HTML5 & CSS3
JavaScript

By
Mr. subba raju Sir

HTML5 & CSS3

Introduction to web : Collection of electronic pages is called web, father of web is Tim Berners-Lee

Define Network: collection of computers interlinked together. Klein Rock is father of network. First network name is ARPANET (Advanced Research projects Agency network). First protocol in IT industry is FTP (File Transfer protocol).

Internet : International network. father of internet is Vint cerf American computer scientiest.

Email : Electronic mail services. It is a free service to communicate with other internet users. father of email is Shabeer Bhatia, first email service is hotmail.com

SMTP [simple mail transfer protocol] : It takes care of delivering emails from one server to another.

MIME [multipurpose internet mail extensions], it exchange different kinds of data.

Define W3C [world wide web consortium] : partnership

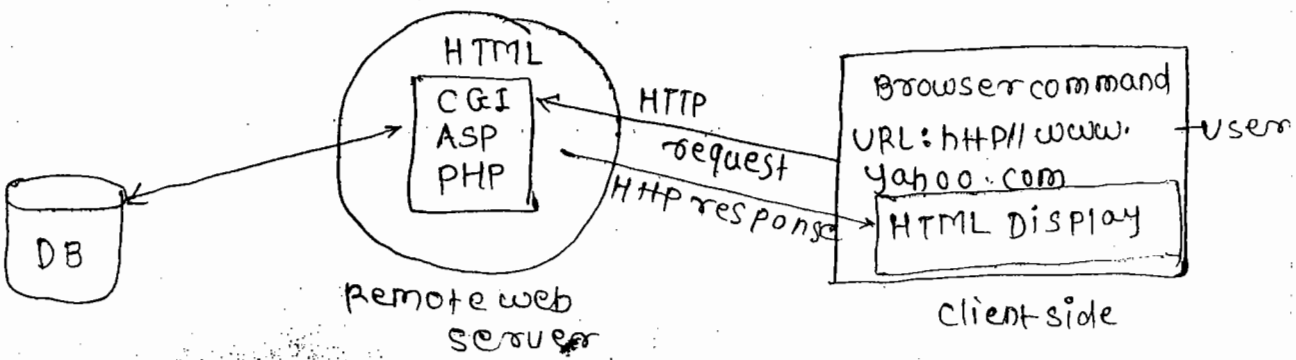
It was founded in 1994 by team Tim Berners-Lee. It has 400+ members partnership.

Well known members are facebook, google, Nokia, samsum, IBM, microsoft etc.

Define webbrowser: It is client side lightweight software, It takes http request from client to server, it renders (brings) http response from server to client.

Web Architecture

1) presentation layer
2) Business Logic Layer
3) Data Access Layer



major webbrowser: In web environment there are 7 webbrowsers

1] Google crome (2008)
2] Mozila firefox (1998)
3] Internet explorer (1995)
4] Mini opera (1995)
5] Safari (2005

1989: GML
1991: SGML
1994: HTML
1998: XHTML
2008: HTML 5
2014: HTML 5.1

one open webframework to develop web pages.

Define Blog: It is daily updating website or webpage, every post displayed in reverse cronological order (latest always first)

Forum: It is online discussion website to exchange resources each other.

Define HTTP: It is a transfer protocol to exchange hypertext documents in the world wide web

Define HTTP(S): secured transfer protocol to exchange hypertext documents with the help of SSL. (ciphertext)

Addons Define addon: Additional feature to the web browser
e.g. Any Toolbar.

Define plugin: It is an additional software to add the web browser
e.g. adobe flash player
adobe pdf reader

Define webpage: It is an electronic page developed on Html. It is classified into two types.
1] static webpage
2] Dynamic webpage

1] Static webpages: A users unable to interact directly with these webpages.
e.g. Html, css.

2] Dynamic webpages: A user can able to interact directly with these webpages.
e.g. Html, css, Javascript.

Define website: collection of webpages or web documents.
These are classified into two types.

1] static website:
- It is completly deveoped on client side technologies
  e.g. Html, CSS, javascript, Html5, XML.

2] Dynamic websites: These websites deveoped on client & server side technologies.
e.g. Html5, CSS3, jquery, php, ASP, JSP etc.

Define script: It is a kind of programing language with light weight feature. scripts are classified into the following two types.
1] client side script:
2] server side script

1] client side scripts: These scripts are executing within the web browser. e.g. java script, jquery, json.

2] Server side scripts: These scripts are running within the web servers Apache, Tomcat, IIS.

Introduction to HTML:

Define HTML: It is specially designed hypertext for web browsers, with meaningfull tags or elements in simple english language.

Features: 1] It is not case sensitive
2] It is browsers mother language.
3] It is global language.
4] It is collection of tags and elements
5] we can create static webpages
6] It is error free english language.
7] It is a markup language. [special text]

```
<html>
<title>
```
markups.

HTML History Line: W3C release the following versions. [Editions]

1989 ⇒ GML [Generalised markup language]
1991 ⇒ Standard generalized markup language
1994 ⇒ HTML (Hypertext markup language)
1998 ⇒ XHTML [XML + HTML] ← sms, mms, Email

2008 ⇒ HTML 5 [Advanced markup language for mobiles]

2014 ⇒ HTML 5.1 [Advanced Hypertext for smart phones], widgets, gadgets] → small electronic devices.

HTML Versions

Ian Hickson

from w3c organization there are several versions released

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]

w3c.org
validators
controls
oceans.mp4

Define Tag : It is special kind of text, placed between left angular brace and right angular brace (< , >)

Syntax: <- - - - -> example : <html>

Types of Tags: Tags are classified into the following two types.

1] paired Tags

2] Nonpaired Tags

1] paired Tags : These tags are having opening and closing tags.

Example: <Html> </Html>

<Head> </Head>

The closing tag starts with a forward slash.

2] Non paired Tags :

The tag that have only opening tags but no closing tags

example:
 break

<hr> horizontal rule

<img> Image

Structure of HTML

As per w3c standard HTML Document has the following detailed structure

```
<html>
    <head>
        <title> ..... </title>
    </head>
    <body>
    ---
    </body>
</html>
```

html Common or critical elements

In html document the following elements are critical or common elements.

1] HTML
2] Head
3] Title
4] Body

How to create a webpage

To create a webpage follow the steps

1] Launch any text editor (Notepad)
2] Enter required html source code

```
<html>
<head>
<title>
my first web page
</title>
</head>
<body>
welcome to web world....
</body>
</html>
```

3] save with .htm or .html extension
4] Right click on the save file open with any web browser
5] If you need to update any text or logic, right click on the file open with Notepad format.
6] Do require changes and save it. Refresh the web page.

HTML 5 comments : comments are non executable statement or ignore statements, with the help of these comment notations we can declare customized statements in the source code.

HTML 5 supports 1 comment notation for single and multiline. Every comments begins with <!-- ——— -->

Example 1 : comments

```
<html>
<head>
<title> HTML comments </title>
</head>
<body>
<!--kspaju--> <- single line comments
<!welcomewelcome
welcomewelcome--> <- multiline comments
</body>
</html>
```

Note: comments are not applicable in the title part

e.g. 1)
```
<html>
<head>
<title>
<!-- HTML comments -->
</title>
</head>
<body>
welcome to HTML5
</body>
</html>
```
Not applicable

Parts in Html documents

Generally every Html documents contains the following parts.

1) version information
2) head section
3) Body section

1) version information

<! DOCTYPE html> represents version of the html. It is the first line in html code. It is related to html5 syntax.

Lang attribute: It represents the primary languages of your webpage It contains two letter country code and language code.

e.g. 1. English (U.S)
```
<html lang="en-US">
.....
</html>
```

2. English (U.K.) Great Britan)
```
<html lang="en-IN")
.....
</html>
```

3. English (I.N)
```
<html lang="en-IN")
.....
</html>
```

Note: If language not declare default U.S. english.

2] head section : This section supports the following list of tags.
<HHe> , <link> , <meta> , <style> , <script>

Title : It defines HHe of the webpage. It is paired tag

syntax: <HHe> ..... </HHle>

```
<html>
 <head>
      <Hle> My first webpage </HHe>
 </head>
 </html>
```

favicon.co.uk

link : It defines the relaHonship between a document and an exter nal resource. It is a non paired tag.

syntax: <link>

Example:

Define favicon : It has the following alise name
1] shortcut icons
2] graphic images
3] url icon
4] page icon etc.

A browser support favicons and displays in the browsers url bar.
The following image extensions are pretered.
1] .PNG : pointable network graphics
2] . GIF : Graphical interchange format
3] ICO : Icon format
These images must be 16*16 or 32*32 pixels.

example1 :

```
<html>
 <head>
     <Hle> My HTML 5 logo
      </HHe>
   <link href="html5.png" rel="icon" type="imagelico">
     </head>
     </html>
```

global image path every browser recognize

example 2 :

```
<html>
<head>
<HHe> My fish floating
 </HHle>
<link href="C:\users\subbaraj\pictures\fish1.gif"
    rel="icon" type="imageli co">
```

file:/// ← file path

HTML5 meta element

Meta means after, metadata means data about data. meta elements are classified into different categories. The following are frequently use

1) meta charset
2) meta keywords
3) meta description
4) meta title
5) meta author

meta charset: utf stands for unicode transformation format, it is used o develop a web page different languages like arebic chinese, japanies, hindi etc.

Syntax: HTML4 syntax: <meta http-equiv="content-Type" content="text/html"; charset=utf-8">

HTML5 syntax: <meta charset="utf-8">

what is unicode: A provides a unique number for every characters. s per w3c standard unicode indicates the following information.

no matter what the platform
no matter what the program
no matter what the language

```
<head>
  <HHe> my web page <1HHe>
<meta charset="utf-8">
<1head>
<body>
<h1 style='color: blue'> Hello how are you </h1>
<h2 style='color: blue'> हैलो हाउ आरे यू ? </h2>
 </1head>
 </1body>
```

meta keywords : These keywords related to search engines to declare meta data

Syntax: <meta name="keywords" content="required list of keywords">

Example: <meta name="keywords" lang="en-IN" content="live cricket scores, cricket, " />

meta description : It defines the description of your page

syntax:

<meta name="description" content="required page description">

Example:

<meta name="description" content="A online beginner java tutorial website". />

meta title : It defines the title of meta content.

