What is JavaScript ?

JavaScript is a lightweight, interpreted **programming language** with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.

The general-purpose core of the language has been embedded in Netscape, **Internet** Explorer, and other **web** browsers

JavaScript is:

- JavaScript is a lightweight, interpreted programming language
- Designed for creating network-centric applications
- Complementary to and integrated with Java
- Complementary to and integrated with HTML
- Open and cross-platform

Client-side JavaScript:

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.

It means that a web page need no longer be static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content.

The JavaScript client-side mechanism features many advantages over traditional CGI server-side scripts. For example, you might use JavaScript to check if the user has entered a valid e-**mail** address in a form field.

The JavaScript code is executed when the user submits the form, and only if all the entries are valid they would be submitted to the Web Server.

JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user explicitly or implicitly initiates.

Advantages of JavaScript:

The merits of using JavaScript are:

- **Less server interaction:** You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- **Immediate feedback to the visitors:** They don't have to wait for a page reload to see if they have forgotten to enter something.

- **Increased interactivity:** You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- **Richer interfaces:** You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

Limitations with JavaScript:

We can not treat JavaScript as a full fledged programming language. It lacks the following important features:

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript can not be used for Networking applications because there is no such support available.
- JavaScript doesn't have any multithreading or multiprocess capabilities.

Once again, JavaScript is a lightweight, interpreted programming language that allows you to build interactivity into otherwise static HTML pages.

JavaScript Development Tools:

One of JavaScript's strengths is that expensive development **tools** are not usually required. You can start with a simple text editor such as Notepad.

A **JavaScript** consists of JavaScript statements that are placed within the <script>... </script> HTML tags in a **web** page.

You can place the <script> tag containing your JavaScript anywhere within you **web page** but it is preferred way to keep it within the <head> tags.

The <script> tag alert the browser program to begin interpreting all the text between these tags as a script. So simple syntax of your JavaScript will be as follows

```
<script ...>
  JavaScript code
</script>
```

The script tag takes two important attributes:

- **language:** This attribute specifies what **scripting language** you are using. Typically, its value will be *javascript*. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.

- **type:** This attribute is what is now recommended to indicate the scripting language in use and its value should be set to *"text/javascript"*.

So your JavaScript segment will look like:

```
<script language="javascript" type="text/javascript">
  JavaScript code
</script>
```

Your First JavaScript Script:

Let us write our class example to print out "Hello World".

```
<html>
<body>
<script language="javascript" type="text/javascript">
<!--
   document.write("Hello World!")
//-->
</script>
</body>
</html>
```

We added an optional HTML comment that surrounds our Javascript code. This is to **save** our code from a browser that does not support Javascript. The comment ends with a "//-->". Here "//" signifies a comment in Javascript, so we add that to prevent a browser from reading the end of the HTML comment in as a piece of Javascript code.

Next, we call a function *document.write* which writes a string into our HTML document. This function can be used to write text, HTML, or both. So above code will display following result:

JavaScript Placement in HTML File

There is a flexibility given to include **JavaScript** code anywhere in an HTML document. But there are following most preferred ways to include JavaScript in your HTML file.

- Script in <head>...</head> section.
- Script in <body>...</body> section.
- Script in <body>...</body> and <head>...</head> sections.
- Script in and external file and then include in <head>...</head> section.

In the following section we will see how we can put JavaScript in different ways:

JavaScript in <head>...</head> section:

If you want to have a script run on some event, such as when a user clicks somewhere, then you will place that script in the head

JavaScript in <body>...</body> section:

If you need a script to run as the page loads so that the script generates content in the page, the script goes in the <body> portion of the document. In this case you would not have any function defined using JavaScript:

JavaScript in External File :

The *script* tag provides a mechanism to allow you to store JavaScript in an external file and then include it into your HTML files.

o use JavaScript from an external file source, you need to write your all JavaScript source code in a simple text file with extension ".js"

JavaScript Popup Boxes

JavaScript has three kind of popup boxes: Alert box, Confirm box, and Prompt box.

A confirm box

is often used if you want the user to verify or accept something.

When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.

If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

Prompt Box

A prompt box is often used if you want the user to input a value before entering a page.

When a prompt box pops up, the user will have to click either "OK" or "Cancel" to proceed after entering an input value.

If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null.

