a collection of all the images in the document
document.lastModified	Returns the date and time the document was last modified
document.links	Returns a collection of all the links in the document
document.referrer	Returns the URL of document that loaded current document
document.title	Sets or returns the title of the document
document.URL	Returns the full URL of the document
document.write()	Writes HTML expressions or JavaScript code to a document
document.writeln()	Same as write(), but adds a newline character after each statement

Examples : Modifying content

```
<body>
<p id=“p1">Click the button to change the text.</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
  document.getElementById(“p1").innerHTML="Hello World";
};
</script></body>
```

```
<body>
The title of the document is:
<script>
document.write(document.title);
document.title="another title“  //change the title
</script>
</body>
```

Example : Modifying styles



```
<html>
<body>

<p id="p1">Hello World!</p>
<p id="p2">Hello World!</p>

<script>
document.getElementById("p2").style.color = "blue";
document.getElementById("p2").style.fontFamily = "Arial";
document.getElementById("p2").style.fontSize = "larger";
</script>

<p>The paragraph above was changed by a script.</p>

</body>
</html>
```

Mouse events

```
<SCRIPT>
function changeColor(para){
   para.style.color="blue";
   para.style.backgroundColor = "lightgray";
   para.style.font = "italic bold 30px arial,serif";
}
function revertColor(para){
    para.style.color="black";
    para.style.backgroundColor = "white";
    para.style.font = "12px arial,serif";
}
</SCRIPT>
<BODY>
   <p id="p1" onmouseover="changeColor(this)"
            onmouseout="revertColor(this)">Hover with mouse to see color change</p>
</BODY>
```

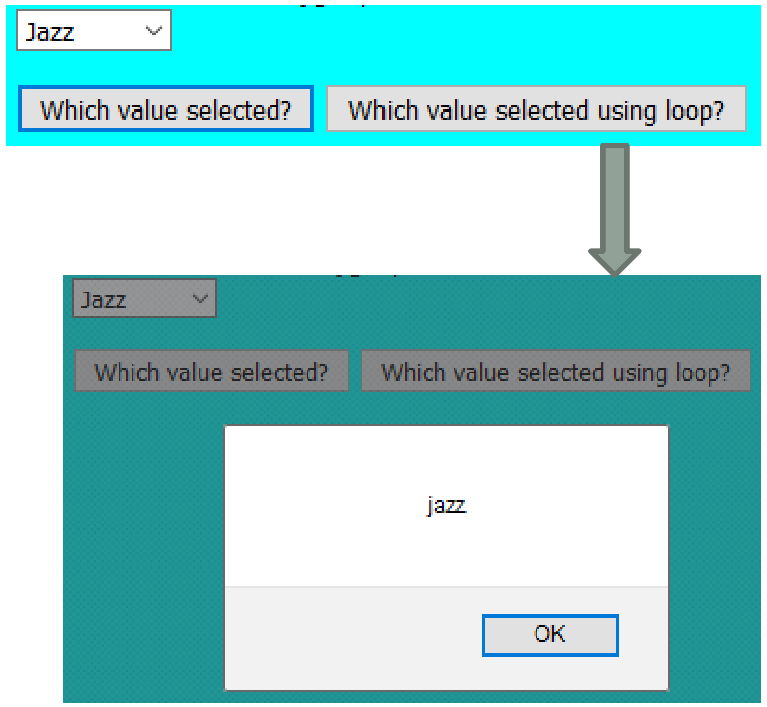
Form validation

```
<html>
<head>
<script>
function validate(){
    var x=document.getElementById("fname").value;
    if (x == null || x == "") {
         alert("Name must be filled out");
         return false;
     }
 }
</script>
</head>
<body>
<form id="form1"  onsubmit="return validate()">
Name: <input type="text" name="fname" id="fname" /><br
<input type="submit" value="validate Name"  />
</form>
</body></html>
```

```
<body>
< Input a number between 1 and 10:</p>
<input id="numb">
<button type="button" onclick="myFunction()">Submit</button>
<p id="demo"></p>
<script>
function myFunction() {
   var x, text;
  x = document.getElementById("numb").value;
  if (isNaN(x) || x < 1 || x > 10) {
      text = "Input not valid";
   } else {
      text = "Input OK";
   }
   document.getElementById("demo").innerHTML = text;
}
</script>
</body>
```