Syntax: <meta name="title" content="required meta title">

Example: <meta name="title" content="welcome to Naresh IT Hyderabad."/>

meta author : It defines author of the web page.

Syntax: <meta name="author" content="name of the author">

Example: <meta name="author" content="kssutbaraj"/>

Style: It is used to declare style information for a html document.

example: <head>
<style type='text/css'>
- - - - - - - -
- - - - - - -
</style>
</head>

Script tag : It is used to define client side script such as java script

<head>
<script tag='text/javascript' language="javascript">
- - - - - - - - -
- - - - - - - - -
</script>
</head>

Body section : It is most essential section, it contains text, hyperlinks, images, special characters, tables, frames, forms etc. It is a paired tag.

Syntax: <body> ..... </body>

attributes	Parameters	
bgcolor	~~Parameters~~	Color Name / code
background	~~Color Name / color code~~	image path
text	Color Name / color code	

Example 1:

```
<body text = "red" bgcolor = "yellow">
HTML5 is New HyperText for mobile APPS....!
</body>
```

(parameter, Tag, attribute)

Example 2: with color code

```
<body text = "FF0000" bgcolor = "FFFF0">
HTML 5 is new Hypertext for mobile APPS...!
</body>
```

Example 3: background attribute

```
<body background = "html5.PNG">
HTML5 is new hypertext for mobile APPS!!
</body>
```

Example 4: `<body background = "file:///C:\users\subbaraj\pictures\simly5.jpg">`

```
HTML5 is new Hypertext for mobile apps!!
</body>
```

Example: background with http path

```
<body background = "http://www.google.com/images/logos/google_logo_41.png">
HTML5 is new hypertext for mobile APPS!
</body>
```

* Hexadecimal color code system

As per w3c standard hexadecimal color system before, the following are web safe colors

Black	Gray	silver	white
yellow	Lime	Aqua	Fuchsia
Red	Green	Blue	purple
Maroon	olive	Navy	Teal

Color values: HTML5 colors are define using hexadecimal notation. It is six digit representation of a color. It is the combination of RGB.

Hexa decimal

6 (10 = 16)

A-F
a-f

0-9

RGB

Red ⇒ 0-255 ⇒ 256
Green ⇒ 0-255 ⇒ 256
Blue ⇒ 0-255 ⇒ 256

16 million number
16 million color

R ⇒ Red ⇒ 2digits
G ⇒ Green ⇒ 2digits
B ⇒ Blue ⇒ 2 digits
6digits

least color ⇒ 00 (No color - 00)
Highest color ⇒ FF (dark color - FF)

RGB ⇒ #FF0000 ⇒ RED
RGB ⇒ #00FF00 ⇒ Green
RGB ⇒ #0000FF ⇒ Blue
RGB ⇒ #000000 ⇒ Black
RGB ⇒ #FFFFFF ⇒ white
RGB ⇒ #FFFF00 ⇒ yellow
RGB ⇒ #FF00FF ⇒ pink (dark)

Introduction to HTML attributes

Attributes are special features of a tag, every tag contains none or one or many attributes.

Parameters: parameters are the values that we assign to an attribute

Syntax: <tag attribute="parameter">

<body bgcolor="blue">

Types of attributes: In HTML5 attributes are classified into different types.

1] Element specific attribute:
2] Global attributes
3] Java script Event attributes
4] optional attributes

1] Element specific attributes: These attributes are exclusively for a specific element of tag

e.g. <body> → bg color, background, text
<img> → src, width, height, alt

2) Global attributes: It has the following alise names

1] Standard attributes
2] Dynamic attributes
3] common attributes etc.

e.g. 1] class 2] id 3] lang 4] insert 5] spellcheck 6] style

```
<body>
  <b style='color:blue'> It is in bold format </b>
  </body>
```

Event Attribute : These attributes are related to javascripting

Example: 1] onclick 2] oninput 3] onprogress 4] onchange

5] oninvalid 6] onLoad.

```
<body>
   <button onclick = "alert ('welcome to HTML5')" > click me
  </button>
 </body>
```

Optional Attributes: These attributes HTML5 treating as optional or not required

example:

HTML 4.01 - `<link rel = "stylesheet" type = 'text/css' href = "hi.css">`

HTML5 - `< link rel" stylesheet " href = "hi.css" >`

HTML5 special characters : These are popularly known as character entities. All entities begin with an "&" and ends with ";"

example: © ®

™
←
→
♠
♥

HTML 4 drawbacks:

1] Not supporting graphics
2] poor in video/audio support
3] Requires external plugins to run flash, media controls
4] very few input controls are available.
5] Need to use external language like javascript to validate controls.
6] Does not support 2D and 3D Animations.

Deprecated Element

The following list of elements are deprecated from HTML5

1] <acronym>
2] <Applet>
3] <basefont>
4] <big>
5]
6] <dir>
7] <font>
8] <frame>
9] <frameset>
10] <noframes>
11] <noframes>
12] <s>
13] <strike>
14] <tt>
15] <u>
16] <xmp>

1] <acronym> : It defines abbreviation or full form from initial letters It is a brand paired tag.

syntax: <acronym> </acronym>

```
<html>
<body>
Hi, friends reply < acronym title="As soon As possible" > ASAP
</acronym>
</body>
</html>
```

<abbr> : It indicates an abbreviation or an acronym It is a paired tag. syntax: <abbr> ..... </abbr>

```
<body>
Hi, friends reply <abbr title="As soon as possible"> ASAP
</abbr>
</body>
```

(attribute)

2] <applet>: It is a small program in java language. It requires additional plugins to work. It is a paired tag

syntax: <applet> ..... </applet>

<object> : It defines an embeded object within a html document using this element we can embed audio, video, java applets, activex, pdf etc. It is a paired tag.

<object> ..... </object>

```
<body>
<object data="windows xp shutdown.wav">
<param name="autoplay" value="true" />
</object>
</body>
```

<basefont> : It defines default font color, size, family for all the text in a document. It is a nonpaired tag.

Syntax: <basefont>

It supports only Internet Explorer web browser.

```
<head>
<basefont color="blue" size="5" face="tahoma" />
</head>
<body> <p> This is a paragraph </p> </body>
```

Example 2 with css:

```
<head>
<style type='text/css'>
div
{
font-size:20px; font-family: tahoma color: blue;
font-weight: bold;
}
</style>
<body>
<div> welcome to css font properties.... </div>
</body>
```

<big> : It is used to make text bigger. It is a paired tag

Syntax: <big>..... </big>

```
<body>
<big> welcome to Big Tag </big>
</body>
```

Example 3: css

```
<head>
<style type='text/css'>
div
{ font size: 10px;
}
</style>
```

```
</head>
<body>
<div> welcome to css font property </div>
</body>
```

 : It displays the text in page center. It is a paired tag. syntax: .....

example:
```
<body>
<center> welcome to center property </center>
</body>
```

Example (css):
```
<head>
<style type = 'text/css'>
div
{
text align = "center";
}
</style>
<body>
<div> welcome to center property </div>
</body>
```

<dir> : It is used to list titles. It is a paired tag.
syntax: <dir>..... </dir>

Example1:
```
<body>
<dir>
<li> HTML5 </li>
<li> CSS </li>
</dir>
</body>
```

Example2:
```
<body>
<ul type = "square" >
<li> HTML5 </li>
<li> CSS </li>
</ul>
</body>
```

: It is used to format the text such as changing text size color, family etc. It is a paired tag

syntax: .....

eg. 1]
```
<body>
<font color="blue" size="5" face="tahoma">
Bye to font tag </font>
</body>
```

eg. 2 (CSS):
```
<body>
<head>
<style type='text/css'>
div
{
font size="10" color="blue" font-family="tahoma";
}
</style>
</head>
<body>
<div> welcome to css font properties </div>
</body>
```

<frameset>: using this tag we can divided the web page as multiple frames. It is paired tag.

<frameset> ..... </frameset>

<frame>: To place the files in frames, we use frame tag. It is a nonpaired tag.

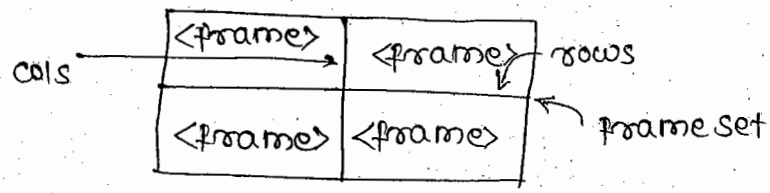
syntax: <frame>

<noframes>: It is used to specify an error message, if the frame fails to load. It is a paired tag.

syntax: <noframes> ..... </noframes>

Example]:
```
<frameset rows="50%,50%" cols="40%,*">
<frame src="http://www.Naresh IT.com">
<frame src="http://www.Naresh IT.in">
<frame src="http://www.nacreservices.com">
<frame src="http://www.sheshajobs.com">
```

```
    </frameset>
  <body>
  <noframes>
  <p style="color:red"> oops your Browser Not supporting
   </p>   </noframes>
  </body>
```



<iframe> : i stands for inline. It enables you to present another resource within the same page. It always occupies small portion of the document. It is mainly usefull for web advertising. It is a paired tag.

syntax: <iframe> ..... </iframe>

attributes	parameters
src	The path of url/image
scrolling	auto, yes, no
align	left, right, top, middle, bottom
height	pix or %
width	pix or %
name	name
frameborder	1,0

Example:

```
<body>
    <iframe src="htm.png" width=150px height=150px
scrolling="no" name="hframe" frameborder="0">
<p style='color:red"> oops your browser not supporting
 </p>
 </iframe>
<iframe> src="html5.png" width=150px height=150px
align="right" name="vframe">
<p style='color:red'> oops your browser not supporting </p>
 </iframe>
  </body>
```

<strike> : It displays strike through text : It is a paired tag

Syntax: <strike> ..... </strike>

```
<body>
<s> It is removed content from the page </s>
<br>
<strike> It is removed content from the page</strike>
<br>
<del> It is removed content from the page </del>
</body>
```

<tt> : stand for teletype text. It is used to declare special formatted text, it is a paired tag.

