HTML CSS & JAVA SCRIPT

CLASS NOTES

BY

MR. SUBBA RAJU SIR

NARESH TECHNOLOGIES

SR/N

HTML:

Web :- Collection of e-papers
father of web - Tim Berners Lee

Web + Network = Internet
Resource H/W + S/W
father of internet - Vint Cerf

Email :- Father is Sabeer Bhatia

SMTP - Simple mail transport protocal
MIME - Multipurpose internet mail extantion
FTP - file transfer program
TELNET - Telecommunication network.

Introduction to web environment

Web :- Collection of electronic pages or e-pages is called web.
Father of web is Tim Berners Lee.

Network :- Collection of hardwore and softwore resources.

Internet :- International network
father of Internet is Vint Cerf

W3C :- World wide web consorium (partnership)
It was founded in 1994 by Tim berners Lee.
It has four hundred plus members the following or popular

IBM, Microsoft, AOL, Apple, Adobe, Micromedia etc.

Email :- Electronic mail service

It is an inexpensive way to communicate with other internet users around world. It is working based on the following two

1] SMTP :- Simple mail transfer protocol

2] MIME :- Multipurpose internet mail extension

TELNET -

Telecommunications network or telephone n/w. It allows a user to log into a remote computer through a local system

FTP -

File transfer program later changed as file transfer protocol. It allows a user to transfer every kind of fight that can be stored from one system to another

HTTP :-

Hypertext transfer protocol to transfer html document in the world wide web.

Web Browser :-

It is client side software, it takes request from client to server. It brings response from server to client.

Popular web Browsers

In web environment the following list of web browser frequently used

Google crome (2008)

Internet explorer (1994)
Opera (1994)
(2005)

Define html.

It is specially designed text for web browser in english language. It has a following list of features
1] It is not case sensitive
2] It is simple english language we can create statistic web pages.
4] It is global languages
5] Browsers mother language
6] Simple and error free language

HTML Versions history :-

As per W3C standard there are following list of versions

1. html 1.0 (1994)
2. html 2.0 (1995)
3. html 3.0 (1997)
4. html 4.0 (1999)
5. html 5.0 (2008)
6. html 5.1 (2014) etc.

Define Tag :- It is popularly known as element the text placed between left angular brace (<) & right angular brace (>) is called as tag.

syntax < ----- >

Types of tags :-
Tags are classified into the following two types

1. Paired Tags
2. Non-paired Tags

1. Paired Tags :-
The tag that have both opening and closing tags are called as paired tags.

Example :- < html > --------- </html>
< body > --------- </body>

2. Non paired tags :-
The tag that have only opening no closing.

<hr>
<ing>

Example 1 :- How to create a web page.
To create web page we should follow following list of steps.

Step 1 :- Launch any text editor

Step 2 :- Enter required HTML source code

```
< html>
< head>
< title>
My First Webpage
</ title>
< / head>
< body>
Welcome to web world...
< /body >
< / html>
```

Step 3 :- Save with .html extension

Step 4 :- Right click on the saved file, open with any layer major web browser

Step 5 :- If any update required on the page, open with notepad format, do required changes and save it.

Step 6 :- Go to web page, Refresh it

In the above example every we page contain the following four certical elements..

1] < html>
2] < head>
3] < title>
4] < body>

HTML Basic elements :-

1]
 tag.

br stands for break. It is use to break a line shift to next line It is a non-paired tag br

Syntax :-

Example :-

```
<html>
<head>
<title>
br tag
</title>
</head>
<body>
Welcome to HTML...<br>
Welcome to HTML...
</body>
</html>
```

2]

Non-breaking space it is used to add were spaces between characters and words.
It is an entity or special charalter.

Syntax :-

Example :- 1 nbsp = 1 space

```
< html>
<head>
<title>
nbsp character
</ title>
</head>
< body>
 Welcome
 to & nbsp & nbsp & nbsp & nbsp & nbsp HTML
 </ body>
 </ html>
```

o/p Welcome to HTML

Working with presentational tag
This are popularly known as formated tags the following tags frequently we are using

bold	<b>	</b>
	<strong>	</strong>
italic	<i>	</i>
	<em>	</em>
Stricking effect	<s>	</S>
del	<del>	</del>
	<u>	</u>
Superscript	[	]
Subscript	_	
blockquote	<blockquote>	</ blockquote>

Small	< Small>	</small>
big	< big>	</big>
teletype	<tt>	</tt>
<q>	<q>	</q>
<Center>	<Center>	</center>

example :-

```
< html>
< head>
<title>
 Formated Tags
< /title>
< /head>
 < body>
 <b> It is in bold formate </b><br>
 < Strong > It is also in Bold formate </ Strong><br>
 <i> It is in Italic format </i><br>
 <em> it is also in italics format </em><br>
 <s> it is removed contents from page </s> <br>
 < Strike> it is removed contents < strike> <br>
 <del> it is removed contents </del> <br>
 <u> it is ~~removed~~ in underline format </u><br>
 it is the power of (100) < sup > 2 < /sup> <br> it is the
 Base of (100) < sub > 2 < / sub > <br>
< blockquote > it is Always special... </blockquote> <br>
< small> small font </small> <br>
<big> big font </big> <br
<tt> it is in teletyped format </tt> <br>
 <q> it is in Quotes </q> <br>
 < center> Pagecenter </center> <br>
     </body>
     </html>
```

* Attributes and Parameters

HTML attributes these are popularly known as properties, these properties can satisfied the following list of statements.

1] Attributes are always specified in the start tag
2] Attributes values or enclose in a single or double codes.
3] Attributes are special feature of tags
4] Each & every tag having its own attributes etc.

Parameters :-

These are the values assigned to attributes

Syntax

```
< tag attribute = "parameter">.
   ↑        ↑          ↑
```

Example < body bg color = "pink">.

```
<html>
<head>
<title>
Body tag
</titles>
</head>
<body>
It is the body seebion.....!!
</body>
</html>
```

* Working with body tag :-

It is a major element it contains text, hyperlink , special characters, tables, frames, forms etc It is a paired tag.

<body>.......</body>

* Body tags attributes and Parameters

Attributes	Parameters
bgcolor	color name / hexa decimal no
background	image path
text	color name / hexa decimal no.

Example :-

```
< html>
< head>
<title>
Body tag with attributes ....!!!
< / title>
< / head>
< body bgcolor = " Lightblue" text = "red">
 it is the body selection......!!
< /body>
 < / html>
```

Example :-

```
<html>
<head>
<title>
Body tag with attributes ---!!
</title>
</head>
<body background = "html5.png">
 It is the body section---!!
</body>
</html>
```

Example :-

```
<html>
<head>
<title>
Body tag with attributes ---!!
</title>
</head>
<body background = "C:\Users\Subbaraj\Pictures\
  fish1.gif ">
</body>
</html>
```

Example :-

```
< / html>
< head>
 < title>
 Body tag with attributes ---!!
 < / title>
 < / head>
  < body background =" file : /// C : \ Users \ subbaraj \
   pictures \ fish 1 . gif ">
  < / body>
   < / html>
```

⁕ Paragraph tag.

It is used to divide into different paragraph It is paired tag.

Syntax : <P> ---------- <\P>

Attributes	Parameters
align	left , right , center justify

```
<html>
<head>
<title>
paragraph tag.
</title>
</head>
<body>
<P>
HTML5 is a ....... (write any paragraph)</P>

HTML5 is a ...... </P>
</body>
</html>
```

Example :-

```
<html>
<head>
<title>
paragraph tag with Attributes
</title>
</head>
<body>
<P align = "left"> some text </P>
<P align = "center"> some text</P>
<P align = "right"> some text </P>
<P align = "justify"> some text </P>
</body>
</html>
```

❋ Font tag :-

It is used to display formatted tag it is paired tag.

< Font > ----- </font>

Attributes	Parameters-
Color	any color name or hexadceimal
size	1 to 7
face	arial , tahoma , ---- etc.

Example :-

```
<html>
<head>
<title>
font tag with Attributes
</title>
</head>
<body>
< font color = "blue" size = "5" face = "tahoma">
  welcome to formated text --- </font>
</body>
</html>
```

* Heading in html :-

There are six heading. all are paired tag

<h1>	</h1>
<h2>	</h2>
<h3>	</h3>
<h4>	</h4>
<h5>	</h5>
<h6>	</h6>

Example :-

```
<html>
<head>
<title>
Headings in HTML
</title>
</head>
<body>
<h1> Javascript </h1>
<h2> Javascript </h2>
<h3> Javascript </h3>
<h4> Javascript </h4>
<h5> Javascript </h5>
<h6> Javascript </h6>
```

Attributes	Parameters
align	Left , right , center

Example :-

```
< html>
< head>
< title>
 Headings with Attributes

 < /title>
 </head>
 < body>
 < h1 align = " left " > Javascript < /h1>
 < h2 align = " center"> Javascript </h2>
 < h3 align = " right "> Javascript </h3>
  < /body>
  < /html>
```

It is used to draw a line across the web page
It is non-paired tag.

<hr>

color	any color name/ hexa decimal
size	Pix
width	% or pix
align	left , right , center
noshade	noshade.

When we put noshade then color must be removed.

```
<body>
<hr color = "blue" size = "2px" width = "100px"
 align = "left">
<h1> javascript </h1>
 <hr color = "red" size = "4px" width = "200px">
 <h2> javascript </h2>
 <hr color = "green" size = "6px"
  width = "300px" align = "right">
 <h3> javascript </h3>.
```

The default width of the horizontal rule is 100%

Default alignment of the HR is center

noshade attribute will be applied only when we are not specifying the color attribute.

* Marquee tag :-

Using this tag we can create a scrolling text or scrolling image from left to right, right to left, top to bottom and bottom to top.

It is a paired tag.

Syntax : < marquee> -------</ marquee>

Example :-

```
< html>
< head>
<title>
Marquee tag with Attributes.
< /title>
< / head>
< body>
< Marquee> Text scrolling </ marquee>
< /body>
< / html>
```

* Attributes and Parameters :-

Attributes

behavior	" slide "	Start and stop as soon as text touches the margin
	" scroll"	Start completely and off one side (Default)
	"alternate"	Text bounce as soon as touch both sidemorgin
bg color	Color code	Speified the color as background
direction	" Left" " right" " up" " down"	Left to Right Right to left Bottom to Top Top to Bottom
width	"size-px"	Specifies width in marquee
hight	"size-px"	Specifies hight in morquee
Loop	"number"	Loop continues in Limited times
Scrollamount	"number"	Specifies speed to scrollon the text

Example :-

```
< html>
< head>
<title>
Marquee tag with Attributes
< / title>
< /head>
< body >
 < marquee behavior = " scroll"> SCROLL < / marquee>
 < marquee behavior = " slide" > SLIDE < / marquee>
 < marquee behavior =" alternate "> ALTERNATE </ marquee>
 < / body >
 < / html>
```

Example :-
②

```
< html>
< head>
<title>
Marquee tag with more Attributes
< / title>
< / head>
<body>
< Marquee behavior = " scroll" bgcolor = " orange">
 SCROLL / marquee>
< Marquee behaviour = " slide"> SLIDE </ marquee>
< Marquee behaviour = " alternate" bgcolor = "Lightgreen"
  width = "150px" height = " 400 px" direction =
  "down > ALTERNATE < / marquee>
 < / body>
 < / html>
```

Example :-

```
< html>
< head>
<title>
Marquee tag with more Attributes
< /title>
</head>
< body>
< marquee behavior = "scroll" bgcolor = "orange"
scrollamount = "1">
SCROLL < / marquee >
< Marquee behavior = " slide " Loop = "5"
scrollamount = "25"> SLIDE < / marquee>
< marquee behavior =" alternate" scrollamount
= "50"> ALTERNATE
< / marquee>
< / body>
</ html>
```

Example :- Marquee tag with JS Events.

```
<html>
<head>
<title>
Marquee tag with JS Events... !!
</title>
</head>
<body>
 < Marquee behavior = "Scroll" bg color = " Orange"
   Scrollamount = "1 "> SCROLL </ Marquee>
 < Marquee behavior = " Slide" loop = "5"
  scrollamount = "25" onmousedown = " this.stop ( )"
  onmouseup = " this.start ()"> SLIDE </ marquee>
< marquee behavior = "alternate" scrollamount = "50"
 onmouseover = " this.stop ()" onmouseout =
 "this. start () "> ALTERNATE </ marquee>
 </body>
 </html>
```

* Marquee tag with JS events .... !!

We can use also for the above example.

1] onmousedown = " this. stop ()"
2] onmouseup = " this. start()"
3] on-mouse over = " this. stop()"
4] onmouseout = " this. start ()"

* Pre tag :-

pre tag stands for pre formated text it displays unformated text on the web page including spaces, line breaks, taps & enters, It is a paired tag

Syntax :- <pre> --------- </Pre>

Example :-

```
<body>
<pre>
H  T    M     L
</pre>
</body>
```

o/p - H T M L

* < img > tag :-

It is used to insert images on the web page, it is a non-paired tag .

Syntax :-

<img>

Attributes	Parameters.
src	image path
border	pix
hight	pix or %
width	pix or %
align	left , right top , middle bottom
alt	any text
title	any text

Example :-

```
<body>
<img src " good morning gift " width = "200px"
height = " 200 px" alt = " sorry Img Not Existed"
  title = KSRaju '' border = "2 px">

< / body>
```

* HTML Links :-

Links are used to navigate easily from webpage to webpage or webpage to website etc

In html links are classified into following two types

1] Internal Links
2] External Links

① Internal Links

Linking within the page and within the website is called as internal linking

② External Links

Linking to external files like other documents other website or other webpages called external linking.

To create links we use Anchor tag. It is a paired tag.

Syntax :-

```
<a> -------- </a>
```

Attributes	Parameters
href	url (uniform resource locat
name	any name
target	-blank , -parent, anyname

Text links :-

A text link allows programmer to create text that acts as a link, so that when it is clicked on by a user, it will transfer them to another web page

Example① :-

```
<html>
<head>
<title>
 Text Anchors
</title>
</head>
<body>
<a href = "http:// www.nareshit.com">
                NareshIT </a>
< a href = "http:// www, nareshit.in">
             NareshIN </a>
< a href = "http:// www.seshajobs.com"> ITJobs
                </a>
 </body>
 </html>
```

O/p: Nareshit / NareshIN / ITJobs

Target Attribute :-
This attribute is used to display a page or website in a specific location.

Anchor tag with Target Attribute.

```
<html>
<head>
<title>
 Anchor tag with target Attribute
 </title>
 </head>
 <body>
 <a href = "http://www.nareshit.com">
    NareshIT </a>
 <a href = " http : // www. nareshit . in"
      "target = " _blank> Naresh IN </a>
 <a . href = " http:// www. seshajobs . com"
    target = " _parent "> IT Jobs </a>
  < a href = " http:// www . nareshservices . com"
    target = "      "> Talent
   Test</a>
   <body>
   <html>
```

* Picture links

A picture link allows the programer to create a picture that act as a link so that when it is click on by user it is transfer them to another page or site.

example :- Anchor tag with Image

```
<html>
<head>
<title>
 Anchar Tag with Images
</ title>
 </ head>
  <body>
  <a href = " http:// www. w3c . org ">
  < img src = " html . png " width = 100px height =
  100px > </a>
   <a href = " http : // www. whatwg . org ">
   < img src = " html5. png " width = 100 px
   height = 100 px>
   < /a>
    < / body>
    <html>
```

* Local Links :-

These links allows programmer to connect to client system resources

Local Resource example

```
<html>
<head>
<title>
Anchar Tag with LocalResources...
</title>
</head>
<body>
<a href = "file:///D:\HTML Class materials\
HTML Materials\images\phunny3.jpg">
Local Links </a>
<a href = "file :///D: \HTML Class
materials \ HTML Materials \images\ cat-fish.
jpg"> Local Links
  </a>
  </body>
  </html>
```

Working with html Tables :-

Tables are used to represent our data in a tabular format. The best way to split a page, with the help of tables. It is a paired tag.

Syntax :-

```
<table> ..... </table>
```

example :-

```
<table>
---
---
</table>
```

Table rows :-

Horizontal links represent as rows to represent table rows we use for tr tag
It is a paired tag.

Syntax :-

```
<tr> --------- </tr>
```

Example :-

```
<table>
<tr>
---
---
</tr>
</table>
```

Cells :~

Each row consist of a number of cells. Each cell defined by a tag. the tag looks like <td> The storting tag is <td> and the closing tag is </td> The intersection of rows and columns are called as cells. To represent table data we use td tag. td tag is a paired tag.

Syntax:- <td> ----- </td>

Table Headings.

