Unit 11 JAVA SCRIPT Programming – 1

Structure:

11.1 Introduction

Most of the internet applications are built using many interoperable languages; hence HTML alone will not be sufficient in making these web applications interactive. HTML is static, so you cannot have loops, variables, and interactive functionality. But HTML being a static language, it can be easily incorporated with different languages to develop a complex page. In previous unit, you have learnt how to develop a web page using PHP functions. You also became familiar with Object oriented features of PHP and an excellent backend database named MYSQL.

In this unit you will learn one more scripting language that is JavaScript. JavaScript is one of the most powerful languages used in web development. This unit contain introduction to JavaScript, basic programming concepts like variable declaration, data types, operators, looping statements, functions and JavaScript objects.

Objectives:

After reading this unit, you should be able to learn:

- describe utility of JavaScript
- differentiate between client and server side JavaScript
- define operators of JavaScript
- list out different loop statements
- describe the calling of functions.
- define various objects of JavaScript

11.1.1 Introduction to Java script Programming

Java script is a programming language with direct support to object oriented methodologies. We can use JavaScript to perform a variety of functions on your website. Some of the functions are just functional, while the others are greater in nature, providing interactivity for our website visitors.

- **JavaScript gives HTML designers a programming tool –** HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax. Almost anyone can put small "script" of code into their HTML pages.
- **JavaScript can react to events –** A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.
- **JavaScript can read and write HTML elements –** A JavaScript can read and change the content of an HTML element.
- **JavaScript can be used to validate data –** A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing.
- **JavaScript can be used to detect the visitor's browser –** A JavaScript can be used to detect the visitor's browser, and depending onthe browser-load another page specifically designed for that browser.
- **JavaScript can be used to create cookies –** A JavaScript can be used to store and retrieve information on the visitor's computer.

11.1.2 Utility of JavaScript

We are building more and more complex web applications these days, and high interactivity either requires Flash (plugins) or scripting. JavaScript is disputably the best way to go, as it is a web standard, it is supported

natively across browsers (more or less some things differ across browsers), and it is compatible with other open web standards.

- JavaScript is very easy to implement. All you need to do is put your code in the HTML document and tell the browser that it is JavaScript.
- JavaScript works on web users' computers even when they are offline.
- JavaScript can help fix browser problems or patch holes in browser support for example fixing CSS layout issues in certain browsers.
- JavaScript allows you to create highly responsive interfaces that improve the user experience and provide dynamic functionality, without having to wait for the server to react and show another page.
- JavaScript can load content into the document if and when the user needs it, without reloading the entire page.

11.1.3 Evolution of the Java Script language

Around 1992, a company called Nombas began developing an embedded scripting language called C-minus-minus (Cmm for short). The idea behind Cmm was simple: a scripting language powerful enough to replace macros, but still similar enough to C (and C++) that developers could learn it quickly.

As World Wide Web gained popularity, a gradual demand for client-side scripting languages developed. At the time, most Internet users were connecting over a 28.8 kbps modem even though Web pages were growing in size and complexity. Adding to users' pain was the large number of round-trips to the server required for simple form validation. Imagine filling out a form, clicking the Submit button, waiting 30 seconds for processing, and then being encountered with a message telling you that you forgot to complete a required field. Netscape, at that time on the cutting edge of technological innovation, began seriously considering the development of a client-side scripting language to handle simple processing.

Brendan Eich, who worked for Netscape at the time, began developing a scripting language called LiveScript for the upcoming release of Netscape Navigator 2.0 in 1995, with the intention of using it both in the browser and on the server (where it was to be called LiveWire). Netscape entered into a development alliance with Sun Microsystems to complete the implementation of LiveScript in time for release.

Just before Netscape Navigator 2.0 was officially released, Netscape changed the name to JavaScript in order to capitalize on Java as a new Internet buzzword.

Java Script Versions and Browser Support

The first browser to support JavaScript was Netscape Navigator 2.0 beta version. Below is the list of JavaScript versions and supported browser versions.