Syntax: <tt> ..... </tt>

```
<body>
<p> This text is normal </p>
<tt> This text is teletype text </tt>
</body>
```

<u>: underline - It defines underline text. It is paired tag.

Syntax: <u> ..... </u>

Eg.

```
<body>
<u> It is underline format </u>
</body>
```

```
<head>
<style>
div
{
text-decoration : underline → overline → line-through
}
</style>
</head>
<body>
<div> welcome to css </div>
</body
```

<IsIndex> : It is used to display single line text input control It is a non paired tag.

syntax: <isindex>

Eg.
```
<body>
<isindex prompt ="Enter one or more search terms" >
<\body>
New
<body>
<input promt = " Enter one or more search terms">
<\body>
```

<xmp> : The xmp tag defines unformatted text within an information region. It is paired tag.

syntax: <xmp> ..... </xmp>

```
<body>
<xmp>
  1   2   3   4
        5
<\xmp>
<pre>
  1 2 3  4
      5
<\pre>
<\html>
```

Features of HTML5 : It is the latest version developed by w3c and WHATWG. It is has the following list of unique features

1] HTML5 symantic elements
2] HTML inline elements
3] Aduanced forms
4] Aduanced form input types
5] Latest form attributes
6] Updated input attributes
7] New form elements
8] HTML5 multimedia

HTML5 Web RTC (Real Time communication)

phonegap.org
vimeo.com
webplatform.org

10] HTML5 graphics (2D and 3D)
11] HTML5 canvas
12] HTML5 SVG (Scalable vector Graphics)
13] MathML (mathematical markup languages)
14] HTML5 Geolocation
15] HTML5 drag and drop
16] HTML5 web storage
17] server sent events
18] HTML5 web workers
19] HTML5 application cache
20] HTML5 speech input
21] performance and integration

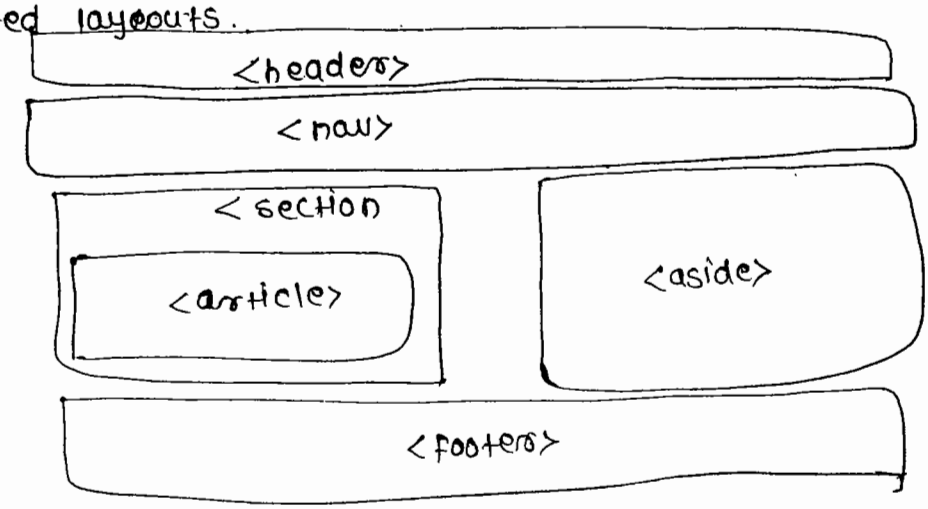
HTML4 page Layouts:

In html4 page layouts are as follows

```
<div id = "header">
<div id = "nav">
<div id = "section>          <div id = "aside">
<div id = "article">
                <div id = "footer">
```

HTML5 page Layouts:

HTML5 supports symantics, with the help of semantics we can design updated layoouts.



★ HTML5 structure:

HTML5 has the following detailed document structure

```
<!DOCTYPE html>
<html lang = "en-US">
<head>
 <meta charset = "UTF-8"/>
  <title> pageTitle </title>
 </head>
 <body>
 <header>..... </header>
  <nav> ..... </nav>
  <section>
  <header>..... </header>
  <article>
    <p>.... </p>
  </article>
  <footer>..... </footer>
   </section>
  <aside>..... </aside>
  <footer >..... </footer>
</body>
</html>
```

★ HTML5 symantics: HTML5 supports the following new element for better structure

1] <section>
2] <article>
3] <header>
4] <footer>
5] <hgroup>
6] <aside>
7] <command>
8] <details>
9] <summary>
10] <figure>
11] <figcaption>
12] <nav>
13] <wbr>

14] <bdi>
15] <bdo>
16] <ruby>
17] <rt>
18] <rp>

13] General semantics
2] Arabic semantics
3] Chinese semantics.

1] General Semantics

i] section: It defines section in a document such as chapters, header, footers any other section of the document. It is a paired tag.

Syntax: <section>..... </section>

Attributes: No Element specific.

ii] header: It is used to display subheadings, version information, navigational controls etc. It is a paired tag.

syntax: <header>..... </header>

Attributes: No element specific

NOTE: A <header> tag cannot be placed within a <footer>

iii] footer: It is used to define footer of an html document or section it is a paired tag.

syntax: <footer>..... </footer>

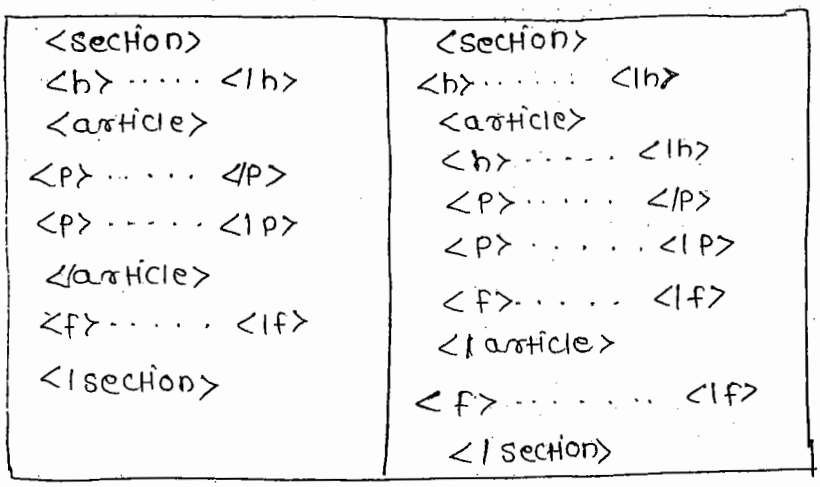
Attributes: Element specific attributes are none.

iv] Article: It is used to represent an article, it is an independant content on the webpage. It is a paired tag.

syntax: <Article>..... </Article>

Attributes: Element specific attributes are none.

Architecture for semantics



example1

```
<!doctype html>
<html lang = "en-IN">
<head>
<title>
welcome to semantics
</title>
<meta charset= "utf-8">
<meta name = "keywords" content = "List of required keywords
 for page">
<meta name = "description" content = "List of Required page
    Description>
<meta name = "title" content = "Required title for meta content">
<meta name = "author" content = "ksraju">
<style type= 'text/css'>
</style>
<script type = 'text/javascript' language = "javascript">
</script>
</head>
<body>
<section>
<header> power of HTML5 .... </header>
<article>
<header> Introduction to HTML5 ... </header>
<p> </p>
<p> </p>
```

<footer> All copyright Reserved © </footer>
</article>

<footer> content right protected &reg ; </footer>
</section>
</body>
</html>

<hgroup> :

It is used to group a set of one or more h1 to h6 elements. It is a paired tag.

Syntax: <hgroup> ..... </hgroup>

Attributes: Element specific attributes are none.

example1 :

```
<body>
 <hgroup>
  <h1> power of HTML5 .... </h1>
  <h2> Introduction to web ... </h2>
 </hgroup>
 <P>      </P>
 <P>      </P>
 </body>
```

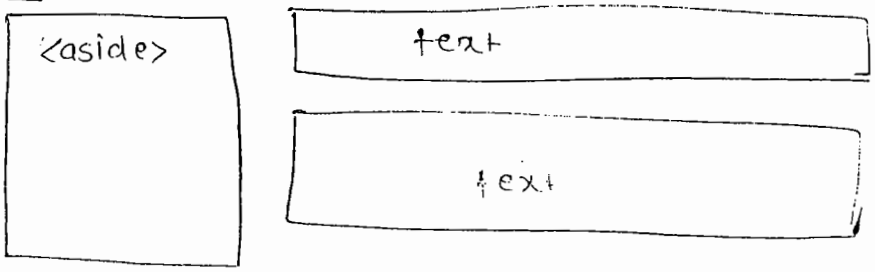
<aside> : It is used to represent content that is related to surrounding content within an article or a web page. It is paired tag.

Syntax: <aside> ..... </aside> ↓ could still stand alone in its own right.

example1 :

```
<body>
<aside> HTML5 is new hypertext .... </aside>
<P> some text </p>
</body>
```

Example2 :



```
<body>
<aside style="font-size:larger; font-style:italic; color:
blue; float:left; width:120px;">
    Some text
  </aside>
<p> some text </p>
<p> some text </p>
  </body>
```

example: aside with images

```
<body>
<aside style="font-size:larger; font-style:italic; color: blue
 float: left; width: 120px">
<img src="good morning.gif" width="100px" height="100px">
 </aside>
<p align="justify"> ..... </p>
<aside style="font-size: larger; font-style: italic; color: blue
    float: right; width: 120px">
<img src="simiy.jpg" width="00px" height="100px">
 </aside>
  <p align="justify"> ..... </p>
 </body>
```

<command>: It defines a command that the user can invoke. It is a paired tag.

<syntax>: <command> ..... </command>

```
<body>
<button onclick="alert('welcome to events')"> click me
   </button>
<command onclick="alert('welcome to events')"> click me
    </command>
 </body>
```

<details> : It is used to create an interactive widget that the user can open and close. It is paired tag.

Syntax : <details> ..... </details>

Attribute

Attribute	value	Description
open	open	specifies that the details should be visible (open) to the user.

<summary> : It defines the visible heading for the <details> element. The heading can be clicked to view/hide the details. It is paired tag.

Syntax : <summary> ..... </summary>

Attribute: Element specific attributes are none.