To represent table headings we are using th tag

Syntax :~

<th> ---------- </th>

Example :~

```
<body>
<table border = 1 px>
<tr>
<th> StdNO </th>
<th> StdName </th>
</tr>
<tr>
<td> 1001 </td>
<td> Kumar </td>
</tr>
<tr>
<td> 1002 </td>
<td> Scott </td>
</tr>
</table>
</body>
```

Table tag Attributes and Parameters

Attributes	Parameters
border	pixels
bordercolor	any color / color code (hexa)
bgcolor	any color / color code (hexa)
background	image path
height	pixels or %
width	pixels or %
align	left , right , center
valign	top , middle , bottom
rules	rows , cols , all , none
cellspacing	pixels
cellpadding	pixels
rowspan	number
colspan	number

Table tag with more Attribute.

```
<body>
<table border = 1px border color = "blue". bgcolor
 = " yellow" background = " chrome.png">
 <tr>
  <th> StdNO </th>
  <th> StdName </th>
  </tr>
   <tr>
```

```
<td> 1001 </td>
<td> kumar </td>
</tr>
<tr>
<td> 1002 </tdt>
<td> scott </td>
</tr>
</table>
</body>
```

Table attributes at cell level

```
<body>
<table border = "1 px" bordercolor = "red">
<tr>
< th > StdNo </th>
<th> StdName </th>
</tr>
<tr>
<td> 1001 </td>
<td. bgcolor = " lightgreen " background
    = " chrome . png " > kumar </td>
</tr>
<tr>
<td> 1002 </td>
< td bgcolor = " lightblue"
    background = " html5 . png ">
    scott </td>
</tr>
</table>
</body>
```

Table tag with width and height property.

```
<body>
<table border = "1px" bordercolor = "red"
   width = 250px height = 200px>
<tr>
<th> Std NO </th>
<th> StdName </th>
</tr>
<tr>
<td> 1001 </td>
<td. height = 100px width = 100px.
   background = "chrom.png"> kumar </td>
</tr>
<tr>
<td> 1002 </td>
<td> scott </td>
</tr>
</table>
</body>
```

Rules Attribute :-

It support the following list of values

1. rows
2. cols
3. all
4. none

StdNo	StdName
1001	kumar
1002	scott

```
<body>
 <table rules="all" border = 1px>
 <tr>
  <th> StdNo </th>
  <th> StdName </th>
  </tr>
  <tr>
   <td> 1001 </td>
   <td> Kumar </td>
   </tr>
    <tr>
    <td> 1002 </td>
    <td> scott </td>
    </tr>
     </table>
      </body>
```

Cell Spacing :-

This attribute contrals the distance between cell

Syntax :

```
< TABLE CELLSPACING = "X">
```

Cellpadding :-

It control the distanee between text in the cell and the edge of the cell

Syntax :

```
< TABLE CELLPADDING = "X">
```

e.g with spaccing and padding.

```
<body>
<table border = 1px cell spacing = 10px
    cellpadding = 10px>
<tr>
 <th> Std No </th>
 <th> StdName </th>
 </tr>
 <tr>
 <td> 1001 < /td>
 <td> Kumar < /td>
 </tr>
 <tr>
 <td> 1002 < /td>
 <td> scott < /td>
 < /tr>
 ~~</body>~~
 < /table> </body>
```

Table tag with entities

Entities are special characters, these characters developed for special meaning. If you remove the text within the cell, cell will be disappere on the web page that time we should implement non breaking space for all browser compatibility.

```
<body>
<table border = 1 px>
<tr>
<th> Std No </th>
<th> StdName </th>
</tr>
<tr>
<td> 1001 < /td>
<td> &nbsp </td>
</tr>
<tr>
<td> 1002 </td>
<td> &nbsp </td>
</tr>
<table>
</body>
```

```
| StdNO |  | Std Name |

| 1001  |  |          |

| 1002  |  | scott.   |
```

Colspan and Rowspan

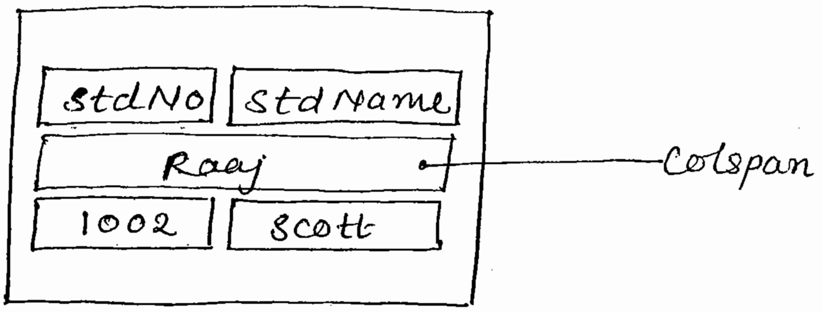
Using these attributes we can extend columns and rows across multiple other coloms and rows.

1. Column Span.

It extends cells on a horizontal row (left & right)

Syntax :-

```
<TD COLSPAN = "X">
```



e.g :-

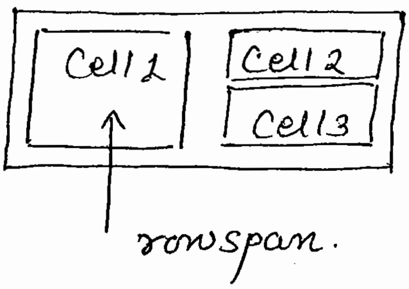
```
<body>
<table border = 1px>
<tr>
<th> stdNo </th>
<th> Std Name </th>
</tr>
<tr>
<td colspan = 2 align = "center">
Raaj </td>
</tr>
<tr>
```

```
<td> 1002 </td>
<td> Scott <td>
 </tr>
</table>
 </body>
```

Rowspan :-

It extends row on a vertical row up & down.

Syntax :- <TD ROWSPAN ='x'>



```
<body>
<table border = 1px>
<tr>
 <td rowspan ="2" valign ="top">
 Cell 1 </td>
  <td> cell2 </td>
  </tr>
  <tr>
  <td> Cell3 </td>
  < /tr>
  </table>
  </body>
```

Table data alignment :~

To align the data inside the table in the following two ways.

1. Horizontally.
2. Vertically.

(1) Horizontally :~

We can align the text horizontally in three way left (default), center and right

(2) Vertically :~

We can align the text vertically in the following three ways.

1. Top
2. middle.
3. bottom.

Working With html frames :~

Frames are used to embed multiple html files in a single browser window.

< frameset >' tag :~

Using this tag we can divided the webpage as a multiple frames. In each frame we can display another web site. frameset tag' is paired tag.

Syntax :~

< frameset > ---------- </ frameset>

Attributes	Parameters
rows	pix, %
cols	pix, %
border	pix
bordercolor	any color name/ Hexadecimal.

< frame> tag :~

This tag is used to called external webpages. It contains src property to specify the path of external web page.

Using frames we can place and view multiple files in a single window.
It is a non paired tag.

Syntax :- < frame>

Attributes	Parameters.
src	File path, External resource
name	any name
scrolling	yes, no, default

example :-

```
< frameset rows = " 50% , 50% ">
 < frame src = " http:// www. nareshit . com">
  < frame src = " http :// www. nareshit . in">
  < /frameset>
```

```
  < frameset rows = "50%, 50%" cols = "50%, 50%">
1 < frame src = " http: // www. nareshit . com">
2 < frame src = " http: // www. nareshit . in">
3 < frame src = " http:// www. nareservices . com">
4 < frame src = " http: // www. seshajobs . com">
   < / frameset.
```

| ① | ② |
| ③ | ④ |

Frame scroll bar :~

scrollbar attribute support the following parameters.

Yes - Turns the scroll bar ON
No - Turns the scroll bar off
Auto - Web page detect if needed.

Scrolling Attribute :~

```
< frameset rows = "50% , 50% ")
< frame src = " http : // www . nareshit . com"
        scrolling = "yes")
< frame src = "http : // www . nareshit . in"
        scrolling = "No")
< / frameset)
```

frame error :~

The mejority of browser not supporting frames that time we should keep a msg to the enduser while frames fails to Lowser it is a paired tag.

Syntax :~ < noframes) ..... < noframes)

Example :-

Frames doesn't supports body section.

```
< frameset rows = "50% , 50% ">
 <  frame src  =  " http :// www. nareshit . com ">
  <  frame  src =  "http :// www. nareshit . in">

      </ frameset>
      < body>
       < no frames>
       < P  style = ' color : red "> OOPs
    Your Browser  not  supporting  frames  Update and
    Try  ------  </P>
        < / noframes>
        < / body>
```

Browser support only upper or lower port, never support both port

Frameborder :-

This property support different frame borders with different sizes.

Syntax :-

< FRAMESET BORDER = " # ">

Example :-

```
< frameset rows ="50% , 50% " border = 20px
  bordercolor = "red">
< frame src = "http:// www. nareshit.com">
< frame src = " http:// www. nareshit. in">
< /frameset>
```

Working with html forms :-

Forms are used to create dynamic website end user able to interact directly with application It is a paired tag

Syntax :- <form> - - - - - - - </form>

Attributes :-

Form support the following list of attributes

Attributes	Parameters
name	any name
method	get, post
action	url (uniform resource locator)

Form tags :- Form support the following list of tag controls.

Tag	Description
<form>	defines a form for user input
<input>	Defines on input field data
<button>	Defines push button
< textarea >	Defines a text area (a multiline text input box)
< Label>	Defines a label to the description
< fieldset >	Defines a border to the input dat
<legend>	Defines a caption name write into fieldset.
<select>	Defines a drop-down select list box.
<option>	Defines an option value in the drop down box

* Types of form fields :-

Form fields are classified into the following two types.

1] Input fields
2] Select fields

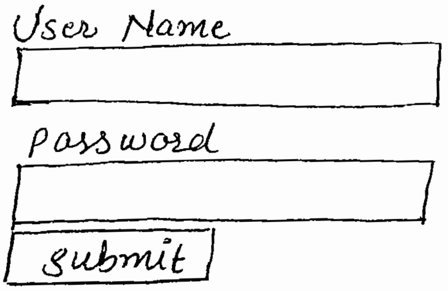
① Input fields :-

Form support the following list of i/p fields.

Field name	Keyword	Syntax
textbox	text	< input type = " text">
password box	password	< input type = "password">
cheekbox	cheekbox	< input type = "cheekbox">
radio button	radio	< input type = "radio">
submit button	submit	< input type = "submit">

reset button	reset	<input type="reset">
text area	textarea	< textarea > </textarea >

Example :-



```
< body>
< form>
< label> User Name : < / label> < br/>
< input type = ' text ' > < br/>
< label> password : < / label > < br/>
< input type = ' password '> < br/ >
< input type = ' submit '>
< / form>
< / body
```

Attributes	Parameters
name	any name
value	any value
size	pixels
maxlength	number
rows	number
cols	number
readonly	true, false
disabled	disabled
cheeked	cheeked
multiple	true false.

Example② form with more attributes

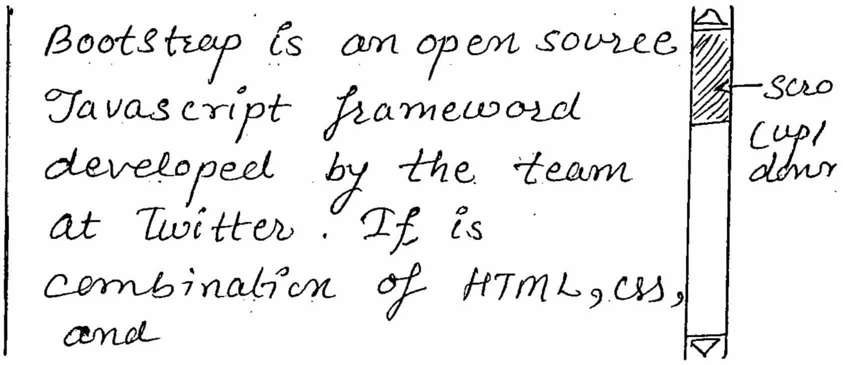
```
<body>
<form>
< label> User Name : < / label> < br/>
< input type = ' text ' name = " uname" value =
   " Enter Name " size = "5px" maxlength = "6"
  readonly = " true"> <br/>
  < label> password: < / label> < br/>
 < input type = ' password' name = " pwd"
 value = " Enter password "> < br/>
 < input type = 'submit' name = "sn"
 value = " Login " disabled = " disabled">
 < / form>
```

* Text area tag attributes

Attributes	Value	Description
cols	"Number"	Define the specify number of character visible in one line of text area
rows	"Number"	Define the specify number of lines visible in text area
name	" message"	Specify unique name for the input element

Example :-

```
<body>
 <form>
 < textarea rows = "6" cols = "23" name = " tarea"
   id = " tare 1 ">
   Some text ....!!
   < /textarea>
   < /form>
   < /body>
```



Bootstrap is an open source Javascript framework developed by the team at Twitter. It is combination of HTML, CSS, and

* Checkboxes :~

Check the required option(s)...

☑ Cricket □ Watch TV

example :~

```
<body>
<h3> check the required option (s).
<form>
<input type = "checkbox" name = "cricket"
 value = " cricket" checked = "checked"
 disabled / > Cricket
<input type = " checkbox" name =" watch TV"
 value = " watch " /> Watch TV
 </select>
 </form>
  </body>
```

* Radio Buttons

Select the Required Option

◉ Cricket ○ watch TV

```
<body>
<h3> select the Required Option ...
 --- </h3>
<form>
 <input type = "radio" name = "      " checked
 = " checked" />
 Cricket
 <input type = "radio" name = "      "
  disabled /> Watch TV
 </select>
 </form>
 </body>
```

* <fieldset> :-

It defines a group of form elements as being logically related. The browser draws a box around the set of fields to indicate that they are related.

It is a paired tag.

Syntax :-

```
<fieldset> ............ </fieldset>
```

* Legend :-

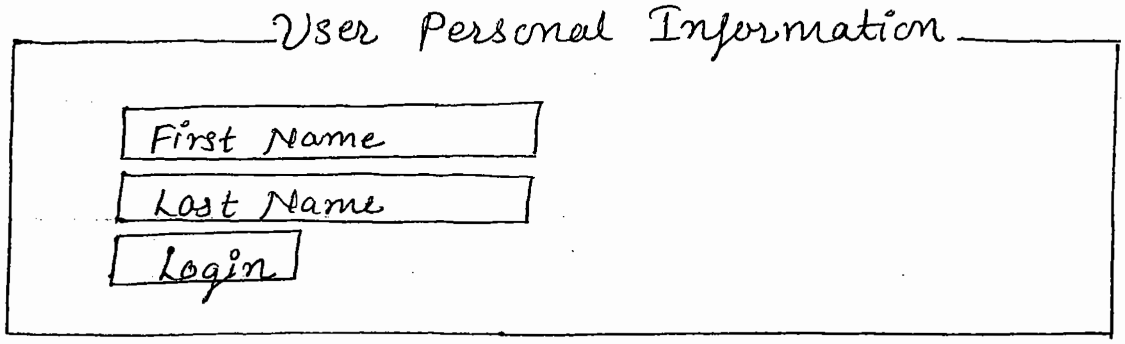
It is used with < field set> to give a tittle to each set of fields.

It is a paired tag.

