JavaScript

Dr. Arul Xavier V M

Assistant Professor

Introduction

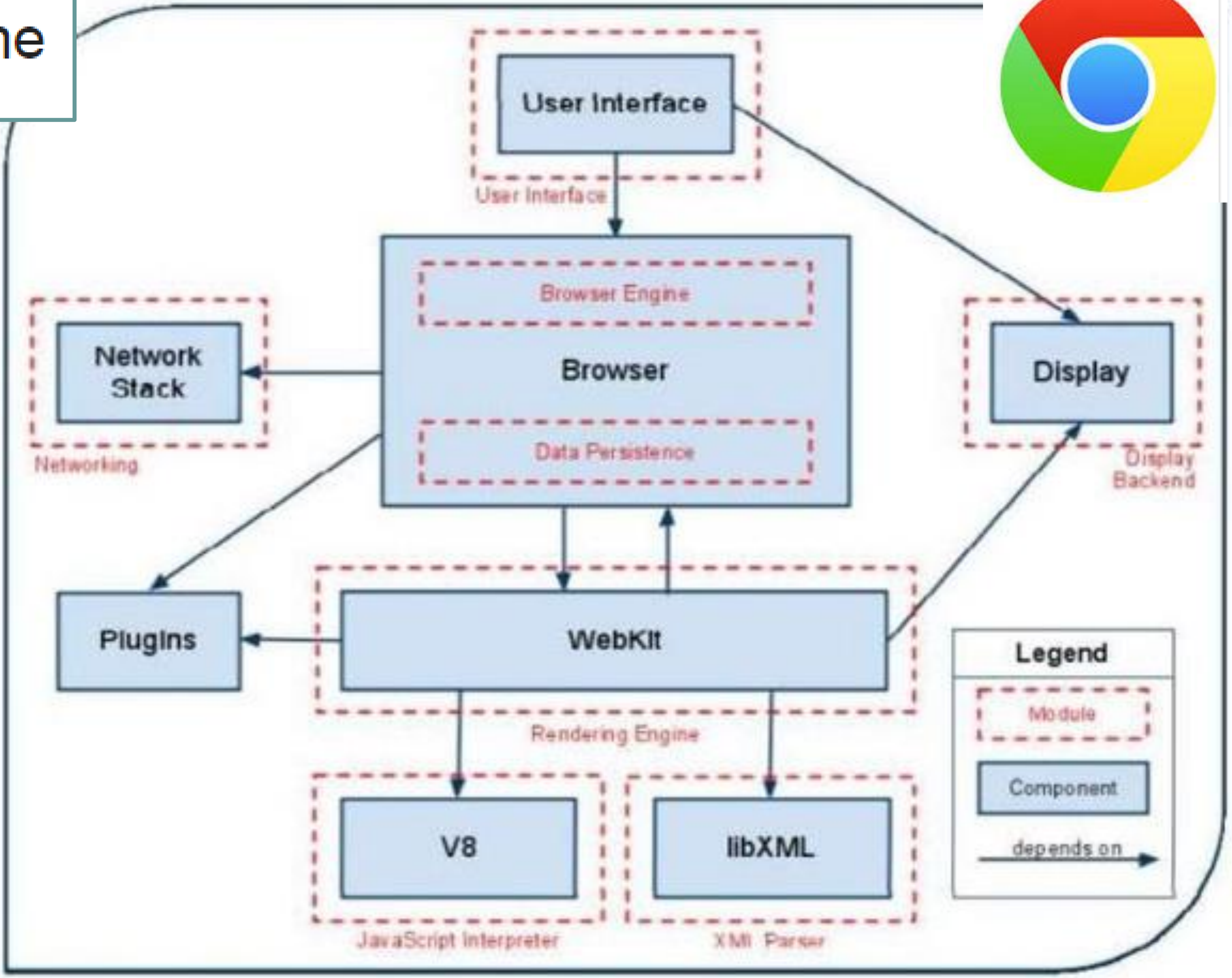
- JavaScript is the world's most popular programming language.
- JavaScript is *an* ***object-based scripting language*** that is lightweight and cross-platform.
- **JavaScript** for beginners and professionals to create interactive client side dynamic pages.
- JavaScript is not compiled but translated.
- The JavaScript Engine (**embedded in browser**) is responsible to translate the JavaScript code.

JavaScript engine

- A JavaScript engine is a **program or interpreter** which executes JavaScript code.
- A JavaScript engine may be a **traditional interpreter.**
- Every browser has an in built Javascript engine. Popular Javascript names are given below.
  - Google – V8
  - Firefox – SpiderMonkey
  - Safari – JavaScriptCore

Google Chrome

Web browser



History of JavaScript

- JavaScript was created in 10 days in May 1995 by **Brendan Eich.**
- The original name was ***“Mocha”,*** and then changed to ***“LiveScript”.***
- Later, upon receiving a trademark license from Sun Microsystems, the name **JavaScript** was adopted.

