JAVASCRIPT





- JavaScript is the programming language of the Web.
- JavaScript is easy to learn.
- Using the JavaScript we can change and manipulate HTML Elements, Styles Etc...



- It is a client-side scripting language.
- Can be directly embedded into a Web page by writing the code inside the *script*> tag.
- Code can also be written in an external JavaScript (.js) file.
- JavaScript is one of the 3 languages all web developers must learn:
 - 1. *HTML* to define the content of web pages
 - 2. *CSS* to specify the layout of web pages
 - 3. *JavaScript* to program the behavior of web pages

The SCRIPT Tag

The *script* tag alerts a browser that JavaScript code follows.

It is typically embedded in the HTML.

<script>
statements
</script>

JavaScript code from an external text file can be written as;.

<script src="filename.js"></script>



The SCRIPT Tag

The *script*> tag alerts a browser that JavaScript code follows.

It is typically embedded in the HTML.

JavaScript code from an external text file can be written as;.



JavaScript can be placed in the **<body>** the **<head>** sections of an HTML page.

In HTML, JavaScript code must be inserted between <script> and </script> tags.

JavaScript Output

JavaScript can "display" data in different ways:,

- 1. Writing into an alert box, using window.alert().
- 2. Writing into the HTML output using document.write().
- 3. Writing into an HTML element, using .innerHTML.
- 4. Writing into the browser console, using console.log().

Using document.write() after an HTML document is fully loaded, will delete all existing HTML elements.

The document.write() method should be used only for testing.

Using window.alert()

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

Using document.write()

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



Using innerHTML

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



Using console.log()

```
<!DOCTYPE html>
<html>
   <head></head>
   <body>
     <h1>My First Web Page</h1>
     My first paragraph.
     <script>
        console.log(5 + 6);
     </script>
  </body>
</html>
```



Events

Events are things that happen, usually user actions, that are associated with an object.

The "event handler" is a command that is used to specify actions in response to an event.

Below are some of the most common events:

- 1. **onLoad** -occurs when a page loads in a browser
- 2. **onUnload** occurs just before the user exits a page
- 3. onMouseOver occurs when you point to an object
- 4. onMouseOut occurs when you point away from an object
- 5. onSubmit occurs when you submit a form
- 6. *onClick* occurs when an object is clicked

```
Eg:
```



JavaScript Can Change HTML Content

One of many HTML methods is *getElementById()*.

This example uses the method to "find" an HTML element (with *id="demo"*), and changes the element content (*innerHTML*) to "*Hello JavaScript*":

```
Eg: <a href="left"><a href="left"><a
```



JavaScript Can Change HTML Styles (CSS)

Changing the style of an HTML element, is a variant of changing an HTML attribute:

```
<!DOCTYPE html>
<html>
<head></head>
<body>
JavaScript can change the style of an HTML element.
<script>

var x = document.getElementById("demo");
x.style.fontSize = "25px";
x.style.color = "red";
</script>
</body>
</html>
```



Javascript Variables

- JavaScript variables are containers for storing data values:
- All variables must be identified with unique names.
- A variable can hold several types of data.
- In JavaScript you don't have to declare a variable's data type before using it.
- Creating a variable in JavaScript is called "**declaring**" a variable.
- You declare a JavaScript variable with the var keyword:

Eg: var carName;

- After the declaration, the variable has no value. (Technically it has the value of undefined)
- To assign a value to the variable, use the equal sign:

```
Eg: var carName = "Volvo";
var price = 100;
```



The general rules for constructing names for variables (unique identifiers) are:

- 1. Names can contain letters, digits, underscores, and dollar signs
- 2. Names must begin with a letter.
- 3. Names can also begin with \$ and _ .
- 4. Names are case sensitive (y and Y are different variables).
- 5. Reserved words (like JavaScript keywords) cannot be used as names.

When string are added, strings will be concatenated (added end-to-end):

If you add a number to a string, the number will be treated as string, and concatenated.





Javascript Operators

- > Arithmetic Operators.
- Comparison Operators
- Logical Operators
- > Assignment Operators



Arithmetic operator

- Is used to perform arithmetic operations on variables and literals.
- Contain following types:

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
	Decrement



Comparison operators

- Is used to compare two values and perform an action on the basis of the comparison.
- Contains following types:

<

>

<=

>=

==

!=

===

! = =

Less than

Greater than

Less than equal to

Greater than or equal to

Equal to

Not equal to

equal value and equal type

not equal value or not equal type



Logical operators

- Is used to evaluate complex expressions.
- It returns a Boolean value.

Contains following types:

&&	And
!	Not
11	Or



Assignment operators:

- Is used to perform arithmetic operations and assign the value to the variable at the left side of the operator.
- Contains following types;

Operator	Example
=	x = y
+=	x += y
-=	x = y
*=	$\boldsymbol{x} *= \boldsymbol{y}$
/=	$x \neq y$
% =	<i>x</i> %= <i>y</i>

Same As x = y x = x + y x = x - y x = x * y x = x / y

x = x % y

Conditional Statements

Conditional statements are used to perform different actions based on different conditions.

JavaScript supports the following conditional statements;

- 1. if statement
- 2. if...else statement
- 3. if...else if... statement
- 4. switch statement



if statement

if statement is used 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
}
```



```
Eg:
```

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



else Statement

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
```



```
<body>
Eg:
                 <button onclick="myFunction()">Try it</button>
                 <p id="demo">
                 <script>
                      function myFunction() {
                            var greeting; var time = new Date().getHours();
                            if (time < 12) {
                                  greeting = "Good morning";
                            } else {
                                  greeting = "Good evening";}
                            document.getElementById("demo").innerHTML =
greeting;
                 </script>
           </body>
```

else if Statement

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

Eg:

```
<body>
   <button onclick="myFunction()">Try it</button>
   <script>
      function myFunction() {
          var greeting; var time = new Date().getHours();
          if (time < 10) {
             greeting = "Good morning";
          }else if (time < 20) {
             greeting = "Good day"
          } else {
             greeting = "Good evening";}
          document.getElementById("demo").innerHTML = greeting;
   </script>
</body>
```



Switch Statement

The switch statement is used to perform different actions based on different conditions.

Syntax

switch(expression) {

case n: code block

break;

case n: code block

break;



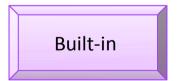
```
<body>
         <input type="text" id="myInput" >
         <button onclick="fun()"></button>
    <script>
         function fun()
             var text;
             var fruits = document.getElementById("myInput").value;
             switch(fruits)
                  case "Banana" : text = "Banana is good!";
                                  break:
                  case "Orange": text = "Orange is Good.";
                                  break;
                   case "Apple" : text = "Apple is Awesome";
                                   break:
                        Default: text = "I have never heard of that fruit...":
    </script>
</body>
```

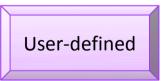
Eg:



Functions

- Functions are used to write the code that needs to reused.
- They optimize the performance of the code.
- Are a self-contained block of statements that have a name.
- Are of the following types:





User-defined functions:

Are defined according to the need of the user

• Functions:

- Are created by using the keyword, function, followed by the function name and the parentheses.
- Are normally defined in the head section of a Web page.
- Can optionally accept a list of parameters.
- Are created using the following syntax:

```
function [functionName] (Variable1, Variable2)
{
//function statements
}
```

A function is called by using the following syntax:

• A function returns a value by using the return statement as displayed in the following example:

```
function functionName()
{
var variable=10;
return variable;
}
```