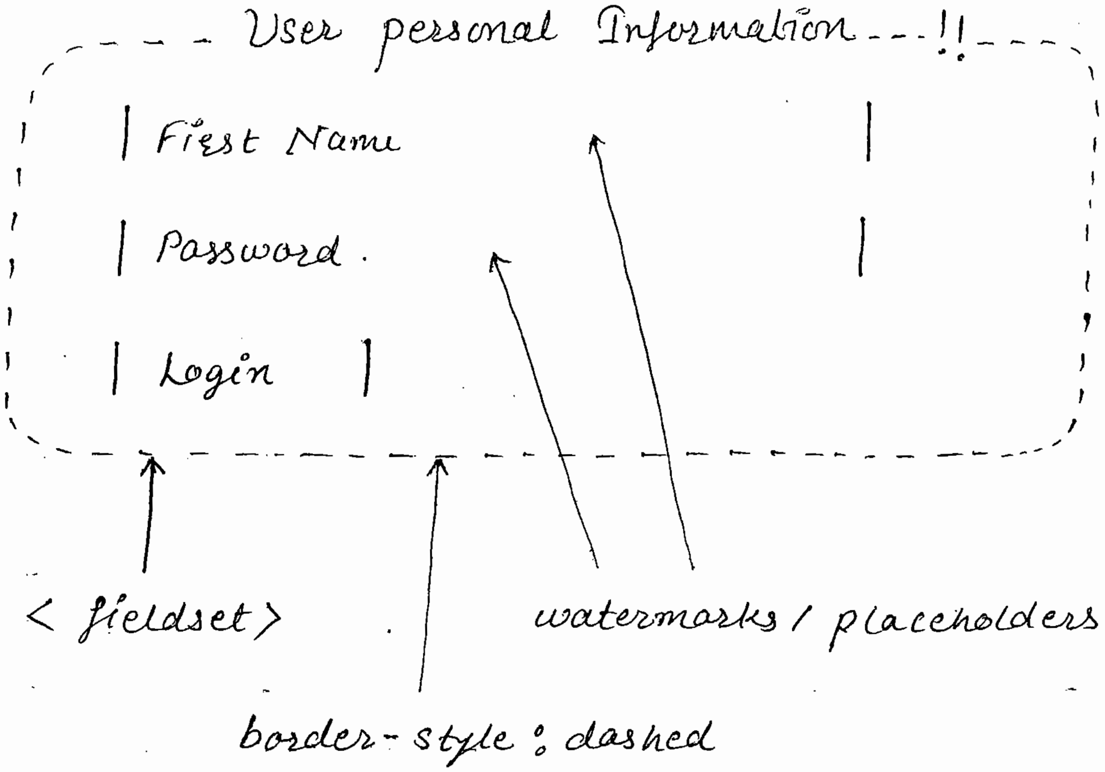
Syntax :-

```
<legend> -------- </ legend>
```

Attributes	Parameters
align	right , center , left



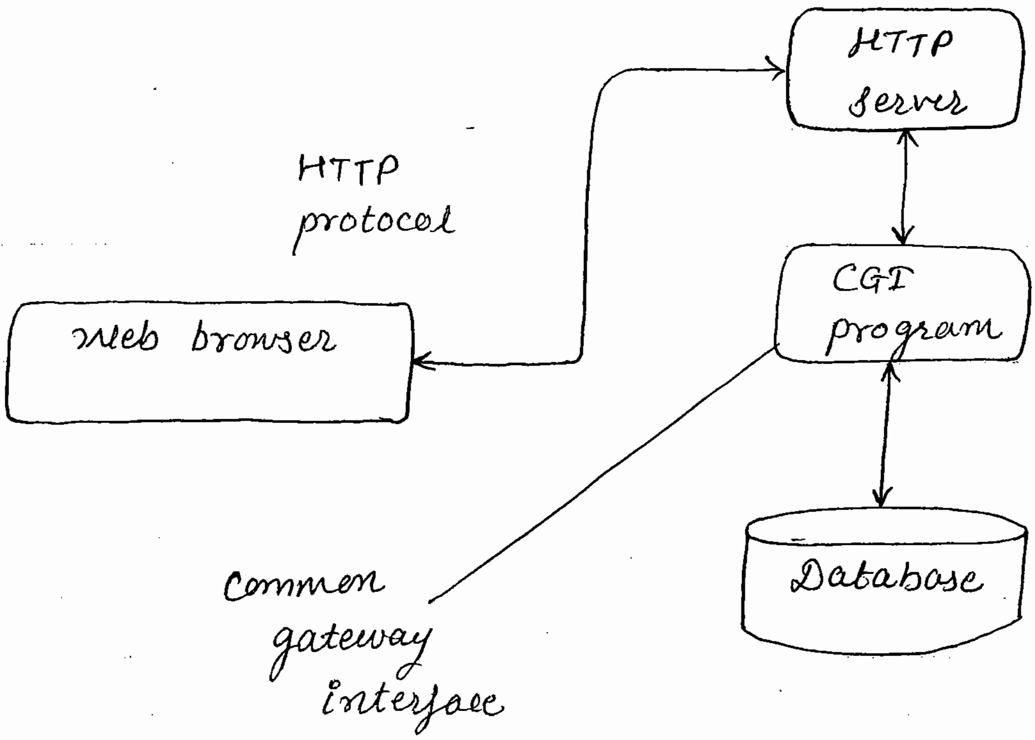
```
< body>
< form>
 < fieldset>
 < legend align = "center"> User personal Information..!!
 < / legend>
  < input type = 'text' name = "uname" value
     = "firstName">
   < /br >
   < input type = 'text' name = "uname" value
       = "LastName">
     < br/>
< input> type = ' submit' name = "      " value =
 " Login">
< / fieldset>
```



```
< body>
<form>
< fieldset style= ' border : 2px dashed #FF00FF;
   border - radius : 25px'>
 < legend align = " center" style = ' color : blue ;
  background - color : yellow'> User personal
  Information .. !! </ legend>
 < input type = ' text' name = ' uname" placeholder =
     ' firstName">
   < br />
   < input type = password' name = " uname"
   placeholder = " password">
```

```
< br/>
< input type = ' submit ' name = " sn" value =
 " Login">
 < / fieldset>
 < / form>
 < / body>
```

* Architecture of http.



Example ①

```
< body>
< form action = " nit. html " method = "get">
< input type = 'text' name = " user">
 < br/>
 < input type = " password " name = " pass">
  < br/>
  < input type = " submitt" value = " sign In">
  < / form>
  < / body>
```

* Get method satis the following list of statement.

1] GET request can be
2] GET request remain in the browser history.
3] GET requests can be bookmarked.
4] GET requests should never be used when dealing with sensitive data.
5] GET requests have length restrictions.

* Action Attribute :-

This attribute is used to specify the URL of the server page to which we want to send our form data.

Syntax :-

```
< form action = " ServerResource">
```

* Introduction to http.

It is designed to enable communication between clients and servers. It is TCP/IP based communication protocol, which is used to deliver virtually all files and other resources on the world wide web

* http Request Methods :-

http supports the following two request. methods

1] GET
2] POST

① GET Method :-

In this method we don't have security for our data and only limited data can be sent to the server page.

- This is the default method of the form.

Syntax :-

< form action = "nit. html" method = "get">

Note :-

Get method carry raw data between client to server

* Difference between get & post

GET	POST
1. Data is visible on URL address.	1. Not visible post information
2. Unsecured	2. Highly secured.
3. Excellent performance	3. Good performance.
4. Transfer limited amount of data	4. Transfer huge amount of data.
5. Unable to upload file	5. we can upload files

* http status messages.

In web environment these are several error messages frequently displayed, these messages are http status messages.

1XX - 199 → Information related messages
2XX - 299 → Successfull messages
3XX - 399 → Redirection messages
4XX - 499 → client side messages
5XX - 599 → server side messages

* Post Method :-

In this method, we have security for our data and we can send bulk of data to the server page.

Syntax :-

```
< form action = "nit.html" method = "post">
```

Explain :-

```
<body>
< form action = "nit.html" method = "post">
< input type = 'text' name = " user">
< br/>
< input type = " password" name = "pass">
< br/>
< input type = " submit " value = " sign in">
< / form>
< / body>
```

Post method can satisfy the following list of points.

1. Post requests are never catched
2. Post requests do not remain in the browser history.
3. post requests cannot be bookmarked.
4. Post requests have no restrictions on data length

Example ①~

```
<body>
<01>
<li> Javascript </li>
<li> Live script </li>
<li> VBScript </li>
<li> HTML5 </li>
<li> CSS3 </li>
<li> Bootstrap </li>
</ol>
</body>
```

Attributes	Parameters
type	i, I, a, A, 1
start	any number

Example ②

```
<body>
<01 type ="a">
<li> JavaScript </li>
<li> LiveScript </li>
<li> VB script </li>
<li> HTML5 </li>
<li> CSS3 </li>
<li> Bootstrap </li>
</01>
</body>
```

* Working With html List

In html list are classified into the following three types

1] Ordered list
2] Unordered list
3] Definition list

* Ordered list

It is also called as numbered list. It is used to give unumbering to the list items It is a paired tag.

Syntax :-
<ol> ------ </ol>

To specify the list items we use li tag. It is also a paired tag.

Syntax :-
< li> ------ </ li>

```
o/p :-   a  Javascript
         b
         c
         d
         e
```

start attribute only applicable for numbers.

example ③ :

```
<body>
<01 start = "5">
<li> Javascript </li>
<li> Live script </li>
<li> VB ...      </li>
<li> HTML5       </li>
<li> css3        </li>
<li> .....
</01>
</body>
```

* Unordered List :-

It is also called as bulleted list. It is used to give bullets to the list items. It is a paired tag.

Syntax :-

```
<ul> ........ </ul>
```

To specify the list items we use li tag. It is also a paired tag.

e.g ①

```
<body>
<ul>
<li>
<li>
<li>
<li>
<li>
<li>
</ul>
</body>
```

o/p :-

```
• Javascript
• Livescript
•
•
•
• Bootstrap
```

Attributes	Parameters.
type	disc, circle, square.

example :~ with attribute.

```
<body>
<ul type = 'circle'>
<li> ...
 ----
 ----
 ----
</ul>
 </body>
```

o/p :~

```
o Javascript
o ---

o ---
o ---
```

* Defination list

It is also called as descriptive list It is used to give definitions to defination terms. It is a paired tag

Syntax :~

```
<dl>-------</dl>
```

To specify definition data, we use dd tag It is a paired tag.

Syntax :~

```
<dd> ---------- </dd>
```

To specify definition term we use dt tag It is paired tag.

Syntax :~

```
<dt> ---------- </dt>
```

```
<body>
 <dt>
  <dt> Bootstrap </dt>
  <dd> sometext ..</dd>
  <dt> HTML </dt>
  <dd> some text </dd>
  <dt> CSS3  </dt>
   <dd> some text </dd>
   </dl>
   </body>
```

* HTML address tag :~

It is used for indicating an address the address renders in italic format.
It is a paired tag.

Syntax :~

```
<address > ........ </address>
```

Example :~

```
<body>
<address>
KS subbaRaj , <br>
Sr. Faculty Member ,<br>
Naresh iTechnologyes , <br>
Ameerpet , <br>
Hyderabad , <br>
TELANGANA.
</address>
</body>
```

* CODE :~

It allows the user to specify code or a command that generates a different font to signify the code. It is a paired tag.

Syntax :~

```
<code> ........ </ code>
```

```
<body>
<P> Normal text. It is related default font
<code>
HTML5 is New hypertext for mobile
Applications ..!!
</code>
</body>
```

* <NOBR> :-

It display a line continuously without any break. It is a paired tag.

<nobr> ------- </nobr>

Example :-

```
<body>
<nobr>
Line Never End! Line never End..!
-----
-----
</nobr>
</body>
```

* HTML Meta Tag Reference :-

Define meta tag :-

meta tags can be very useful for web developers with the help of meta elements we can declare meta data,

These elements are classified into several types but the following are frequently used.

1. keywords
2. description.

① Keywords :-

These keywords are implemented inside the head section, these are very usefull for search engines.

Syntax :-

< meta name = ' keywords ' content = List of related keywords" >

Example :-

< metaname = " keywords" content = " live cricket scores , cricket, ... cricket" / >

② Description :~

It is used to declare the description of web-page, it is very useful for search-engine results.

Syntax :~

< meta name = 'description' content = " Required Description">

Example :~

< meta name = " description content = " Check out live cricket scores, cricket headlines "/>

* html span tag :~

It is used to apply inline styles to a specific text.

It is used to display formated text on the web page.

It is a paired tag.

Syntax :~

< span> ------- </span>

HTML span tag Attributes

1. Style
2. Class.

e.g ①
```
<body>
<span>
Welcome to formated text - - <br>
Welcome to formated text
</span>
</body>
```

e.g ②. Span tag with style attribute.

```
<body>
< span style = ' color : blue ; font - family :
 tahoma ; background - color : yellow; font -size
 20 px' >
Welcome to formulated text - - <br/>
 welcome to - - - - - -
 </Span>
 </ body>
```

* HTML Div tag :~

div stands for division block, it can held other html element by using div tag we can design the web as multiple divisions.

It is a paired tag.

Syntax :-

<div> ---------- </div>

It supports the following list of attributes

1. align :-

align attribute accepts value as left, right or center to align to the content.

2. Style :-

It allows to provide the CSS inline styles div tag.

3. id :-

It allows to access the div element in scripting languages like Javascript.

4. Class :-

It accepts the name CSS class to apply the style defines in the CSS files.

① example :-

```
<body>
<div> DIV1 </div>
<div> DIV2 </div>
<div> DIV3 </div>
</body>
```

o/p :- DIV 1
DIV 2
DIV 3

② example :- using align attribute.

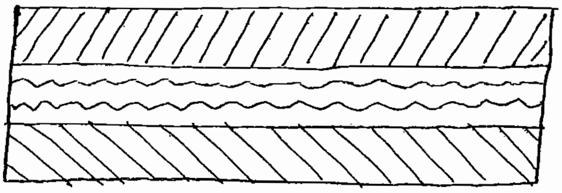
```
<body>
<div align = "left"> DIV 1 </div>
<div align = "center"> DIV2 </div>
<div align = "right"> DIV3 </div>
</body>
```

③ example :-

```
<body>
<div align = "left">
<b> If is in Bold format </b>
<br/>
<i> It is in Italic format </i>
</div>
```

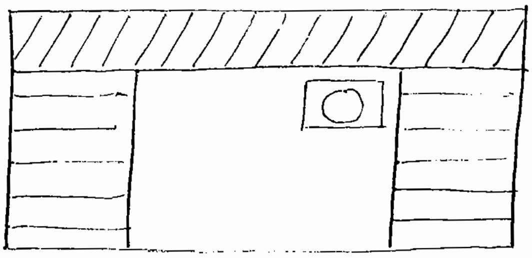
```
< div align = "center">
< b> It is in Bold format </b>
< br/>
<i> It is in Italics format </i>
</div>
< div align = "right">
< b> It is in Bold format </b>
< br/>
<i> It is in Italics format </i>
< /div>
< /body>
```

Example ④ :-



```
<body>
< div style = ' background - color : orange ; width :
    100% ; height : 100px'>
 < / div>
 < div style = background - color : white ; width :
    100% ; height : 100px'>
  < / div>
  < div style = ' background - color : green ;
   width ; 100% height : 100px'>
   < /div>
   </ body>
```

⑤ Example for



```
<body>
<div style =' background-color : orange ; width :
 100% ; height : 100px'>
<  /div>
 <div style = ' background- color : light-green ;
 width : 150 px ; height : 400 px ; float : left >
<  /div>
 <div style =' background-color : light blue ; width
355 px ; height ; 400 px ; float : left>
<marquee>
<  img src = " smiley6 . jpg" width = 100px height
  = 100px>
 </ morquee>
 </div>
 < div style =' background - color : Light-green :
   width ; 150 px ; height : 400 px ; float ;-right';
  </div>
  </ body>
```

* Working with table Layouts :-

Layouts are related to advanced tables we, can impliment inside the table header part menupart, footer part etc.

Header part

Menu Part   Main body part

List 1

List 2

List 3

List 4

Footer Part

```
<body>
<table width = "500px" border = "0">
<tr>
<td colspan = "2" style = " background-color:
  # FFFF00; height : 30px;">
Header Part
</td>
</tr>
<tr>
 <td style = " background - color : # 008029 ;
  width : 120px ; " >
  Menu Part <br/>
  List 1 <br/>
List 2 <br/>
List 3 <br/>
List 4
</td>
<td style =" background-color : # FF9900; height
200 px ; width : 380 px ;">
Main body part
  </td>
  </tr>
  <tr>
 <td colspan = "2" style = "background-color :
 # 000000 ; color : # FFFFFF ;">
 Footer Port
 </td>
 </tr>
 </table>
```

Introduction to dynamic hypertext markup language

It is collection of technology used together to Create interactive and animated website. It is a combination of following three technology

1. html
2. css
3. Java script

```
< html>
< head>
 <title>
  DHTML = HTML + CSS + Javascript
  </ title>
   < Style type = 'text/css'>
     div
   {
        color : blue ;
        background - color : light-blue ;
        font - size : 20px;
        font - weight : bolder
        }

        </ Style>
       < script type = ' text/javascript ' language
              = " javascript ">
        function MyMsg ()
       {
          alert (" welcome to DHTML");
           alert (" Bye ... ");

       }
           document . get Element By Id ("Myd").
            inner HTML = Date ();
       }
         </script>
         </head>
         <body>
         <P> click the button to display the
```

```
alert msg ....</P>
< button onclick =" My Msg ()"> click Me
                              </button>
<P id ='Myd'> Click the button to display
the system Date & Time ....</P>
< button onclick = " MyDate()"> ClickMe </button>
< marquee>
<img src =" goodmorning.gif" width= 100px>
< marquee>
 < div> welcome to DHTML Tech .... </div>
 < /body>
  < /html>
```

CSS

CSS :-

It is style design to design user Interface more effectively.

Features of CSS.

