Session: 14

Loops and Arrays

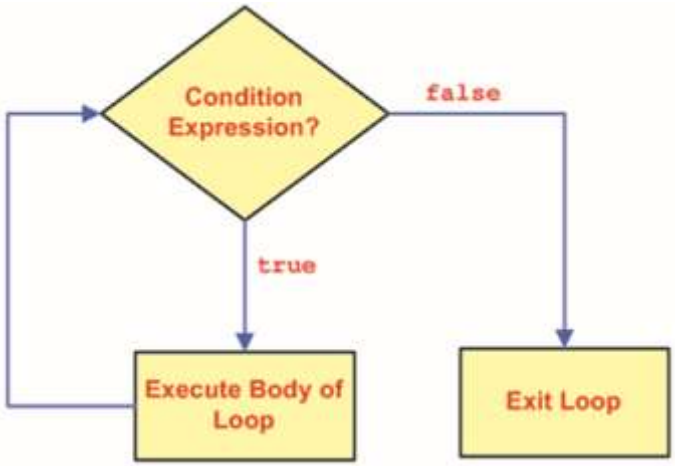
Objectives

- Explain while loop
- Explain for loop
- Explain do..while loop
- Explain break and continue statement
- Explain single-dimensional arrays
- Explain multi-dimensional arrays
- Explain for..in loop

Introduction

- Loops allow you to execute a single statement or a block of statements multiple times.
- A loop construct consists of a condition that instructs the compiler the number of times a specific block of code will be executed.
- If the condition is not specified within the construct, the loop continues infinitely. Such loop constructs are referred to as infinite loops.
- JavaScript supports three types of loops that are as follows:
  - while Loop
  - for Loop
  - do-while Loop

while Loop



Syntax:

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

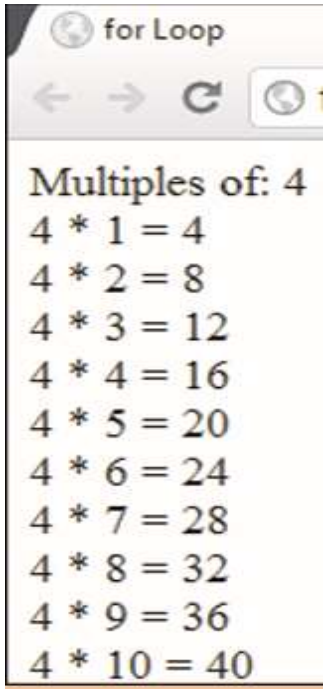
```
<script>
var i =0, sum= 0;
while (i <=10)
{
    sum += i;
    i++;
    alert("sum of first ten numbers:"+sum);
}
</script>
```

for Loop

Syntax:

```
for (initialization; condition; increment/decrement)
{
    // statements;
}
```

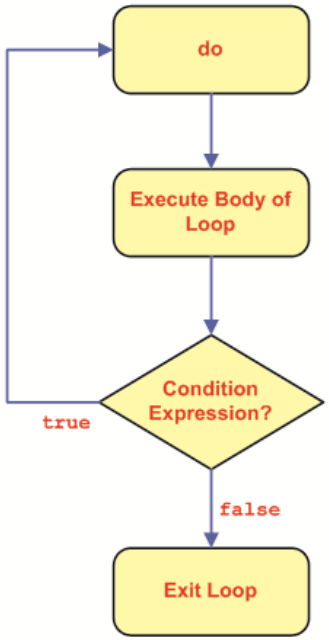
```
<script>
var inputNum = prompt('Enter any number:');
var result = 0;
document.write ('Multiples of: ' + inputNum + '<br />');
for (var i=1; i<=10; i++)
{
   result = inputNum * i ;
   document.write (inputNum + '  *  ' + i + '  =  ' +
        result + '<br />');
}
</script>
```