1. JavaScript 1.0:- supported by Netscape navigator 2.0 and Internet explorer 3.0 (1996)
2. JavaScript 1.1:- supported by Netscape navigator 3.0 (1996)
3. JavaScript 1.2:- supported by Netscape navigator 4.0 (1997)
4. JavaScript 1.3:- supported by Netscape navigator 4.06 and Internet explorer 4.0 (1998)
5. JavaScript 1.4:- supported by Netscape navigator server
6. JavaScript 1.5:- supported by Netscape navigator 6.0 and Mozilla Firefox 1.0 and IE 5.5 to IE 8 (2000)
7. JavaScript 1.6:- supported by only by Mozilla Firefox 1.5 (2005)
8. JavaScript 1.7:- supported by Mozilla Firefox 2.0 (2006)
9. JavaScript 1.8.1:- supported by Mozilla Firefox 3.6 (2009)
10. JavaScript1.8.5:- supported by Mozilla Firefox 4.0 and IE9 (2010)

Client-Side JavaScript vs. Server Side Java Script

There are two major types of JavaScript namely:

Client-Side JavaScript (CSJS):

Client side Java script comprises the basic language and predefined objects which are relevant to running Java script in a browser. The client side java script is embedded directly by in the HTML pages. This script is interpreted by the browser at run time. It is JavaScript that enables the web pages on browsers to run active online content.

Server-Side JavaScript (SSJS)

Server side java script also resembles like client side java script. It has relevant java script which is to run in a server. The server side java scripts are deployed only after compilation. It is JavaScript that enables back-end access to databases, file systems, and servers.

Self Assessment Questions

1. The first browser to support JavaScript was _____
2. Which of the following JavaScript that enables back-end access to databases file systems, and servers?
   a) client side JavaScript
   b) Server side JavaScript
   c) Core JavaScript

11.2 Programming with Java Script

JavaScript can be used in different ways within an HTML file.

1. Header scripts

JavaScript is embedded between <SCRIPT> tags within the <HEAD> tag and is used for initializing variables and or create functions that can be called from anywhere of the entire script. JavaScript in header is executed once when the page is loaded.

2. Body scripts

In body script, JavaScript is embedded between <SCRIPT> tags within the <BODY> tag. JavaScript within Body Tags are used for handling various events.

3. Called directly when certain events occur

In this type, JavaScript are called when certain events occur. Example of events include onLoad, onClick, onError, onSelect, onSubmit, onChange etc.

11.2.1 Variable declarations

A variable is a data item that can be manipulated at any time. In JavaScript the variables are same as the other programming language variables. If we give a value to variable then this is called assignment.

Example

a=10

b=20 +a

aand bare two variables. Name given to variable called variable name. In JavaScript there are some rules for naming variables.

1. A variable name should begin either with a digit, underscore or a letter.
2. Names in JavaScript are case-sensitive.

3. As with all programming languages, reserved keywords in JavaScript can't be used as variable names.

Data Types

JavaScript data types can be classified into mainly four types

1. **Boolean Data type**: This data type consists of two logical values: Trueand False.
   Example: var snowing =false
2. **String data type:** In this data type JavaScript, information is written indouble quotes.
   Example: var name= "Sikkim "
3. **Null data type:** The user can use this data type when he/she feels itdoes not want to initialize the value of particular variable.
   Example: var age=null;
4. **Number data type:** This data type is just the opposite of string data type.The value is declared without double quotes.

Example: var n1= 30
Var rate=300

Example:

```
<HTML>
<HEAD>
  <TITLE>My First JavaScript</TITLE>
</HEAD>
<BODY>
  <H1 ALIGN=CENTER>
  <BR /><BR /><BR />
  <SCRIPT LANGUAGE="JavaScript">
  Document. Write ("My First JavaScript")
  </SCRIPT>
  </H1>
</BODY>
</HTML>
```

The following figure 11.1 depicts the output of the above program**.**

NOTE: Document, Write, predefined code segment, which directly displays the text followed by it on the web browser.

Output

Figure 11.1: Example of JavaScript program output

11.2.2 Operators and Statements

An operator, as the name suggest, performs some action. JavaScript language supports following type of operators.

- Arithmetic Operators
- Logical Operators
- Comparison Operators
- Assignment Operators
- String operators
- Conditional Operators

Arithmetic Operators

Arithmetic operator performs basic mathematical operation

Table 11.1 arithmetic operators

Operator	Description
+	Performs addition of two operands
-	Performs subtraction of two operands
*	Performs multiplication of two operands
/	Performs division of two operands
%	Performs modulus operation and gives reminder as result.