JavaScript vs Java

- Javascript is not a Java
- It is not a light version of Java
- It was not based on Java
- It does not matter if you know Java
- Note:
  - Javascript is not all related to Java Programming Language

How to include JavaScript code in HTML?

- HTML provides 2 places to put your JavaScript code:
  - Internal JavaScript
    - Using <script> tag
      - Inside <head> tag   or
      - Inside <body> tag
  - External JavaScript file.
    - Using external file with extension “.js”.
    - Then include the file using <script> tag

Internal JavaScript

- You can include the JavaScript code internally by embedding using <script> tag in your HTML page.
- You can place the <script> tag either inside the <head> tag, or inside the <body>.

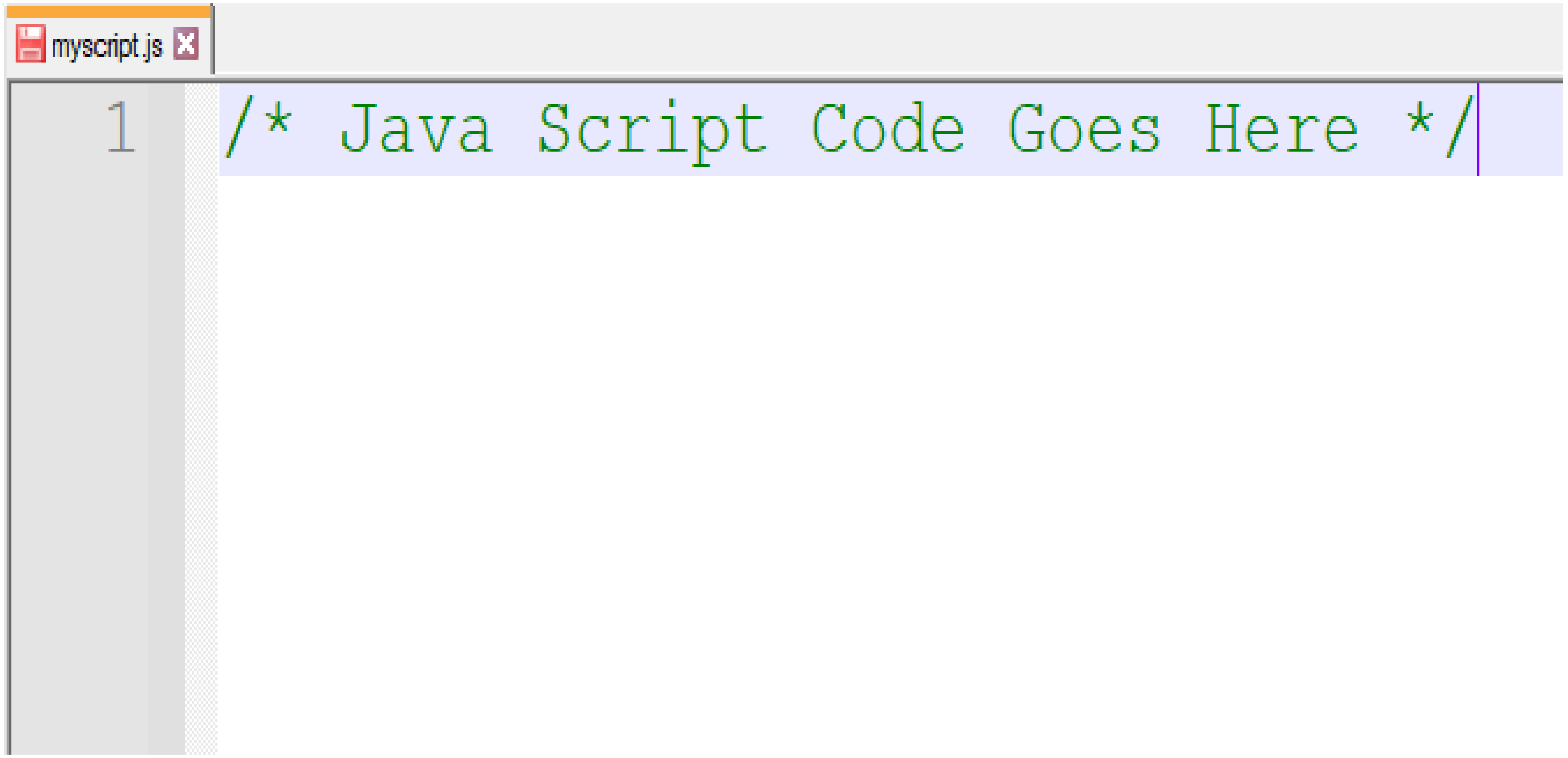
```
<!DOCTYPE html>
<html>
    <head>
        <title>My Website</title>
        <script>
            <!-- Javascript code goes here -->
        </script>
    </head>
    <body>
       <script>
           <!-- Javascript code goes here -->
       </script>
    </body>
</html>
```

External Javascript

- Scripts can also be placed in external files.
- External scripts are practical when the same code is used in many different web pages.
- JavaScript files have the file extension .js.
- To use an external script, put the name of the script file in the src (source) attribute of a <script> tag.

