Unit: 3 Web Scripting JavaScript

Q. 1 What is javaScript ?

Ans:

1. JavaScript is a cross-platform, object-oriented scripting language.
2. JavaScript is a small, lightweight language. It is not useful as a standalone language, but is designed for easy embedding in other products and applications, such as web browsers.
3. Inside a host environment, JavaScript can be connected to the objects of its environment to provide programmatic control over them.

Q.2 What is Difference between client side java script and server side java script?

Ans:

- **Client-side** JavaScript extends the core language by supplying objects to control a browser (Navigator or another web browser) and its Document Object Model (DOM).
  For example, client-side extensions allow an application to place elements on an HTML form and respond to user events such as mouse clicks, form input, and page navigation.
- **Server-side** JavaScript extends the core language by supplying objects relevant to running JavaScript on a server.
  For example, server-side extensions allow an application to communicate with a relational database, provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

Q.3 Use of Java Script.

Ans:

1. **Developing Multimedia Applications**
   The users can use JavaScript to add multimedia elements. With JavaScript you can show, hide, change, resize images and create images rollovers. You can create scrolling text across the status bar, thus making multimedia applications more interactive.
2. **Create Pages Dynamically**
   Based on the user's choice, the date or other external data, JavaScript can produce pages that are customized to the user.
3. **Interact with the User**
   JavaScript can do some processing of forms and can validate user input when the user submits the form.
4. **JavaScript Objects are Similar to Dictionaries**

In JavaScript, objects are just collections of name-value pairs. JavaScript objects are considered as a dictionary with string keys. The users can get and set the properties of an object using either the familiar "." (dot) operator, or the "[ ]" operator, which is typically used when dealing with a dictionary.

Q.4 Explain the features of JavaScript?

Ans:

Browser Support

All browsers have accepted JavaScript as a scripting language and provide integrated support for it. For example, to access flash content, you need to install flash plug-in in your browser. But to use JavaScript, you don't have to use any plug-in at all.

JavaScript can be used on Client Side as well as on Server Side

JavaScript has access to Document Object Model DOM of browser. You can change the structure of web pages at runtime. Thus, JavaScript can be used to add different effects to WebPages. On the other hand, JavaScript could be used on the server side as well.

Functional Programming Language

In JavaScript, function could be assigned to variables just like any other data types. A function can accept another function as a parameter and can also return a function. You can have functions with no name as well. This provides you the ability to code in functional programming style.

Support for Objects

JavaScript is an object oriented language. However, the way JavaScript handles objects and inheritance is bit different from conventional object oriented programming languages like C++/ Java.

Run-time Environment

JavaScript typically relies on a run-time environment (e.g. in a web browser) to provide objects and methods by which scripts can interact with "the outside world".

Vendor-specific Extensions

JavaScript is officially managed by Mozilla Foundation, and new language features are added periodically. However, only some JavaScript engines support these new features.

Object based Features Supported by JavaScript

JavaScript supports various features related to object based language and JavaScript is sometimes referred to as an object-based programming language

Object Type

JavaScript supports the development of object types and in this context JavaScript supports both predefined and user-defined objects. It is possible to assign objects of any type to any variable. It is possible to instantiate the defined object types to create object instances in JavaScript, which is a very powerful feature of Object based language.

Q.5 What is script tag?

Ans:

The <script> tag alerts a browser that JavaScript code follows. It is typically embedded in the HTML.

<script language="javaScript">

Statements....

</script>

Q.6 What is the current version of javaScript?

Ans: JavaScript 1.5 is fully compatible with ECMA-262, Edition 3.

Q.7 How to Save and Run your Program in JavaScript?

Ans:

1. Open any editor like notepad and write the program

2. Save the program in a file with .html extension in a proper folder or subfolder on a drive.

3. Open the web browser like internet explorer or Mozilla Firefox

4. Open the file you created and save in step 2, to execute the program like to execute C:/JAVA/firstprogram.html

Where C is the drive, JAVA is the folder name and firstprogram.html is the name of program which is saved in the Java folder

Q.8 Explain Operators in javaScript?