Logical operators

Logical operator works with logical values and returns a Boolean value. i.e. true or false.

Table 11.2: Logical operators

Operators	Description
&&(Logical AND)	If both the operands are true then condition becomes true
||(Logical OR)	If both the operands are false then condition becomes false
!(Logical Not)	If condition is true then logical not operator will make false. i.e. used to reverse the logical state of its operand.

Comparison operators

Compares the values of the operands and returns a Boolean value.

Table 11.3: Comparison operators

Operator	Description
==	It checks the value of two operands are equal or not, if yes then it returns true
!=	It checks the value of two operands are equal or not, if values are not equal then it returns true
>	It checks the value of left operand is greater than the value of right operand, if yes then it returns true
<	It checks the value of left operand is less than the value of right operand, if yes then it returns true
>=	It checks the value of left operand is greater than or equal to the value of right operand, if yes then it returns true
<=	It checks the value of left operand is less than or equal to the value of right operand, if yes then it returns true

Bitwise operators

Bitwise operator performs operation on the bit representation (zeros and ones) of a value and returns the result as a numeric value.

Table 11.4: Bitwise Operators

Operator	Description
&(Bitwise AND)	Returns value one in each bit position if both corresponding Bit are value one
|(Bitwise OR)	Returns value one in each bit position if one or both corresponding Bits are value one
^(Bitwise XOR)	Returns value one in each bit position if one, but not both, of the corresponding Bits are value one
~(Bitwise NOT)	One become zero; Zero become one i.e. inverts the bits of operands
<<(Left shift)	It moves all bits in its first operand to the left by the number of places specified. In the second operand.
>>(Right shift)	It moves all bits in its first operand to the right by the number of places specified In the second operand.
>>>(Bitwise shift right with zero operator)	This operator is just like the >> operator, except that the bits shifted in on the left are always zero.

Assignment operator

Assignment operator assigns a value to the left operand based on the value of the right operand.

Table 11.5: Assignment operators

Operators	Description
=	Assigns values from right side Operands to left side operand
+=	It adds right operand to the left operand and Assign the result to left operand
-+	It subtracts right operand from the left operand and Assign the result to left operand
*=	It multiplies right operand with the left operand and Assign the result to left operand
/=	It divides right operand with the left operand and Assign the result to left operand
%=	It takes modulus using two operands and assign The result to left operand

String operator

String operator takes a value as their operands and returns a string as their value.

Concatenation is a very common string operation, and works as shown below

Operand1 + Opernad2

A sting is actually an object, so it can be said that the string operator operates on string object. It joins them together as in

"Nature "+ "is"+ "beautiful"

The string operator can operate on more than two operands. An expression consisting of numerous string operators evaluates to a single string

Document. Write ("I have" + two + "cookies")

Conditional Operators

A condition operator works based on certain condition is true or false, here the condition is an expression. If condition is true it returns value one and it executes true expression, if condition is false, it executes false block expression.

Syntax:

[Condition]? [True expression]; [false expression]

11.2.3 Conditional statements

JavaScript enables us to manage the flow of control, to choose which statements are executed and which are not, and also determines the number of times a set of statements is performed.

If statement

The if statement is a conditional statement. The simplest form of flow of control structure is if statement.

Syntax:

```
If(condition)
Statement;
```

Example:

```
Var Str_Day= "Monday "
If(Str_Day== "Monday ")
```

Document.write("Today is Monday ");

Else statement

The else statement can be used in conjunction with the if statement, but it cannot be used on its own.

Syntax

If(condition)

Statement

Else

statement;

Example

Var Str_Day= "Friday "

If(Str_Day== "Monday ")

Document.write("Today is Monday ");

Else

Document.write("Today is other than Monday");

Here the condition is false then the statements of else part will be performed.

Switch

The switch statement allows us to select an option from a variable. The variable can be a string or an integer and the content of the variable are matched against a list of values known as cases within the switch statement. If a case is found to be true, then the statement associated with that case are performed

Syntax:

Switch (variable)

Case value1:statement;

Case value2:statement;

Case value N :statement;

Default :statement;

A default case can be included that is performed if none of those other cases are found to be true.