External Javascript

- Create a .js file and keep some Javascript code



```
/* Java Script Code Goes Here */
```

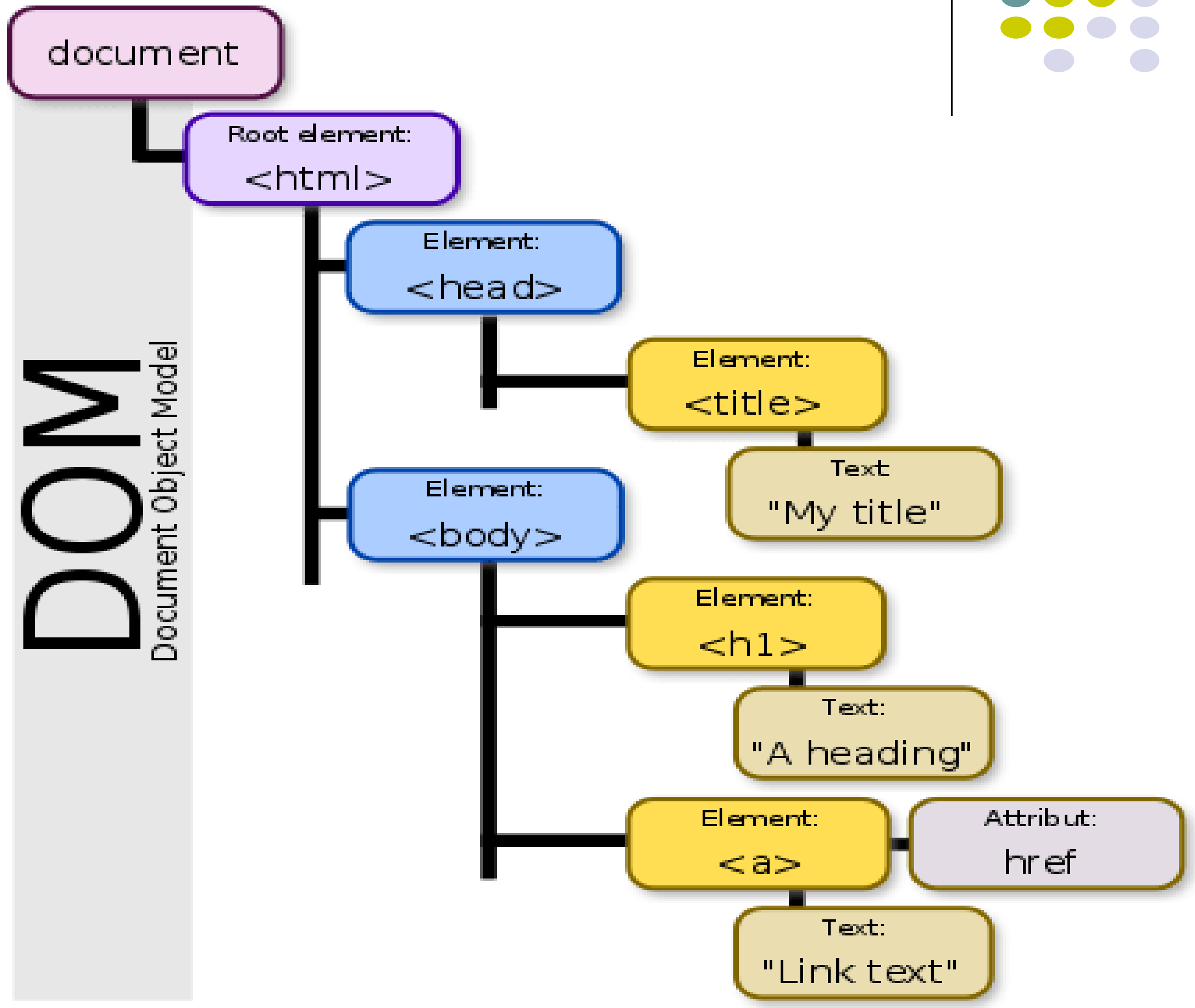
Link external Javascript file in HTML

- Link the external .js File in HTML code via
  - <script src= “myscript.js” >

```
<!DOCTYPE html>
<html>
    <head>
        <title>My Website</title>
           <script src="myscript.js"> </script>
    </head>
    <body>

    </body>
</html>
```

Document Object Model (DOM)

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



Use of DOM with Javascript

- With the object model, JavaScript gets all the power it needs to create dynamic HTML:
- JavaScript can access, change, add or remove
  - HTML Elements or Tags
  - HTML Attributes
  - HTML Events
  - React to HTML Events
  - CSS Styles

Working with JavaScript

- JavaScript is object based language.
- In JavaScript, functions, events and properties are major elements to make the web page more interactive.
- There 2 variants of functions
  - Built-in functions
  - User-defined function
- JavaScript functions are also called as “Methods” (Both are same)

