**Javascript**

It is an object-based scripting language that is lightweight and cross-platform

1)Internal JavaScript

2)External JavaScript

1)Internal Javascript:-

a)Between body tag of Html:-

<!DOCTYPE html>

<html>

<body>

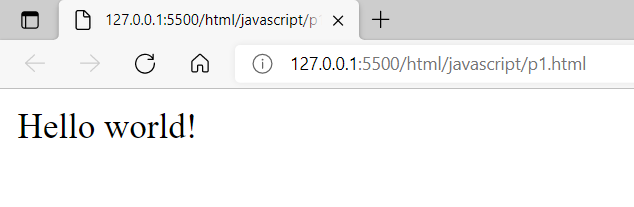
    <script type="text/javascript">

        document.write('Hello world!');

    </script>

</body>

</html>



b)in head tag:-

<!DOCTYPE html>

<html lang="en">

<head>

    <script type="text/javascript">

        function msg(){

            alert('Hello world');

        }

    </script>

</head>

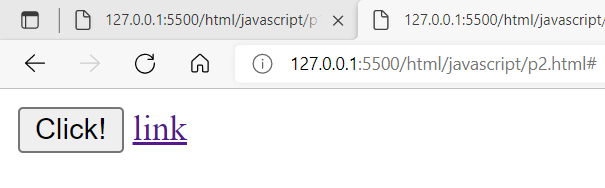
<body>

    <input type="button" value="Click!" onClick="msg();"/>

    <a href="#" onClick="msg();">link</a>

</body>

</html>



2)External JS:-

Step1:- create external js file “message.js”

function msg(){

    alert('Hello world');

}

Step2:-create html file and link js file

<!DOCTYPE html>

<html lang="en">

<head>

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

</head>

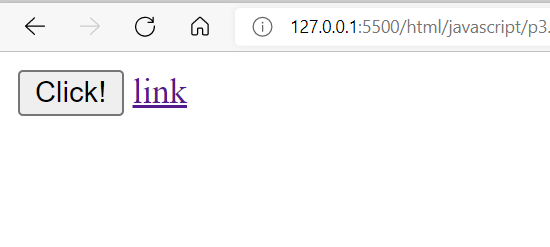
<body>

    <input type="button" value="Click!" onClick="msg();"/>

    <a href="#" onClick="msg();">link</a>

</body>

</html>



**Variables In JS**

* It is a container for storing data value
* Reserved words for variables

Var,let and const

* Identifiers are case-sensitive.

**Var keyword:-**

Ex:-

<!--var keyword-->

<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

    <script>

        var val='Hello JavaScript';

      console.log(val);

        //it can be updated

        val="Hello LWC";

        console.log(val);

        val=100;

        console.log(val);

        val=true;

        console.log(val);

//it can be redeclared

        var val=1000;

        console.log(val);

        //scope

        //its supports 2 type of scope

       // 1.global

       var subject="LWC";

       console.log("subject="+subject);

        //2.functional

          function abc(){

              var x=10;

              console.log("X="+x);

          }

          abc();

         // console.log(x);//error=> x is not defined

        //its not support to block level scope

       if(10==10){

            var y=20;

        }

        console.log('Y='+y);

    </script>

</body>

</html>

**Let keyword:-**

<!-- let keyword -->

<!DOCTYPE html>

<html lang="en">

<head>

    <script>

       let x=10;

       console.log('X='+x);

       //it can be updated

       x=20;

       console.log('X='+x);

       //can't be redeclared

     //  let x=30;//error=> x has already been declared

     //scopes

     //its supports global,fuctional and block scope

     //1.global scope

     let y=10;

       console.log('Y='+y);

       //2.functional scope

       function scope(){

           let z=20;

           console.log('Z='+z);

       }

       scope();

      // console.log('Z='+z);//error=> Z is not defined

      //3.block level scope

      if(10==10){

          let val='i am in block level scope';

          console.log('val='+val);

      }

    //  console.log('val='+val);//error=> val is not defined

      var temp1='i am in temp1';

      let temp2='i am in temp2';

      console.log('temp1='+temp1);

      console.log(window.temp1);//output

      console.log(window.temp2);//undefined

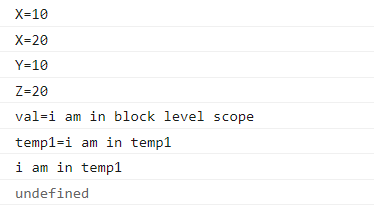
    </script>

</head>

<body>

</body>

</html>

****

**const keyword:-**

<!-- const keyword -->

<!DOCTYPE html>

<html lang="en">

<body>

    <script>

        const PI=3.14;

        console.log(PI);

        //it can not be updated

         //PI=6.14;//error

         //it can not be redeclared

        // const PI=4;//error=> PI has already declared

         //scope

         //it support global,functional and block level scope

    </script>

</body>

</html>

****

**Datatypes**

1.Number

2.String

3.boolean

4.bigint

5.undefined

6.null

7.object

Note:- Rest all the types are objects i.e.

Array,Date,Math,string etc…..