JavaScript Variables and DataTypes

JavaScript DataTypes:

One of the most fundamental characteristics of a programming language is the set of data types it supports. These are the type of values that can be represented and manipulated in a programming language.

JavaScript allows you to work with three primitive data types:

- Numbers eg. 123, 120.50 etc.
- Strings of text e.g. "This text string" etc.
- Boolean e.g. **true or false**.

JavaScript also defines two trivial data types, *null* and *undefined*, each of which defines only a **single** value.

In addition to these primitive data types, JavaScript supports a composite data type known as *object*. We will see an object detail in a separate chapter.

Note: Java does not make a distinction between integer values and floating-point values. All numbers in JavaScript are represented as floating-point values. JavaScript represents numbers using the 64-bit floating-point format defined by the IEEE 754 standard.

JavaScript Variables:

Like many other **programming languages**, JavaScript has variables. Variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container.

Before you use a variable in a JavaScript program, you must declare it. Variables are declared with the **var** keyword as follows:

You can also declare multiple variables with the same **var** keyword as follows:

Storing a value in a variable is called variable initialization. You can do variable initialization at the time of variable creation or later point in time when you need that variable as follows:

For instance, you might create a variable named *money* and assign the value 2000.50 to it later. For another variable you can assign a value the time of initialization as follows:

Note: Use the **var** keyword only for declaration or initialization.once for the life of any variable name in a document. You should not re-declare same variable twice.

JavaScript is *untyped* language. This means that a JavaScript variable can hold a value of any data type. Unlike many other languages, you don't have to tell JavaScript during variable declaration what type of value the variable will hold. The value type of a variable can change during the execution of a program and JavaScript takes **care** of it automatically.

.JavaScript Operators

What is an operator?

Simple answer can be given using **expression** *4 + 5 is equal to 9*. Here 4 and 5 are called operands and + is called operator. **JavaScript** language supports following type of operators.

- Arithmetic **Operators**
- Comparision Operators
- Logical (or Relational) Operators
- Assignment Operators
- Conditional (or ternary) Operators

JavaScript Functions

A function is a group of reusable code which can be called anywhere in your programme. This eliminates the need of writing same code again and again. This will help programmers to write **modular** code. You can divide your big programme in a number of small and manageable functions.

Like any other advance **programming language**, **JavaScript** also supports all the features necessary to write modular code using functions.

You must have seen functions like *alert()* and *write()* in previous chapters. We are using these function again and again but they have been written in **core** JavaScript only once.

JavaScript allows us to write our own functions as well. This section will explain you how to write your own functions in JavaScript.

Function Definition:

Before we use a function we need to define that function. The most common way to define a function in JavaScript is by using the function keyword, followed by a unique **function name**, a

Calling a Function:

To invoke a function somewhere later in the script, you would simple need to write the name of that function as follows:

Function Parameters:

Till now we have seen function without a parameters. But there is a facility to pass different parameters while calling a function. These passed parameters can be captured inside the function and any manipulation can be done over those parameters.

A function can take multiple parameters separated by comma.

The *return* Statement:

A JavaScript function can have an optional *return* statement. This is required if you want to return a value from a function. This statement should be the last statement in a function.

JavaScript Events

<body> and <frameset> Level Events:

There are only two attributes which can be used to trigger any javascript

when there is any event occurs on document level.

Attribute	**Description**
onload	Script runs when a HTML document loads
onunload	Script runs when a HTML document unloads

NOTE: Here script refer to any VBScript or JavaScript function or piece of code.

<form> Level Events:

There are following six attributes which can be used to trigger any javascript when there is any event occurs on form level.

Attribute	Description
onchange	Script runs when the element changes
onsubmit	Script runs when the form is submitted
onreset	Script runs when the form is reset
onselect	Script runs when the element is selected
onblur	Script runs when the element loses focus
onfocus	Script runs when the element gets focus

Keyboard Events

There are following three events which are generated by keyboard. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Description
onkeydown	Script runs when key is pressed
onkeypress	Script runs when key is pressed and released
onkeyup	Script runs when key is released

Other Events:

There following other 7 events which are generated by mouse when it comes in contact of any HTML tag. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, title elements.

Attribute	Description
onclick	Script runs when a mouse click

ondblclick	Script runs when a mouse double-click
onmousedown	Script runs when mouse button is pressed
onmousemove	Script runs when mouse pointer moves
onmouseout	Script runs when mouse pointer moves out of an element
onmouseover	Script runs when mouse pointer moves over an element
onmouseup	Script runs when mouse button is released

JavaScript RegExp Object

RegExp Object

A regular expression is an object that describes a pattern of characters.

