Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)
18th April 2022
(Offline) Lecture 11

Padre Conceicao College of Engineering

Verna Goa 403722 India

Web Technology

RC 2019-20

Unit 2

UNIT 2

Topic	Subtopics
JavaScript:	Introduction to client side scripting, documents, forms, statements, comments, variables, operators, conditional statements, loops, events, objects, functions.

UNIT 2 :

Sr.No	Title
1	Introduction to client side scripting
2	documents
3	forms
4	statements
5	comments
6	variables
7	operators
8	conditional statements
9	loops
10	events
11	objects
12	functions

What is JavaScript?

- JavaScript is a text-based programming language
- JavaScript is used both on the client-side and server-side that make web pages interactive.
- Javascript is used by programmers across the world to create dynamic and interactive web content like applications and browsers.
- JavaScript is so popular that it's the most used programming language in the world, used as a client-side programming language by 97.0% of all websites.

Java language v/s JavaScript Language

Sr.No	Java language	JavaScript Language
1	It is a Programming language.	It is a scripting language.
2	It is one of the complex languages to learn.	It one of the easy languages to learn.
3	It requires a large amount of memory.	It does not require large amount of memory.
4	Java is stored on the host machine as the "Byte" code.	JavaScript is stored on the Host machine (client Machine) as the "source" text.
5	Java is a pure Object Oriented Programming Language.	JavaScript is Object-Based Language.
6	Java is a Standalone language.	JavaScript is not a standalone language, as it needs to be integrated into an HTML program for execution.
7	Java program should be compiled before execution.	JavaScript needs to be integrated into the HTML program for the execution.
8	The web-browser is not required to run java programs.	The web-browser is essential to run the JavaScript programs.

Java language v/s JavaScript Language

Sr.No	Java language	JavaScript Language
9	Java is a strongly typed language, which means that the user has to decide the data type of the variable before declaring and using it. Example "int a", the variable "a" can store the value of integer type only.	JavaScript is a loosely typed language, which means that the user does not have to worry about the data-type of the variable before and after the declaration. Example "var a", the "a" variable can store the value of any data-type.
10	In Java, by utilizing the Multi-threading, users can perform complicated tasks.	In JavaScript, user is not able to perform complicated tasks.
11	Java programming language was developed by the "Sun Microsystems."	JavaScript programming language was developed by the "Netscape."
12	In Java programming language, programs are saved with the ".java" extension.	On the other hand, programs in JavaScript are saved with the ".js" extension.

Advantages of JavaScript

1. Gives the ability to create rich interfaces.
2. Popularity: JavaScript is used everywhere on the web
3. Simplicity: JavaScript is relatively simple to learn and implement.
4. Server Load: Being client-side reduces the demand on the website server.
5. Interoperability: JavaScript has the ability to support all modern browsers and produce an equivalent result.
6. Speed: Client-side JavaScript is very fast because it can be run immediately within the client-side browser. Unless outside resources are required, JavaScript is unhindered by network calls to a backend server.
7. Less Bandwidth: Regardless of where you host JavaScript, it always gets executed on client environment to save lots of a bandwidth and make execution process fast.

Advantages of JavaScript

8.Global companies support community development by creating projects that are important. An example is Google (created Angular framework) or Facebook (created the React.js framework).

9.In JavaScript, XMLHttpRequest is an important object that was designed by Microsoft. The object call made by XMLHttpRequest as a asynchronous HTTP request to the server to transfer the data to both sides without reloading the page

10.There are some ways to use JavaScript through Node.js servers. It is possible to develop a whole JavaScript app from front to back using only JavaScript.

Disadvantages of JavaScript

1. Client-Side Security: Because the code executes on the users' computer, in some cases it can be exploited for malicious purposes. This is one reason some people choose to disable Javascript.
2. Browser Support: JavaScript is sometimes interpreted differently by different browsers. This makes it somewhat difficult to read and write cross-browser code.
3. No matter what proportion fast JavaScript interpret, JavaScript DOM (Document Object Model) is slow and can be never fast rendering with HTML.
4. If the error occurs in the JavaScript, it can stop to render the whole website. Browsers are extremely tolerant of JavaScript errors.
5. This continuous conversions takes longer in conversion of number to an integer. This increases the time needed to run the script and reduces its speed

Disadvantages of JavaScript

6.This may be difficult to develop large applications, although you’ll also use the TypeScript overlay.

7.This applies to larger front-end projects. The configuration is often a tedious task to the amount of tools that require to figure together to make an environment for such a project. This is often directly associated with the library’s operation.