do-while Loop

Syntax:

```
do {
    ...
    statements;
    ...
} while(condition);
```



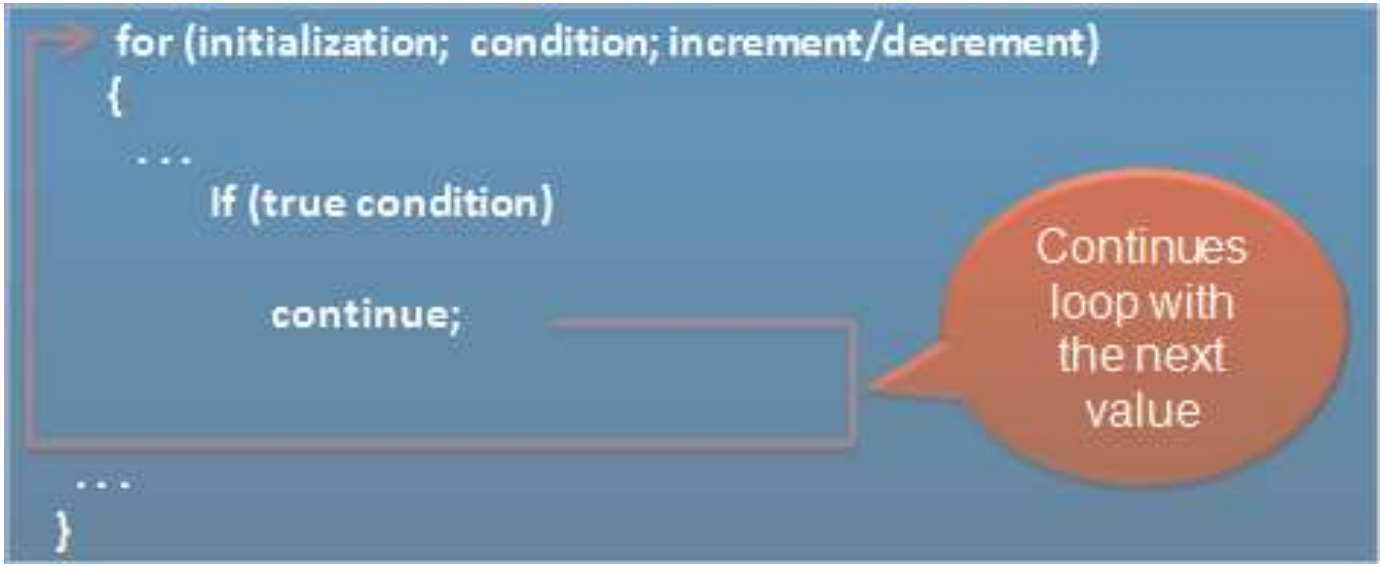

```
<script>
  var answer = '';
  do
  {
       answer = prompt('Capital of United States:', '');
  }while(answer!='Washington');
alert('Capital of United States: ' + answer);
</script>
```

break statement

- can be used with switch-case and loop constructs such as for and while loops.
- is used to exit the loop without evaluating the specified condition.
- The control is then passed to the next statement immediately after the loop.

continue statement 1-2

- terminate the current execution of the loop and continue with the next repetition by returning the control to the beginning of the loop.
- not terminate the loop entirely, but terminates the current execution.

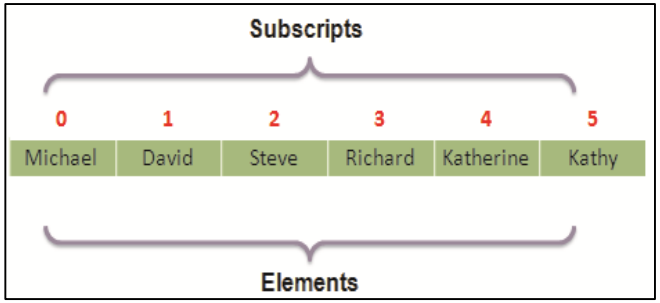


continue statement 2-2

- The Code Snippet displays even numbers from 0 to 15.

```
<script>
  var result = '';
  for (var i = 0; i <= 15; i++)
  {
      if((i%2) != 0)
      {
        continue;
      }
      result = result + i + '\n';
  }
  alert('Even Numbers:\n' + result);
</script>
```

Arrays



- Array is a collection of values stored in adjacent memory locations.
- The values of an array variable must be of the same data type.
- These values that are also referred to as elements can be accessed by using subscript or index numbers.
- JavaScript supports two types of arrays that are as follows:
  - Single-dimensional array
  - Multi-dimensional array
- The array variable can be created using the Array object and new keyword along with the size of the array element.

Single-dimensional Array

```
var variable_name = new Array(size);
variable_name[index] = 'value';
```

```
<script>

//Declaration using Array Object and then Initialization
   var marital_status = new Array(3);
   marital_status[0] = 'Single';
   marital_status[1] = 'Married';
   marital_status[2] = 'Divorced';

//Declaration and Initialization
   var marital_status = new
     Array('Single','Married','Divorced');

//Declaration and Initialization Without Array
   var marital_status = ['Single','Married','Divorced'];

</script>
```

Accessing single-dimensional Arrays

➢ Accessing Array Elements Without Loops

- An array element can be accessed by specifying the array name followed by the square brackets containing the index number.

```
<script>

    var names = new Array("John", "David", "Kevin");
    alert('List of Student Names:\n' + names[0] + ',  '
    + names[1] + ', ' + names[2]);

</script>
```

Accessing single-dimensional Arrays

➢ Accessing Array Elements With Loops

```
<script>

    var sum = 0;
    var marks = new Array(5);
    for(var i=0; i<marks.length; i++)
    {
      marks[i] = parseInt(prompt('Enter Marks:', ''));
      sum = sum + marks[i];
    }
    alert('Average of Marks: ' + (sum/marks.length));

</script>
```

Multi-dimensional Array 1-2

- A multi-dimensional array stores a combination of values of a single type in two or more dimensions.