NOTE : The <summary> tag is currently only supported in Chrome & Opera.

The <summary> element should be first child element of <details> element.

▷ power of HTML5.... !!

widget / gadget for interactive (show/Hide)

```
<body>
<details open="open">
<summary> power of HTML5 </summary>
<P> file system APIS </P>
<P> Geolocation </P>
<P> semantics </P>
</details>
</body>
```

example 2

```
<body>
<details>
<summary style='color:#000ff'> power of HTML5 !!
</summary>
<ul type='square' style='color:green;font-family:
tahoma;'>
<li> file system APIs </li>
<li> Geolocation </li>
<li> device orientation </li>
</ul>
</details>
</body>
```

<figure>: It is used to apply a unit of element like images and group of text. It is a paired tag.

Syntax: <figure> .... </figure>

Attribute: Element specific attribute are none

<fig caption> : It defines a caption for figure element. This element represents a caption or a legend for a figure. It is a paired tag.

Syntax: <figcaption> ..... </figcaption>

Attributes: Element specific attributes are none.

```
<body>
<figure>
<img src="html5.png" width="300" height="250" />
<figcaption> It is html5 logo from w3c and whatwg
</figcaption>
</figure>
</body>
```

<nav>: It is used to declare navigational section of html document. It is a paired tag.

Syntax: <nav> ..... </nav>

Attributes: Element specific attributes are none.

```
<body>
<nav>
<a href = "http://www.nareshit.com"> Naresh IT </a>
<a href = "http://www.nareshit.IN "> NareshIN </a>
</nav>
</body>
```

<wbr>: It is used to word break opportunity. It is a non paired tag.

Syntax: <wbr>

Attributes: Element specific attributes are None.

```
<body>
<p> To learn Html5 you must be familiar with the HTML.
<wbr> CSS <wbr> JS... </p>
</body>
```

<bdo>: It is used to overright the current text direction. It can be useful when displaying hebrew and other languages It is a paired tag

Syntax: <bdo> .... </bdo>

Attributes

Attribute	value	Description
dir	ltr	specifies the text direction
	rtl	specifies the text direction.

```
<body>
<bdo dir = "rtl"> HTML5 is New hypertext markup
language
</bdo>
</body>
```

<bdi>: bidirectional isolation

This can be useful when displaying right to left text. It is a paired tag.

Syntax:

<bdi> ..... </bdi>

Attributes

Element specific attributes are none.

```
<body>
<ul>
<li> user <bdi> suresh </bdi> </li>
<li> user <bdi> Daniel </bdi> </li>
</ul>
</body>
```

Arabic

* Chinese Semantics

1] <ruby> : It used for specifying ruby annotations, which is used in East Asian typography. It is a paired tag.

Syntax: <ruby> ..... </ruby>

Attributes: Element specific attributes are none

2] <rt> (Ruby text) : It marks the ruby Text component of a ruby annotation. Ruby annotations are used in East Asian typography. It is a paired tag.

Syntax: <rt> ..... </rt>

Attributes : Element specific attributes are none.

3] <rp> : The HTML <rp> is used in ruby annotations for the benefit of browsers that dont support ruby annotations. It is a paired tag.

Syntax: <rp> ..... </rp>

Attributes: Element specific attributes are none.

```
<html>
<body>
<p> <ruby> ... <rt> ... </rt> ... <rt ... </rt> </ruby>
..... </p>
</body>
</html>
```

HTML5 Inline Elements

HTML5 supports the following inline elements

1] <mark>
2] <meter>
3] <progress>
4] <time>

1] <mark>: It is used for indicating text as marked or highlighted for reference purposes. It is a paired tag.

Syntax: <mark>..... </mark>

Attributes: Element specific attributes are none.

```
<body>
  <mark> HTML5 </mark> is new Hypertext markup language
for mobile apps....!
  <br/>
  <HTML5 is new hypertext markup language for mobile
  <mark> apps..... </mark>
<mark style='background-color: lightblue'>
```

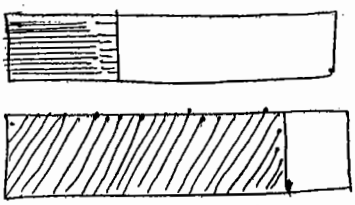
2] <meter> : It defines a scalar measurement within a known range This is also known as gauge. It is paired tag.

<syntax>:

<meter> ..... </meter>

Attributes

Attribute	value	Description
form	form_id	Speci one or more forms
high	number	Spe the rage that considered to be high.
low	number	Spe the range that is considered to be low.
max	number	Specifies the maximum value of the range.
min	number	Specifies the minimum value of the range
optimum	number	What value is the optimum value for gauge
value	number	Specifies the current value for the gauge.



↑ widgets or gadgets of meter tag.

```
<body>
 <meter value="2" max="10">
  <p style='color:red'> oops your Browser not supporting
   meter tag </p>
   </meter>
   <br/> </br>
    <meter value="g" max="10">
  <p style='color:red'> oops your Bowser not supporting
     meter element </p>
      </meter>
      </body>
```

NOTE: 1] If value is higher than high the gauge is yellow (when low available)

2] when value is lower than low if optimum is lower than low the gauge is Green.

3] if value is more than high then optimum is Red (when max available)

He got a [Red] on the exam.

He got a [yellow] on the exam.

He got a [green] on the exam.

```
<body>
 <meter <p> He got a <meter low="69" high="80" max="100"
 value="100">-B </meter> on the exam </p>
 <p> He got a <meter high="80" max="100" value="84">
     B</meter> on the exam </p>
 <p> He got a <meter max="100" value=50 min=10>
      B </meter> on the exam </p>
    </body>
```

<progress>:

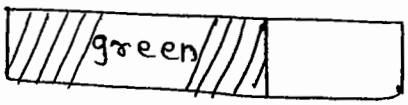
It is used to represent progress of a task. It is a paired tag.

<progress> ..... </progress>

Attributes

Attribute	value	Description
max	number	specifies how much work the task requires in total.
value	number	specifies how much of the task has been completed.

eg.




```
<body>
 <progress> value = "5"  max = "10">
  <p1 style = 'color-red' > oops your browser not supporting
   progress tag </p>
 </progress>
 </body>
```

<time>:

This element is used to presenting dates and times in machine readable format. It is a paired tag.

Syntax: <time> ..... </time>

Attributes

Attribute	value	Description
datetime	datetime	Gives the date/time being specified

e.g.

```
<body>
 we arrived at <time> 9:00 </time>
</body>
```

Working with HTML4 forms

→ These are the basic forms, form is a container it can hold other controls. It is a paired tag

Syntax: <form> ..... </form>

form attributes	parameters
Attributes	any name
name	
method	get, post
action	url (uniform resource locator)

Form Tags

Tag	Description
<form>	Defines a form for user input.
<input>	Defines an input field data
<button>	Defines a push button
<textarea>	Defines a text area
<label>	Defines a label to the description.
<fieldset>	Defines a border to the input data.
<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

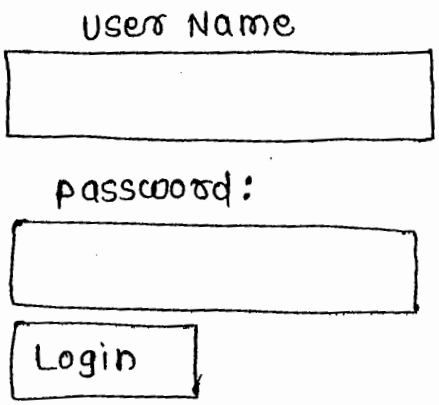
It is classified into the following two types.

1) Input fields
2) select fields

1) **Input fields:** The following table displays list of input field.

Field Name	Keyword	Syntax
text box	text	<input type = "text">
password box	password	<input type = "password">
Checkbox	Checkbox	<input type = "checkbox">
radio button	radio	<input type = "radio">
submit button	submit	<input type = "submit">
reset button	reset	<input type = "reset">
text area	textarea	<textarea> </textarea>

e.g.1




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

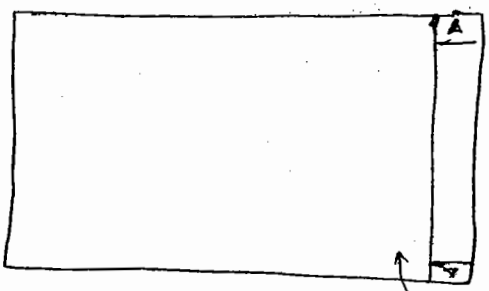
Input field Attributes

Attributes	parameters
Name	Any name
Value	any value
Size	pixels
maxlength	number
rows	number
cols	number
readonly	true, false
disabled	disabled
checked	checked
multiple	true, false

e.g. Input types with Attributes

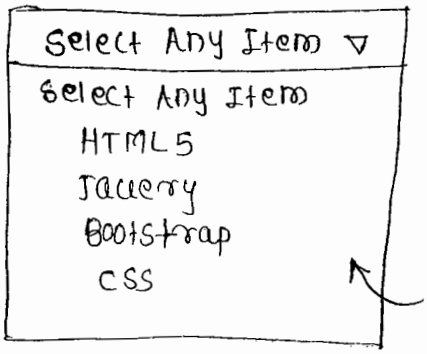
```
<body>
  <form>
  <label> User Name </label>
  <input type="text" name="uname" value="Enter your name"
   size="6px" maxlength= 6 readonly="true"><br>
  <label> password </label>
  <input type="text" name="pwd" value="Enter password">
    <br>
  <input type="submit" value="signIn" name="sn"
    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

Textarea



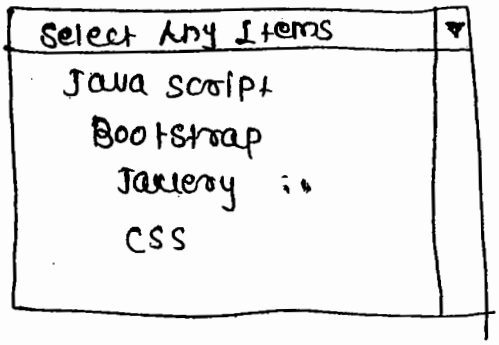