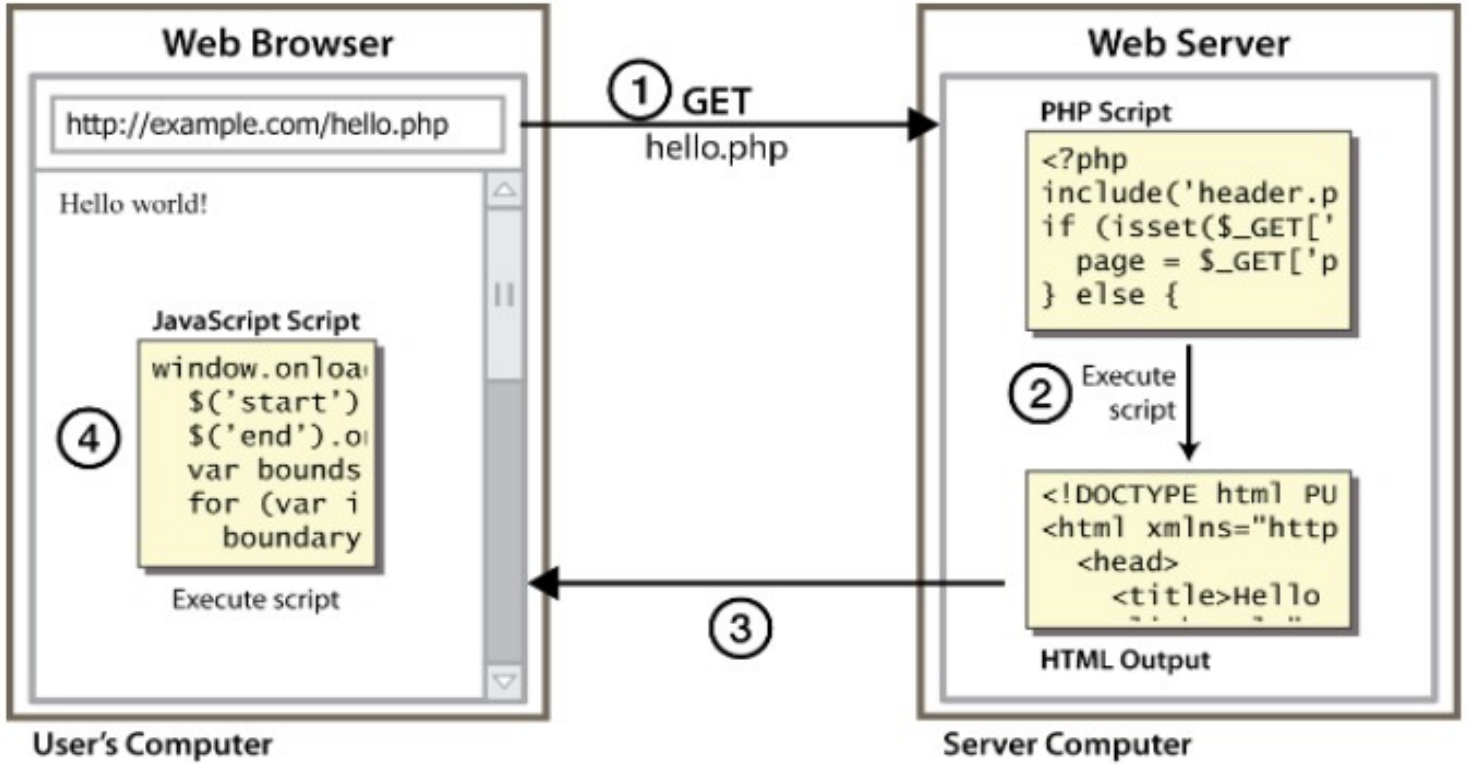
8.The main problem or disadvantage in JavaScript is that the code is always visible to everyone anyone can view JavaScript code.

9.Though some HTML editors support debugging, it’s not as efficient as other editors like C/C++ editors . Hence difficult for the developer to detect the matter

Introduction to client side scripting

- Client-side scripting generally refers to the class of computer programs on the web that are executed client-side, by the user’s web browser, instead of server-side (on the web server).
- This type of computer programming is an important part of the Dynamic HTML (DHTML) concept, enabling web pages to be scripted; that is, to have different and changing content depending on user input, environmental conditions (such as the time of day), or other variables.
- These kinds of scripts are small programs which are downloaded , compiled and run by the browser.
- JavaScript is an important client-side scripting language and widely used in dynamic websites.

client side scripting



Client Side scripting v/s Server Side scripting

Sr.No	Client-Side scripting	Server-Side scripting
1	It runs on the user’s computer.	It runs on the webserver.
2	Source code is visible to the user	Source code is not visible to the user because its output of server-sideside is an HTML page.
3	It does not provide security for data.	It provides more security for data.
4	It usually depends on the browser and its version.	In this any server-side technology can be used and it does not depend on the client.
5	It is a technique used in web development in which scripts run on the client’s browser.	It is a technique that uses scripts on the webserver to produce a response that is customized for each client’s request.
6	There are many advantages linked with this like faster response times, a more interactive application.	The primary advantage is its ability to highly customize, response requirements, access rights based on user.
7	HTML, CSS, and javascript are	PHP, Python, Java, Ruby are used.

Internal and External JavaScript

- You can use JavaScript code in two ways.

1. Internal: You can either include the JavaScript code internally within your HTML document itself
2. External: You can keep the JavaScript code in a separate external file and then point to that file from your HTML document.

Example of Internal JavaScript?

day.htm

```
<html>
<head>
  <title>My First JavaScript code</title>
  <script type="text/javascript">
    // Create a Date Object
    var day = new Date();
    // Use getDay function to obtain todays Day.
    // getDay() method returns the day of the week as a number like 0 for Sunday, 1 for Monday,….., 5
    // This value is stored in today variable
    var today = day.getDay();
    // To get the name of the day as Sunday, Monday or Saturday, we have created an array named weekday and stored the values
    var weekday = new Array(7);
    weekday[0]="Sunday";
    weekday[1]="Monday";
    weekday[2]="Tuesday";
    weekday[3]="Wednesday";
    weekday[4]="Thursday";
    weekday[5]="Friday";
    weekday[6]="Saturday";
    // weekday[today] will return the day of the week as we want
    document.write("Today is " + weekday[today] + ".");
  </script>
</head>
<body>
</body>
</html>
```

output

Today is Wednesday.

What is External JavaScript?

- You decide to display the current date and time in all your web pages. If you wrote the code and copied in all your web pages (eg 100 pages),but later, you want to change the format in which the date or time is displayed.
- In this case, you will have to make changes in all the 100 web pages. This will be a very time consuming and difficult task.
- Therefore, save the JavaScript code in a new file with the extension .js. It is assumed that the .js file and all your web pages are in the same folder. If the external.js file is in a different folder, you need to specify the full path to your file in the src attribute.
- Then, add a line of code in all your web pages to point to your .js file like this <script type="text/javascript" src="currentdetails.js">

How to Link HTML file to an external JavaScript file

- script tag should be placed in HTML page's head
- script code is stored in a separate .js file
- JS code can be placed directly in the HTML file's body or head (like CSS)
  - but this is bad style (should separate content, presentation, and behavior)
  - Syntax

```
<script src="filename" type="text/javascript"></script>
```