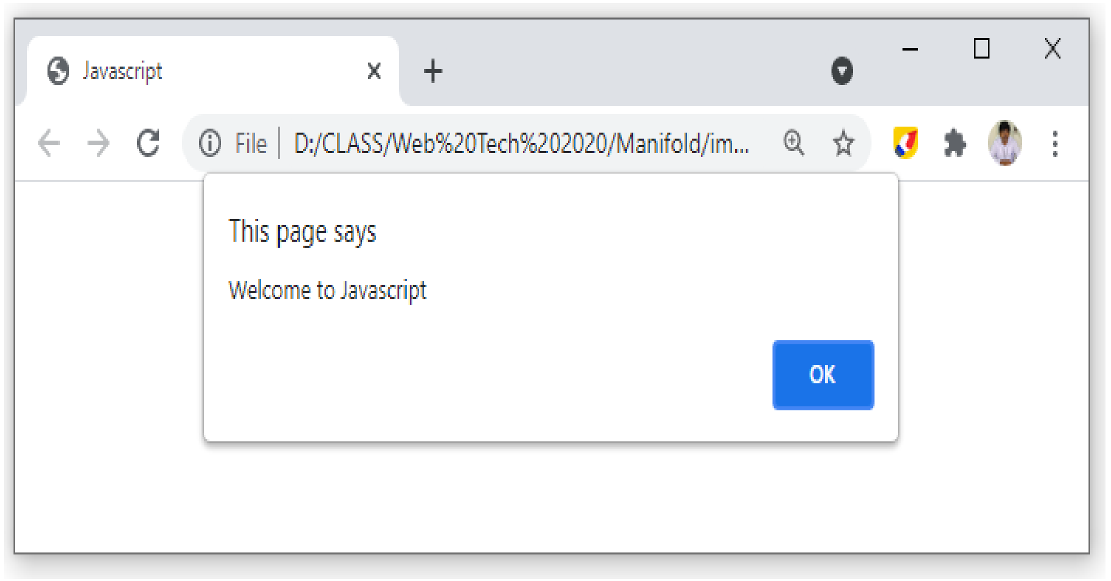
Display Data in Javascript

- JavaScript can "display" data in different ways using its built-in functions/methods.
  - Display using an alert box, using window.alert().
  - Display inside web page using document.write().
  - Display in browser console, using console.log().
  - Display inside HTML element using innerHTML property.
  - Display inside HTML Form Tex Box using value property.

To display data in a Alert or Message box

- The alert() method displays an alert box with a specified message and an OK button.
- The alert() method is defined by window object

```
<!DOCTYPE html>
<html>
    <head>
        <title>My Website</title>
        <script>
            alert("Welcome to Javascript");
        </script>
    </head>
    <body>

    </body>
</html>
```



This page says

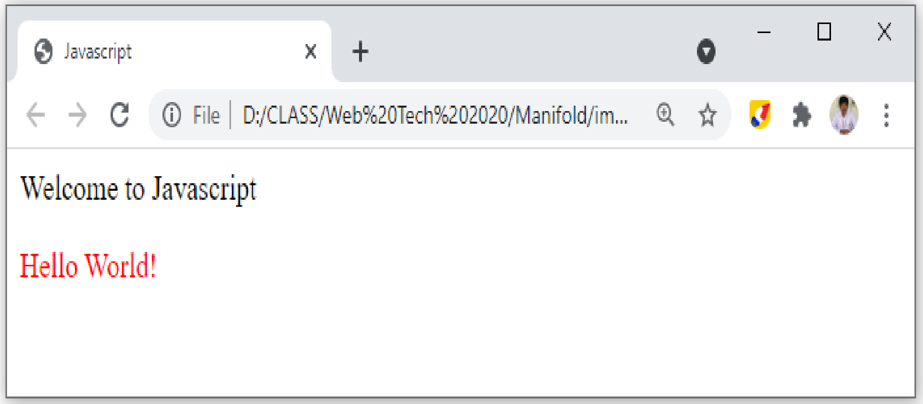
Welcome to Javascript

Display data into the Web Page.

- The HTML DOM model provides an object called "document", which includes set of methods.
- One of the method is write(), which can be used to display any data inside the web page using Javascript.

Syntax: document.write()

```
<!DOCTYPE html>
<html>
    <head>
        <title>My Website</title>
        <script>
      document.write("Welcome to Javascript");
      document.write("<p style='color:red;'>Hello World!</p>");
        </script>
    </head>
    <body>

    </body>
</html>
```