Regular expressions are used to perform pattern-matching and "search-and-replace" functions on text.

Syntax

var patt = new RegExp(*pattern,modifiers*);

var patt = /*pattern*/*modifiers*;

- pattern specifies the pattern of an expression
- modifiers specify if a search should be global, case-sensitive, etc.

Modifiers

Modifiers are used to perform case-insensitive and global searches:

Modifier	Description
i	Perform case-insensitive matching
g	Perform a global match (find all matches rather than stopping after the first match)
m	Perform multiline matching

Brackets

Brackets are used to find a range of characters:

Expression	Description
[abc]	Find any character between the brackets
[^abc]	Find any character NOT between the brackets
[0-9]	Find any digit between the brackets
[^0-9]	Find any digit NOT between the brackets
(x|y)	Find any of the alternatives specified

Name:
Alphabets, numbers and space(' ') no special characters min 3 and max 20 characters.

var ck_name = /^[A-Za-z0-9]{3,20}$/;

Email
Standard email address
var ck_email = /^([\w-]+(?:\.[\w-]+)*)@((?:[\w-]+\.)*\w[\w-]{0,66})\.([a-z]{2,6}(?:\.[a-z]{2})?)$/i

UserId
Supports alphabets and numbers no special characters except underscore('_') min 3 and max 20 characters.
var ck_username = /^[A-Za-z0-9_]{3,20}$/;

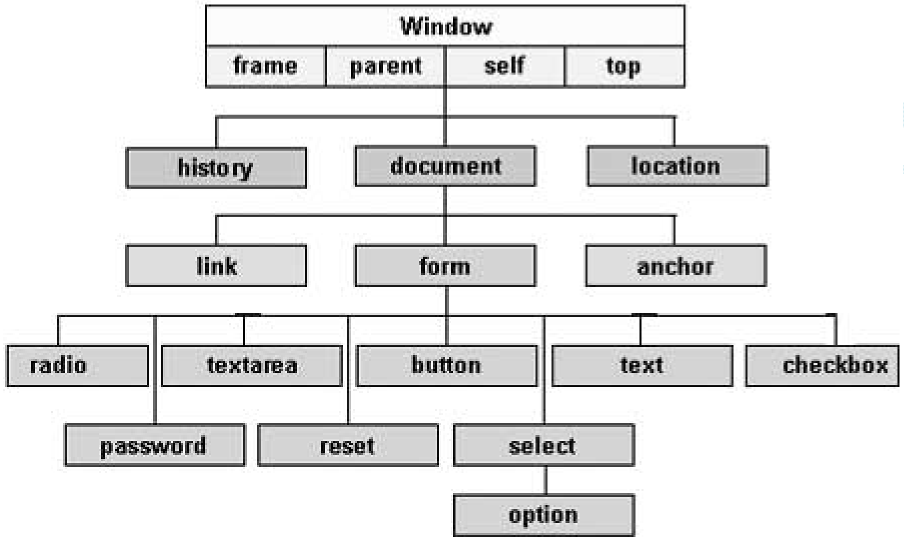
Password
Password supports special characters and here min length *6* max 20 charters.
var ck_password = /^[A-Za-z0-9!@#$%^&*()_]{6,20}$/;

The HTML DOM (Document Object Model)

A Document object represents the HTML document that is displayed in that window. The Document object has various properties that refer to other objects which allow access to and modification of document content.

The way that document content is accessed and modified is called the **Document Object Model**, or **DOM**. The Objects are organized in a hierarchy. This hierarchical structure applies to the organization of objects in a Web document.

- **Window object:** Top of the hierarchy. It is the outmost element of the object hierarchy.
- **Document object:** Each HTML document that gets loaded into a window becomes a document object. The document contains the content of the page.
- **Form object:** Everything enclosed in the <form>...</form> tags sets the form object.
- **Form control elements:** The form object contains all the elements defined for that object such as text fields, buttons, radio buttons, and checkboxes.



Finding HTML Elements

Method	Description
document.getElementById()	Find an element by element id
document.getElementsByTagName()	Find elements by tag name
document.getElementsByClassName()	Find elements by class name