Ans:

JavaScript operators can be used to perform various operations such as:

- Arithmetic Operators
- Comparison Operators
- Logical Operators
- Relational Operators
- Assignment Operators
- Conditional Operators

Arithmetic Operators

The arithmetic operators are used to perform common arithmetical operations, such as addition, subtraction, multiplication etc. Here's a complete list of JavaScript's arithmetic operators:

Operator	Description	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y.
*	Multiplication	$x * $y	Product of $x and $y.
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y

JavaScript Code Example for Arithmetic Operators

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Arithmetic Operators</title>
</head>
<body>
    <script>
    var x = 10;
    var y = 4;
    document.write(x + y); // Prints: 14
    document.write("<br>");

    document.write(x - y); // Prints: 6
    document.write("<br>");

    document.write(x * y); // Prints: 40
    document.write("<br>");

    document.write(x / y); // Prints: 2.5
    document.write("<br>");

    document.write(x % y); // Prints: 2
```

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

JavaScript Comparison/Relational Operators

The comparison operators are used to compare two values in a Boolean fashion.

Operator	Name	Example	Result
==	Equal	x == y	True if x is equal to y
===	Identical	x === y	True if x is equal to y, and they are of the same type
!=	Not equal	x != y	True if x is not equal to y
!==	Not identical	x !== y	True if x is not equal to y, or they are not of the same type
<	Less than	x < y	True if x is less than y
>	Greater than	x > y	True if x is greater than y
>=	Greater than or equal to	x >= y	True if x is greater than or equal to y
<=	Less than or equal to	x <= y	True if x is less than or equal to y

Code example of Comparison Operators

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Comparison Operators</title>
</head>
<body>
    <script>
    var x = 25;
    var y = 35;
    var z = "25";

    document.write(x == z);  // Prints: true
    document.write("<br>");

    document.write(x === z); // Prints: false
    document.write("<br>");

    document.write(x != y);  // Prints: true
    document.write("<br>");

    document.write(x !== z); // Prints: true
    document.write("<br>");

    document.write(x < y);   // Prints: true
    document.write("<br>");

    document.write(x > y);   // Prints: false
    document.write("<br>");

    document.write(x <= y);  // Prints: true
    document.write("<br>");

    document.write(x >= y);  // Prints: false
    </script>
</body>
```

```
</html>
```

JavaScript Logical Operators

The logical operators are typically used to combine conditional statements.

Operator	Name	Example	Result
&&	And	x && y	True if both x and y are true
||	Or	x || y	True if either x or y is true
!	Not	!x	True if x is not true

Code example of Logical Operator

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Logical Operators</title>
</head>
<body>
    <script>
    var year = 2018;

    // Leap years are divisible by 400 or by 4 but not 100
    if((year % 400 == 0) || ((year % 100 != 0) && (year % 4 == 0))){
        document.write(year + " is a leap year.");
    } else{
        document.write(year + " is not a leap year.");
    }
    </script>
</body>
</html>
```

JavaScript Assignment Operators

The assignment operators are used to assign values to variables.

Operator	Description	Example	Is The Same As
=	Assign	x = y	x = y
+=	Add and assign	x += $	x = x + y
-=	Subtract and assign	x -= y	x = x - y
*=	Multiply and assign	x *= y	x = x * y
/=	Divide and assign quotient	x /= y	x = x / y
%=	Divide and assign modulus	x %= y	x = x % y

Code example of assignment operators

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Assignment Operators</title>
</head>
<body>
    <script>
    var x;    // Declaring Variable

    x = 10;
    document.write(x + "<br>"); // Prints: 10

    x = 20;
    x += 30;
    document.write(x + "<br>"); // Prints: 50

    x = 50;
    x -= 20;
    document.write(x + "<br>"); // Prints: 30

    x = 5;
    x *= 25;
    document.write(x + "<br>"); // Prints: 125

    x = 50;
    x /= 10;
    document.write(x + "<br>"); // Prints: 5

    x = 100;
    x %= 15;
    document.write(x); // Prints: 10
    </script>
</body>
</html>
```

JavaScript Logical Operators

The logical operators are typically used to combine conditional statements.