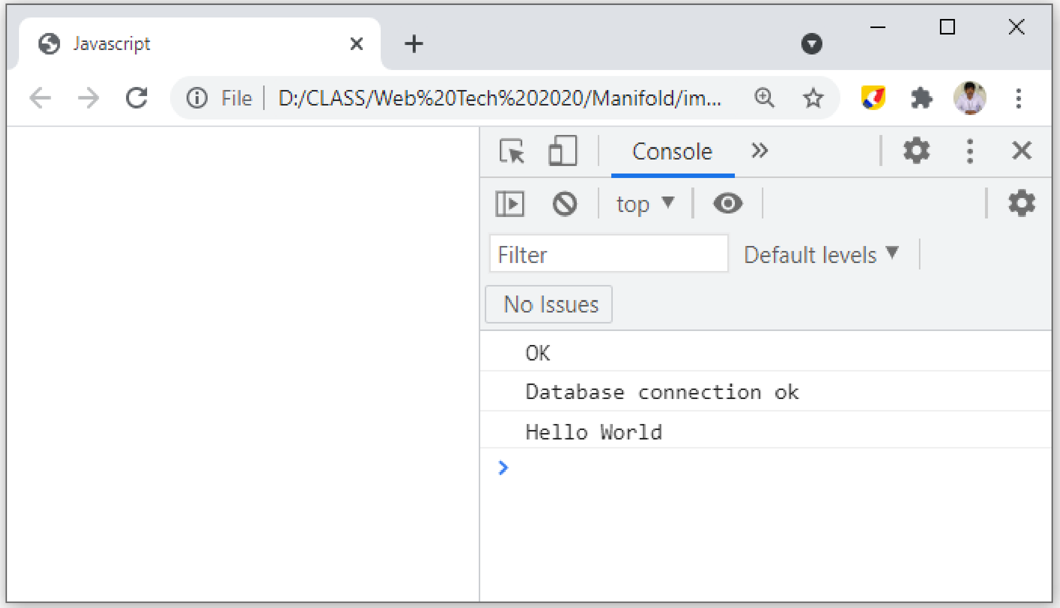


```
Welcome to Javascript

Hello World!
```

Display data in a Browser's Console Window

- For debugging purposes, you can call the console.log() method in the browser to display data.

```
<!DOCTYPE html>
<html>
    <head>
        <title>My Website</title>
        <script>
       console.log("OK");
       console.log("Database connection ok");
       console.log("Hello World");
        </script>
    </head>
</html>
```

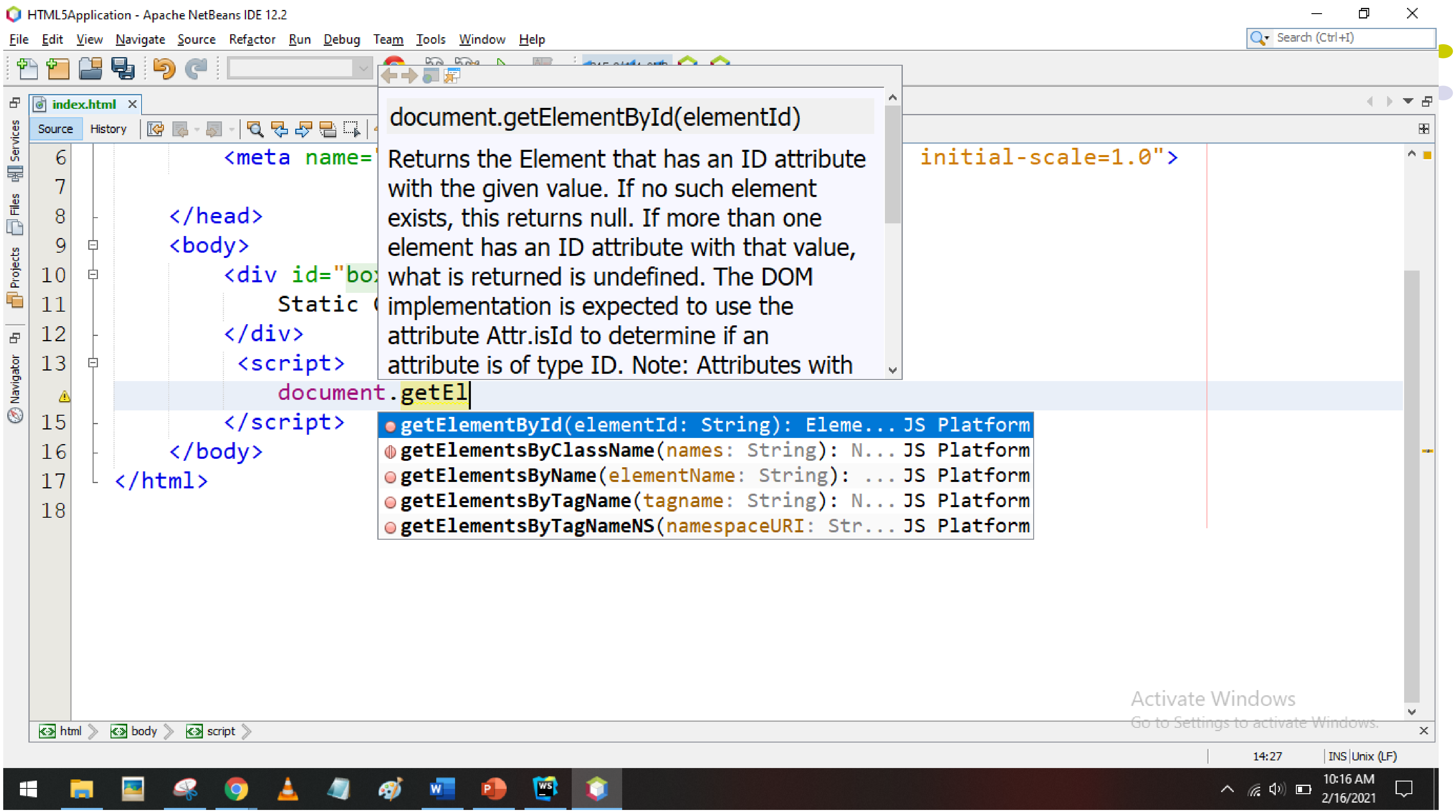


```
OK
Database connection ok
Hello World
>
```