CSS support the following list of features

1] flexibility
2] Code Rendering
3] Accessibility
4] Easy manage
5] Global change
6] Save a lot of time
7] Easy maintainance
8] Inline / Style sheet
9] Internal style sheet
10] External style sheet etc.

Structure of CSS.

As per W3C standard CSS has the following detaile structure

```
<html>
<head>
        <style type = " text / CSS ">

  {
    ----
    ----
  }
```

```
< / style>
< / head>
     < body>
     - - - -
     - - - -
< / body>
< / html>
```

CSS syntax :-

In CSS syntax is divided into the following three parts

1] selector
2] property
3] value

① Selector :-

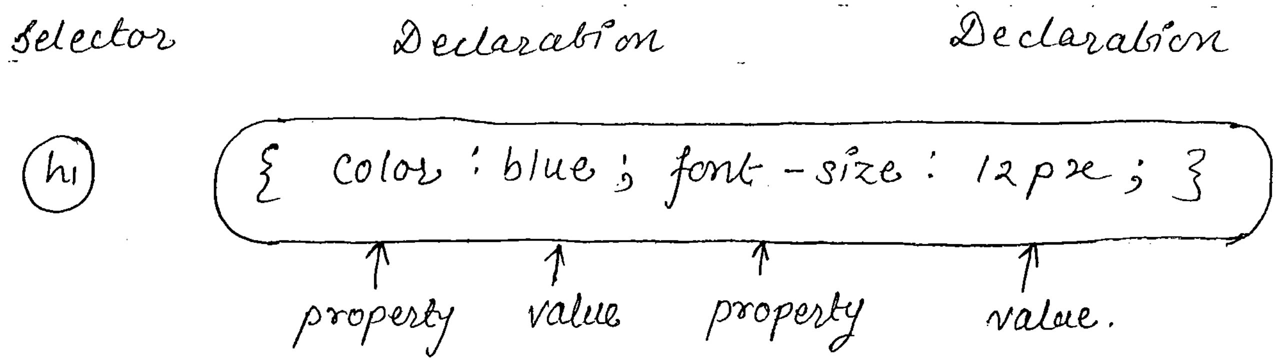
It is normally the html element.

② property :-

It is a kind of attribute you wish to change.

③ value :-

Every property has the value



Example :-

```
<html>
<head>
<title>
 Cascading style sheets
</title>
<style type = 'text/css'>
   h1          ← selector
 {

   color : blue;    ← value

 }
 </style>            property
 </head>
 </ body>
 <h1> Welcome to style sheets .. </h1>
 </body>
 </html>
```

Inline style sheet :-

We can specify styles inside the tag in the body port, these styles will be applied only for that perticulor line.

e.g.

Example :~

```
<body>
<i style = 'color : blue'> welcome to style
                                    sheets </i>
<p style = 'color : green'> welcome to style
                              sheets </P>
< b style = 'color : red'> welcome to style
                                 sheets </P>
</body>
```

* Internal Style Sheets :~

These are popularly known as embedded style sheets. If u specify the style in our HTML file itself then they are called as internal style.

These styles can not be used in other files

Syntax :~

```
<html>
<head>
    <style type =" text/css">
    </style>
</head>
 <body>
 </body>
 </html>
```

Example :~

```
<head>
 < style type = 'text / css'>
 p
 {
   color : blue ; font - weight : bold ;
   background - color : yellow ;
   font - size : 30px ; font - family : tahoma ;

  }
   < / style >
   < / head >
   < body >
   <p> welcome to internal styles --- </p>
   <p>
   <p>
    < / body >
    < / head >
```

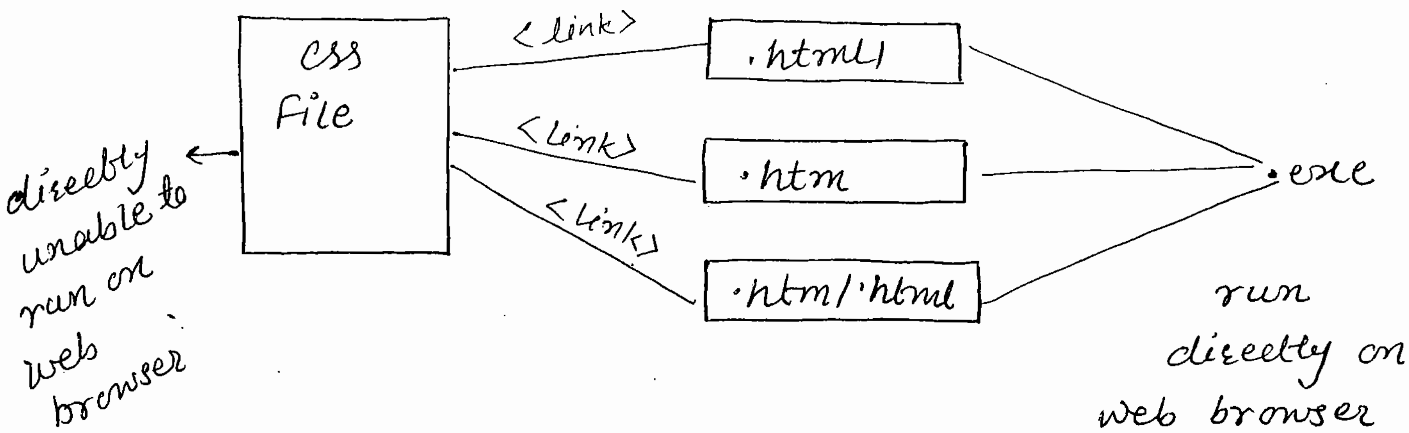
* External style sheet :

If we declare the styles out side our html file, then they are called external styles
These styles can be reusable on more than one file.
These file extension is .css

Syntax :~

```
        <head>
          < Link rel = " style sheet " href = " # " type
   = " text / css ">
        </ head>
```

External CSS Architecture


CSS File
<link>
<link>
<link>
.html
.htm
.htm/.html
.exe
directly unable to run on web browser
run directly on web browser

① Creating .css file

```
div
{
 font - size : 3cm;
 font - family : tahoma;
 text - decoration : underline;
 Color : blue;
 }
```

Save with .css extension

② Creating HTML file

```
<html>
<head>
< link rel=" stylesheet" href=" Hello.css">
< /head>
<body>
<div> welcome to external style sheets <div>
< /body>
< /html>
```

Save with .htm or .html extensions & run on any major web browser ---- !!

* Working with CSS selectors.

Selector means styles reusability, css supports difficult types of selectors

1] tag selectar
2] ID selector
3] Class selector
4] Grouping selectors
5] Universal selector

1] tag selector :-

These are popularly known as time selectors. It matches the name of document lagguage element type

Syntax :

```
        div
        {
          Styles
          Styles
          Styles
        }
```

Example :-

```
<head>
< Style type = ' text / css'>
p          ←      Tag or type selector
{
 color : blue;
background - color : yellow;
 font - family : tahoma;
 font - size : 30px

}
 < / style>
 < /head>
  < body>
 <P>  Welcome to Tag or Type selector </P>
 <P>  ———— " ——                       </P>
 <P>  ———— " ——                       </P>
  </body>
```

2] ID selectors.

It is used to specify a style for a single or unique element.

The id selector uses the id attribute of the HTML element, and is defined with #.

Syntax :

```
<style>
 # div
{
  styles
 styles
 styles
}
< /style>
```

Example :-

```
<head>
<style type = 'text/css'>
# h2       ←   ID selector
{
   color: blue; font-weight = bolder;
   font - size : 30px;
   font - family : tahoma;
}
< /style>
< / head>
< body>
```

```
<p  id = "h2"> Welcome to ID selectors </P>
<P> ID selector not affected </P>
</body>
```

* ID selector with prefix :-

```
<head>
<style type = 'text/css'>
div # h2
 {
   color : blue ; font - weight : bold;
  font - size : 30px; font - family : tahoma;
   text - documentation : overline;

 }

  < /style>
  < / head>
   < body>
   < div id = " h2"> Welcome to ID selectors
   < /div>
    <P> ID selectors not effected </P>
    < / body>
```

ID Selector with JavaScript

```
< head>
< style type = 'text / css'>
# div 1
 {
   color : blue ; font - family : Arial Rounded MT;

 }
  < / style>
  < script type = 'text / javascript'>
   function Myvalue ()
  {
    document . get Element By Id ("div 1) .inner
       HTML = "jQuery";
  }
  < / script>
  < / head>
  <body>
  < P> click the Button to change the name of
   the course .... </P>
  < div id = "div> Bootstrap < / div>
  < div id = "div 1> javascript < / div>
  < button onclick = "My value ()"> click Me
       < / button>
  < / body>
```

3) Class selector :-

It is used to specify style for group of elements it is always defined with a " . "

Syntax :-

```
        div
        {
          styles
          styles
          Styles
        }
```

Example :-

```
    <head>
    <style type = ' text/css'>
     p.div
      {
       color : red;
       font-size : 20px.;
       }
       </style>
       </head>
       <body>
       <p class = "div"> welcome to class selectors
         </P>
       </body>
```

* class selector with Javascript :-

```
<head>
<style type =' text / css'>
 .div
{
  color : blue ; font-family : Arial Rounded MT ;
}
 </style>
 <script type = 'text / javascript'>
  function myvalue ()
 {
  document .getElement ById ("div"). inner
   HTML = " jQuery";
 }
  < / script>
  < / head>
  < body>
  <P> Click the button to change the name
   of the course ----- </P>
  < div id = " div " class = "div"> Bootstrap
    < /div >
 <  div id = "div1" class = "div"> Javascript
                                    </div>
  < button onclick = "myvalue ()"> Click Me
                                   </button>
  < /body>
```

* ID & Class selectors

```
<head>
< Style type =' text/css'>
# div
{
  Color : green;
  font - family : Arial Rounded MT ;
  font-size : 20 px;

}
 .div1
 {
  color : blue;
  font - family : tahoma;

}
 </ Style>
 </ head>
 <body>
 <div id = "div"> welcome to multiple selectors
                         ..... </div>
< div class = "div1"> Welcome to multiple
                     selectors ..... </ div>

 </ body>
```

* Grouping selectors :-

These selectors are seperated with the help of " . " notation. The following example express grouping selector usage.

Example :-

```
h1 { font - family : sans - serif }
h2 { font - family : sans - serif }
h3 { font - family : sans - serif }
```

is equivalant to :

```
h1, h2, h3 { font - family : sans - serif }
```

Example :-

```
<head>
<style type = 'text/css'>
#div , .div1
{
   color : blue;
   font - family : tohoma;
}
</Style>
</head>
<body>
<div id = "div") welcome to grouping selectors...
     </div>
<div class = "div1 "> welcome to grouping
     selector .... </div>
</body>
```

Working with CSS border :-

CSS supports the following list of border property.

1. Border color :- It specify the color of border.

2. The border-style :- It specifies whether a border should be solid, dashed line, double line, or one of the other possible values.

3. The border width :- It specifies the width of a border.

Example :-

```
<head>
<style type =' text/css'>
div
{
  border-color : blue;
  border - style : solid;
  border - width : 3px;
}
 p
 {
   border - color : green;
   border - style : double;
   border - width ; 3px;
}
```

```
h6
{
   border-color : red
   border-style : dashed
   border-width : 3px;

}
  </style>
  </head>
  <body>
  <div> It is solid Border ..! </div>
   <p> It is double Border ..! </p>
   <h6> It is dashed Border ..! </h6>

   </body>
```

Java Script

* Introduction to script :-

Script is a type of programing language that can be used and client location

Types of script
Scripts are classified into the following two types

1] Client side script
2] Server side script

Client side script
These scripts are getting executed within the web browser

e.g :- Java script live script, VB script

Server side script
A script which executes within the web servers like

IIS → Internet information services
Apache , Tomcat etc
.NET → iNetpub --> IIS
PHP --> htdocs
AJAVA --> WebApps

ex : php , Jsp , Asp

Difference between Scripts and languages.

Scripting	language
] weakly typed programming or loosely typed programming or lightweight	1] Strictly typed programming
] Easy to understand	2] Complex to understand.
] Simple to develop	3] Trouble to develop
] no headers files required	4] headers files mandatory.
] no libraries require	5] libraries compulsory
] no special compiler require	6] special compiler mandatory.
] Client side validation	7] server/ client side validation / verification.
] poor graphics	8] rich graphics.
] Ex. live script, Java script, VB script, perl script, shell script Jscript, Python script etc.	9] Ex. FORTRAN, BASIC, COBOL, PASCAL, ALGOL, CPL, BCPL, B, C, C++ Java, C# etc.

Introduction to Javascript :-

Javascript is power scripting language of the web. It supports all modern web browsers, modern devices.

Features of Javascript :- Javascript supports the following list of features.

1] It is client side validation purpose.

2] It can react to events

3] It can be use to validate data.

4] It can be use to create cookies.

5] It is designed with light weight features

6] It is open source or cross platform ....
etc.

Javascript syntax :- Javascript consists of javascript statements that are placed within the script.

```
Syntax : < script language = "javascript" type = " text/javascrip.
           -----
           -----
         < /script>
```

```
Syntax : < script type = "text/ javascript">
           statements
           statements
           statements
           </ script>
```

```
Syntax : < script language = " javascript">
             Statements
             statements
             Statements
           < / Script>
```

Syntax 4 :-

```
<script>
statements
statements
statements
</script>
```

History of Javascript :- Javascript versional name is live script, It was developed by netscape corporation later it is renamed as java script developed by Brendan Eich. These syntaxes are very close to "C" programming language.

Java script structure :- As per W3C std, javascript has the following detailed structure.

```
<html>
<head>
<title> example </title>
<script language = "javascript">
<!--
  ----- >
</Script>
</head>
<body>
 ....
 ....
</body>
</html>
```

Example :-

```
<html>
 <head>
 <title>
  working with JS
  </title>
  < script type = 'text / javascript' language = "javascript";
   function MyMsg()
   {
     alert ("welcome to JS");
   }
   < / script>
   </head>
   <body>
   <P> click the button to display the alert msg ... </P>
   < button onclick = "MyMsg()"> click Me < / button>
   </body>
   </html>
```

* save the file with .html or .htm extension & Run on any major web browser.

Javascript Comments :- Comments are non-executable statements or ignore statements using these comment notation we can declare customized statements or user defined statements within the source code.

Types of comments :- Javascript supports follow two types of comments

1. Single line comments
2. Multi line comments

1] Single line comments :- These comments are restricted to a specific line. These are denoted with "//"

example :-
```
< head>
< script language = "javascript">
// alert ("Welcome to LS");
< / script>
< / head>
```

o/p = nothing.

2] Multi line Comments These comments are applicable to one or more lines. These are denoted with /* */

e.g.
```
< head>
<script>
/* alert (" welcome to LS");
alert (" welcome to LS");
< / script>
< / head>
```

document write () method :-

The write () method writes HTML expressions or javascript code to a document

Syntax : document write (exp1, exp2, exp3 ....)

Example :
```
< head>
< script type = 'text / javascript'>
document write (" welcome to javascript");
< / script>
< / head>
```

1] document → object → webpage
2] write () → is method → webpage level.

example
```
<head>
      <script type = 'text / javascript'>
      document . write ("<h1> Hello word ! </h1>
      <P> Have a nice day ! </P>");
      </script>
      </head>
```

example with

```
      <head>
      <script type = ' text / javascript '>
      document . write ( " welcome to JS ");
      document . write ( " < br/>");
      document . write (" welcome to LS");
      </ script>
      </head>
```

document writeln() method :-

The writeln() method is identical to the write(method, with the addition of writing a newline character after each statement.

Syntax :- document . writeln (exp 1, exp 2, exp 3, ...)

Example:
```
<head>
      < script type = 'text / javascript '>
      document . writeln (" Welcome to JS ");
      document . writeln (" Welcome to LS");
      < / script>
      < / head>
```

* DHTML

```
<head>
<script type='text/javascript'>
document.write("<h1 style='color:blue;
font-size : 35px ; font - family: tahoma'>
Welcome to JS </h1>");
document.write ("<font color='green' size='6'
face=' century Gothic'>
Welcome to LS </font>");
</Script>
</head>
```

Working with javascript string :-

In javascript a string should be in single or double quotes double quotes inside single quotes valid, single quotes inside double quotes. valid.

Example :-

```
<head>
<script type ='text/javascript'>
document.write ("javascript is client side script");
document.write ('Livescript is javascript');
 document.write ("<br/>");
 document.write ("<br/>");
document.write (" Livescript is 'java' script");
document.write ('<br/>');
document.write (' Livescript is "Java" script');
</script>
</head>
```

Javascript strings with escape sequences.