Example

```
Var Str_Day= "Friday "
Switch (Str_Day")
Week<br/>
Case "   Monday "      Document.write("Today is Monday "); <br/>
Case "   Tuesday "      Document.write("Today is Tuesday "); <br/>
Case" Wednesday"  Document.write("Today is Wednesday"); <br/>
Default : Document.write("Today is Weekend !"); <br/>
```

Self Assessment Questions

3. Which of the following data type, user can use when he/she feels it does not want to initialize the value of particular variable.
   a) String b) number c) null d) Boolean
4. _____performs operation on the bit representation.

11.3 Looping Statements ,Functions and Java script Objects

11.3.1 Types of Loops

A common need when you develop in JavaScript is the ability to loop through a task a number of times or until a specified condition changes.The types of loops are:

- For loop
- While loop
- Do while loop
- For-in loop

➢ **For loop**

For loop tends to be the most commonly used JavaScript looping construct. It lets you traverse, a number of items quickly and easily.

Example: For (var i=2; i<=10; i++)

{

Document. Write (“ even numbers” +I);

}

➢ **While loop**

A while loop is simple pretest loop comprised of a pretest condition and loop

body.

Before loop body executed, first it test the condition. If condition is true, then loop body is executed, otherwise loop body is skipped.

Example: While (item>10){

Document. Write(“ even numbers” +Item);

Item+2;

}

- **Do while loop**

A do-while loop is the only post-test loop. The loop body executed at least once, and the post-test condition determines whether the loop should be executed again.

Example:do{

Document.write(“ even numbers” +Item);

Item+2;

} While (item>10)

- **For-in loop:**

This loop has a very special purpose. It enumerates named properties of any object.

Syntax

For (var prop in object) {

//loop body

}

The following **example** specifies window object property of web browser

Var i=0; a= “ ”;

For(property *in* window)

{

a+ = Property +“ …”;

}

11.3.2 Implementing Functions

A function is defined as a set of actions. Functions are invoked by event handlers or by statements elsewhere in the script. Good functions are designed for reuse in other file.

A function is capable of returning a value to the statement that invoked it. When a function does return a value, the calling statement treats the function call like any expression persisting in the returned value right where the function call is made.

11.3.3 Defining Functions

A function is defined with the keyword function, usually followed by the name of the function, and then by parentheses that contain optional arguments or parameters to be used.

Function function_name(argument1, argument2…argument N){

//statements

}

One of the difference between JavaScript and other languages is that in JavaScript, you don't need to specify the number of arguments being passed into a function, nor do the number of arguments being passed in need to match those that are defined in function definition. When invoked, the function is given an object named arguments. An argument holds the arguments sent into the function, which can be helpful when you don't know the number of arguments being sent in.

Example: function myfunction ()

{

Var firstarg= argument [0];

Var secondarg= argument[1];

}

Return values

When a function finishes executing its code, it can return a value to the caller by using the return keyword.

Example:

Function mulnum(x) {

Return x*2;

}

Var num=10;

Var result=mulnum(num);

The easiest way to think about methods is that they are functions defined as part of an object. You can use method of an object by using dot operator (.). Function such as the alert () functions are actually just methods of window object, and then can be called as window. Alert () or just alert ().

The alert ( ) method of window object is used to display an alert dialog box with a message and an OK button.

11.3.4 Calling Functions

In most cases, the calling statement and the function exchange information. That is the calling statement sends some information that the function uses or the function returns some information that the calling statement uses.

Calling function from a link: a function can be called directly from a link, by using the JavaScript psedo protocol. The JavaScript protocol and functioncall are placed within quotes and assigned to the href attribute of the <a> tag.

Example:

```
<Html>
<Head>
<script language=JavaScript>
Function. Greeting () {
Document.bgcolor =“light blue”;
Alert(“greeting to you!”);
</script>
</head>
<body>
<a href=”JavaScript. Greeting ()”> click here
</a>
</body>
</html>
```

Calling a function from an Event: an event is triggered when a user performs some action, like clicking on a button or moving his mouse over a link.