Operator	Name	Example	Result
&&	And	x && y	True if both x and y are true
||	Or	x || y	True if either x or y is true
!	Not	!x	True if x is not true

Code Example of logical operators:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Logical Operators</title>
</head>
```

```
<body>
   <script>
   var year = 2018;

   // Leap years are divisible by 400 or by 4 but not 100
   if((year % 400 == 0) || ((year % 100 != 0) && (year % 4 == 0))){
      document.write(year + " is a leap year.");
   } else{
      document.write(year + " is not a leap year.");
   }
   </script>
</body>
</html>
```

Q.9 What is Data Types in javaScript?

Ans:

A data type is a classification of the type of data that a variable or object can hold. Data type is an important factor in virtually all computer programming languages, including visual basic, C#, C/C++ and JavaScript.

Q.10 Numbers in javaScript.

Ans:

We can convert a string to an integer using the built-in parseInt() function. This takes the base for the conversion as an optional second argument, which you should always provide:

parseInt("123", 10)

123

parseInt("010", 10)

10

parseInt("010")

8

It happens because the parseInt() function decided to treat the string as octal due to the leading 0.

If you want to convert a binary number to an integer, just change the base:

parseInt("11", 2)

3

Q.11 Explain String in javaScript.

Ans: Strings in JavaScript are sequences of characters. More precisely, they're sequences of Unicode characters, with each character represented by a 16-bit number. If we want to represent a single character, we just need to use a string of length 1.

To find the length of a string, access its length property:

"hello".length

5 //length of given string “hello”

The strings are represented using objects and they have methods as well:

"hello, world".replace("hello", "goodbye") //by using replace() method

goodbye, world

"hello".toUpperCase() //by using toUpperCase() method

HELLO

Q.12 What is array in javaScript?

Ans: Arrays in JavaScript are actually a special type of object. They work similar to regular objects but they have one magic property called 'length'. The length of the array (size of the array) is always one more than the highest index in the array.

Code Example of JavaScript

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <title>Creating Arrays in JavaScript</title>
</head>
<body>
   <script>
   // Creating variables
   var colors = ["Red", "Green", "Blue"];
   var fruits = ["Apple", "Banana", "Mango", "Orange", "Papaya"];
   var cities = ["London", "Paris", "New York"];
   var person = ["John", "Wick", 32];

   // Printing variable values
   document.write(colors + "<br>");
   document.write(fruits + "<br>");
   document.write(cities + "<br>");
   document.write(person);
   </script>
</body>
</html>
```

Q.13 What is Function?

Ans: A function is a group of statements that perform specific tasks and can be kept and maintained separately form main program. Functions provide a way to create reusable code

packages which are more portable and easier to debug. Here are some advantages of using functions:

- **Functions reduces the repetition of code within a program** — Function allows you to extract commonly used block of code into a single component. Now you can perform the same task by calling this function wherever you want within your script without having to copy and paste the same block of code again and again.
- **Functions makes the code much easier to maintain** — Since a function created once can be used many times, so any changes made inside a function automatically implemented at all the places without touching the several files.
- **Functions makes it easier to eliminate the errors** — When the program is subdivided into functions, if any error occur you know exactly what function causing the error and where to find it. Therefore, fixing errors becomes much easier.

Code example of Function

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <title>JavaScript Define and Call a Function</title>
</head>
<body>
    <script>
    // Defining function
    function sayHello() {
        document.write("Hello, welcome to this website!");
    }

    // Calling function
    sayHello(); // Prints: Hello, welcome to this website!
    </script>
</body>
</html>
```

Some Important Programs

//Program to search for a text in a string and return the text if found using match()

```
<html>
<body>
<script>
var str="Honesty is the best policy";
document.write(str.match("policy")+"<br>");
document.write(str.match("Police")+"<br>");
document.write(str.match("pollicy")+"<br>");
document.write(str.match("policy")+"<br>");
</script>
</body>
</html>
```

Output
Policy
Null
Null
Policy

//Program to replace characters in a string using replace()