An escape character is consist of backslash "/" symbol with an alphabet. The following are frequently using escape characters

1]	\n	:	Inserts a new line
2]	\t	:	Inserts a tab
3]	\r	:	carriage return
4]	\b	:	Backspace
5]	\f	:	form feed
6]	\'	:	single quote
7]	\"	:	Double quote
8]	\\	:	Backslash

Example :
```
<head>
   <script type = ' text / javascript'>
    document . write (" Livescript is \" Java \" Script ");
    document . write ('<br/>');
    document . write (' Livescript is \' Java \' Script ');
    < / script>
    < / head>
```

* Difference between window . document . write & document. write

There is no difference between these two statements , window is highest level object , it contains child objects & their methods

child object / sub object
↓
window . document . write ( );
↓ ↓ ↓
Browser page method

document . write ( );
↓ ↓
page method

Browser is default object, master object, super object

write() is a method related to document object

Example

```
<head>
<script type='text/javascript'>
window. document. write (''LiveScript is java script");
document. write ("<br>");
document. write ('Livescript is java script');
</script>
</head>
```

⁕ Javascript semicolon (;)

In java script every statements ends with semicolon (;). It is an optional notation.

Example.

```
<head>
<script type='text/javascript'>
 document. write =(" Livescript is java script")
 </script>
 </head>
```

Example.

```
<head>
<script type='text/javascript'>
 document. write ( " java script");
 document. write (' Livescript ');
 document. write
 (' Livescript is java script')
  </script>
  </head>
```

Note :- 1) In the above script semicolon (;) is mandatory.
2) It is a good programming practice to use the semicolon.

* Java script place in HTML file :-

There is a flexibility given to include Javascrip code any where in a HTML document but the follow ways are most prefered in the line environment.

1] Script in <head> ..... </head> section
2] script in <body> ..... </body> section
3] Script in <body> .... </body> & <head> ...... </head> section.
4] Script in & external file & then include in <head> ..... </head> section..

Example :-

```
<head>
<script type = 'text / javascript'>
document . write ( " welcome to Head Section");
</ Script>
</ head>
 <br/>
 < body>
 < script language = " javascript ">
  document. write ( " welcome to the Body Section")
 < / script>
 </body>
```

* External javascript :-

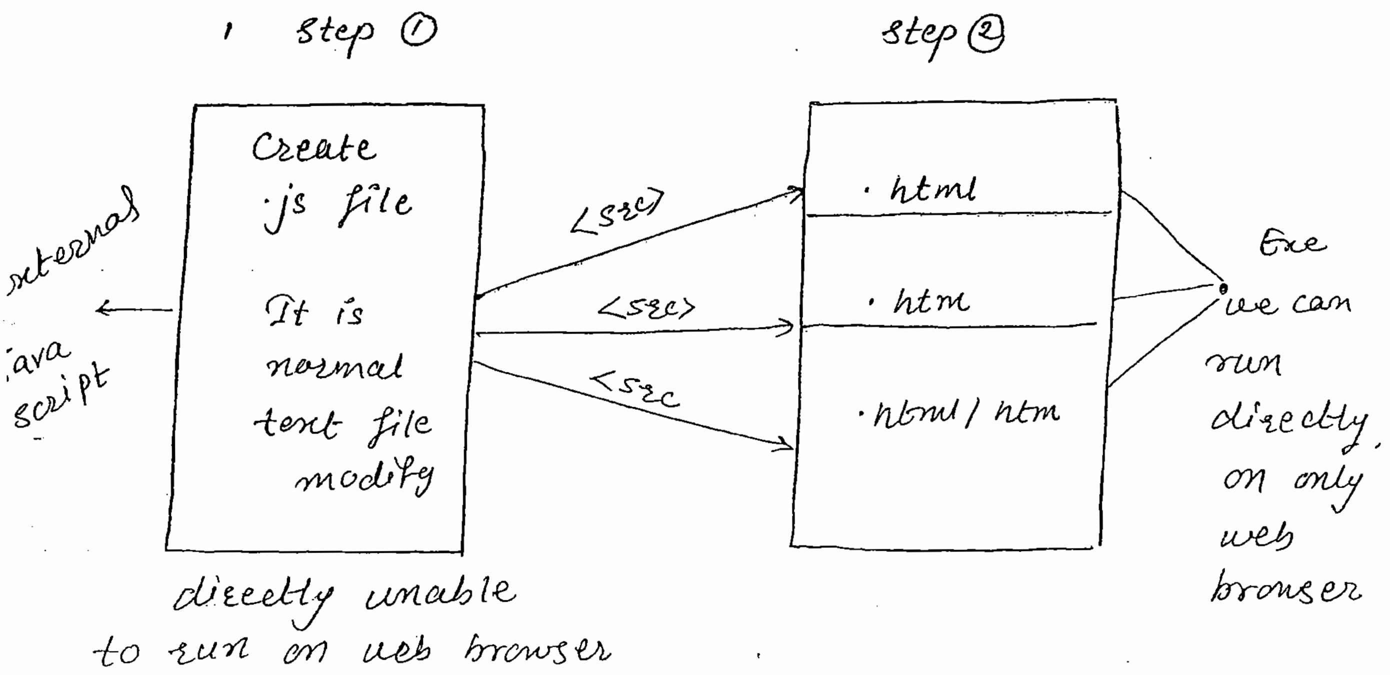
Javascript can also be placed in a external files, these files contains javascript code, this code we can apply on diff. webpages. External javascript files extensions is .JS

Note :-

1] External script cannot contain the <script> </script> tags!

2] To use an external script, point to the .js file in the "src" attribute of the <script> tag.

* how to run external JS :-



* Creating javascript file :-

```
        document . write ( "<h1 style ='color : blue'>
  welcome to external java scripts .... ! </h1>");
   document . write ("<br/>");
  document . write (" Thank U ...");
```

Save with Example .js extension @ any location.

* Creating HTML files :-

```
<html>
<head>
<script type = "text/ javascript " src =" Example
   . js  "> </script>
</head>
<body>
</body>
</html>
```

save with .html or .htm extensions .....

* Java Script code :-

It is a sequence of javascript statements, each statement is executed by the browser in the sequence they are returned.

```
<head>
<script type =" text / javascript ">
document . write ("<P> This is paragraph </P>")
</script>
 </head>
```

Js code

* Javascript blocks :-

Javascript sentences can be group together in blocks. blocks starts with a left curly bracket { & end with a right curly bracket }

The purpose of your block is to make the sequence of statements execute together.

```
<head>
<script type = " text / javascript">
 {
  document. write (" This is a Block");
 }
 </script>
 </head>
```

* Javascript Popup Boxes :-

Javascript has 3 kind of popup boxes.

1] Alert Box
2] Confirm Box
3] Prompt Box.

Alert Box :- An Alert Box is often used if you want to make sure information comes through the user. when an alert box pops up, the user will have to click "OK" to proceed.

Syntax :- alert ("Message");

Example :-

```
  <body>
  <script type = 'text / javascript'>
   alert ( "Invalid Entry");
   < / script >
   </ body>
```

* How to display multiple line on the alert.

we cannot the use <BR> tag here because alert is a method of the windows object, that cannot be interpret HTML tag.

Insted we use the new line escape character.

```
   <head>
   < script type = "text / javascript'>
    alert (" JavaScript \nis \na \n client - side
    \n programming \n language");
    </script>
    </ head>
```

Blank

Example ③ Alert with functions

```
<head>
<script type = 'text/javascript'>
function MyAlert()
{
    alert ("JavaScript\nis\ns\nclient-side\
n programming \nlanguage");
alert ("1\n\t2\n\t\t3');

}

 </script>
 </head>
 <body>
  <P> Click the button to display alert
 Messages .... </P>
 <button onclick = "myAlert()"> ClickMe </butt
 </body>
```

* Prompt Box :- Confirm Box

It is often used, if you want the user to verify and accept something.

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

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

Syntax :-

```
confirm("Message");
```

example ① :-

```
<body>
<script type='text/javascript'>
 confirm ("Click OK or Canele");
 </script>
 </head>
```

Example ② :-

```
<body>
<script type = 'text/javascript'>
 var  X = confirm ("Click OK or Cancel");
 alert (" User Selected Option is : " +X);
 < /script>
 < / head>
```

```
<body>
<script type = 'text /javascript'>
 var X = confirm("Click OK or Cancel");
 alert (" User selected option is: " +x);
 if (x = = true)
 {
    alert (" User Clicked on OK Button");
 }
  else
 {
  alert (" User Clicked on Cancel Button");
}
< /script>
< /head>
```

Example :- Confirm with function

```
< body>
< script type='text/javascript'>
function myConfirm()
{
  var x = confirm("click OK or Cancel");
  alert(" User selected option is : " +x);
  if (x == true)
  {
   alert("User Clicked on OK Button");
  }
 }
 </script>
 </head>
 < body>
  <P> Click the button to display the user
  Selected Result..</P>
  < button onclick = "myConfirm()"> Confirm
  </ button>
  </ body>
```

Prompt Box :-

It is used to, 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 value. If the user clicks "Cancel" the box returns null.

Syntax :-

```
prompt ("sometext", defaultvalue");
```

Example :- ①

```
<head>
<script type ='text/javascript'>
prompt ("Enter Any Number:");
</script>
</hehad>
```

Example :- ②

```
<head>
<script type = 'text / javascript'>
 var MyVal = prompt ("Enter Any Number:");
 alert ("User Entered Value is : " + MyVal);
 < /script>
 < / head>
```

Example :-

```
<head>
< script type = 'text / javascript'>
 var MyVal = prompt ("Enter Any Number : ", "123");
 alert ("User Entered Value is : " + myVal);
 if (myVal > 100)
 {
  alert ("User Entered Value is Big");
 }
  else it (myVal < = 100)
  {
   alert ("User Entered Value is Small or Equal");
  {   </script>
      </hehad>
```

Eg :- prompt with function :-

```
 <head>
 < script type = 'text / javascript'>
   function My Result ( )
  {
   var Myval = prompt (" Enter Any Number : ", "123")
   if my val > 100)
  {
    alert ("User Entered Value is Big");
  }
    else if (my Val <= 100)
    {
```

```
alert ("User Entered value is small or Equal");
}
}
 </script>
  </head>
   <body>
   <P> Click the button to display user Entered
   value ... </P>
  <button onclick = "myResult()"> Prompt
  Box </Button>
   </body>
```

Note :-

If the above script, if click on cancel button it return null, that is unable to handle with directly condition.

* External Javascript with popup boxes :-

Step 1 :- Create a required JS File

```
function MyAlert()
{
  alert ("Welcome to ExternalJS");
 }
 function My Confirm()
 {
   confirm ("Click Ok or Cancel");
  }
   function MyPrompt()
  {
    prompt ("Enter any Value:");
  }
```

Save with .js Extension @ any location....!!

Step 2:- Prepairing required html file.

```
<html>
<head>
<script type = "text/javascript" src = "myscript.js"
 </script>
 </head>
  <body>
  <p> Click the button to display alert Message..</
  < button onclick = "MyAlert()"> Alert </button>
  <P> Click the button to display Confirm
   Message..</P>
  <button onclick = "MyConfirm()"> Confirm</butto
  <P> Click the button to display Prompt Value..</P
  <button onclick = "MyPrompt()"> Prompt </button
   </body>
   </html>
```

* Working with javascript variables :-

In Java script values are working with variables remember the following statements

1. Every variable should starts with an alphabets
2. Variable should not contains unnecessary special charcters.
3. Variables or case sensitive
4. Every variable should have reasonable length
5. keyboords should not be used as variables
6. In Java Script every variable should starts with var keywords.

In software envirenment we can declare the variables the following two ways.

1] Implicit declaration
2] Explicit declaration

1] Implicit declaration :-

In every scripting it is the default declaration

example :- y = 100

2] Explicit declaration :-

All programing languages default declaration

example :- int a = 5

Scripts are able to support implecit & explicit declaration but languages are only explicite declaration.

Note : Explicit detlaration is allways reccomman-ded or a good programming practice

* JavaScript datatypes :-
In Java script Data types are classified into the following two types.

1] Premitive datatypes
2] Non-premitive data types.

1] Premitive data types :-
Java script has a five primitive data types

- 1] String
- 2] Number
- 3] Boolean
- 4] Undefined
- 5] Null

2] Non-premitive datatypes :-
These are populearly known as reference or composite data types.

✻ Premitive data types.

① Javascript strings :-

In java script a string should be within a single or double quotes

```
:-      var name = " nit";
                 = 'nit';
```

② No. data type

Java script has only one type of numbers, they can be return with or without decimals

```
var x1 = 34.00;    with decimals
var x2 = 34.       without decimals.
```

③ Boolean data type

It is used to represent a boolean value, These are as follows

```
true // equivalent to true, yes, or on
false // equivalent to false, no, or off.
```

(4) Undefined :-

It is a value of variable with ~~k~~ no value.

```
var x ;        // Now x is undefined.
```

(5) Null :-

Variables can be emptied by setting the value to Null

Ex,

```
var x = null ;    // Now x is null
```

* Dynamic data types :-

Java script has dynamic types. This means tha the same variable can be used as different types.

Example,

```
var x ;             // Now x is undefined
var x =5;           // Now x is a Number
var x = "Raaj";     // Now x is a String.
```

* Non-primitive data types :-

When a variable is declaired with the Keyword new, the variable is an object.

Example,

```
var name = new String();
var x    = new Number();
var y    = new Boolean();
```

Example :-

```
<head>
< script type = ' text / javascript' >
 alert ("welcome to JS");
</script>
</head>
<body>
<noscript>
<p style = ' color : red' > OOPs Your Browser not
supporting Javascript Update / Change the
script settings and
Try .. </P>
</noscript>
</body>
```

↑

* < noscript> tag :-

It is used to provide an alternate contents for users when script is disabled or not supporting, It is a paired tag.

It is allways declaired within the body section

Syntax :-

< noscript > -------- </ noscript>

* Javascript operators :-

Java script supports the following list of operators.

1] arithmetical operators :-
2] comporison operators
3] logical operators etc

* Arithmetical operators

Using these operators we can perform the arithmatical operations.

The following table describes the operator, descriptions, e.g.

operator	Description	Example
+	Addition	j + 12
-	Subtraction	j - 22
*	Multiplication	j * 7
/	Division	j / 3.14
%	Modulus	j % 6
++	Increment	++j
--	Decrement	--j

* Comporison operator

Using these operator we can compair the following table describes the

operator	Description	Example
==	is equal to	j==42
!=	is not equal to	j!=17
>	is greater than	j>0
<	is less than	j<100
>=	is greter than or equal to	j>=23
<=	is less than or equal to	j<=13

* Logical operators :-
Using these operators we can work with logical expresions.
The following table describes the operator expression, descriptions, example

operator	Description	Example
&&	And	j==1 && k==2
||	OR	j<100 || j>0
!	Not	!(j==k)

* Javascript conditional, control statements,
Java script supports the following list of conditional controls

1] if statement
2] if .... else statement
3] if .... elseif .... else statement
4] switch statement

* 1] if statement

Use this statement we can execute through block of statements.

Syntax :

```
if (condition)
{
  True Statements
  True statements

}
```

Example :-

```
<head>
<script type = 'text/javascript'>
 var  x = prompt ("Enter Any Number!");
 if  (x> 100)
 {
   alert ("Number is Big");
 }
  </script>
  </head>
```

* 2] if .... else statements :-

In this conditional controls statements, If the given condition is true, true block can executed othere wise else block executed. blocks means collection of logical statements

Syntax :

```
        if (condition)
        {
            True Block Statements.
            True Block Statements.
        }
        else
        {
             false Block statements
             false Block statements

        }
```

Example :-

```
      <head>
      < script type = 'text / javascript'>
       var  x = prompt (" Enter Any Number");
       if  (x > 100)
        {
            alert (" Number is big");
         }
            else
         {
            </script>
            </head>
```

* if .... else if .... else statements :-

In this conditional control we can select any one of block among the several.