Example

```
<Html>
<Head>
<script language=JavaScript>
Function. Greeting (){
```

```
Document.bgcolor =”light blue”;
Alert(“greeting to you!”);
</script>
</head>
<body>
<form>
<input type=”button” onclick=Greeting ();”>
</form>
</body>
</html>
```

11.3.5 Java Script Objects

Java Script is an Object Oriented Programming (OOP) language. An OOP language allows us to define our own objects and make our own variable types. But here in this unit we will discuss built-in objects. JavaScript supports programming with objects. Objects are a way of organizing the variables. The different screen elements such as Web pages, forms, text boxes, images, and buttons are treated as objects.

Every object has its own property and methods. Property defines the characteristics of an object whereas Methods are the actions that the object can perform or that can be performed on the object.

There are various types of objects in java script that we are going to discuss below:

1. Naming Objects

Objects are organized in a hierarchy. To refer to an object use: objectName

- To refer to a property of an object use: objectName.propertyName
- To refer to a method of an object use: objectName.methodName()

2. Built-In Objects

Some of the built-in language objects of JavaScript offer more advanced operations such as:

- **Math object –** provides for math calculations like round(), random(),max(), min() etc.
  Syntax: var x=Math.PI;
  var y=Math.sqrt(16);

- **Date object –** provides date and time information
  There are four ways of instantiating a date object:
  i) new Date()
  ii) new Date(milliseconds)
  iii) new Date(dateString)
  iv) new Date(year, month, day, hours, minutes, seconds, milliseconds)
- **String object –** The String object is used to manipulate a stored piece of text. Here are two ways to define new strings.one is without using the name of string and another one is using the name of string.
  var myText1 = "This is my Manipal ";
  var myText2 = new String("This is my Manipal.");

3. Document Object

The Document object represents the Web page that is loaded in the browser window, and the content displayed on that page, including text and form elements. Since in HTML there is no DOCUMENT element, the document object in JavaScript contains a few different attributes of the BODY element and it also includes most of the elements contained within the body of the page.

For example if we want to change the background color of the document, just use its bgColor attribute. We can define colors using hexadecimal code or standard names.

```
document.bgColor = color;
document.bgColor = "black";
or
document.bgColor = "#000000";
```

Document Methods

We can use the methods of the document object to work on a Web page. Here are the most common document methods:

- write() - write a string to the Web page
- open() - opens a new document
- close() - closes the document

4. Window object

The window is a top level object. It pertains to a window on the computer screen, such as a browser window that contains a document. The window object represents an open window in a browser. If a document contain frames (<frame> or <iframe> tags), the browser creates one window object for the HTML document, and one additional window object for each frame.

Property of window object

There are many properties of window object but here we are going to discuss some important or frequently used properties.

a) Closed Property

This property is used to return a Boolean value that determines if a window has been closed. If it has, the value returned is true.

Syntax: window.closed

b) defaultStatus Property

This property is used to define the default message displayed in a window's status bar.

Syntax: window.defaultStatus = "Message"

c) Document Property

This property's value is the document object contained within the window. See Document object.

Syntax: window.document

d) Frames Property

This property is an array containing references to all the named child frames in the current window.

Syntax: window.frames = "frameID"

e) History Property

This property's value is the window's History object, containing details of the URL's visited from within that window. See History object.

Syntax: window.history

f) Location Property

This property contains details of the current URL of the window and its value is always the Location object for that window.

Syntax: window.location

5. History object

The History object contains an array of previously visited URLs by the visitor. To simulate the browser's back button, for example, we can use the History object:

```
<a href="javascript:history.go(-1)">Go back</a>
```

Using JavaScript to work with a web page document

The document is the web page itself, and with Javascript it is possible to modify the title of the web page:

```
document.title = "A Dynamic Web Page";
```

The contents of that web page can also be created by using Javascriptcode; for instance the web page can be written to by using the document's write method:

```
//Create a heading
document.write ("<h1 id=h1>Hello Students </h1>");
//Create a paragraph with text in it
document.write ("<p id=p1>This is the distance education.</p>");
//Create an empty paragraph
document.write ("<p id=p2></p>");
```

This code actually creates some additional objects (h1, p1 and p2) and they can now be accessed using Javascript code.