  - Example

```
<script src="example.js" type="text/javascript"></script>
```

currentdetails.htm and Output

currentdetails.htm - Notepad

File Edit Format View Help

```
<html>
        <head>
           <title>My External JavaScript Code!!!</title>
           <script type="text/javascript" src="currentdetails.js">
           </script>
        </head>
        <body>
        </body>
</html>
```

output

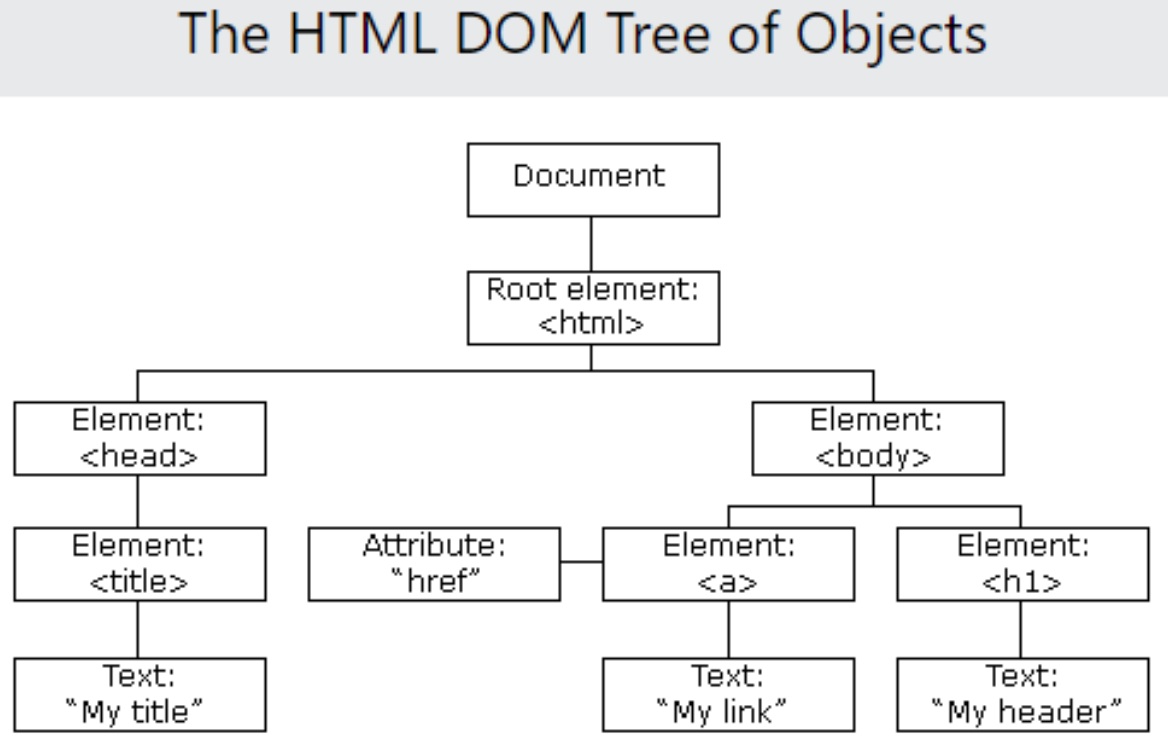
Today is 13 - April - 2022.
Current time is 01:11:52 PM.

When to Use Internal and External JavaScript Code?

- If you have only a few lines of code that is specific to a particular webpage, then it is better to keep your JavaScript code internally within your HTML document.
- On the other hand, if your JavaScript code is used in many web pages, then you should consider keeping your code in a separate file.
- In that case, if you wish to make some changes to your code, you just have to change only one file which makes code maintenance easy.
- If your code is too long, then also it is better to keep it in a separate file. This helps in easy debugging.

Document **O**bject **M**odel (DOM)

- When a web page is loaded, the browser creates a **DOM** of the page. The **HTML DOM** model is constructed as a tree of **Objects**:



What is the DOM?

- The DOM is a W3C (World Wide Web Consortium) standard. The DOM defines a standard for accessing documents: "The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."
- The W3C DOM standard is separated into 3 different parts:

1. Core DOM - standard model for all document types
2. XML DOM - standard model for XML documents
3. HTML DOM - standard model for HTML documents

What is the HTML DOM?

- The HTML DOM is a standard object model and programming interface for HTML. It defines:

1. The HTML elements as objects
2. The properties of all HTML elements
3. The methods to access all HTML elements
4. The events for all HTML elements

- In other words: The HTML DOM is a standard for how to get, change, add, or delete HTML elements.

What can JavaScript create with DOM?

With the object model, JavaScript gets all the power it needs to create dynamic HTML:

1. JavaScript can change all the HTML elements in the page
2. JavaScript can change all the HTML attributes in the page
3. JavaScript can change all the CSS styles in the page
4. JavaScript can remove existing HTML elements and attributes
5. JavaScript can add new HTML elements and attributes
6. JavaScript can react to all existing HTML events in the page
7. JavaScript can create new HTML events in the page

JavaScript Forms

- JavaScript Form Validation: HTML form validation can be done by JavaScript.
- Example: If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted

HTML form validation by JavaScript

jsform.htm

jsform - Notepad

File Edit Format View Help

```
<html>
<head>
<script>
function validateForm() {
  let x = document.forms["myForm"]["fname"].value;
  if (x == "") {
    alert("Name must be filled out");
    return false;
  }}
</script>
</head>
<body>
<h2>JavaScript Validation</h2>
<form name="myForm" action="/action_page.php" onsubmit="return validateForm()" method="post">
  Name: <input type="text" name="fname">
  <input type="submit" value="Submit">
</form>
</body>
</html>
```

output

JavaScript Validation

Name: Submit

This page says

Name must be filled out

OK

JavaScript Can Validate Numeric Input

numeric.htm

numeric - Notepad

File Edit Format View Help

```
<html>
<body>
<h2>JavaScript Validation</h2>
<p>Please 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() {
  // Get the value of the input field with id="numb"
  let x = document.getElementById("numb").value;
  // If x is Not a Number or less than one or greater than 10
  let text;
  if (isNaN(x) || x < 1 || x > 10) {
    text = "Input not valid";
  } else {
    text = "Input OK";
  }
  document.getElementById("demo").innerHTML = text;
}
</script>
</body>
</html>
```

output

JavaScript Validation

Please input a number between 1 and 10:

11 Submit

Input not valid

Automatic HTML Form Validation

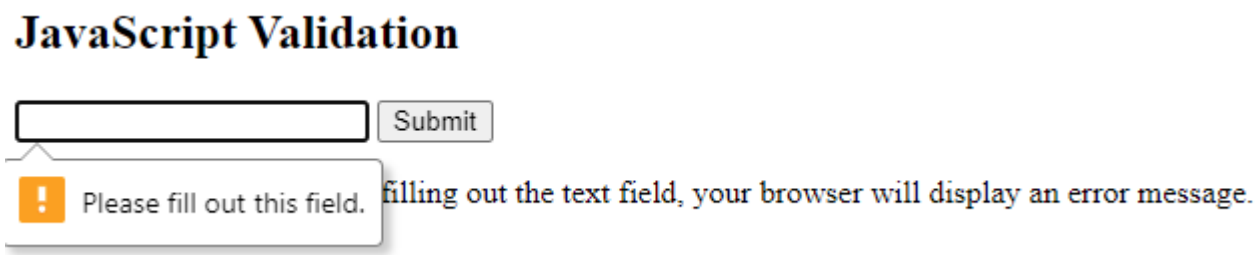
formValid.htm

formValid - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Validation</h2>

<form action="/action_page.php" method="post">
  <input type="text" name="fname" required>
  <input type="submit" value="Submit">
</form>

<p>If you click submit, without filling out the text field,
your browser will display an error message.</p>

</body>
</html>
```

output



Data Validation

- Data validation is the process of ensuring that user input is clean, correct, and useful.
- Typical validation tasks are:

1. has the user filled in all required fields?
2. has the user entered a valid date?
3. has the user entered text in a numeric field?

- Validation can be defined by many different methods, and deployed in many different ways.

1. **Server side validation** is performed by a web server, after input has been sent to the server.
2. **Client side validation** is performed by a web browser, before input is sent to a web server.

JavaScript Statements

- JavaScript statements are composed of:

1. Values
2. Operators
3. Expressions
4. Keywords
5. Comments.

A JavaScript program is a list of programming statements. Most JavaScript programs contain many JavaScript statements. The statements are executed, one by one, in the same order as they are written. JavaScript programs (and JavaScript statements) are often called JavaScript code.

JavaScript Statements

- This statement tells the browser to write "Hello unit2 JavaScript." inside an HTML element with id="statement":

statement.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Statements</h2>

<p>In HTML, JavaScript statements are executed by the browser.</p>

<p id="statement"></p>

<script>
document.getElementById("statement").innerHTML = "Hello unit2 JavaScript.";
</script>

</body>
</html>
```

output

JavaScript Statements

In HTML, JavaScript statements are executed by the browser.

Hello unit2 JavaScript.

Comments

- single line comment start with //
- Any text between // and the end of the line will be ignored by Javascript and will not be executed.

comment.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1 id="myH"></h1>
<p id="myP"></p>

<script>
// Change heading:
document.getElementById("myH").innerHTML = "JavaScript Comments";
// Change paragraph:
document.getElementById("myP").innerHTML = "This is unit2";
</script>

</body>
</html>
```

output

JavaScript Comments

This is unit2

Comments

- Multi-line Comments start with /* and end with */
- Any text between /* and */ will be ignored by JavaScript

multicomment.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1 id="myH"></h1>
<p id="myP"></p>

<script>
/*
The code below will change
the heading with id = "myH"
and the paragraph with id = "myP"
*/
document.getElementById("myH").innerHTML = "JavaScript multiline Comments";
document.getElementById("myP").innerHTML = "This is unit2";
</script>

</body>
</html>
```

output

JavaScript multiline Comments

This is unit2

Variables

- Variables are containers for storing data
- 4 Ways to Declare a JavaScript Variable:

1. Using var
2. Using let
3. Using const
4. Using nothing

- Examples

1. var name = expression;
2. var age = 32;
3. var weight = 127.4;
4. var clientName = "Connie Client";

When to Use JavaScript var?

Always declare JavaScript variables with var,let, or const.
The var keyword is used in all JavaScript code from 1995 to 2015.
The let and const keywords were added to JavaScript in 2015.
If you want your code to run in older browser, you must use var.

Variables declared using Var keyword

- In the below example,a,b and c variables are declared using the var keyword

vardemo.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1>JavaScript var Variables</h1>
<p>In this example, a, b, and c are variables.</p>
<p id="vardemo"></p>

<script>
var a = 1;
var b = 2;
var c = a + b;
document.getElementById("vardemo").innerHTML =
"The value of c is: " + c;
</script>

</body>
</html>
```

output

JavaScript var Variables

In this example, a, b, and c are variables.

The value of c is: 3

When to Use JavaScript let?

- If you think the value of the variable can change,

letdemo.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1>JavaScript let Variables</h1>
<p>In this example, a, b, and c are variables.</p>
<p id="letdemo"></p>

<script>
let a = 1;
let b = 2;
let c = a + b;
document.getElementById("letdemo").innerHTML =
"The value of c is: " + c;
</script>

</body>
</html>
```

output

JavaScript let Variables

In this example, a, b, and c are variables.