```
<Body>
  <form>
    <textarea rows="5" cols="24" name="tarea" id="tar1">
        some text.....
    </textarea>
    </form>
  </body>
```

Combobox:



```
<body>
 <form>
  <select>
     <option> Select Any Item </select>
     <option> HTML5 </option>
      <option> Jquery </option>
     <option> Bootstrap </option>
       <option> CSS </option>
     </select>
     </form>
     </body>
```

Listbox :




```
<Body>
  <form>
  <select multiple = " multiple">
   <option> Select Any Item </option>
    <option> HTML5 </option>
    <option> Jquery </option>
    <option> CSS </option>
  </select>
   </form>
    </body>
```

Radio Buttons :

What is your favourite web browser

O Internet Explorer 11 O Google Chrome

```
<body>
<p> what is your favourite web browser </p>
<form>
<input type="radio" name="browser" value="IE">
Internet Explorer 11
<input type="radio" name="browsers" value="GC">
Google chrome
</form>
</body>
```

checkboxes:

what is your favourite web browser

☐ Internet Explorer 11 ☐ Google chrome

```
<body>
<p> what is your favourite web browser </p>
<form>
<input type="checkbox" name="browsers"/> Internet
Explorer 11
<input type="checkbox" name="browsers"/>
Google chrome
</form>
</body>
```

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

syntax:

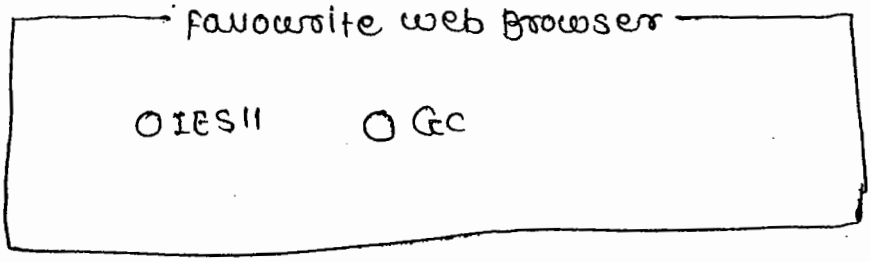
```
<fieldset> ..... </fieldset>
```

```
<body>
<p> what is your favourite browser </p>
<form>
<fieldset>
<input type="radio" name="browser"/> IE 11
<input type="checkbox" name="browser"/>
GC
</form>
</body>
```

<legend>: It is used with fieldset to give a title to each set of fields. It is a paired tag.

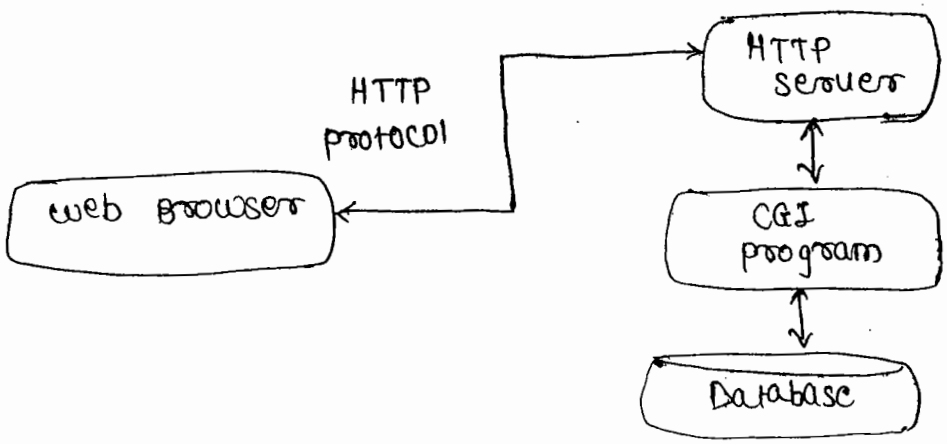
Syntax: <legend> ..... </legend>

Attributes	parameters
Align	right, center, left




```
<Body>
 <form>
  <fieldset>
     <legend> favourite web browser </leged>
  <input type= "radio" name =" browser"> IE 11 <br>
  <input type= "checkbox" name= "browser"> Gc <br>
   </fieldset>
   </form>
   </body>
```

HTTP: It is transfer protocol, it is design to enable communications between clients and servers. It has the following detailed architecture



HTTP supports the following two request methods

1] get

2] post

1] get : In this method we dont have security for our data and only limited data can be sent to the server page. It is carrying raw data between client-server. This is the default method of the form.

·Syntax :

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

e.g. 1

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

→ get method satisfies the following list of point.

1] GET request can be catched.

2] GET request remain in the browser history

3] GET request can be bookmarked

4] GET request should never be used when dealing with sensitive data.

5] GET request have length restriction.

★ Action Attribute

It is used to specify the url of the server page to which we want to send our data.

Syntax: `<form name = "myform" action = "user.aspx">`

post method

→ In this method we have security for our data and we can send bulk of data to the server page. It transfers encrypted data from client to server.

Syntax:

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

Example:

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

Working with Aduanced forms:

HTML5 supports the following aduanced form control, these are latest input types in web 2.0.

1] color (color chooser)
2] date (popup calender)
3] datetime (datetime chooser)
4] datetime-local (datetime chooser)
5] Email (Email Entry)
6] month (month chooser)
7] number (spinner)
8] range (slider)
9] search (search query input)
10] Tel (Telephone input)
11] Time (Time selector)
12] url (URL Entry)
13] week (week chooser)

NOTE: All major browsers supports all the new input types. If they are not supported, will behave as regular text fields.

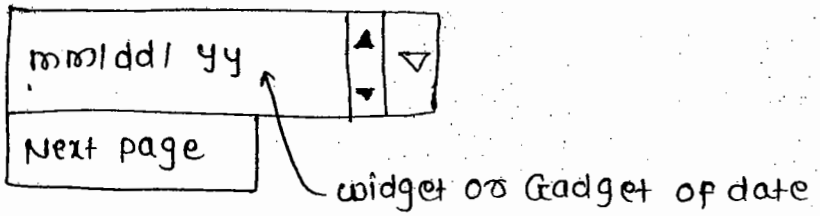
Date pickers

1] date
2] datetime
3] datetime local
4] month
5] time
6] week

1] input type date: It allows the user to select date. In real time applications it is very usefull for ticket booking or taking appointments ordering food items etc.

Syntax: input type = name

- Select valid Date

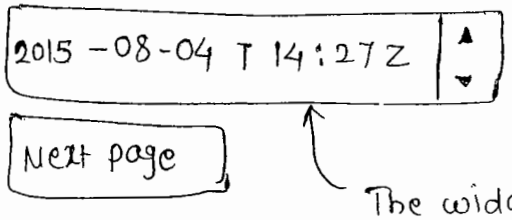


ex:
```
<body>
<form action= "nit.html" name = "myform" id = "form1">
<label style = 'color: blue'> Select valid Date.... ! </label><br/>
<input type = 'date' name ="dt" > <br/>
<input type = 'submit' value = " Next page">
</form>
</body>
```

2] datetime : This input type allows the user to select date and time with time zone.

syntax: input type = name

- Select valid Date and time

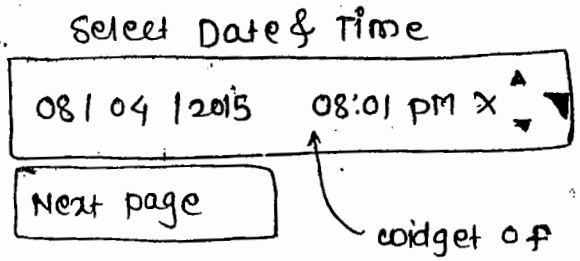


```
<body>
<form action = "nit.html" name = "myform" id = "form1">
<label style = 'color: blue'> select valid Date & Time....!
</label> <br>
<input type= 'datetime' name = "dt" ><br>
<input type = 'submit' value = "Next page">
</form>
</body>
```

3] dateHime local : This input type allows the user to select date and time without time zone.

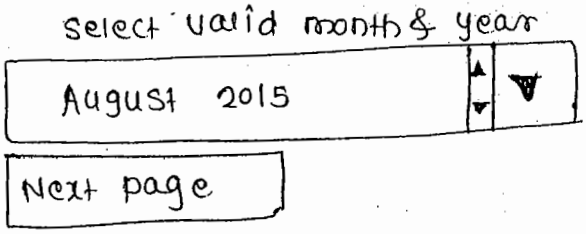
Syntax : Input type = name.

~~<body>~~
~~<form action = 'nit.html'~~




```
<body>
<form action = 'nit.html' name = "myform" id = "form1" >
<label style = 'color : blue' > select valid date time local !
</label> <br>
<input type = 'datetime-local' name = "dt1" ><br>
<input type = 'submit' value = "Next page" >
</form>
</body>
```

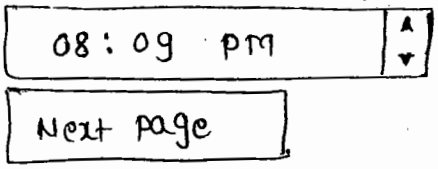
4] month : This input type allows the user to select month and year. Syntax : input type = name




```
<body>
<form action = "nit.html" name = "myform" id = "form1">
 <label style = 'color: blue'> select valid month & year ... !
 <label> <br1>
 <input type = 'month' name = "my"> <br1>
 <input type = 'submit' value = "Next page">
 '</form>
  </body>
```

5] time : This input type allows the user to select time with hours and minutes

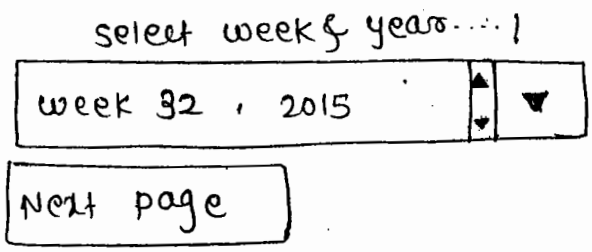
Syntax: input type=name




```
<body>
<form action='nit.html' name = "myform" id = "form1">
<label style = 'color: blue'> select valid time .... ! </label> <br1>
 <input type = 'time' name = 'tm'> <br1>
 <input type = 'submit' value = "Next page">
    </form>
    </body>
```

week : This input type allows user to select week & year

Syntax: input type = name




