JavaScript is an object-based scripting language which is lightweight and cross-platform.

JavaScript is not a compiled language, but it is a translated language. The JavaScript Translator (embedded in the browser) is responsible for translating the JavaScript code for the web browser.

What is JavaScript

JavaScript (js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document. It was introduced in the year 1995 for adding programs to the webpages in the Netscape Navigator browser. Since then, it has been adopted by all other graphical web browsers.

## Application of JavaScript

JavaScript is used to create interactive websites. It is mainly used for:

* Client-side validation,
* Dynamic drop-down menus,
* Displaying date and time,
* Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),
* Displaying clocks etc.

**Example:**

<html>

<body>

<h2>Welcome to JavaScript</h2>

<script>

document.write("Hello JavaScript by JavaScript");

</script>

</body>

</html>

# **JavaScript Where To**

## JavaScript in <head>

In this example, a JavaScript function is placed in the <head> section of an HTML page.

The function is invoked (called) when a button is clicked:

### **Example**

<!DOCTYPE html>  
<html>  
<head>  
<script>  
function myFunction() {  
  document.getElementById("demo").innerHTML = "Paragraph changed.";  
}  
</script>  
</head>  
<body>

<h2>Demo JavaScript in Head</h2>  
  
<p id="demo">A Paragraph</p>  
<button type="button" onclick="myFunction()">Try it</button>

</body>  
</html>

## JavaScript in <body>

In this example, a JavaScript function is placed in the <body> section of an HTML page.

The function is invoked (called) when a button is clicked:

### **Example**

<!DOCTYPE html>  
<html>  
<body>  
  
<h2>Demo JavaScript in Body</h2>  
  
<p id="demo">A Paragraph</p>  
  
<button type="button" onclick="myFunction()">Try it</button>  
  
<script>  
function myFunction() {  
  document.getElementById("demo").innerHTML = "Paragraph changed.";  
}  
</script>  
  
</body>  
</html>

# **JavaScript Output**

## JavaScript Display Possibilities

JavaScript can "display" data in different ways:

* Writing into an HTML element, using innerHTML.
* Writing into the HTML output using document.write().
* Writing into an alert box, using window.alert().
* Writing into the browser console, using console.log().

## Using innerHTML

To access an HTML element, JavaScript can use the document.getElementById(id) method.

The id attribute defines the HTML element. The innerHTML property defines the HTML content:

### **Example**

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Web Page</h1>  
<p>My First Paragraph</p>  
  
<p id="demo"></p>  
  
<script>  
document.getElementById("demo").innerHTML = 5 + 6;  
</script>  
  
</body>  
</html>

## Using document.write()

For testing purposes, it is convenient to use document.write():

### **Example**

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Web Page</h1>  
<p>My first paragraph.</p>  
  
<script>  
document.write(5 + 6);  
</script>  
  
</body>  
</html>

## Using window.alert()

You can use an alert box to display data:

### **Example**

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Web Page</h1>  
<p>My first paragraph.</p>  
  
<script>  
window.alert(5 + 6);  
</script>  
  
</body>  
</html>

## Using console.log()

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

You will learn more about debugging in a later chapter.

### **Example**

<!DOCTYPE html>  
<html>  
<body>  
  
<script>  
console.log(5 + 6);  
</script>  
  
</body>  
</html>

# Difference Between Var, Let, and Const in Javascript

In the early days of JavaScript, there was only one way of declaring variables and that was using the **var** keyword. A variable declared with **var** is defined throughout the program. One of the issues with using the **var** keyword was redeclaring a variable inside a block will also redeclare the variable outside the block.

With the introduction of ES6 in 2015 two more keywords, let and const came into the picture. var and let are both used for variable declaration in javascript but the difference between them is that var is function scoped and let is block scoped. Variable declared by let cannot be redeclared and must be declared before use whereas variables declared with var keyword are hoisted.

The **var**, **let**, and **const** is the keyword to declare variables in javascript. The var variable is an old method to declare a variable in javascript. In modern javascript, we use the let and const variable, which was introduced in the ES2015(ES6) update; now, the let and const variable is used more frequently in modern javascript as compared to the var variable.

# JavaScript let Vs var

In this tutorial, you will learn about the difference between let and var in JavaScript with the help of examples.

In JavaScript, both the keywords var and let are used to declare variables.

The let keyword was introduced in the later version of JavaScript known as **ES6(ES2015)**. And it's the preferred way to declare variables.

## JavaScript let Vs var

Here's the overview of the differences between let and var.

|  |  |
| --- | --- |
| let | var |
| let is block-scoped. | var is function scoped. |
| let does not allow to redeclare variables. | var allows to redeclare variables. |
| Hoisting does not occur in let. | Hoisting occurs in var. |

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

/\* var name="chandu";

document.write(name); \*/

/\* let n="siva";

document.write(n); \*/

/\* const name="raghu";

document.write(name); \*/

/\* const name="vinod";

name="ramu";

console.log(name); \*/

function name(print){

if(print){

/\* let name="chandu";

const sname="rahul"; \*/

var name="rahul";

}

console.log(name);

}

name(print);

    </script>

</body>

</html>

**Example 1:**Here we will see the use of var.

<script>

    console.log(x);

    var x=5;

    console.log(x);

</script>

<script>

    console.log(x);

    let x=5;

    console.log(x);

</script>

**Output:**

ReferenceError: Cannot access 'x' before initialization

<script>

    // calling x after definition

    var x = 5;

    document.write(x, "\n");

    // calling y after definition

    let y = 10;

    document.write(y, "\n");

    // calling var z before definition will return undefined

    document.write(z, "\n");

    var z = 2;

    // calling let a before definition will give error

    document.write(a);

    let a = 3;

</script>

**Output:**