The value of c is: 3

Variables declared using nothing

undeclareddemo.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1>JavaScript undeclared Variables</h1>
<p>In this example, a, b, and c are undeclared variables.</p>
<p id="undeclareddemo"></p>

<script>
a = 1;
b = 2;
c = a + b;
document.getElementById("undeclareddemo").innerHTML =
"The value of c is: " + c;
</script>

</body>
</html>
```

output

JavaScript undeclared Variables

In this example, a, b, and c are undeclared variables.

The value of c is: 3

When to Use JavaScript const?

If you want a general rule: always declare variables with const.

In this example, rs1, rs2 are constant values and cannot be changed. variable total is declared with Let keyword.

constdemo.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h1>JavaScript Const Variables</h1>

<p>In this example, rs1, rs2, and total are variables.</p>

<p id="constdemo"></p>

<script>
const rs1 = 1;
const rs2 = 2;
let total = rs1 + rs2;
document.getElementById("constdemo").innerHTML =
"The total is: " + total;
</script>

</body>
</html>
```

output

JavaScript Const Variables

In this example, rs1, rs2, and total are variables.

The total is: 3

JavaScript Arithmetic Operators

Sr. No	Operator	Description
1	+	Addition
2	-	Subtraction
3	*	Multiplication
4	**	Exponentiation
5	/	Division
6	%	Modulus (Division Remainder)
7	++	Increment
8	--	Decrement

JavaScript Assignment Operators

Sr.No	Operator	Example	Same As
1	=	x = y	x = y
2	+=	x += y	x = x + y
3	-=	x -= y	x = x - y
4	*=	x *= y	x = x * y
5	/=	x /= y	x = x / y
6	%=	x %= y	x = x % y
7	**=	x **= y	x = x ** y

JavaScript Comparison Operators

Sr. No	Operator	Description
1	Operator	Description
2	==	equal to
3	===	equal value and equal type
4	!=	not equal
5	!==	not equal value or not equal type
6	>	greater than
7	<	less than
8	>=	greater than or equal to
9	<=	less than or equal to

JavaScript Logical Operators

Sr. No	Operator	Description
1	&&	logical and
2	||	logical or
3	!	logical not

JavaScript Type Operators

Sr. No	Operator	Description
1	typeof	Returns the type of a variable
2	instanceof	Returns true if an object is an instance of an object type

JavaScript Bitwise Operators

Sr.No	Operator	Description	Example	Same as	Result	Decimal
1	&	AND	5 & 1	0101 & 0001	0001	1
2	|	OR	5 | 1	0101 | 0001	0101	5
3	~	NOT	~ 5	~0101	1010	10
4	^	XOR	5 ^ 1	0101 ^ 0001	0100	4
5	<<	left shift	5 << 1	0101 << 1	1010	10
6	>>	right shift	5 >> 1	0101 >> 1	0010	2
7	>>>	unsigned right shift	5 >>> 1	0101 >>> 1	0010	2

JavaScript Bitwise Operators

- Bit operators work on 32 bits numbers.
- Any numeric operand in the operation is converted into a 32 bit number.
- The result is converted back to a JavaScript number.
- The examples in the table uses 4 bits unsigned examples.
- But JavaScript uses 32-bit signed numbers. Because of this, in JavaScript, ~ 5 will not return 10. It will return -6.
  ~00000000000000000000000000000101 will return
  11111111111111111111111111111010

The **multiplication** * operator

mul.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript multiplication Arithmetic</h2>
<h3>The * Operator</h3>

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

<script>
let a = 1;
let b = 2;
let c = a * b;
document.getElementById("demo").innerHTML = c;
</script>

</body>
</html>
```

output

JavaScript multiplication Arithmetic

The * Operator

2

JavaScript String Operators

- The + operator can be used to concatenate strings

string.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Operators</h2>

<p>The + operator concatenates (adds) strings.</p>

<p id="string"></p>

<script>
let text1 = "John";
let text2 = "Sparrow";
let text3 = text1 + " " + text2;
document.getElementById("string").innerHTML = text3;
</script>

</body>
</html>
```

output

JavaScript Operators

The + operator concatenates (adds) strings.

John Sparrow

JavaScript Concatenation Operator

- The += assignment operator can also be used to concatenate strings:

stringconcat.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Operators</h2>

<p>The assignment operator += can concatenate strings.</p>

<p id="concat"></p>

<script>
let text1 = "This is TE IT ";
text1 += "Web Technology Class";
document.getElementById("concat").innerHTML = text1;
</script>

</body>
</html>
```

output

JavaScript Operators

The assignment operator += can concatenate strings.

This is TE IT Web Technology Class

Adding Numbers and Strings

- Adding two numbers, will return the sum, but adding a number and a string will return a string:

strno.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Operators</h2>

<p>Adding a number and a string, returns a string.</p>

<p id="strno"></p>

<script>
let a = 5 + 5;
let b = "5" + 5;
let c = "Hello" + 5;
document.getElementById("strno").innerHTML =
a + "<br>" + b + "<br>" + c;
</script>

</body>
</html>
```

output

JavaScript Operators

Adding a number and a string, returns a string.

10
55
Hello5

Conditional statements

In JavaScript we have the following conditional statements:

1. Use if to specify a block of code to be executed, if a specified condition is true
2. Use else to specify a block of code to be executed, if the same condition is false
3. Use else if to specify a new condition to test, if the first condition is false
4. Use switch to specify many alternative blocks of code to be executed

The if Statement

- Use the if Statement to specify a block of JavaScript code to be executed if a condition is true.
- Syntax

```
if (condition) {
  //  block of code to be executed if the condition is true
}
```

- Example
- Make a "Good day" greeting if the hour is less than 18:00(6 pm)

```
if (hour < 18) {
  greeting = "Good day";
}
```

The if Statement

if.htm

if - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript if</h2>
<p>Display "Good day!" if the hour is less than 18:00:</p>
<p id="demo">Good Evening!</p>

<script>
if (new Date().getHours() < 18) {
  document.getElementById("demo").innerHTML = "Good day!";
}
</script>
</body>
</html>
```

output

JavaScript if

Display "Good day!" if the hour is less than 18:00:

Good day!

The else Statement

- Use the else Statement to specify a block of code to be executed if the condition is false.
- Syntax