```
<html>
<body>
<p>click the button to replace the characters</p>
<p id="demo">hello prachi</p>
<button onclick="myFunction()">try it</button>
<script>
function myFunction()
{
var str=document.getElementById("demo").innerHTML;
var n=str.replace("hello","good morning");
document.getElementById("demo").innerHTML=n;
}
</script>
</body>
</html>
```

//Program to round off any number using round()

```
<html>
<body>
<p id="demo">click the button to round the no.to its nearest integer.</p>
<button onclick="myFunction()">try it</button>
<script>
function myFunction()
{
document.getElementById("demo").innerHTML=Math.round(8.7896);
}
</script>
</body>
</html>
```

//Program to return a value random number between 0 and 1 using random()

```
<html>
<body>
<p id="demo">click the button to display a number</p>
<button onclick="myFunction()">try it</button>
<script>
function myFunction()
{
document.getElementById("demo").innerHTML=Math.random();
}
</script>
</body>
</html>
```

//Program to return the number with highest value of two specified numbers using max()

```
<html>
<body>
<p id="demo">Click the button to return the highest no. between 5 and 10.</p>
<button onclick="myFunction()">try it</button>
<script>
function myFunction()
{
document.getElementById("demo").innerHTML=Math.max(5,10);
}
</script>
</body>
</html>
```

//Program to return the number with the lowest value of two specified number using min().

```
<html>
<body>
<p id="demo">Click the button to return the lowest no. between 77 and 9.</p>
<button onclick="myFunction()">try it</button>
<script>
function myFunction()
{
document.getElementById("demo").innerHTML=Math.min(77,9);
}
</script>
</body>
</html>
```

//Program to join two arrays using concat().

```
<html>
<body>
<p id="demo">Click the button to join three arrays</p>
<button onclick="myFunction()">Click me</button>
<script>
function myFunction()
{
var fruits=["Apple","Orange"];
var vegetables=["Cucumber","Carrot","Raddish"];
var grains=["Wheat","Maize"];
var mix=fruits.concat(vegetables,grains);
var x=document.getElementById("demo");
x.innerHTML=mix;
}
</script>
</body>
</html>
```

//Program to reverse the order of the elements in the array.

```
<html>
<body>
<p id="demo">Click the button to reverse the order of the element in the array.</p>
<button onclick="myFunction()">Click me</button>
<script>
var alphabet=["z","k","j","h","e"];
function myFunction()
{
alphabet.reverse();
var x=document.getElementById("demo");
x.innerHTML=alphabet;
}
</script>
</body>
</html>
```

//Program to sort the array.

```
<html>
<body>
<p id="demo">Click the button to sort the array</p>
<button onclick="myFunction()">Click me</button>
<script>
function myFunction()
{
var fruits=["Banana","Orange","Apple","Mango"];
fruits.sort();
```

```
var x=document.getElementById("demo");
x.innerHTML=fruits;
}
</script>
</body>
</html>
```

Q.15 What is events in javaScript explain it.

Ans: An event is something that happens when user interact with the web page, such as when he clicked a link or button, entered text into an input box or textarea, made selection in a select box, pressed key on the keyboard, moved the mouse pointer, submits a form, etc. In some cases, the Browser itself can trigger the events, such as the page load and unload events.

Below are some of the most commonly used events:

- onLoad - occurs when a page loads in a browser
- onUnload - occurs just before the user exits a page
- onMouseOver - occurs when you point to an object
- onMouseOut - occurs when you point away from an object
- onSubmit - occurs when you submit a form
- onClick - occurs when an object is clicked

//Execution of javascript immediately after a page has been loaded.

Some important events program

//Execution of javascript immediately after a page has been loaded.

```
<html>
<head>
<script>
function myFunction()
{
confirm("Welcome to the loaded browser");
}
</script>
</head>
<body onload="myFunction()">
<h1>Event handling!</h1>
</body>
</html>
```

//Execute a javascript when a button is clicked.

```
<html>
<head>
<script>
function myFunction()
{
```

```
document.getElementById("demo").innerHTML="Hello World";
}
</script>
</head>
<body>
<p>Click the button</p>
<button onclick="myFunction()">Click me</button>
<p id="demo"></p>
</body>
</html>
```