Ex:-

<!-- Data types in JS -->

<!DOCTYPE html>

<html lang="en">

<head>

    <script type="text/javascript">

        //number

        var x=10.5;

        console.log(x);

        console.log(typeof x);//number

        //String

        var str='silvercloud';

        console.log(typeof str);//string

        //boolean

        var isActive=true;

        console.log(typeof isActive);//boolean

        //bigint

        var y=12345n;

        console.log(typeof y);//bigint

        //undefined

        var z;

        console.log(typeof z);//undefined

        //null

        var w=null;

        console.log(typeof w);//object

        //objects

        var obj={name:"abc",age:23,"DOB":"01/06/1989"};

        console.log(obj.age);

        console.log(typeof obj);//object

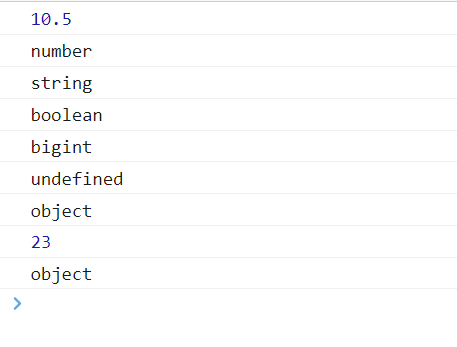
    </script>

</head>

<body>

</body>

</html>



Diff bet Null and undefined:-

Null:-

It’s a special data type which is used to represent nothing or empty value

Its defined explicitly

Its use with typeof then it returns object

Undefined:-

If a variable is declared but not initialized or assigned any value then js automatically initializes its value with undefined

Its use with typeof then returns undefined

Q=>Diff bet == and ===

<!-- Difference between == and === -->

<!DOCTYPE html>

<html lang="en">

<head>

    <script>

        //==

        console.log(3==3);//true

        console.log(3=="3");//true

        //===

        console.log('===');

        console.log(3===3);//true

        console.log(3==="3");//false

    </script>

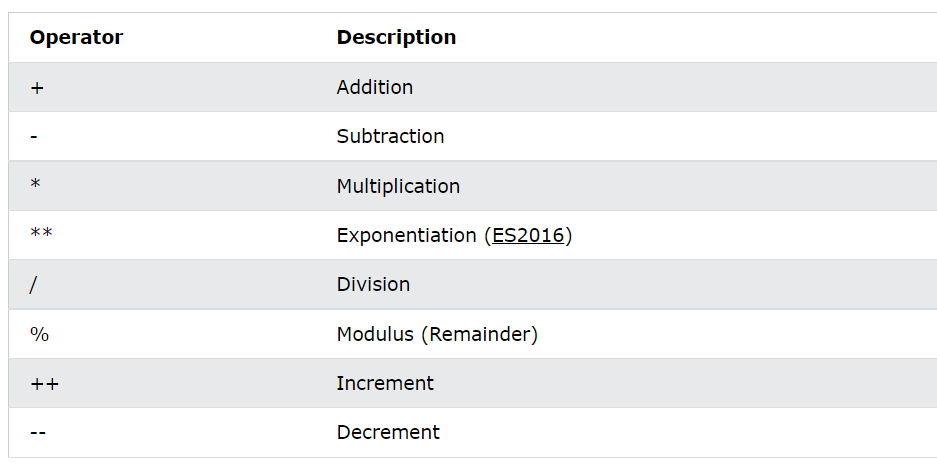
</head>

<body>

</body>

</html>

Operators in JS:-



**Function in JS**

1.No Argument Function:-

<!-- no argument function  -->

<!DOCTYPE html>

<html lang="en">

<head>

    <script>

        function msg(){

            alert('This is alert message');

        }

    </script>

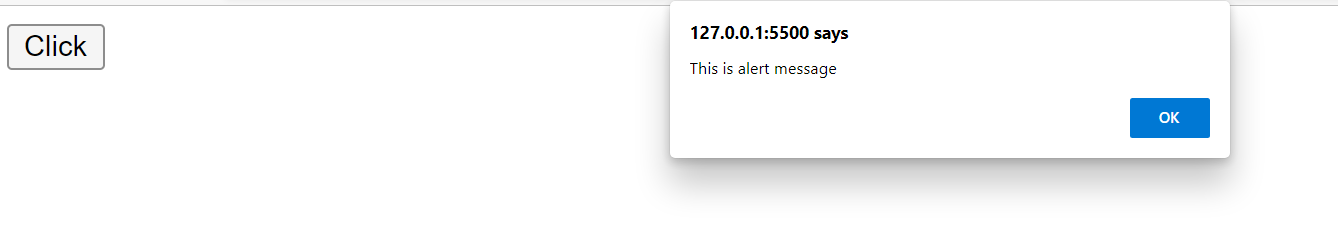
</head>

<body>

   <button onClick="msg();">Click</button>

</body>

</html>



2.Argument Function:-

a.myfun.js

function getCube(number){

    alert('Cube='+(number\*number\*number));

}

b.p11.html

<!DOCTYPE html>

<html lang="en">

<head>

    <script src="myfun.js"></script>

</head>

<body>

    <button onClick="getCube(2);">GET CUBE</button>

</body>

</html>