```
if (condition) {
  //  block of code to be executed if the condition is true
} else {
  //  block of code to be executed if the condition is false
}
```

- Example

If the hour is less than 18, create a "Good day" greeting, otherwise "Good evening":

The else Statement

else.htm

else - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript if .. else</h2>
<p>A time-based greeting:</p>
<p id="demo"></p>

<script>
const hour = new Date().getHours();
let greeting;

if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}

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

</body>
</html>
```

output

JavaScript if .. else

A time-based greeting:

Good day

The else if Statement

- Use the else if Statement to specify a new condition if the first condition is false.
- Syntax

```
if (condition1) {
  //  block of code to be executed if condition1 is true
} else if (condition2) {
  //  block of code to be executed if the condition1 is false and condition2 is true
} else {
  //  block of code to be executed if the condition1 is false and condition2 is false
}
```

- Example
- If time is less than 10:00, create a "Good morning" greeting, if not, but time is less than 20:00, create a "Good day" greeting, otherwise a "Good evening":

The else if Statement

elseif.htm

elseif - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript if .. else</h2>
<p>A time-based greeting:</p>
<p id="demo"></p>

<script>
const time = new Date().getHours();
let greeting;
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
document.getElementById("demo").innerHTML = greeting;
</script>

</body>
</html>
```

output

JavaScript if .. else

A time-based greeting:

Good day

The JavaScript Switch Statement

- Use the Switch Statement to select one of many code blocks to be executed.
- Syntax

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

- Example
- The getDay() method returns the weekday as a number between 0 and 6.

(Sunday=0, Monday=1, Tuesday=2 ..)

Refer to switch.htm file

The JavaScript Switch Statement

switch - Notepad

File Edit Format View Help

switch.htm

```
<html>
<body>
<h2>JavaScript switch</h2>
<p id="demo"></p>
<script>
let day;
switch (new Date().getDay()) {
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
    day = "Tuesday";
    break;
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case  6:
    day = "Saturday";
}
document.getElementById("demo").innerHTML = "Today is " + day;
</script>
</body>
</html>
```

output

JavaScript switch

Today is Monday

Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)
19th April 2022
(Offline) Lecture 12

Padre Conceicao College of Engineering

Verna Goa 403722 India

Loops

- JavaScript supports different kinds of loops:

1. for - loops through a block of code a number of times
2. for/in - loops through the properties of an object
3. for/of - loops through the values of an iterable object
4. while - loops through a block of code while a specified condition is true
5. do/while - also loops through a block of code while a specified condition is true

The For Loop

- Syntax

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
}
```

- Statement 1 is executed (one time) before the execution of the code block.
- Statement 2 defines the condition for executing the code block.
- Statement 3 is executed (every time) after the code block has been executed.

The For Loop

for.htm

for - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript For Loop</h2>
<p id="demo"></p>

<script>
let text = "";

for (let i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}

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

</body>
</html>
```

output

JavaScript For Loop

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4

JavaScript For In

- The JavaScript For In statement loops through the properties of an Object
- Syntax

```
for (key in object) {
  // code block to be executed
}
```

Example Explained

1. The **for in** loop iterates over a **person** object
2. Each iteration returns a **key** (x)
3. The key is used to access the **value** of the key
4. The value of the key is **person[x]**

JavaScript For In

forin.htm

forin - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript For In Loop</h2>
<p>The for in statement loops through the properties of an object:</p>
<p id="demo"></p>

<script>
const person = {fname:"John", lname:"Gate", age:1};

let txt = "";
for (let x in person) {
  txt += person[x] + " ";
}

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

</body>
</html>
```

output

JavaScript For In Loop

The for in statement loops through the properties of an object:

John Gate 1

For In Over Arrays

- The JavaScript For In statement can also loop over the properties of an Array
- Syntax

```
for (variable in array) {
  code
}
```

- Example

For In Over Arrays

forinArray.htm

forinArray - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript For In</h2>
<p>The for in statement can loops over array values:</p>
<p id="demo"></p>

<script>
const numbers = [1, 12, 8, 14, 21];

let txt = "";
for (let x in numbers) {
  txt += numbers[x] + "<br>";
}

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

</body>
</html>
```

output

JavaScript For In

The for in statement can loops over array values:

1
12
8
14
21

Array.forEach()

- forEach() method calls a function (a callback function) once for each array element.
- Note that the function takes 3 arguments:

1. The item value
2. The item index
3. The array itself

Array.forEach()

forEach.htm

forEach - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Array.forEach()</h2>
<p>Calls a function once for each array element.</p>

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

<script>
const numbers = [4, 14,2, 19, 3];

let txt = "";
numbers.forEach(myFunction);
document.getElementById("demo").innerHTML = txt;

function myFunction(value, index, array) {
  txt += value + "<br>";
}
</script>

</body>
</html>
```

output

JavaScript Array.forEach()

Calls a function once for each array element.

4
14
2
19
3

JavaScript For of

- The JavaScript For of statement loops through the values of an iterable object. It lets you loop over iterable data structures such as Arrays, Strings, Maps, NodeLists, and more.
- Syntax

```
for (variable of iterable) {
  // code block to be executed
}
```

variable - For every iteration the value of the next property is assigned to the variable. *Variable* can be declared with const , let or var

iterable - An object that has iterable properties.

Looping over an Array

forOf.htm

forOf - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript For Of Loop</h2>
<p>The for of statement loops through the values of any iterable object:</p>

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

<script>
const cars = ["BMW", "Volvo", "Mini"];

let text = "";
for (let x of cars) {
  text += x + "<br>";
}

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

</body>
</html>
```

output

JavaScript For Of Loop

The for of statement loops through the values of any iterable object:

BMW
Volvo
Mini

Looping over a String

forOfStr.htm

forOfStr - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript For Of Loop</h2>

<p>The for of statement loops through the values of an iterable object.</p>

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

<script>
let language = "JavaScript";

let text = "";
for (let x of language) {
  text += x + "<br>";
}

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

</body>
</html>
```

output

JavaScript For Of Loop

The for of statement loops through the values of an iterable object.