```
<body>
 <form action = "nit.html"  name = "myform"  id = "form1" >
  <label style = 'color: blue' > select week & year </label>
     <br/>
  <input type = 'week' 'week'
```

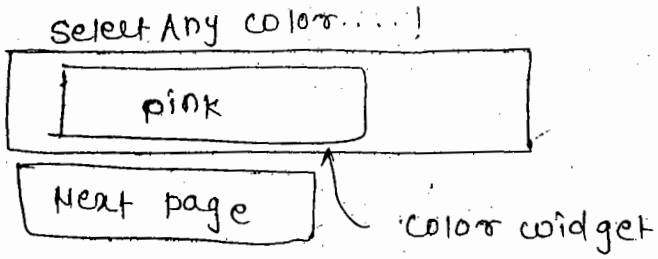
General form controls

1] color
2] email
3] number
4] range
5] search
6] tel
7] url

1] color: This i/p type allows the user to display color picker

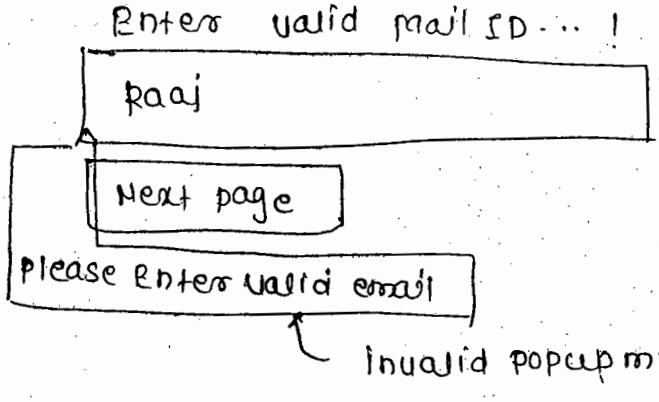
syntax: input type = name

```
<body>
<form  action = "nit.html"  name = "myform"  id = "form1" >
<label style = 'color: blue' > select any color.....! </label>
 <br/>
<input type = 'color' name = "clr" > <br/>
 <input type = 'submit' value = "next page" >
</form>
</body>
```



2] email : This i/p type allows user to enter valid email address

Syntax : input type = name



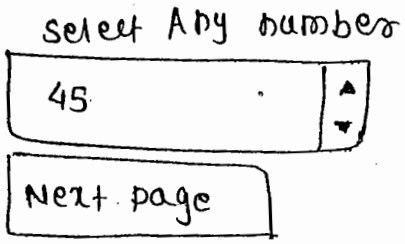

```
<body>
<form action="nit.html" name="myform" id="form1">
<label Color style='color:blue'> Enter valid Email ID </label>
<br>
<input type='email' name="e1"><br>
<input type='submit' value="Next page">
</form>
</body>
```

number : This input type allows the to display numerical spinners

Syntax: input type = name

Attributes:

- max: specifies the maximum value allowd
- min: specified the minimum value allowed
- Step: specifies the legal number intervals
- value: specifies the default value.




```
<body>
 <form action = "nit.htm"  name = "myform"  id = "form1" >
<label > Select Any number </label> <br>
 <input type = 'number' name = 'nn' value = "0" min=5 max=
"104" step = "5" > <br>
  <input type = 'submit' value = "Next page">
    </form>
     </body>
```

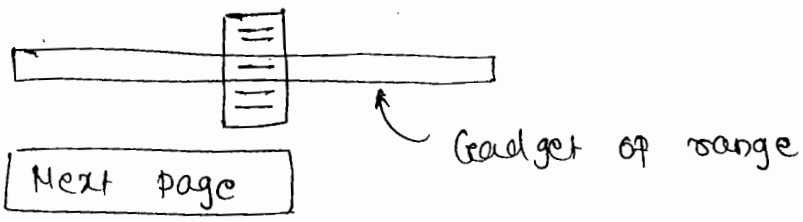
3] range : This i/p type allows the user to display slider.

Syntax: input type = name

Attribute

- max
- min
- Step
- value

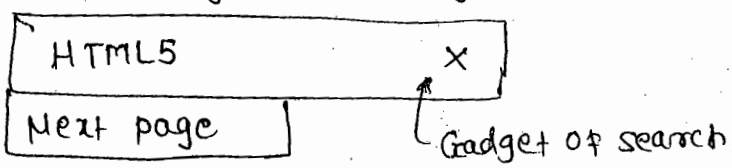
find the slider...)



```
<Body>
  <form action= "nit.html" name= "myform" id="form1">
<label style='color:blue'> Select find the slider </label><br>
  <input type= "range" name="rg" value="0" min=5
      max="104" step="5"> <br/>
  <input type= "submit" value= "Next page" >
  </form>
  </body
```

5] search : This i/p allows user IN HTML5 we can define textbox as search box instead of a normal textbox Notice there is a blue "cross" sign appears in the textbox when you input something in the search box when you click on the "cross" yours input string will be clear and you can start to type a new string.

syntax : input type= name

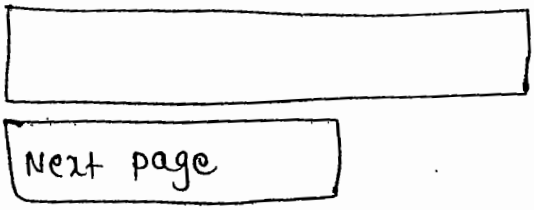



```
<body>
  <form action= "nit.html" name= "myform" id= "form1">
  <label> Enter any search keyword </label><br/>
  <input type= 'search' value name='key"><br/>
  <input type= "submit" value= "next page" > </br>
  </form>
  </body>
```

6] Tel : This i/p type accept only phone numbers it does not confirm any specific pattern

Syntax: input type= name

Enter Any Mobile Number



Next page

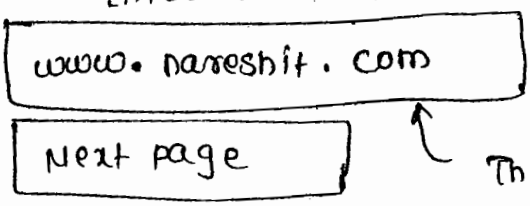
```
<body>
 <form action="nit.html"  name="myform"  id="form1">
 <label> Enter Any Mobile number </label><br>
 <input type="tel" name="tno" pattern = "[0-9]{10}"> <br>
 <input type="submit"  value="Next page">
     </form>
      </body>
```

attribute

Url: It is used to validate url address. It is very useful for mobile applications

Syntax: input tye= name

Enter valid URL



The widget of URL

```
<body>
  <form action="nit.html"  name="myform"  id="form1">
 <label> Enter valid URL </label><br>
 <input type='url' name="ur"> <br>
  <input type = "submit"  value="next page">
    </form>
    </body>
```

HTML5 New form attributes

HTML5 supports the following list of new form attributes

1) autocomplete
2) novalidate

1) **autocomplete** : It specifies whether a form or input field should have autocomplete on or off. when autocomplete is on the browser automatically complete values based on values that the user has entered before.

Syntax :

```
<form autocomplete="on|off">
```

NOTE : Default autocomplete is on.

```
<body>
<form action="nit.html" autocomplete="off">
First Name: <br/>
<input type="text name="fname" /> <br/>
email : <br/>
<input type="text" name value="Login" />
</form>
</body>
```

2) **Novalidate** : It is a boolean attribute. when present it specifies that the form data (input) should not be validated when submitted.

Syntax : `<form novalidate = "novalidate">`

```
<body>
<form action = "nit.html" novalidate = "novalidate">
First name: <br/>
<input type = "text" name="uname" /><br/>
email : <br/>
<input type = "text" value = login required = "required">
<br/>
<input type = "submit" value = 'validate'>
</form>
</body>
```

HTML5 Input attributes

In HTML5 input contains the following list of new attributes:

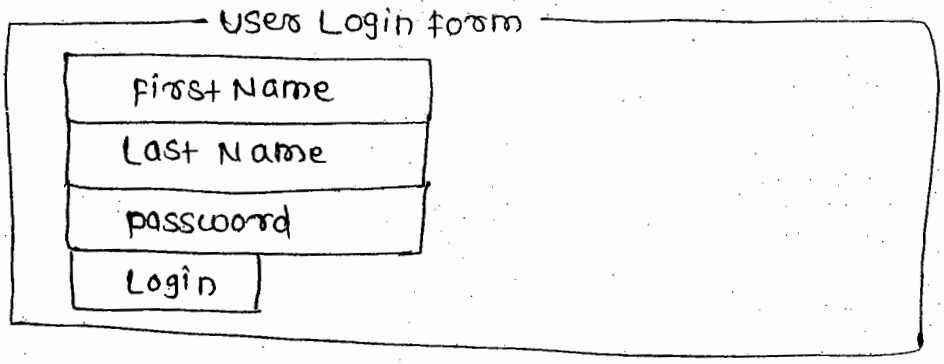
1] placeholder (Text fields with temporary entry)
2] autofocus
3] required (Text field with required values [non ~~value~~empty])
4] autocomplete
5] form
6] formaction
7] formenctype
8] formmethod
9] formnovalidate
10] formtarget
11] height and width
12] list
13] min and max
14] multiple
15] pattern
16] step
17] Spellcheck
18] contenteditable
19] accesskey

1] placeholder Attribute :

It specifies a short hint that ~~values of an~~ describes the expected value of an input field. The hint is displayed in the input field when it is empty and disappears when the field gets focus.

Syntax: <input placeholder = "text/hint">

Attribute Value	values Description
text	specifies a short hint that describes the expected value of the input field.