Accessing Objects within a Web Page Document

Each object that's created must be given a unique ID, and once that has been done then each object can be accessed using the document's **getElementById** method

for example, the code above creates an empty paragraph (p2) - the contents of that paragraph can be changed by usingthe **getElementById** method to access the paragraph and then updating it's i**nnerHTML** property:

```
var p2 = document. GetElementById ('p2');
p2.innerHTML = "text for empty paragraph";
```

Dealing with Events in a Dynamic Web Page

Each object in a web page has a number of events associated with it - these events are things such as:

- **onmouseover** – this event occurs when the mouse pointer is placed overan object
- **onmouseout** – the mouse pointer is moved away from an object
- **onclick** – the user has clicked on an object

These events can then be used to run JavaScript functions - in this example the text in a paragraph changes colour and is displayed in a second paragraph:

Example using events:

```
<p id=p3
Onmouseover="overme('p3')"
Onmouseout="away ('p3')">
An example of a web page event</p>
<p id=p4></p>
<script>
var p4 = document. GetElementById ('p4');
function oversee(me) {
var object = document. GetElementById(me);
object.style.color = "blue";
p4.innerHTML = "Your mouse is over "' + object. InnerHTML + '"";
}
function away(me) {
var object = document.getElementById(me);
object.style.color = "black";
p4.innerHTML = "";
}
</script>
```

In the above example there are two paragraphs with id p3 and p4 respectively. In the first paragraph there is a text **"An example of a web page event"** and in the second paragraph text is **"Your mouse is over"**. After execution of this program the text in a paragraph changes colour and is displayed in a second paragraph. When the mouse will be on paragraph

then the colour of the paragraph will be blue otherwise when it will be away the colour of the paragraph will be black.

Self Assessment Questions

5. When a function finishes executing its code, it can return a value to the caller by using the_____keyword.
6. The_____is used to manipulate a stored piece of text.
7. The_____contains an array of previously visited URLs by the visitor.

11.4 Summary

- Around 1992, a company called Nombas began developing anembedded scripting language called C-minus-minus (Cmm for short). The first browser to support JavaScript was Netscape Navigator 2.0 betaversion.
- The most widely used lightweight, interpreted, and compiled programming language is JavaScript. Both client-side and server-side developments are compatible with it.
- In JavaScript, variables are storage spaces for reused data. It serves as a program's fundamental unit of storage.
- To carry out repeated operations based on a condition, loops are used. A loop will continues until the defined condition becomes false. the sorts of loops are for loop, while loop, do-while loop and for-in loop.

- A function is defined as a set of actions. Functions are invoked by event handlers or by statements elsewhere in the script.
- Function can be called by using a) function can be called directly from a link, by using the JavaScript psedoprotocol b) Function can be called from an event.
- An event is triggered when a user performs some action, like clicking on a button or moving his mouse over a link.

11.5 Terminal Questions

1. Differentiate between Client side JavaScript and server side JavaScript?
2. Explain different types of operators used in JavaScript?
3. Explain different types of conditional and loop statement?
4. What is function? How many ways we can call a function in JavaScript?
5. Briefly explain the types of Objects in JavaScript's?

11.6 Answers

Self Assessment Questions

1. Netscape Navigator 2.0
2. B) server side script
3. C) NULL
4. Bitwise operator

5. Return
6. String Object
7. History Object

Terminal Questions

1. The client side java script is interpreted by the browser at run time. The server side java scripts are deployed only after compilation. For more details refer section 11.1.1.
2. JavaScript provides different types of operators. For more details refer section 11.2.2.
3. JavaScript provides different types of control and loop statements. For more details refer section 11.3..
4. A function is defined as a set of actions. Function can be called in two ways, first way is calling a function from link and second way is calling a function by event. For more details refer section 11.3.4.
5. Objects are a way of organizing the variables. There are various types of objects in Java script. For more details refer section 11.3.5.

11.7 References

- Nichaolas C. Zakas (2010), *High performance Javascrip*t. O'reilly
- Jennifer Niederest (2003). Learning web design: A beginner's guide to Html and JavaScript.
- Paul wilfon, Jermy McPeak (2009). Beginning JavaScript. Ninth edition. Wrox press.