J
a
v
a
S
c
r
i
p
t

JavaScript While Loop

- While Loop loops through a block of code as long as a specified condition is true.
- Syntax

```
while (condition) {
  // code block to be executed
}
```

- Example

JavaScript While Loop

while.htm

while - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript While Loop</h2>

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

<script>
let text = "";
let i = 0;
while (i < 5) {
  text += "<br>The number is " + i;
  i++;
}
document.getElementById("demo").innerHTML = text;
</script>

</body>
</html>
```

output

JavaScript While Loop

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4

The Do While Loop

- The Do While Loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.
- Syntax

```
do {
  // code block to be executed
}
while (condition);
```

The Do While Loop

doWhile - Notepad

File Edit Format View Help

doWhile.htm

```
<html>
<body>

<h2>JavaScript Do While Loop</h2>

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

<script>
let text = ""
let i = 0;

do {
  text += "<br>The number is " + i;
  i++;
}
while (i < 10);

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

</body>
</html>
```

output

JavaScript Do While Loop

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9

JavaScript Events

- HTML events are "things" that happen to HTML elements.
- When JavaScript is used in HTML pages, JavaScript can "react" on these events.

HTML Events

- An HTML event can be something the browser does, or something a user does.
- Here are some examples of HTML events:

1. An HTML web page has finished loading
2. An HTML input field was changed
3. An HTML button was clicked

- Often, when events happen, you may want to do something.
- JavaScript lets you execute code when events are detected.

HTML Events

- HTML allows event handler attributes, with JavaScript code, to be added to HTML elements.
- With single quotes:

*<element event=**'some JavaScript'**>*

- With double quotes:

*<element event=**"some JavaScript"**>*

HTML Events

- In the following example, an onclick attribute (with code), is added to a <button> element:

event.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<button onclick="document.getElementById('event').innerHTML=Date()">The time is?</button>

<p id="event"></p>

</body>
</html>
```

output

The time is?

Wed Apr 13 2022 16:14:01 GMT+0530 (India Standard Time)

Common HTML Events

Sr.No	Event	Description
1	onchange	An HTML element has been changed
2	onclick	The user clicks an HTML element
3	onmouseover	The user moves the mouse over an HTML element
4	onmouseout	The user moves the mouse away from an HTML element
5	onkeydown	The user pushes a keyboard key
6	onload	The browser has finished loading the page

JavaScript Event Handlers

Event handlers can be used to handle and verify user input, user actions, and browser actions:

- Things that should be done every time a page loads
- Things that should be done when the page is closed
- Action that should be performed when a user clicks a button
- Content that should be verified when a user inputs data

...

Many different methods can be used to let JavaScript work with events:

- HTML event attributes can execute JavaScript code directly
- HTML event attributes can call JavaScript functions
- You can assign your own event handler functions to HTML elements
- You can prevent events from being sent or being handled

JavaScript Objects

- In real life, a car is an object. A car has properties like weight and colour , and methods like start and stop. All cars have the same properties, but the property values differ from car to car.
- All cars have the same methods, but the methods are performed at different times

object	Properties	Methods
car	car.name = Fiat car.model = 500 car.weight = 850kg car.color = white	car.start() car.drive() car.brake() car.stop()

JavaScript Objects

1. Booleans can be objects (if defined with the new keyword)
2. Numbers can be objects (if defined with the new keyword)
3. Strings can be objects (if defined with the new keyword)
4. Dates are always objects
5. Maths are always objects
6. Regular expressions are always objects
7. Arrays are always objects
8. Functions are always objects
9. Objects are always objects

Object Definition

- You define and create a JavaScript object with an object literal
- Example

```
const person = {
firstName:"Irwin",
lastName:"Rebello",
rollno:16,
};
```

Object Definition

obj.htm

obj - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>

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

<script>
// Create an object:
const person = {firstName:"Irwin", lastName:"Rebello", rollno:16};

// Display some data from the object:
document.getElementById("demo").innerHTML =
person.firstName + " is student with rollno" + person.rollno;
</script>

</body>
</html>
```

output

JavaScript Objects

Irwin is student with rollno16

Object Properties

- The named values, in JavaScript objects, are called properties.

Property	Value
firstName	Irwin
lastName	Rebello
rollno	16

Accessing Object Properties

- You can access object properties in two ways:

1. objectName.propertyName

or

2. objectName["propertyName"]

Accessing Object Properties using *objectName.propertyName*

acc1obj.htm

acc1obj - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>

<p>There are two different ways to access an object property.</p>

<p>You can use person.property or person["property"].</p>

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

<script>
// Create an object:
const person = {
  firstName: "Anacia",
  lastName : "Dias",
  rollno    : 3
};

// Display some data from the object:
document.getElementById("demo").innerHTML =
person.firstName + " " + person.lastName;
</script>

</body>
</html>
```

output

JavaScript Objects

There are two different ways to access an object property.

You can use person.property or person["property"].

Anacia Dias

Accessing Object Properties using *objectName["propertyName"]*

acc2obj.htm

acc2obj - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>

<p>There are two different ways to access an object property.</p>

<p>You can use person.property or person["property"].</p>

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

<script>
// Create an object:
const person = {
  firstName: "Abdul",
middleName:"Gafur",
  lastName : "Sayed",
  irollno   :  2
};

// Display some data from the object:
document.getElementById("demo").innerHTML =
person["firstName"] + " " +person["middleName"]+" "+ person["lastName"];
</script>