Display inside HTML element, using innerHTML property.

- JavaScript is a object-based language.
- Every HTML element is consider as an “object”.
- “**object**” consists of “**properties**” and “**methods**”.
  - innerHTML is a one of the property of HTML elements(limited!!)
  - It is used to write data to specific element’s content area directly.
- To access any HTML element, JavaScript can use the following method

  document.getElementById(**id**);
- The **id** attribute defines a unique identity of the HTML element.
- The **innerHTML** property defines the content of the particular element.


HTML5Application - Apache NetBeans IDE 12.2

File Edit View Navigate Source Refactor Run Debug Team Tools Window Help

index.html

```
<meta name="              initial-scale=1.0">

</head>
<body>
    <div id="bo
        Static (
    </div>
     <script>
        document.getEl
    </script>
</body>
</html>
```

document.getElementById(elementId)

Returns the Element that has an ID attribute with the given value. If no such element exists, this returns null. If more than one element has an ID attribute with that value, what is returned is undefined. The DOM implementation is expected to use the attribute Attr.isId to determine if an attribute is of type ID. Note: Attributes with

```
getElementById(elementId: String): Eleme... JS Platform
getElementsByClassName(names: String): N... JS Platform
getElementsByName(elementName: String): ... JS Platform
getElementsByTagName(tagname: String): N... JS Platform
getElementsByTagNameNS(namespaceURI: Str... JS Platform
```

html > body > script

14:27 INS Unix (LF)

Activate Windows
Go to Settings to activate Windows.

10:16 AM 2/16/2021

Example

```
<!DOCTYPE html>
<html>
   <head>
      <title>Javascript</title>
   </head>
   <body>
      <div id="box1" style="color:blue;"></div>
      <script>
         document.getElementById('box1').innerHTML = "Welcome User!!";
       </script>
   </body>
</html>
```

Closer look at document.getElementById()…

Method

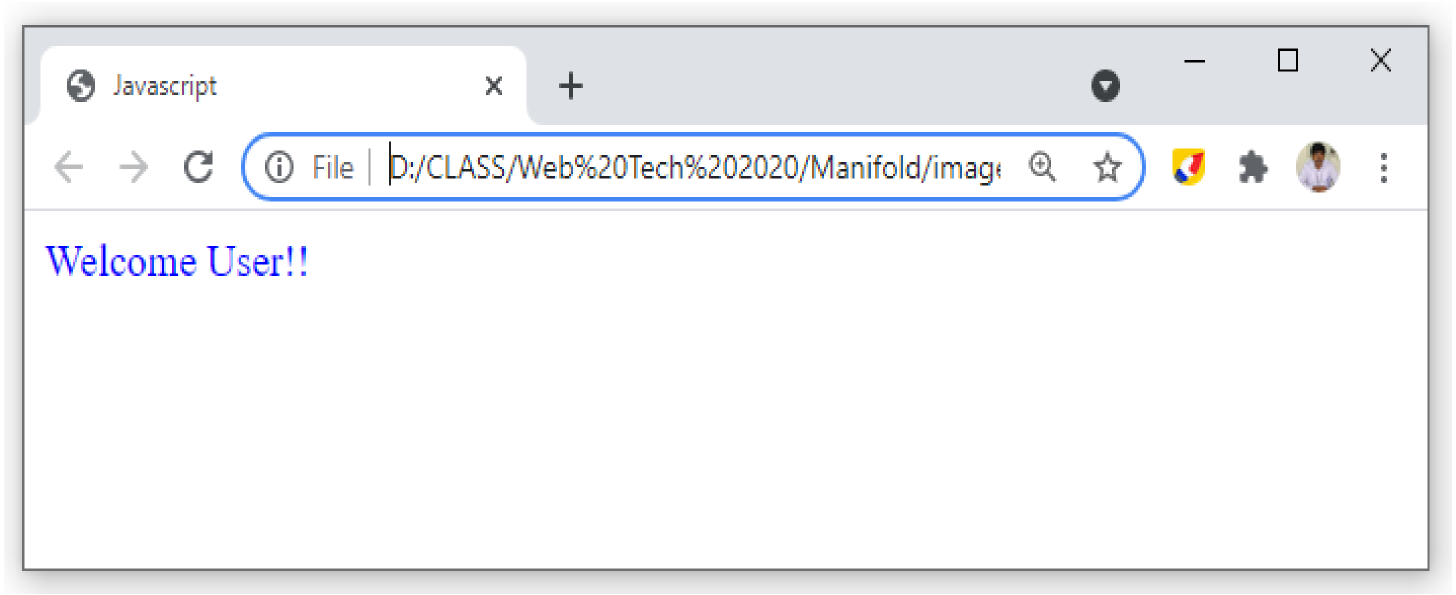
Property to access the content area of DIV element

```
document.getElementById('box1').innerHTML = "Welcome User!!";
```