Syntax :-

```
if (condition 1)
 {
   code to be executed if condition1 is true.
 }
  else if (condition2)
 {
    code to be executed if condition2 is true.
  }
   else
  {
    code to be executed if neither
    condition 1 nor condition is true.
  }
```

Example :-

```
 <head>
 < script type = 'text/ javascript'>
  var  X = prompt (" Enter Any Number ");
   if ( X > 100)
  {
    alert ("Number is Big");
  }
   else if (x < 100)
```

```
{
  alert (" Number is small");
}
   else if (x = = 100)
 {
  alert (" Number is equal ");
 } else
 {
  alert (" Invalid Input ");
 }
  < /script>
  < / head>
```

Example :-

```
  < head>
   < script type = ' text / javascript'>
   function My Course ()
   { -
     var course = prompt (" Enter Any Course Name
    (HTML5, CSS3 , Bootstrap ,jQuery ):"," HTML5");
    if (course = = " HTML5")
    {
      alert (" Your are Selected : " + course);
     }
      else if ( course = = " CSS3 ")
     {
      alert (" Your are selected : " +course);
     }
    else if (course == " Bootstrap")
  {
    alert ("Your are selected:"+ course);
  }
    else if ( course == "jQuery")
  {
    alert ("Your are Selected:" + course);
  }
   else
  {
    alert ("Course not Existed");
  }
  }
  </script>
  </head>
  <body>
   <P style ='color : blue'> Click the button to
display the User Entered Course Name : </P>
   < button onclick ="my Course ()"> Select_Course
 _</button>
  </body>
```

* Switch - Conditional Control :-

Use the switch statement to select one of many blocks of code to be executed.

It is a basically an enhanced version of if else statement.

Syntax :-

```
switch(x)
 {
  case 1:
    executed code block 1
    break;
   Case 2:
     executed code block 2
     break;
    default:
     code to be executed if x is different from
     case 1 and 2.
   }
```

Example :-

```
  <head>
  < Script type = 'text / javascript'>
     function My course()
  {
   var course = prompt (" Enter any course name
    (HTML5, CSS 3, Bootstrap, JQuery) : ", "JS").
    switch ( course)
     {
      case (course)
     {
      case 'HTML' :
      alert (" You are Selected:" + course );
      break;
       case 'JS':
      alert ("You are Selected: "+ course);
       break;
       case 'CSS':
```

```
alert ("You are Selected : "+ course);
    break;
    case 'jQuery':
    alert (" You are Selected : "+ course);
    break;
    default:
     alert ("You are selected Wrong Course");
     }
     }
     </script>
     </head>
     <body>
      <P> Click the button to display the course
    Name : </P>
    <button onclick = "My course()"> Click Me </button>
    </body>
```

* Javascript key words :-

These are popularly known as reserved words, they can not be used as variables, functions, methods, lebles and object name.

In javascript several keyword existed the following are frequently used.

abstract, boolean, break, byte, case, catch, Char, class, const, continue, debugger, default delete, do , else , enum, export

* Javascript looping controls :-

The java script supports the following looping controls.

1. for
2. while
3. do - while .....etc

1] for

It execute a block of statements repeatedly until the given condition false.

Syntax :-

```
for (initialization ; test condition ; iteration
   statement)
  {
    Statement(s) to be executed if test condition
   is true
    Statement(s) to be executed if test condition
    is true
  }
```

Example :-

```
     <head>
     <script type='text/javascript'>
     for (i=1 ; i<=10 ; i++)
      {
       document.write("The Value is : "+i);
       document.write("<br/>");
      }
     </script>
     </head>
```

Example :- 2nd method

```
<head>
< script type = 'tent / javascript'>
for ( i=1 ; i< =10 ; i++)
 {
   document . write ( " The Value is : " +i ); .
  document
 } document . write (" The value is : "+i);
   </script>
   </head>
```

ex,

```
<head>
<script type = ' tent/ javascript'>
for (i=1 ; i< =6 ;  i++)
{
 document. write ("<h" +i+ "> This is
    heading " +i);
 document write ("</h" +i +">");
 }
 </script>
 </head>
```

* break and continue statements.

1] Break statement

The statement will break the loop at a specific condition

1) Example :-

```
<head>
<script type='text/javascript'>
 for (i=1; i<=10; i++)
  {
   if (i==5)
   {
    break;
   }
   document.write("This value is" + i);
   document.write("<br/>");
   }
   </script>
   </head>
```

```
o/p        1
           2
           3
           4
```

* Continue :-

The statement will break the current loop & continue with next value.

Example

```
   <head>
    <script type=' text/javascript'>
    for (i=1; i<=10; i++)
     {
      if (i==5)
      {
      continue;
     }
```

```
document. write (" This Value is " + i);
document . write (" <br/>");
}
  </script>
  </head>
```

* While loop :-

It execute the block of statement repeatedly n number of times.

Syntax :-

```
    While (variable <= endvalue)
      {
        code to be executed
        code to be executed
      }
```

Example :-

```
  <html>
  <body>
  <script type ="text/javascript">
   var i=1;
    while (i<=10)
  {
   document write (" The number is " +i);
   document. write ("<br/>");
     i++;
   }
    </script>
    </body>
```

* do-while :-

It execute a block of statement repeatedly n+1 time

Syntax :-

```
do
{
  code to be executed
  code to be executed

}
 while ( variable <= endvalue);
```

Example :-

```
< script type = "text/javascript">
var i = 1 ;
do
{
document.write ( "The number is" + i );
document.write ("< br/ >");
i++;
}
 while (i <= 10);
 < /script>
 < / body>
```

Working with Javascript function :-

Function is a block of code that will be executed only by an occurance of an event, that time function is called. Function is group of reusable code which can be called anywhere in your program. It eliminates the need of writting same again and again.

Syntax :-

```
function functionname ( var1 , var2 , ... varx)
{
  specify some code
  specify some code
  specify some code

}
```

Example 1 :-

```
<head>
<script type = 'text / javascript'>
 function My msg ()
 {
  alert ("Welcome to functions");

 }
   </script>
   </head>
   <body>
   <P> Clickt the button to display the Alert
    Msg ... </P>
  < button Onclick = " MyMsg ()"> ClickMe < /button
  < / body>
```

The return statement :-

It is used to specify the value that is returned from the functions, It is an optional statement

Example 1 :-

```
<head>
<script language =" javascript">
 function MyReturn()
{
 return ("Welcome to Return Statement");
}
 </script>
 </head>
<body>
 < script type ='text / javascript'>
  document . write (MyReturn());
 < / script>
  < / body>
```

Example 2 :-

```
<head>
<script language =" javascript">
 function MyReturn (x,y)
 {
   return x+y;
 }
```

```
</script>
</head>
<body>
<script type='text/javascript'>
document.write ("sum of Two Number is:
" + MyRebuen (2,3));
</script>
</body>
```

Lifetime of javascript Variables.

A variable declared within a javascript function becomes a local, only accessed by within that function, (the variable has local scope)

You can have local variables with the same name in different functions, because local variables are only recognized by the functi in which they are declared.

Local variables are deleted as soon as the function is completed.

Global Javascript Variables :-

Variables declared outside a function become global, all scripts and functions on the web page can access it.

Global variables are deleted when you close the page.

Example :-

```
<head>
<script>
var x=5; ←  global scope
 function MyVal()
 {
   Var a=5; ←  Local scope
   ----.
   ----.
 }
  function MyVal1()
  {
    var b=6;
    ----.
    ----
  }
```

bgcolor property :- Using this property we can change backgroung color of page, It is a property inside document object.

Syntax :-

```
document . bg color = " colorName / colorCode"
```

```
<head>
<script type =' text/javascript'>
 function MyBlue ()
```

```
{
 document . bgColor = "blue";
}
 function MyGreen()
 {
  document. bgColor = "# 00FF00";
 }
 </script>
 </head>
 <body>
  <P> Click the button to change the background.
  color of the page..!! </P>
  < button onclick = "MyBlue()"> Blue </ button>
 <P> Click the button to change the
 background color of the page..!! </P>
 < button onclick = "MyGreen()" > Green </button>
 </body>
```

* Working with javascript events.

Events or actions that can be effected by javascripts. Events are normaly used in combination with functions.

The following list of events frequently used

Event	Description.
click	Occurs when the user clicks on a link or form element

error	Occurs when an error happens during loading of doc.
focus	Occur when input focus is given to a form element.
load	Occurs when a page is loaded into Nevigator
mouseout	Occurs when the user moves the pointer off
nouseover	Occurs when the user moves the pointer over
reset	When the user clears a form using the reset button
select	Occurs when the user selects a form elements field
submit	Occurs when a form is submitted
unload	Occurs when the user leves a page

1] On click events

This events fires when the user clicked on a element

Syntax :- In HTML :

< element onclick = " Some JavaScript Code">

In Java script :

object . onclick = " Some

Non Supported tags :

The Tags are not supported onclick event : < base> , <bdo> ,
 , <head>, <html> <iframe>, <meta>, <param>, <script>, <style> & <title>

Supported JS objects:
Document, window

Example

```
    <body>
    < button onclick = "alert ('Welcome to JS
  Events')" > Events
      </button>
      <script type ='text ='text/javascript>
      function mymsg()
    {
      alert ("Welcome to Events");
    }
     </script>.
     <P> Click the button to display the alert
     msg...! </P>
     <button onclick = "Mymsg ()"> Click Me</bubbon
```

2] On double click

This events fires when the user double click on the element

In HTML

```
< element ondblclick = " Some JavaScript Code">
```

In Javascript

```
object.ondblclick =" some JavaScript code"
```

Example

```
< head>
< script type = ' text /javascript '>
 function MyDate()
  {
   document . getElementById ("dt"). innerHTML
  = Date ();
  }
    </ script>
    < /head>
    < body >
    <p id = "dt"> Double Click on the Button
    and Observe the output ..</p>
    < button ondblclick = "MyDate()"> Click Me
       < / button>
    < input type = 'button' value = " Clickme"
       ondblclick = " MyDate ()"/>
     </body>
```

3] Onload

- This event fires when the object has been loaded. It is often used within the body elements on load.

Syntax

In HTML :

```
< element onload = " SomeJavaScriptCode">
```

In JavaScript :

```
Object onload = "Some JavaScript Code"
```

Example

```
<head>
<script type ='text/javascript'>
function ImgLoad()
{
  alert ("ImageLoadedSuccessfully");
}
 </script>
 </head>
 <body>
 <P> Refresh the page and Observe...</P>
 <img src = "html5.png" width =150px height
 =150 px onload =" ImgLoad()" alt =" sorryNoImg
 </body>
```

~~Exa~~ 4] On error

This event is triggered if an error occurs while loading an external file (e.g a document or an image

Syntax In HTML:

```
<element onerror = "someJavaScript Code">
```

In JavaScript :

```
Object . onerror = " Some JavaScript Code"
```

Example

```
    <head>
    <script type = 'text / javascript'>
      function ImgErr()
      {
         alert (" Image FailtoLoad");

       }

       </script>
       </head>
       <body>
        <P> Refresh the page and observe ... </P>
        <img src = "tml5 . png" width = 150px
         height = 150px onerror = "ImgErr()" alt
        = "SorryNoImg"/>
       </body>
```

Onmouseover :-

These events fires when you overmouse curser to an element

Syntax:

In HTML:

```
<element onmouseover = " some JavaScriptCode">
```

In JavaScript:

```
Object . onmouseover = " someJavaScriptCode"
```

Onmouseout :-

This event fires when mouse pointer of frame an element

Syntax :

In HTML

```
< element onmouseout = "Some JavaScript Code">
```

In Javascript : `object.onmouseout = "Some JavaScript code"`

Example (i) :-

```
<head>
<script type = 'text/javascript'>
 function mover()
 {
  alert("Mouseover");
 }
  function Mout()
  {
   alert ("Mouseout");
 }
  < /script >
  < /head>
  < body>
  < div style = 'color : blue' onmouseover = "mover()"
  onmouseout = "Mout()"> Bring The Mouse Pointer...
  </div>
  </body>
```

Example : mouseover

```
  <body>
   <h1 onmouseover = " style.color = 'red''
  onmouseover = " style.color = 'black">
  Mouse over this text </h1>
  <body>
```

Form Events :-

1]

1] Onblur :- This event occurs when an object loss the focus, It is most often used with form validation code (when the user leaves a form field)

Note :- The onblur event is the opposite of the onfocus event

Syntax :- In HTML :

```
        < element onblur = "Some JavaScript Code">
```

In JavaScript :- objeet. onblur = "Some JavaScript Code"

Non Supported HTML events :- onblur event unable to Support the following HTML elements
< bose > , < bdo > , < br > , < head > , < html >
< iframe >, < meta > , < param > , < script > , < style >, &
< title >.

Supported JS objects :- Document , window

Example ① :-

```
        <head>
        < script type = 'text / javascript'>
         function upperCase()
         {
            Var x = document . getElementById (" fname");
            x value = x. Value . toUpperCase ();
        }
          < / script>
          </head>
          < body>
          Enter your name <br/>
```

```
< input type = "text" id = "fname" onblur = "upperCase()".
 </body>
```

Onfocus Event :- This event fires when an element ~~got~~ gets a focus it is most often used in,
→ It is most often used in input ~~get~~ select & anchors <input> <select> &<
It is opposite of onblur event

Syntax :- In HTML :

```
        < element onfocus = "Some JavaScript Code">
```

In Javascript :

```
      object . onfocus = "Some Javascript Code"
```

Nonsupported HTML elements :- The following elements are non supporting onfocus events.
<base>, <bdo>, <br>, <head>, <html>, <iframe>, <meta>, <param>, <script>, <style>, & <title>

Supported JS Objects :- Document, windows

```
<head>
< script type = 'text / javascript'>
function setStyle (x)
 {
   document . getElementById
  (x) . style . background = "yellow";
 }
 </script>
</ head>
 < body>
 First name : <br>
< input type = "text" id = "fname" onfocus =
    "setStyle ( this . id)"/>
```

```
<br/>
Last name : <br/>
<input type = "text"id = "lname" onfocus ="set-style
  (this.id)"/>
  </body>
```

Onselect :- This event fires after some text has been selected. This event frequently used for within:

```
        <input type= "file">,
        <input type= "password">,
        <input type = "text"> , and <textarea>
```

Syntax :-

```
      <element onselect =" script">
```

```
    <head>
    <script type ='text/javascript'>
    function MySelect ()
     {
      alert ("sorry Text Should Not Select");
     alert ("Content Write Protected");
     }
      </script>
      </head>
      <body>
      <form>
      <input type ='text' value =" Javascript"
     onselect  =  " MySelect()"/>
      <br/>
      <textarea rows = "5" cols ="23" onselect =
 "MySelect()"> HTML5 is New Hyper Text Markup for
   mobile Apps... </textarea>
          </body>
```

Onresize Event : when the size of an element has been changed then this event fires

Syntax :-

In HTML :

```
<element onresize = "some JavascriptCode">
```

In Javascript:

```
Object. onresize = "some JavaScript Code"
```

Example :-

```
<head>
<script type ='text/javascript'>
function myPage()
{
  alert ("sorry Page Resized");
}
</script>
</head>
<body onsesize = "myPage()">
<p> Resize the Page and Observe ...</P>
<P> It displays Warning Msg ..</P>
</body>
```

Types of errors in Javascript :- Javascript supports the following list of errors, these are divided into. a) Syntax error
b) Runtime error
c) Logical error.

Java Script supports following three types of errors :-

Syntax Error :- Is called as parsing errors, occurs at compile time for traditional programming languages at interpret time for JavaScript.

following example causes a syntax error because it is missing a closing parenthesis.

Example :

```
<body>
    <script type = "text/javascript">
    document.write (;  ↖
    < script                  syntax error..!
```

Runtime Errors :- These are called exceptions, and these errors occurred at execution time.

The following example causes a run time error because here syntax is correct but at run time it is trying to call a non existed method :

```
<body>
 <script type = "text/javascript">
 document.writeLi( "Hello Welcome");
  </script>
   </body>
```

Logical errors :- These can be most difficult error to find. This errors occured, if you make a mistake in the business logic. These errors unable to handle.

try·

< head>

* Exception handling in Java script :-

(1)

Try... catch statement This statement allows the you to test a block of code for errors. The try block contains the code to be run, & the catch block contains the code to be executed if an error occurs.

Example(1)
```
< head>
    <script type = 'text/ javascript'>
    alert ("Welcome to Exceptions");
    alert ("Thank U");
    < / Script)
    < / head>
```

The above script get executed successfuly.

Example (2)
```
< head>
    < script type = 'text/ javascript'>
    alert ("Welcome to Exceptions");
    alst ("Thank U");
    < /script>
    < / head>
```

O/P :- Welcome to exceptions.

Example (3)
```
<head>
    < script type = 'text/ javascript '>
    alet ("Welcome to Exceptions");
    alert ("Thank U");
    < /script>
    < / head>
```

NO. O/P

In the above example we need to apply the try catch block.