```
<body>
 < form action = "nit.html" >
  < fieldset>
   <legend align =" center"> User Login form .... ! </legend>
   <input type = "text" name= "fname" placeholder = "first Name">
       <br/>
  <input type = "text" name = "lname" placeholder = " Last Name" >
    <br/>
  <input type = "password" name = "pwd" placeholder = "password">
    <br>
   <input type= "submit" value = "Login">
  </fieldset>
  </form>
  </body>
```

2) Autofocus : It is a boolean attribute. when present it specifies that an <input> element should automatically get focus when the page loads.

Syntax: <input autofocus = "autofocus" />

or

<input autofocus>

<input autofocus = " ">

Note: autofocus attribute works on any input level

example :

```
<body>
  <form action= "nit.html">
    first name: <br/>
   <input type = "text" name = "fname" autofocus = "autofocus">
<br/>
    Last name: <br/>
   <input type = "text" name = "lname"><br>
  <input type = "submit" value = "Next page" >
 </form>
  </body>
```

3] Required attribute :

It is a boolean attribute when present it specifies that an input field must be filled out before submitting the form.

Syntax : <input required="required">

or

<input required>

<input required="">

```
<body>
 <form action="hit.html">
  <label> what is your favourite movie </label>
  <input name="movie" type="text" required="required">
  <br>
 <input type="submit" value="next page">
  </form>
  </body>
```

4] Autocomplete : This attribute specifies whether or not an input field should have autocomplete enabled. Autocomplete allows the browser to predict the value.

Syntax: <input autocomplete="on|off">

Attribute Value	Values
On	Default Specifies that autocomplete is on.
Off	Specifies that autocomplete is off.

Ex:

```
<body>
 <form action = hit.html autocomplete="on">
  First name: <br>
 <input type="text" name="fname"> <br>
  Email : <br>
 <input type = "email" name="email" autocomplete="off">
  <br>
  <input type="submit" value="Login">
 </form>
 </body>
```

6] form : It specifies one or more forms an <input> element belongs to

Syntax: <input form = "id1 name"/>

```
<body>
 <form action="bit.html" id="form1">
 First Name: <br>
  <input type="text" name="fname"> <br>
  <input type="submit" value="submit">
  </form>
  Last name: <br>
   <input type="text" name="lname" form="form1">
    </body>
```

6] formaction : It specifies the URL of a file that will process the input control when the form is submitted. The formaction attribute overrides the action attribute of the <form> element.

Note :

This attribute is used with the following input types

1] "submit"

2] "image"

```
<body>
 <form action="http://www.nareshit.com">
<input type="text" placeholder="uname">
   <br>
<input type="password" placeholder="password"> <br>
 <input type="submit" value="FACTION">
 <input type="submit" value="IACTION" formaction="html5.png">
    </form>
    </body>
```

7] formencytype : It specifies how the form data should be encoded when submitting it to the server. The formenctype attribute overrides the enctype attribute of the <form> element.

Syntax : <input type formenctype = name>

Note : It is used with type = "submit" and type = "image">

```
<body>
  <form action=" nit.html" method="get">
First Name: <br/>
<input type="text" name="fname"><br/>
    password: <br/>
<input type="password" name="pwd"><br/>
<input type="submit" value="G@form">
 <input type="submit" formenctype="form-data"
          value="p@input">
      </form>
      </body>
```

8] formmethod : It specifies defines the http method for sending form data to the action url. This attribute overrides the method attribute of the <form> element.

Note: It can be used with the type="submit" and type="image"

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

ex:
```
<body>
    <form action=" nit.html" method="get">
    First name: <br>
    <input type="text" name="fname"><br/>
    password: <br/>
    <input type="password" name="pwd"><br/>
  <input type="submit" value="@form">
  <input type="submit" formmethod="post" value="@input">
    </form>
    </body>
```

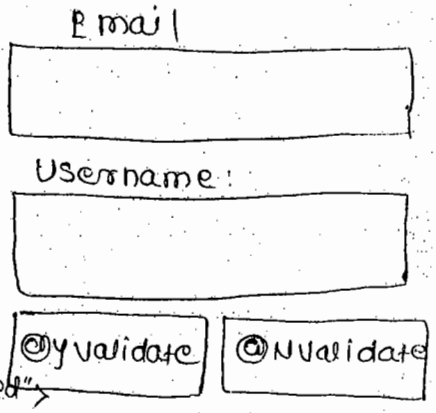
9] formnovalidate : It is a boolean attribute. when present it specifies that the <input> element should not be validated when submitted.

syntax: <input type formnovalidate='on|off'>

Note: It can be used with type="submit".

example : <body>

```
<form action="nit.html" method="get">
 user name : <br/>
<input type="text" name="uname">
 <br/> required="required">
  Email :
<input type="email" name="email">
 novalidate="novalidate" required="required">
  <br/>
<input type="submit" value="@yvalidate">
<input type="submit" value="@Nvalidate" form novalidate=
<form novalidate">
 </form>
 </body>
```



10] formtarget : It specifies a name or keyword that indicates where to display the response that is received after submitting the form.

syntax : <input type formtarget ='target'>

Note : It can be used with type="submit" and type="image">

example : <body>

```
<form="nit.html">
 first name: <br>
<input type="text" name="uname" > <br>
Last Name password: <br>
<input type="text" name="lname"> <br>
<input type="submit" value="@sameTW" >
 <input type="submit" value="@NewTW" formtarget="_blank">
</form>
 </body>
```

11] Height and width attributes : It specify the height and width of the <input> element.

Note : attributes are only used with <input type="image">

syntax : <input type='image' height="10px" width="10px">

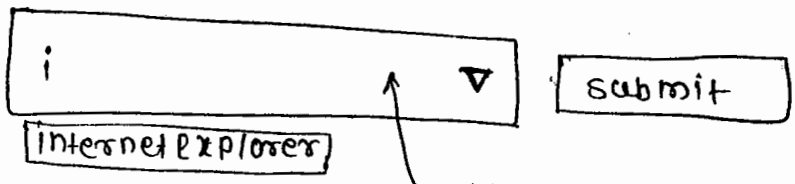
ex: <body>

```
<form = "nit.html">
 first name: <br>
<input type = "text"  name = "fname"> <br>
 password : <br>
 <input type = "password" name = "pwd"> <br>
<input type = 'image' src = "html5.png" width = "25px"
    height = "15px">
</form>
 </body>
```

[2] List attribute : It refers to a <datalist> element that contains predefined options for an <input> element.

Syntax:

```
<input list = "Name">
```




```
<body>
<form = "nit.html" method = "get">
<input type = "list"
 <input list = "browsers" name = "browser">
  <datalist id = "browser">
  <option value = "Internet Explorer">
   <option value = "firefox>
  <option value = "crome">
  <option value = "opera">
  </datalist>
  <input type = "submit">
  </form>
  </body>
```

13] multiple Attribute: It is a boolean attribute. when present it specifies that the user is allowed to enter more than one value in the <input> element.

Note: It supports email and file input types.

Syntax: <input type='file' multiple='on|off'>

ex:
```
<body>
<form action="bit.html">
 select files():
<input type="file" name="img" multiple="on"><br>
 <input type="submit" value="login">
 </form>
 </body>
```

14] pattern attribute: It specifies a regular expression that the <input> elements value is checked against.

Syntax:

```
<input type pattern = regExp>
```

```
<body>
<form action="bit.html">
<input type="text" name="ccode" pattern="[A-z a-z]{2}"
title="Two letter Country code in Text format" placeholder=
"Country code" required>
<input type="submit" value="login">
</form>
</body>
```

15] Spell check attribute: It specifies whether the element is to have its spelling and grammer checked or not.

Syntax:

```
<element spellcheck = "true|false">
```

Attributes	value
true	The element is to have its spelling and grammer checked.
false	The element is not to be checked (default)

```
<body>
<input type="text" spellcheck="true"><br>
<p> <textarea spellcheck="true"> Text area element
</textarea> </p>
</body>
```

14] contenteditable attribute : using this attribute we can modify the content directly on the web page.

Syntax:

```
<element contenteditable = "true|false">
```

Attribute Values	
Value	
true	The element is able to modify.
false	The element content is unable to modify (default).

```
<body>
<p contenteditable = "true" spellcheck = "true"> This is a
paragraph. It is editable </p>
<p contenteditable = "false" spellcheck = "false"> This is a
paragraph. It is not editable <p>
</body>
```

15] accesskey attribute : Accesskeys allows easier navigation by assigning a keyboard shortcut to a link (which will usually gain focus when user presses 'Alt' or 'Ctrl' + the accesskey).

Syntax: `<element accesskey = "character">`

Attribute values

Value	
Character	specifie the shortcut key to activate/ focus the element.

ex:
```
<body>
<a href = "http://www.nareshit.com" accesskey = "n">
NareshIT </a>
<a href = "http://www.nareshit.in" accesskey = "s">
Nit </a>
</body>
```

* HTML5 New form elements

HTML5 support the following list of new elements:

1) <datalist>
2) <keygen>
3) <output>

1) <datalist> : It is used to list a set of data to be used in a list of options like autocomplete feature used in forms. It enables you to provide a list of predefined options to the user as they input data. It is a paired tag.

Syntax: <datalist> ..... </datalist>

Attributes : data ⟹ specifies a xml file that can be used to prefill the data list.