Example

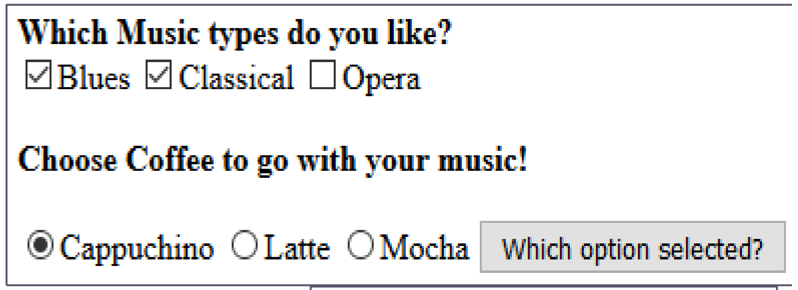
```
<SCRIPT>
function valSelected1(){
  var sel = document.getElementById("musicTypes");
  alert(sel.value);        // prints value, not text
  var opt = sel.options[sel.selectedIndex];
  alert(opt.text);       //option.text prints text
}
function valSelected3(){
  var sel = document.getElementById("musicTypes");    var opt;
    for ( var i = 0, len =; i < sel.options.length; i++ ) {
          opt = sel.options[i];
          if ( opt.selected  == true ) {    break;    }
    }
    alert(opt.value);
}
</SCRIPT>
<FORM NAME="selectForm">
  <SELECT name="musicTypes" id="musicTypes">
    <OPTION VALUE="rnb" SELECTED> R&B </OPTION>
    <OPTION VALUE="jazz"> Jazz </OPTION>
    <OPTION VALUE="blues"> Blues </OPTION>
</SELECT>
<INPUT TYPE="button" VALUE="value selected?"onClick="valSelected1()">
<INPUT TYPE="button" VALUE="value selected using loop?"onClick="valSelected3()">
</FORM>
</BODY>
```

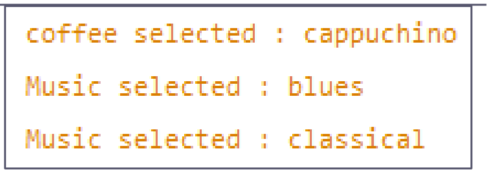

Jazz
Which value selected?
Which value selected using loop?
Jazz
Which value selected?
Which value selected using loop?
OK

jazz

Example

```
<SCRIPT>
function valSelected(){
  var radio = document.getElementsByClassName("r1");
   for(var i = 0; i < radio.length;  i++){
        if(radio[i].checked)   console.log("coffee  selected  : " + radio[i].value);
   }
  var checklist = document.getElementsByClassName("c1");
  for(i=0;i<checklist.length;i++){
      if (checklist[i].checked  == true)   console.log("Music  selected  : " + checklist[i].value);
}
</SCRIPT>
<FORM NAME="selectForm">
<B>Which Music types do you like?</B>
<input type="checkbox"  class="c1"  id="c1"  value="blues">Blues</input>
<input type="checkbox"  class="c1"  id="c2"  value="classical">Classical</input>
<input type="checkbox"  class="c1"  id="c3"  value="opera">Opera</input>
<b>Choose  Coffee  to  go  with  your  music!</b><br>
<INPUT TYPE="radio"  name="coffee"  class="r1"  id="coffee"  VALUE="cappuchino">Cappuchino
</input>
<INPUT TYPE="radio"  name="coffee"  class="r1"  id="coffee"  VALUE="latte">Latte</input>
<INPUT TYPE="radio"  name="coffee"  class="r1"  id="coffee"  VALUE="Mocha">Mocha</input>
<INPUT TYPE="button"  VALUE="Which option  selected?"  onClick="valSelected()">
</FORM>
```


Which Music types do you like?
☑ Blues ☑ Classical ☐ Opera
Choose Coffee to go with your music!
● Cappuchino ○ Latte ○ Mocha
Which option selected?



```
coffee selected : cappuchino
Music selected : blues
Music selected : classical
```