Syntax :-

```
<script>
try
{
code to run [break;]
}
catch(e)
{
Code to run if an exception occurs [ break;]
}
</script>
```

Example :-

```
<head>
< script type =' text / javascript'>
try
{
alert ("Welcome to exceptions");
}
catch(e)
{
alert( e.description);
}
alert ("Thank you");
</ script>
</ head>
```

Above script get executed successfully

eval() :- It is a global function stands for evaluate. It evaluates a numerical values

Syntax :- eval (expression)

e.g ①

```
< head>
< script type = ' text / javascript '>
var X = prompt ( " Enter value to evaluate");
alert ( eval (x));
alert ("Next");
< / script>
</head>
```

In the above script if you enter the numerical value script get executed successfully. otherwise script unable t run. That time we should implement try catch block.

e.g.

```
<head>
<script type = 'text / javascript '>
try
  {
   Var x = prompt ("Enter value to evaluate")
   alert ( eval (x))
  }
  catch (e)
  {
  alert ("Sorry Alpha - Invalid : " + e.description)
  }
  alert ("Next")
  < / script >
  < / head>
```

Finally block :- This block get already executed regardless of an exception occuring.

Syntax :-
```
<script>
 try
 {
  code to run [ break;]
 }
  catch (e)
  {
  code to run if an exception occurs
  [ break;]
  }
   finaly
  {
   Code that is always executed regardless
of // an exception occurring

 }
 </ script>
```

e.g :-
```
<head>
<script type = 'text / javascript'>
 try
 {
  var x = prompt (" Enter value to evaluate")
   alert (eval (x))
  }
   catch (e)
  {
   alert (" Sorry Alpha - Invalid : "+e.description)
  }
  finally
```

```
{
alert ("This Block Always get executed");
}
 alert ("Next")
 </script>
 </head>
```

Throw statement :- This statement allow to you create an exception. if you use this statement together with try catch statement, you can control program flow and generate accurate error message The exception can be string, integer, boolean or an object.

Throw Exception

```
<body>
<
var x = prompt (" Enter Any number").
try {
      if  (x>10)
     {
       throw "Error1' ;}
     else if (x<=10)
     {
       throw " Error 2";
     }
     else if ( isNaN (x))
     {  throw " Error3";
     }
   }
   catch (error)
```

```
{  if ( err = = " Err1")
    {
      document. write (" Error: The value is too high");
    }
      if (err = = "Err3")
       {
         document . write (" Error : The value is not
          a number");
       }
    }  </script>
       </body>
```

Java script Global function :-

isfinite() function :-

The isfinite is used to determine whether a specified number is finite or not. isfinite is a top-level function and is not associated with any object

Syntax : isfinite (number)

Example :

```
<body>
      <script type ='text/ javascript'>
      document . write (isfinite (" Good morning") +
        "<br/>");
      document. write ( is finite (423) + "<br/>");
       </script>                          ↓ true
       </body>
```

is Nan() function :- The isNan function is used to determine whether a value is "NaN" (not a number) or not.

isNan (text value)

```
<body>
<script type = 'text/javascript'>            true
document . write ( is Nan (" Good morning ")) "</br>
                                       true
document . write ( isNan ("2009/10/15 ") "</br>");
document . write (isNan (455) "</br>");
</script>                       False
</body>
```

parseInt() function

It parse a string and returns an integer

Syntax   parse Int (string

Parameter	Description
String	Required The string to be parsed.

```
<head>
<script type = 'text/javascript'>
 var x = 100;
 var y = 200;
 var z = x+y
document . write (z);
</script>
</head>
```

o/p → 300

```
<head>
< script type ='text / Javascript ")
 var    x=100 ;
var  y = "200";
var z  = x+y;
document. write (z);
 </ script>
 < /head>
```

O/P → 100200

```
< head>
< script type = 'text/ Javascript'>
 var x = 100 ;
 var y = " raju";
 var z = x + y parseInt (y);
document. write (z);
< /script>
< / head>
```

→ Nan

```
< head>
< script type =' text / javascript'>
var x = 100;
var y = "200";
var z = x+ parseInt (y);
document. write (z);
< / script>
< / head>
```

→ 300

```
<head>
<script type = 'text/ javascript>
 var x = 100;
 var y = "raju"
 var z = x+y;
 document.write (z);
 </ script>
 </ head>
```

→ 100raju.

```
<head>
<script type =' text / javascript'>
 var x = prompt (" Enter Any No:");
 var y = prompt (" Enter Any No: ");
 var z = parseInt (x) + parseInt (y);
 document.write ("sum of two no is :" + z);
 </script>
 </head>
```

parsefloat () function :-
It parses a string and return a floating value.

- Syntax :-

```
parsefloat ( String)
```

Example :-

```
<head>
<script type = 'text/Javascript'>
var x = 100.54;
vor y = 100.20;
var z = parsefloat (x) + parsefloat (y);
document. write (z);
</script>
</head>
```

Working with Javascript objects :-

Javascript is object based programming, it allows you to define your own objects and make your own variable types. An objects has properties & methods.

Define property :- Properties are values associated with an object

e.g. length, width, height, Name etc.

Methods :- These are actions that can be performed on objects.

e.g. open, close, Resize etc.

```
<body>
<script type = 'text/javascript'>
vor str = "Nareshitechnologies";
document. write ("The length of the string is : "+str.
 length);
document. write ("<br/>");
document.write ("The string in Uppercase: "+str to
 UpperCase())
document.write <"<br/>");
document.write ("The string in Lower Case: "+str
   tolowerCase());
 </script>
 </body>.
```

Commonjavascript objects :-

In Javascript the following list of objects listed

1] Array objects
2] Boolean objects
3] Date objects
4] Math objects
5] String objects
6] Number objects
7] RegExp Objects.

Browser objects :-

Javascript supports following list of browser objects

1] Window objects
2] Navigator objects
3] Screen objects
4] History objects
5] Location objects

Window objects :-

It is highest level javascript object, which corresponds to the web browser window. It has the following list of methods :-

1] open () method :-

The open () method opens a new browser window.

Syntax :- window.open (URL, name, specs, replace)

Example

```
<body>
< button onclick = "window.open ('http://www.nareshit.
 com')">
 Naresh IT </button>
< button onclick = " window.open ('http://www.nareshit.
 in')"> NareshIN </button>
 </body>
```

Example

```
<body>
<head>
< script type = 'text/javascript'>
  function Myopen ()
{
 window.open ("http: //www.nareshit.com");
 window.open ("http://www.nareshit.in");
}
 </script>
 </head>
 <body>
```

```
<P> click the button to open Nareshit .com on a New
tab or window ....</P>
< button onclick = " Myopen()"> ClickMe </button>
</body>
```

Window print() method :-

It prints the contents of the current window

Syntax :- window. print()

Example :-

```
<head>
<script type = 'text / javascript'>
function mypage()
{
  window .print()
}
</script>
</head>
<body>
<P> Click the button to print the current page...!!<//
< button onclick = "myPage ()"> PrintPage </button>
</body>
```

Window stop() Method :-

This method stops windows loading.

Syntax :- window. stop()

Example :-

```
<head>
 <script>
 window.stop();
 </script>
 </head>
```

escape

Navigator objects :-

This objects contains the information about the web browser

It supports the following list of properties :-

Property	Description.
appCodeName	Returns the code name of the browser
appName	Returns the Name of the browser.
appVersion	Returns the version information of the browser
CookieEnabled	Determines whether cookies are enabled in the browser

Example :-

```
<body>
<script type='text/javascript'>
document write (" The Name of the Browser is :
" + navigator . appName);
 document. write ("<br/>");
document.write (" The version of the Browser is:"
 + navigator. appversion);
 </script>
 </body>
```

2] Navigator objects methods.

It supports following list of methods

Java Enabled ();

Specifies whether or not the browser has Java enabled.

Example :-

```
        <body>
        < Script type = 'text/javascript'>
        document. write (" The status of the javascript is :
      "  + navigator. javaEnabled ());
        < /script>
         </body>
```

3] The Screen Object

This contains information about visitor's screen It has following list of properties

Property	Description
avail Height	Returns the height of the screen (excluding the windows task bar)
avail width	Returns the width of the screen (excluding the windows Task bar)

height   Returns the total height of the screen

width   Returns the total width of the screen.

ex.

```
<body>
<script type ='text/javascript'>
document.write (" The width of the screen is:"
 + screen.width);
document.write ("<br/>");
document.write (" The height of the screen is:
 " + screen.height);
</script>
</body>
```

4] History object :- This object contains the URL of the visited by the user, it is a part of window object It has the following list of the properties & methods.

1] length () property :-
It returns number of url in the history list

Syntax :- history.length.

Note :- Internet Explorer & opera start at 0, while firefox, chrome, & safari start at 1.

example :-

```
<body>
<script type = " text / javascript">
document . write (" Number of URLs in history list : "
+ history. length);
</script>
</body>
```

History objects Methods :-

History objects supports the following list of methods.

Method	Description.
1] back()	Loads the previous URL in the history list
2] Forward()	Loads the next url in the history lis
3] go()	Loads a specific URL from the history list

```
<head>
<script type = ' text / javascript'>
function HisBack()
 {
  window . history . back ();
 }
 function Hisfor()
 {
  window . history . forword ();
 }
```

```
< /script>
< /head>
<body>
<P> Click the button to transfer to backword history...!! <IP>
< Button onclick = "HisBack ()"> His_Back </button>
<P> click the button to transfer to forword history..!!<IP>
< button onclick= " Hisfor()"> His-for </button>
< / body>
< /script>
< / head>
```

Location Objects :-

It contains information about the current URL. It is the port of window object. It has the following list of properties

Property	Description
1] hash	Returns the anchor portion of a URL
2] host	Returns the hostname & port of a URL
3] hostname	Returns the host name of a URL.
4] href	Returns the entire URL
5] Pathname	Returns the path name of a URL.

Example

```
<body>
< script type = 'text/ javascript'>
document . write ( location · href);
< / script>
</ body>
```

Recoquize space

output :- URL = http: // D: shweta 20% / sneha

Location Object methods :-

This method supports the following list of methods

Method	Description.
assign()	Loads a new Document
reload()	Reloads the Current document
Replace()	Replaces the current document with a new one.

Example :-

```
<head>
<script type = 'text / javascript'>
 function MyReplace()
{
 window. Location. replace (" http:// www. nareshit. in");
}
 </ script>
  </head>
  <body>
  <P> click the button to replace the current URL...!! </P.
  < button onclick = " MyReplace ()"> ClickMe < / button>
   </body>
```

output : nareshit. in.

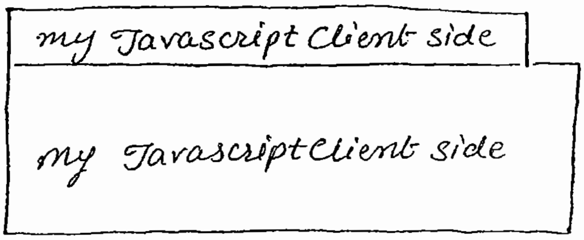
* Document Object :- Each html document loaded into a browser window. It has following list of property.

1] Document title property :-
The title property returns the title of the current document (the text inside the HTML title element)

Syntax :- document.title

Example :-
```
<head>
<title>
 My Javascript client side
 </title>
 </head>
 <body>
 <script type = 'text/javascript'>
  document.write (document.title);
 </script>
 </body>
```

output :-



my Javascript Client side

my Javascript Client side

2] document URL property :-
It returns the URL of the document

Syntax - document.URL;

Example :-

```
<body>
<script type='text/javascript'>
 document.write (document.URL);
</script>                    ↑
</body>               must be upper case
```

O/P :- Same as above example

* Javascript objects :-

1] Array objects : It is used to store multiple values in a single variable.

In Javascript while you are working with array variables you should remember the following list of points.

1] The array is a special type of variable.

2] Values are stored into an array by using the array name & by stating the location in the array you wish to store the value in brackets.

Example

```
myArray [4] = " JavaScript ";
```

3] values in an array are accessed by the array name & location of the value.

E.g : myArray [2];

4] Java script has built in functions for arrays

In Javascript array object supports the following three types of syntaxes :-

① Regular

```
var myNames = new Array();
myNames [0] = "Ravi";
myNames [1] = "smith";
myNames [2] = "Raju";
```

② Condensed

```
var myNames = new Array ("Ravi", "smith",
                          "Raaj");
```

③ Literal

```
var myNames = [ "Ravi", "smith", "Raju"];
```

Array object properties.

Array object supports the following property.

length :- This property is used to display no of elements in an array.

Syntax :- ArrayName.length;

Example :-

```
<body>
     <script type='text/javascript'>
   var myArray = ["html", "css", "JS", "HTML5",
   "css3"];
   document.write ("Number of Array Element are:"
    + myArray.length);
    </script>
    </body>
```

```
<head>
<script type='text/javascript'>
function Mylen()
 {
 var MyArray = ["html", "css", "JS", "HTML5", "css3"];
var x = document.getElementById ("Arr");
 x.innerHTML = myArray.length;
}
</script>
 </head>
 <body>
 <P id = "Arr"> Click the button to display the
number of Array elements ....</P>
```

```
< button onclick.
< body>
```

Array object method.

Array object supports following list of methods

Reverse () : Using this method we can display array elements in reverse order. (last to first)

Syntax : ArrayName . reverse ();

Example 1:
```
<head>
        < script type = 'text / javascript'>
    var MyArray = [ " html " , "css" , "JS", 'HTML5",
                  "CSS3" ]
      document. write (MyArray . reverse());
      < / script>
      < / head>
```

Example 2 :
```
<head>
        < script type = ' text / javascript '>
    var MyArray = [ " html", " css", "JS", "HTML5",
          " css3 " ]
       function MyReverse()
       {
         var myArray [ "html",
         document . getElement ById ("Rev");
         x . innerHTML = MyArray. reverse ();
       }
       <
       < / script>
        < body>
< P  id = "Rev" >
```

Method 2 :- Global variables.

```
<head>
< script type = 'text / Javascript'>
var myArray = [" HTML", "CSS", "JS", "HTML5",
          "CSS3"];
  function myRev()
   {
   document. getElementById
     (" rev") . inner HTML = myArray . reverse();
   }
    </script>
    </head>
    <body>
   <P id = "rev" style ='color : blue; background-
color ; yellow' Click the button to display Array
  Elements in reverse order ...</P>
 <button onclick = " My Rev()"> my -Rev </button>
  </body>
```

pop() :- This method remove the last element of an array that means it remove array elements from right to left directions.

Syntax :- Array. pop();

shift() :- This method remove the array elements from left to right that means 1st to last.

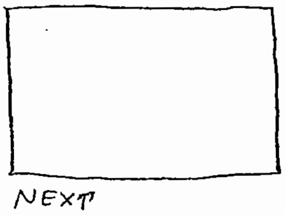
Syntax : array . shift ()

Example 1 :

```
< head>
< script type = 'text /javascript'>
var MyArray = [" html" , "css", " JS ", "HTML5",
                        " css3"];
   function MyRem()
   {
     var x = document.getElementById ("rem")
     x. inner HTML = myArray.pop ();

   }
      < / script>
      < / head>
      < body >
    <P id = "rem" > Click the button to remove
Array elements from right to Left ... !! </P>
<  button onclick = "MyRem ()" > Array-Rem</button>
  < / body >
```

Example 2 :



```
    < html>
     < head>
     < script type = ' text/ javascript '>
      arr = ['fish.jpg', 'fish1.gif', 'nature.jpg',
           'nature1.jpg', 'nature 2.pg']
       i = 0;
      function fun1()
       {
```

```
i++
if (i==5)
{
  alert ("No more images")
}
  else
  {
    document.get ElementById ("img1'). src ="img/"
   + arr[i];
  }
  }
   </script>
   <body>
  <image src = " img/fish.jpg" width = "300" height
   = "250"  id = "img 1">
 <br>
 <input type = "button" value = "Next" onclick = "fun1
 </body>
```

Java script boolean object : It is used to return a boolean value, that value is either zero or one. Here zero represents false 1 represents true.

Syntax : Creating a boolean object
  var my Boolean = new Boolean

Boolean object properties

Note : Except 1 remaining all values returns false.

Example 1

```
<head>
<script type ='text/javascript'>
 vor b1  =  new Boolean (0)
     b2  =  new Boolean (1)
     b3  =  new Boolean (" ")
                        ( null)
                        (NaN);
document.write ("0 is boolean "+b1 +"<br/>");
document.write (" 1 is boolean "+b2 + " <br/>");
              (" An empty string is boolean "+b3+ "<br/>");
              (" null is boolean"  +b4 + "<br/>");
              (" NaN is boolean" +b5 +  "        );
</script>
</head.
```

Example 2 :

```
<head>
<scrip type =' text/javascript'>
vor b1 =new Boolean(0);
document . write (b1);
 </script>
 </head>
```

Java script - The Date Object

It is a data type, date objects are created with new date, once the date object is created we can able to access several properties & methods. we can create the date & objects in following ways.