Employee Salaries → / ↓	0 BASIC	1 HRA	2 ALLOWANCE	3 TOTAL
0	14350	10500	1500	26350
1	34350	4050	1000	39400
2	6150	4500	3250	13900
3	4920	4500	2250	11670
4	12300	9000	2000	23300

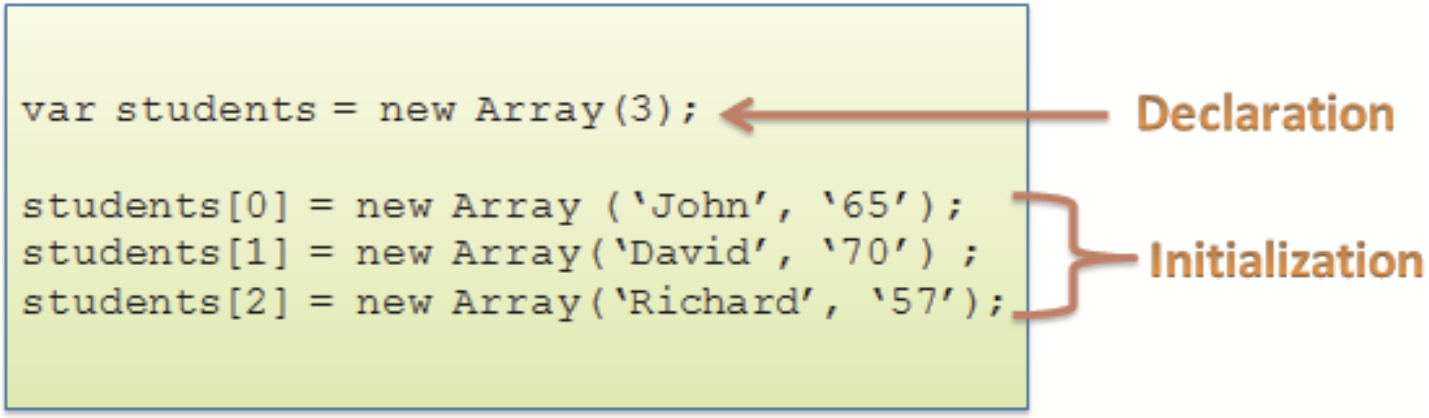
- A two-dimensional array is an array of arrays.
- This means, for a two-dimensional array, first a main array is declared and then, an array is created for each element of the main array.

Multi-dimensional Array 2-2

- The syntax to declare a two-dimensional array is as follows:

```
var var_name = new Array(size);
var_name[index] = new Array('value1','value2'..);
```

- Following figure shows the declaration of a two-dimensional array.



Array Methods 1-2

- In Javascript, array is an object. It has the length property that determine the number of elements in an array.
- The various methods of the Array object allow to access and manipulate the array elements.

Method	Description
concat()	Combines one or more array variables.
join()	Joins all the array elements into a string.
pop()	Retrieves the last element of an array.
push()	Appends one or more elements to the end of an array.
sort()	Sorts the array elements in an alphabetical order.

Array Methods 2-2

```
<script language="javascript">

function f(){
var flowers = new Array('Rose', 'Sunflower', 'Daisy');
document.write('No of flowers: ' + flowers.length + '<br/>');
document.write('Flowers: ' + flowers.join(', ') + '<br/>');
document.write('Orchid and Lily are added:'+flowers.push("Orchid","Lily")+'<br/>');
document.write('Flowers (In Ascending Order):' + flowers.sort() + '<br/>');
document.write('Flowers Removed: ' + flowers.pop() + '<br/>');
}

</script>
```

for .. in Loop 1-2

- is an extension of the for loop.
- It enables to perform specific actions on the arrays of objects.
- The loop reads every element in the specified array and executes a block of code only once for each element in the array.
- Syntax:

```
for (var_name in array_name)
{
     //statements;
}
```

for .. in Loop 2-2

- Example

```
function f(){
    var books = new Array('Beginning CSS 3.0','Introduction to HTML5', 'HTML5 in
MobileDevelopment');

    document.write('<H3> List of Books </H3>');

    for(var i in books)
    {
        document.write(books[i] + '<br/>');
    }
}
```

- Output

Summary

- A loop construct consists of a condition that instructs the compiler the number of times a specific block of code will be executed.
- JavaScript supports three types of loops that include: while loop, for loop,and do-while loop.
- The break statement is used to exit the loop without evaluating the specified condition.
- The continue statement terminates the current execution of the loop and continue with the next repetition by returning the control to the beginning of the loop.
- JavaScript supports two types of arrays namely, Single-dimensional array and Multi-dimensional array.
- The for..in loop is an extension of the for loop that enables to perform specific actions on the arrays of objects.