DOM object

id of div Element

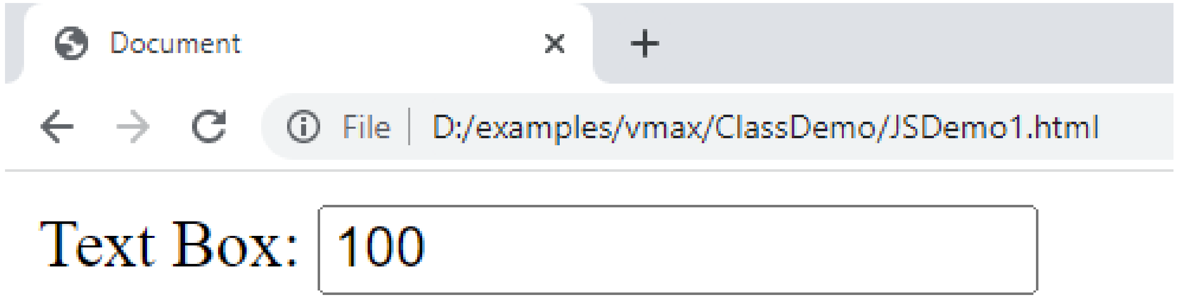
data



Welcome User!!

Display inside a HTML Form Text Box

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    Text Box: <input type="text" id="output">
    <script>
        document.getElementById('output').value = "100";
    </script>
</body>
</html>
```



Text Box: 100

Javascript Programming Features

- JavaScript is a programming language:
- All instructions are called as statements, which must be separated by semicolon.
- JavaScript statements are composed of:
  - Values(literals), Variables, Operators, Expressions, Control Statements, Keywords, and User Defined Functions
  - Values or literals are: Numbers, Strings, Arrays etc..
  - Variables can be used represent Numbers, Strings, Arrays, Objects, etc.

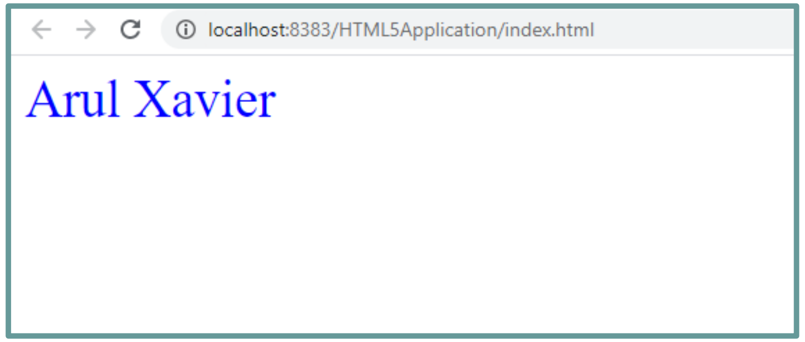
Creating variables using var keyword

```
<script>
  var price=1000;
  var name="Prince";
  var a,b,c;
  var interest=7.5;
</script>
```

String Concatenation

- The + operator can also be used to add (concatenate) strings.

```
<body>
    <div id="box" style="color: blue;font-size: 30px;"></div>
    <script>
        var firstname = 'Arul '
        var lastname ='Xavier'
        var name = firstname + lastname;
        document.getElementById('box').innerHTML = name;
    </script>
</body>
```


localhost:8383/HTML5Application/index.html

Arul Xavier

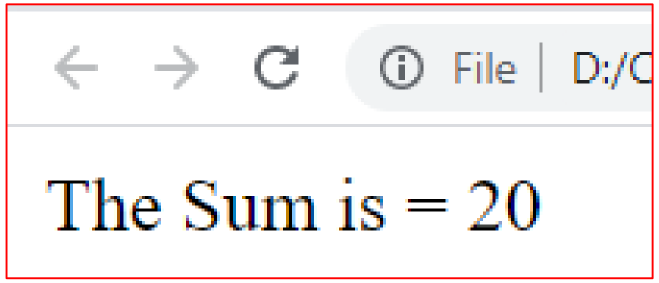
Creating user defined Function

- Functions are created using the keyword function.
- 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 name(parameter1, parameter2, parameter3)
{
  // code to be executed
}
```

Creating Function - Example

```
<script>
    function add(){        // creating a function
        var a = 10;
        var b = 10;
        sum = a+b;
        document.write("The Sum is = " + sum);
    }

    add();      //calling or invoking the function

</script>
```

