

CLIENT SIDE SCRIPTING USING JAVASCRIPT

JAVASCRIPT

- Developed by BRENDAN EICH in 1995 for Netscape Navigator
- Initially known as MOCHA
- Case sensitive
- Uses camelCase names for identifiers for better readability
- It is a program code embedded in < SCRIPT > tag within a HTML page
- It is supported by all browsers
- Javascript programs are embedded in Web pages and executed by browsers using javascript engine.
- Javascript is an interpreted language

Need for client side scripting

1. To reduce workload of the server
2. To reduce network traffic
3. To make pages interactive and dynamic

STARTING WITH JAVASCRIPT

A javascript code can be inserted in either **head or body** section of an HTML. The basic data types supported by javascript are

- Number { eg : -89.-.002,6845 etc}
- String { eg: "A", " Mera Bharath "}
- Boolean { true , false}

Variables are declared using the keyword **var**. Data type of variables is determined as and when values are assigned to these variables.

Operators in JavaScript is similar to C++ in the case of arithmetic, assignment, relational and logical operators

+ also serves as **string concatenation** operator.

Control statements such as **if , if-else, switch, for** and **while** is same as in C++.

There are no input/output statements in javascript.

document.write() ----- To display a text in the body section of web page

to access data from a form text box

document.formname.textboxname.value is used.

A sample javascript in a HTML file.

```
<html>
<head>
<title>
    Java script
</Title>
<SCRIPT language= "javaScript">
    document.write(" Welcome");
</SCRIPT>
</head>
<body>
<SCRIPT language= "javaScript">
    document.write(" Client Side Scripting");
</SCRIPT>
</Body>
</html>
```

Functions are available for common routines. Two types of functions

1. Built-in functions
2. User defined

Built-in Functions

1. alert() – used to display a message on the screen.
Eg: alert(" Welcome");
2. isNaN() - to check the value is a number or not. If not a number returns true otherwise false.
Eg : isNaN("13") returns false
3. toUpperCase() – returns the given string in uppercase.
Eg: var x,y;
x = "javaScript";
y = x.toUpperCase() ;
alert(y) ;
4. toLowerCase() – returns the given string to lowercase.

5. `charAt()` - returns the character at particular position.

```
Eg: var x;  
    x = "JavaScript";  
    y= x.charAt(4) ;  
    alert(y) ;    will display S
```

6. `length` (property) - returns the number of character in the string (a property will not have paranthesis after its name.)

```
Eg: var x;  
    x = "JavaScript";  
    y= x.length ;  
    alert(y) ;    will display 10
```

USER DEFINED Functions

User Defined functions contains a function header and a function call statement.

Syntax of Function header

Function `fn_name()`

```
{  
  
    Statements;  
  
}
```

Syntax of Function call statement

`Fn_name();`

A function header can be inserted in head or body section of HTML. A function can be called with the help of an event. An event is an action that triggers a function call. Commonly used events are

`onClick`

`onMouseEnter`

`onMouseLeave`

`onKeyDown`

`onKeyUp`

A HTML code containing script to find the square of a number using a user defined function.

```
<HTML>  
<HEAD>  
<TITLE>  
    Java script  
</TITLE>
```

```

<SCRIPT language= "JavaScript">
    function showsqr()
    {
        var num1, sqr;

        num1 = Number(document.FRM.txtNum1.value);
        sqr = num1 * num1;
        document.FRM.txtSum.value =sqr;
    }
</SCRIPT>
</HEAD>
<BODY>

<FORM name = "FRM">
<CENTER> PRINT the SQUARE of NUMBER</CENTER>
    Enter a number
        <INPUT TYPE = "text" name="txtNum1">
        <BR>
        <BR>
    Square of the given number is
        <INPUT TYPE = "text" name="txtSum" >
        <BR>
        <BR>
        <INPUT TYPE = "button" value="Square" onClick="showsqr()">

</BODY>
</HTML>

```

Method to include scripts in a webpage

- inside <HEAD> Section
- inside <BODY> section
- as external file (*.js)

When the browser renders an HTML-page and finds a `<script>` tag - it switches into Javascript mode and executes the code inside. Once executed the browser continues rendering the rest of the page.

If the HTML may be large, If you want a script to execute early, before the page is displayed, then the HEAD section is a good place. Putting scripts into HEAD is a common and easy practice

A script can also be at the bottom of page body. In this case it executes after the page is shown. So the contents of the webpage will be displayed faster. But this will not work properly if the script contains code to be executed while loading the page.

Usually, most JavaScript code is put into an external file, which is attached to HTML, like this:

```
<script src="/path/to/script.js">
</script>
```

For example let's us write a script for finding the sum of first n natural numbers and save it as **sumn.js** file. The content of the file will be like this

```
function sumn()
{
    var n, ctr, sum=0;
    n = Number(document.FRMSum.txtNum.value);

    for( ctr = 1; ctr<=n; ctr++)
        sum +=ctr;
    document.FRMSum.txtSum.value = sum;
}
```

Now create HTML file to accept the limit and display the sum.

```
<html>
<head>
<title>
JavaScript Sum function
</title>
<script type="text/javascript" src="sumn.js"> // linking external file
</script>
</head>
<BODY>
    <FORM Name="FRMSum">
        <CENTER> SUM OF FIRST N NATURAL NUMBERS</CENTER>
        Enter the limit.
        <INPUT Type = "text" Name = "txtNum">
        <BR>
        <BR>
        Sum Of first N Natural numbers =
        <INPUT Type = "text" Name = "txtSum">
        <BR>
        <BR>
        <INPUT Type = "button" Value = "Show" onClick="sumn()">
```

</BODY>
</HTML>