1. Nor x = new Date()
2. var x = new Date (milliseconds);
3. var x = new Date (datestring);
4. var x = new Date (year, month, day, hours, minute
   seconds, milliseconds);

E.g (1)

```
<head>
<script type = 'text/javascript'>
vor dt = Date()
document.write ("Current System Date and Time is:'
   +dt );
</script>
</head>
```

Javascript Date object methods :- Once date object is created the following list of methods able to work on date pattern.

1. getTime () - Number of milliseconds.
2. getSeconds () - Number of seconds (0-59)
3. getMinutes () - Number of minutes (0-59)
4. getHours() - Number of hourse (0-23)
5. getDay () - Day of the week (0-6). 0 = Sunday, 6 = Saturda
6. getDate () - Day of the month (0-30)
7. getMonth() - Number of month (0-11)
8. getFullYear() - The four digit year (1970-9999)

Example :-

```
< head>
< script type ='text/javascript'>
 var dt = new Date();
 document . write ( dt. getFullYear ());
 </ script>
 </ head>
```

Example ② :-

```
 <head>
 <script type ='text/ javascript'>
  function MyYear()
 {
  var dt = new Date();
  var x = document .getElementById ("yr");
   x . inner HTML = dt.getFullYear();
  }
    </script>
    </head>
    < body>
    <p id = "yr"> Click the button to display the full
    year .... </P>
     < button onclick = "myYear()"> Full Year </button>
     < / body>
```

Write a script to display the current date month in mm / dd / year format-
(yy)

e.g :- < body>

```
      <h1 style ='color : blue'> Current Date is :
      < script>
```

```
Var current Time = newDate()
var month = current Time.getMonth()+1
var day = current Time.get Date()
var year = Current Time.get FullYear()
document.write(month+"/"+day+"/"+year)
</script>
</h1>
</body>
```

Write a script to display the timestamp.

* Date object set method.

Date object supports all the set method.

1] setDate() Method :-

This method set the day of the month to the date object.

Syntax: Date. setDate (day)

Example :-

```
<head>
<script type='text/javascript'>
 function myFunction()
 {
  var d = new Date();
  d.setDate (17);
 var x = document.getElementById("demo");
  x.innerHTML = d;
 }
  </script>
  </head>
  <body>
  <p id="demo"> Click the button to display the
  date after changing the day of the month</P>
 < button onclick="myFunction()"> Display_setDate
 </button>
 </body>
```

Date Object timing Events :- In javascript it is possible to execute a specific que code at the specified time intervals.

The following methods are frequently used.

1. setInterval() - It executes the functions, over and over again, at specified time intervals.

Syntax - window.setInterval("javascript function", milliseconds);

Example -

```
<head>
<script type='text/javascript'>
 function MyInter()
 {
   setInterval(function() { alert("Welcome to
     Events")} , 3000);

 }

 </script>
 </head>
 <body>
 <p style='color: blue'> Click the button to
display the Alert Msg With Time Interval ...</P>
< button onclick="MyInter()"> Interval </button
</body>
```

2] SetTimeout() - Executes a function, once, after waiting a specified number of milliseconds.

window.setTimeout("javascript function", milliseconds);

Example :-

```
<head>
<script type ='text/javascript'>
 function TDelay()
  {
    window.Location = " http:// www.seshajobs.com";
  }

  </script>
  </head>
  < body onload =" setTimeout ('TDelay()', 5000)'>
  <P> Refresh the page to load the related
   URL... </P>
   <P> once You refresh it takes a few seconds..</P>
   </body>
```

Write a script to display digital clock on webpage.

```
<head>
<script>
setInterval ("fun1()", 1000);
 function fun1()
 {
   var d = new Date
   str = d.getHours ()+":"+d.getminutes()+":"+d.
      getSeconds() document.getElementById('sp1').
        innerHTML = str;
 }
  </script>
  </head>
  < body>
  <span id= "sp1" style =" color : red ; font -size:30">
   </span>
   </body>
```

* Javascript string object. This object is used to work with piece of text. String objects are created with the help of new String.

Syntax :

```
var txt = new string ("string");
 or more simply:
var txt = " string";
```

String object supports the following list of properties.

length property : It returns the length of string in characters.

Syntax : string.length

E.g.

```
<head>
<script type = 'text/javascript'>
 var str = " NareshiTechnologies";
 document . write (str . length);
 </script>
 </head>
```

Example :- with function

```
<head>
 <script type = 'text/javascript'>
 function mylen()
 {
```

[redacted]

```
var str = "Naresh iTechnology";
var x = document.getElementById("ln");
 x.innerHTML = str.length;

}

</script>
</head>
<body>
<P> Click the button to display the length of
 the string is : "</P>
<button onclick = " Mylen()"> ClickMe </button>
</body>
```

String object methods

String object supports the following list of methods.

1] toUpperCase()
2] toLowerCase()
3] charAt()
4] match()
   etc.....

1] toUpperCase()

This method is used to display the given string in capital letters.

Syntax:

```
str.toUpperCase()
```

2] toLowerCase()

This method is used to display a given string in Lowercase character.

Syntax: str . toLowerCase();

```
<body>
<script type ='text/javascript'>
 var str = " NareshiTechnologies";
 document . write (str . toUpper Case ());
 document . write ("<br/>");
 document . write (str . toLower Case ());
document . write ("<br/>");
document . write (str . charAt (5));
<lscript>
 <lbody>
```

* Javascript Math Object :- This object allows you to perform mathematical tasks. . The Math objects supports several properties & the methods

Java script PI property :- The PI property returns the radio of a circle's area to the square of its radiu approximately 3.14.

Syntax :- Math . PI

```
<body>
<script type='text/javascript>
 document. write (Math PI);
  </script>
   </body>
```

PI property must be in upper case.

E.g = with function.

```
    <body>
    <script type='text/javascript'>
     function myPvalue()
     {

     document.getElementById("p").innerHTML=Math.PI;

     }
      </script>
      </head>
      <body>
      <p id="py"> Click the button to display the PI
       Value....</P>
      <button onclick="myPvalue()"> ClickMe</button
        </body>
```

* Java script number Objects : This object is used to work with primitive numeric values.

```
var num = new Number(value);
```

Number object properties.

Property	Description
MAX_VALUE	Returns the largest number possible in Javascript
MIN_VALUE	Returns the smallest number possible in JS.
NaN	Represents a "Not-a-Number" value

JavaScript RegExp Object :- It describes a patterns of characters , simple patterns can be single character ,' complicated pattern can be consists of more character .

Syntax :-

```
var patt = new RegExp (pattern , modifiers);
 or more simply :
 var patt = / pattern / modifiers ;
```

* Brackets

These are used to find the range of Characters.

The following table describes the number of few characters

Expression	Description.
[a b c]	find any character between the brackets.
[^abc]	Find any character not between the brackets.
[0-9]	Find any digit from 0 to 9
[A - Z]	Find any digit from uppercase A to uppercase Z.

* Quantifiers.

Quantifiers	Descriptions
n+	Matches any string that contains at least one n
n*	Matches any string that contains zero or more n's
n?	Matches any string that contains zero or one n.

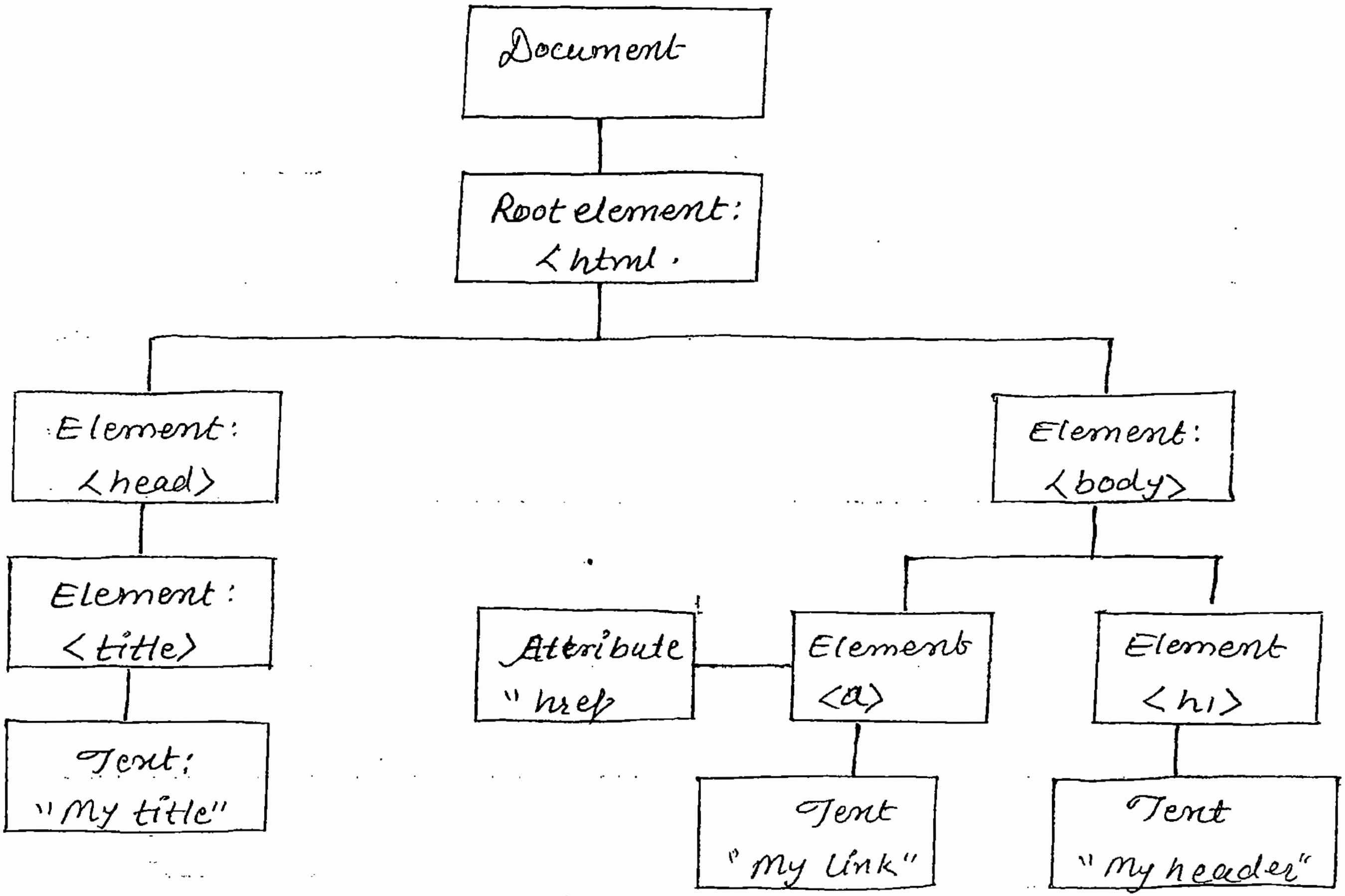
* Metacharacters A metacharacter is simply an alphabetical character preceded by a backslash.

Character	Description
.	a single character
\s	a whitespace character (space)
\S	non-whitespace character
\d	a digit (0-9)
\D	a non digit
\w	a word character (a-z, A-Z, 0-9, _)
\W	a non word character

What is HTML "DOM" (Document Object Model)

Dom is an platform and language nutral interface that allows the programs & scripts to dynamically access & update the content structure and style of the document .

Dom has the following detailed structure.



The above structure represents as follows.

1. The entire document is a document node.

2. Entry HTML element is an element node

3. The text inside HTML elements are text node.

4. Every HTML attribute is an attribute node

5.

```
<body>
< p id = "demo"> Welcome to HTML DOM</P>
< script type = 'text / javascript'>
  document . getElementById ("demo") . innerHTML =
  "Hello World!";
  </script>
 </body>
```

Event.button: It indicates which mouse button cause the event

Syntax: event. button

W3C its values should be :-

Left button - 0
Middle button - 1
Right button - 2.

According to microsoft values should be :-

Left button - 1
Middle button - 4
Right button - 2

Example

```
<head>
< script type = "text / javascript">
  function whichButton ( event)
  {
    if (event. button = = 2)
    {
     alert (" You clicked the right mouse button!");
    }
  }
  </script>
  </head>
  < body onmousedown = "whichButton (event)">
  <P> Click Here Observe ...! </P>
  </body>
```

InnerHTML property :- The innerHTML property is used along with getElementById within your Javascript code to refer to an HTML element and change its contents.

Syntax :-

```
document.getElementById ('{ ID_of element}').
 inner HTML = '{ content }';
```

Containers :- Elements can hold other html Elements / Controls

Example :- Div, P, Table, Span ...!

Non Containers :- Elements can hold only text can not hold html Controls / Elements

Example :- Text box, Button, Radio, Textarea ..!

Note :- All containers are paired tag, But all paired tags are not containers (containers having inner html property , non containers having value property.

Example :-

```
    <head>
    < script type = 'text / javascript'>
     function Mytext()
    {
      var Value = document . get ElementById ('txt1').
     value;
        alert (" The Values is : " + Value);
     }
      < /script>
      </head>
      <body>
     <P> click the button to display the text from
     a text box based on Value property ..</P>
     <input type = 'text' value = " Javascript " id=
     "txt 1"> <br/>
    < button onclick = " mytext ()"> ClickMe </button>
     <body>
```

Example 2.

```
      <head>
     <
     fun
     {
      var value = document . get ElementById ('P1).
         innerHTML ; alert (" The Values is : " + Value);
```

```
}
 </script>
 </head>
 <body>
<P> Click the button to display the text from a text
 box based on value property ....</P>
 <P id = "P1"> <img src = "html5.png" width=100p
  height = 100px></P>
 <button onclick = "Mytext()"> Click Me </button>
 </body>
```

Working with Javascript Validation :-

Javascript can be used to validate the data in HTML forms before sending of the containts to a server.

Javascript form validation is provide a method to check a user entered information before click on submite.

Form validation generaly perform in the following two ways.

1] Basic validation
2] Data formate validation

(1) Basic Validation :- The form must be checked to make sure data was entered into each form fields that required it. This would need just loop through each field in the form and check for data.

②

The data that is entered must be checked for correct form and value. This would need to put more logic to test correctness of data.

Validating Text Box &

```
<head>
<script type ='text / javascript'>
 function notEmpty()
 {
   var my TextField = document.getElementById
   ('my Text');
   if (my TextField.value != " ")
   {
    alert ("You entered :" + myTextfield.value)
   }
    else
   {
    alert " would you please enter some text? ")
    }
   }
    < /script>
    < /head>
    < body>
    <input type ='text' id = 'myText' /> <br/>
    < input type = 'button' onclick = 'notEmpty()'
     Value = 'Form Validate'/>
    </ body>
```

Validating Text-Box with border color :-

UserName :

Password :

```
<head>
< script type ='text/javascript'>
 function funchklen (len , cid)
 {
   if (len<6)
  {
    document.getElement ById (cid) . style . border
   color = "red"
  }
  else
  {
   document.get ElementById
  (cid ) . style . border Color ="silver"
 }
}
   </script> |
   </head>
   <body>
Username : < input type =" text " id = "txt 1"
   onblur =" funchklen (this . value. length , 'txt 1')">
  <br>
Password : < input type =" password " id = "txt2"
 onblur =" funchklen (this.value . length , 'txt 2')">
 </body>
```

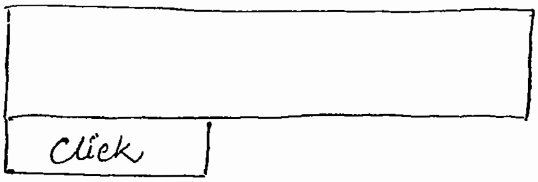
Validating Radio Buttons :-

You are?

○ male ◉ female

[submit]

```
<head>
 < script type = "text / javascript">
   function validate() {
    var V1 = document . get Element By Id ('male').
       Checked;
   var r2 = document . get Element By Id ('female').
    checked;
   if ((r1 == " ") && (r2 == " ")) {
     alert ("select either Male or Female");
   return false;
   }
    return true;
   }
    < /script>
    < / head>
```



```
<head>
<script type ='text/javascript'>
function fun1()
{
  if (document.getElementById('un').value.lengt
= 6 && document.getElementById('pw').value.
length >=6)
  {
   document.getElementById('but1').disabled=fals
  } else
  {
  document.getElementById('but1').disabled = true
  }
}
<\script>
</head>
<body>
<input type ='text' id = 'un' onblur ='fun1()'>
 <br>
  < input type ='passwerd' id = 'pw' onblur ='fun1()'>
  <br/>
   < input type ="button" value = "click" id = "but1"
    disabled>
    </body>
```

nithtmljavascript@gmail.com.

http://www.javascriptkit.com
http://www.webplatform.org.