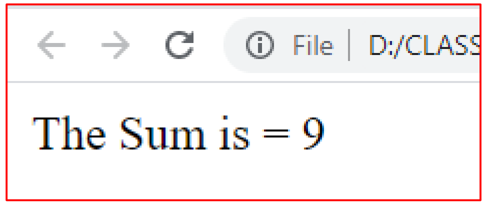


```
The Sum is = 20
```

Adding Parameters to function

- The parentheses may include parameter names separated by commas: (parameter1, parameter2, ...)
- Function arguments are the values received by the function when it is invoked.
- Inside the function, the arguments (the parameters) behave as local variables.

```
<script>
    function add(num1, num2){    // creating function with parameters
        var sum = num1+num2;
        document.write("The Sum is = " + sum);
    }

    add(5,4); //calling or invoking the function with arguments

</script>
```



File | D:/CLASS

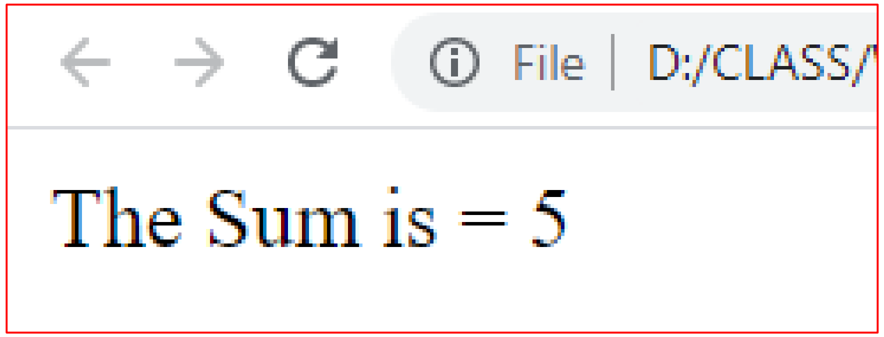
```
The Sum is = 9
```

Return a value from a Function

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

Return a value from a Function - Example

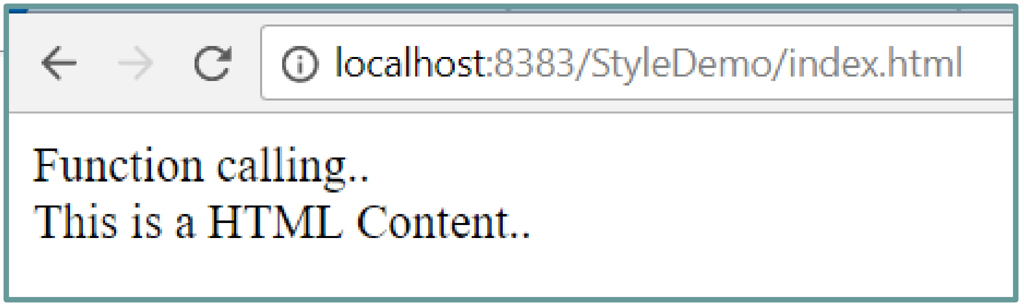
```
<script>
    function add(num1, num2)   // creating function with parameters
    {
        return num1+num2;
    }

    result=add(5,4);   // the value will be return to result variable

    document.write("The Sum is = " + result);
</script>
```



The Sum is = 5

Calling Functions

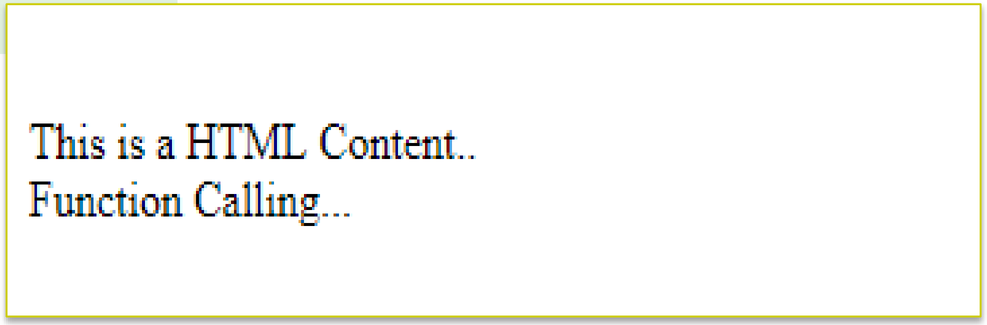
```
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <script>
            show();
            function show(){
                document.write("Function calling..");
            }
        </script>
    </head>
    <body>
        <div>This is a HTML Content..</div>
    </body>
</html>
```



```
Function calling..
This is a HTML Content..
```

Function calling from <body>

```
<!DOCTYPE html>
<html>
    <head>
        <title>TODO supply a title</title>
        <meta charset="UTF-8">
        <script>
            function show(){
                document.write("Function calling..");
            }
        </script>
    </head>

    <body>
        <div>This is a HTML Content..</div>
        <script>
            show();
        </script>
    </body>
</html>
```



```
This is a HTML Content..
Function Calling...
```