</body>
</html>
```

output

JavaScript Objects

There are two different ways to access an object property.

You can use person.property or person["property"].

Abdul Gafur Sayed

JavaScript Object Methods

- Objects can also have **methods**.
- Methods are **actions** that can be performed on objects.
- Methods are stored in properties as **function definitions**.

What is **this**?

- In JavaScript , this keyword refers to an object. Which object depends on how this is being invoked (used or called).this is not a variable. The this keyword refers to different objects depending on how it is used:

1. In an object method, this refers to the object.
2. Alone, this refers to the global object.
3. In a function, this refers to the global object.
4. In a function, in strict mode, this is undefined.
5. In an event, this refers to the element that received the event.
6. Methods like call(), apply(), and bind() can

JavaScript Object Methods

this - Notepad

File Edit Format View Help

this.htm

```
<body>

<h1>The JavaScript <i>this</i> Keyword</h1>
<p>In this example, <b>this</b> refers to the <b>person</b> object.</p>
<p>Because <b>fullName</b> is a method of the person object.</p>

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

<script>
// Create an object:
const person = {
  firstName: "Yusuf",
  lastName: "Khan",
   fullName : function() {
    return this.firstName + " " + this.lastName;
  }
};

// Display data from the object:
document.getElementById("demo").innerHTML = person.fullName();
</script>

</body>
</html>
```

output

The JavaScript *this* Keyword

In this example, **this** refers to the **person** object.

Because **fullName** is a method of the person object.

Yusuf Khan

Object Methods

Property	Property Value
firstName	Yusuf
lastName	Khan
rollno	29
fullName	function() {return this.firstName + " " + this.lastName;}

How to Display JavaScript Objects?

- Displaying a JavaScript object will output **[object Object]**.

displayObj.htm

displayObj - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>
<p>Displaying a JavaScript object will output [object Object]:</p>

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

<script>
const person = {
  name: "Angelo",
  rollno: 4,
  state: "Goa"
};

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

</body>
</html>
```

output

JavaScript Objects

Displaying a JavaScript object will output [object Object]:

[object Object]

Displaying Object Properties

displayObjProp.htm

displayObjProp - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>
<p>Display object properties:</p>

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

<script>
const person = {
 name: "Angelo",
 rollno: 4,
 state: "Goa"
};

document.getElementById("demo").innerHTML = person.name + ", " + person.rollno + ", " + person.state;
</script>

</body>
</html>
```

output

JavaScript Objects

Display object properties:

Angelo, 4, Goa

Displaying the Object in a Loop

- The properties of an object can be collected in a loop as shown below

displayObjloop - Notepad

File Edit Format View Help

displayObjloop.htm

```
<html>
<body>

<h2>JavaScript Objects</h2>
<p>Display object properties:</p>

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

<script>
const person = {
  name: "Annalie",
 rollno: 5,
 state: "Goa"
};

let txt = "";
for (let x in person) {
  txt += person[x] + " ";
};

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

</body>
</html>
```

output

JavaScript Objects

Display object properties:

Annalie 5 Goa

Using Object.values()

- Any JavaScript object can be converted to an array using Object.values()

displayObjArray.htm

displayObjArray - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Objects</h2>
<p>Object.values() converts an object to an array.</p>

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

<script>
const person = {
 name: "Aryan",
 rollno: 6,
 state: "Goa"
};

document.getElementById("demo").innerHTML = Object.values(person);
</script>

</body>
</html>
```

output

JavaScript Objects

Object.values() converts an object to an array.

Aryan,6,Goa

JavaScript Functions

- A JavaScript function is a block of code designed to perform a particular task. A JavaScript function is executed when "something" invokes it (calls it). A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses (). Function names can contain letters, digits, underscores, and dollar signs (same rules as variables). The parentheses may include parameter names separated by commas: (*parameter1, parameter2, ...*)
- The code to be executed, by the function, is placed inside curly brackets: {}
- Function parameters are listed inside the parentheses () in the function definition. Function arguments are the values received by the function when it is invoked. Inside the function, the arguments (the parameters) behave as local variables.

JavaScript Function Syntax

- Syntax

```
function name(parameter1,parameter2)
 {
        statement ;
        statement ;
         ...
        statement ;
}
```

- Example

```
function myFunction()
{
alert("Hello!");
alert("How are you?");
 }
```

Function Invocation

- The code inside the function will execute when "something" **invokes** (calls) the function:

1. When an event occurs (when a user clicks a button)
2. When it is invoked (called) from JavaScript code
3. Automatically (self invoked)

JavaScript Functions

function.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Function</h2>

<p>This example calls a function which performs a calculation, and returns the result:</p>

<p id="function"></p>

<script>
function myFunction(x1, x2) {
  return x1 * x2;
}
document.getElementById("function").innerHTML = myFunction(1, 2);
</script>

</body>
</html>
```

output

JavaScript Function

This example calls a function which performs a calculation, and returns the result:

2

Function Return

1. When JavaScript reaches a return statement, the function will stop executing.
2. If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.
3. Functions often compute a return value.
4. The return value is "returned" back to the "caller"

Function Return

functionreturn.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Functions</h2>

<p>This example calls a function which performs a calculation and returns the result:</p>

<p id="functionreturn"></p>

<script>
var c = myFunction(1, 2);
document.getElementById("functionreturn").innerHTML = c;

function myFunction(a, b) {
  return a * b;
}
</script>

</body>
</html>
```

output

JavaScript Functions

This example calls a function which performs a calculation and returns the result:

2

Functions Used as Variable Values

- Functions can be used the same way as you use variables, in all types of formulas, assignments, and calculations. Instead of using a variable to store the return value of a function. You can use the function directly, as a variable value:

functionvariable.htm - Notepad

File Edit Format View Help

```
<html>
<body>

<h2>JavaScript Functions</h2>

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

<script>
document.getElementById("demo").innerHTML =
"The temperature is " + toCelsius(77) + " Celsius";

function toCelsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
</script>

</body>
</html>
```

output

JavaScript Functions

The temperature is 25 Celsius

Assignment 2

- Q1) Design automatic HTML Form Validation and illustrate the output (5 marks)
- Q2) Write a javascript function to add two numbers and display the output (5 marks)

- *Assignment Announced to students : AA :19th April 2022*
- *Assignment to be Submitted by students : AS: 2nd May 2022*