```
<body>
<label> Enter your favourite color <br/>
<input type = "text" name = "favcolor" list = "colors">
<datalist id = "colors">
<option value = "Green">
<option value = "blue">
<option value = "marron">
</label>
</body>
```

2) keygen : Generate keys to authenticate users. The purpose of the <keygen> element is to provide a secure way to authenticate user.

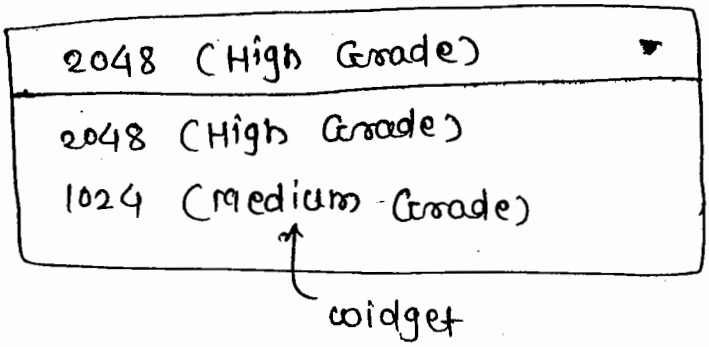
The keygen tag is used to key generator for a form. The private key is stored browser and the public key is sent to the server when the form is click or submitted. It is a non paired tag.

Syntax: <keygen>

Attributes:

Attribute	value	Description
autofocus	autofocus	<keygen> element should automatically get focus when the page loads.
challenge	challenge	<keygen> should be challenged when page is submitted.
disabled	disabled	specifies that a <keygen> element should be disabled.
name	name	defines a name for the <keygen> element.

ex1: with challenge attribute



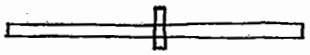

```
<body>
  <form action="hit.html">
   Username: <br/>
  <input type = "text" name="uname" id = "user1" /> <br/>
  Encryption key: <br/>
  <keygen name = "encrypt" challenge= "challenge">
    <br>
   <input type = "submit" value= "Generate" />
     </form>
     </body>
```

output : It is used to display the result of arithmetical calculation, it is a paired tag.

Syntax: <output> ..... </output>

Attributes :

Attribute	value	Description
for	element_id	specify the relation between the result of the calculation
form	form_id	specify one or more forms the output element belongs to
name	name	specifies a name for the output element.

Range  Number + [50] = 105

```
<body>
 <form oninput=" x.value = parseInt(a.value) + parseInt(b.value)"
  >Range
  <input type="range" name="a" value="50" />Number
  + <input type ="number" name="b" value="50" /> =
  <output name="x" for = "a b" > </output>
   </form>
   </body>
```

parseInt global function: The parseInt() function parses the string and returns an integer.

Syntax: parseInt(string)

Parameter	Description
String	Required. The string to be parsed.

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

ex:
```
<head>
<script>
  var x= prompt("Enter Any number");
  var y= prompt("Enter Any number");
 var z= parseInt(x)+ parseInt(y);
        document.write(z);
 </form>
 </body>
```

Working with HTML5 Multimedia

Multimedia comes in many different formats in web environment modern webpages having pictures, music, sound, videos, records films, Animations etc.

Multimedia video formats

The following formats frequently we are using in the web environment.

AUI (.aui) → AUI (Audio video Interleave) was developed by microsoft.

WMV (.wmv) → WMV (Windows media video) was developed by microsoft

MPEG (.mpg or .mpeg) → The MPEG (moving picture expert group)

Flash (.swf or .flv) → It was developed by micromedia

MP4 (.mp4 MPeg-4 (MP4) is the new format for the internet)

Multimedia audio format

The following audio formats commonly used in the web environment

MIDI (.mid or .midi) → MIDI (Musical instrument Digital interface)

MP3 (.mp3) → MP3 files are actually the sound part of the MPEG file.

WAV (.wav) → WAVE (more known as wAV) was developed by microsoft and IBM

WMA (.wma) → WMA (windows media Audio)

HTML5 audio Tag :

It is used to specify audio on an HTML document. It enables audio playback without plugins, The audio can be controlled with native or customized controls, Audio files can be prefetched or downloaded on demand. It is a paired tag.

Syntax : <Audio> ..... </audio>

Note:
Any text between <audio> and </audio> will be displayed in browsers that do not support audio.

Attributes:

Attribute	value	Description
autoplay	autoplay	specifies that the audio will start playing
controls	controls	specifies that the audio controls should be displayed.
loop	loop	specifies that the audio will start over again.

src path of audio specifies the URL of the audio file

examples:

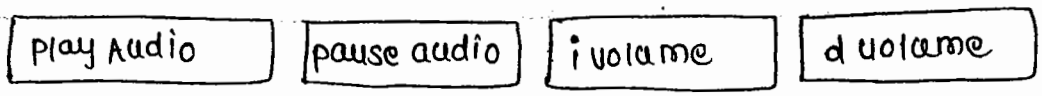
```
<body>
  <audio src = "song.mp3" controls = "controls" loop = "3" autoplay = "autoplay">
  <p> oops your Browser not supporting Audio feature
      update and try </p>
  </audio>
</body>
```

muted = "muted"

ex2:

```
<body>
<audio src = "http://sound26.mp3pk.com/indian/raone/
raone/(www.songs.pk).mp3" controls = "controls"
  autoplay = "autoplay">
<p style='color:red'> oops your Browser not supporting</p>
  </audio>
</body>
```

HTML5 audio Tag with JS controls




```
<body>
<audio <h2 style='color:blue'> HTML 5 audio Tag with JS controls
  </h2>
 <audio  src = "http://sound26.mp3pk.com  id = "myAud">
 </audio>

<div>
     <button  onclick = "document.getElementById ('myAud').play()">
   Play Audio </button>
   <button  onclick = "document.getElementById ('myAud').pause()">
    Pause audio </button>

 <button onclick = "document.getElementById ('myAud').volume += 0.1">
    i volume </button>
```

```
<button onclick="document.getElementById('myAud').volume-=
    0.1"> dvolume </button>
</div>
</body>
```

HTML5 video Tag

It is used to specify video on an HTML document. It is a paired tag.

Syntax: `<video> ..... </video>`

Note: Any text between the <video> and </video> tags will be displayed in browsers that do not support video.

Attribute	value	Description
autoplay	autoplay	specifies that the video will start playing.
controls	controls	specifies that the video controls should be displayed.
src	URL	specifies the URL of the video file
width	pixels	sets the width of the video file.
height	pixel	
loop	loop	Specifies that the video will start again
muted	muted	specifies the audio output of the video should be muted.
poster	URL	specifies an image to be shown while the video is downloading or until the user hits the play button.
preload	auto metadata none	specifies if and how author thinks the video should be loaded when the page loads.

```
<body>
<video src="oceans.mp4" controls="controls" autoplay="autoplay"
    loop="5" width="500px" poster="html5.png" preload="auto">
<p style='color:red'> oops your Browser not supporting update
        and try </p>
</video>
</body>
```

Example: video tag with java script

```
<body>
<video src="oceans.mp4" id="MYVID">
</video>
<div>
<button onclick="document.getElementBYID("MYVID".play())">
Play video </button>
<button onclick="document.getElementBYID("MYVID).pause()">
    Pause video </button>
<button onclick="document.getElementBYID("MYID).Volume+=
0.1">
Incr Ivolume </button>
<button onclick="document.getElementBYID("MYVID).volume-
    =0.1">
    Dvolume </button>
</div>
</body>
```

HTML5 Track Element

It specifies text tracks for media elements. This element is used to specify subtitles, caption files or other files containing text that should be visible when the media is playing. It is a non paired tag.

syntax : <track>

Note : It is not supported by Any of the major Browsers.

WebVTT : It is a format for displaying timed text tracks with the <track> element. The primary purpose of webVTT files is to add subtitles to a <video> web VTT is a text based format. A webVTT file must be encoded in UTF-8 format.

* How To create web VTT file?

1] Launch Any text editor

2] Enter the following tracks

1] 00:00:00.500 → 00:00:02.000
The web is always changing

2] 00:00:02.500 → 00:00:04.300
the way we access it is changing

3] save with .vtt extension @ any location.

Example : Track

```
<body>
 <video src="devstories.webm" width="300" height="150"
      controls>
 <track src="voice.vtt" kind="subtitles" srclang="en"
      label="English">
<p> oops Browser does not support the HTML5 video element </p>
  </video>
  </body>
```

xiph.org

HTML5 Graphics

<canvas> Tag

It is used for creating graphics on the fly. It can be used for rendering graphs, game graphics and other visual images.

<canvas> required java script getElementById method to design graphics. It has several methods and the following list of feature.

1] Graphs and charts
2] Animations
3] Games
4] Diagrams
5] videos and photo gallaries
6] special image effects
7] Drawing applications.
8] User interface enhancement

It is a paired tag

Syntax :

<canvas> ..... </canvas>

Note : Always specify an id attribute and a width and height attribute to define the size of the canvas.

Note : Any text inside the <canvas> element will be displayed in browsers that does not support <Canvas>

Attributes

Attribute	Value	Description
height	pixels	specifies the height of the canvas
width	pixels	specifies the width of the canvas.

Global Attributes

Attribute	Value	Description
id	ID_Name	Declared unique id for the element.
class	predefined_class	used in cascading style sheet
style	style_attributes	CSS code specify inline the HTML tag is presented.

Event attributes

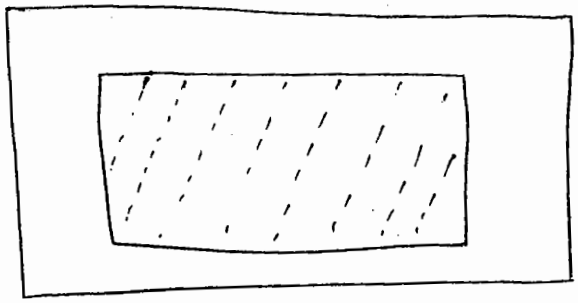
Attributes	value	Description
onfocus	"java-script"	element gets focus on object when script to be run.
onblur	"java_script"	element lose the focus on object when script to run.
onabort	"java-script"	element gets aborted on object when script to run.

Canvas shapes: It supports the following list of shapes.

1] create a canvas
2] Draw onto the canvas with Javascript
3] Canvas Coordinates
4] Canvas - paths (lines, circles)
5] Canvas - Text
6] Canvas - Gradients

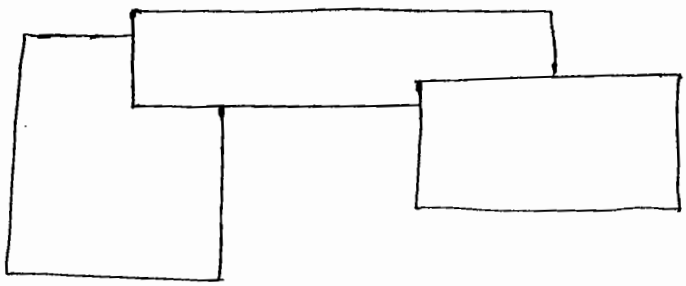
1] How to create a canvas?

```
<!DOCTYPE html>
<html lang="en-IN">
 <head>
 <meta Charset = 'Utf-8'>
 <title> canvas Element </title>
 </head>
  <body>
 <canvas width= "200 px" height ="100px" style ='border:2px
    solid #FF00FF'> id ='mycan' >
 <p style='color:Red'> oops your browser unable to support
       canvas Element </p>
    </canvas>
    </body>
    </html>
```



```
<!DOCTYPE html>
<html lang='en-IN'>
 <head>
 <meta charset='UTF-8'>
 <title> canvas element </title>
 </head>
  <body>
 <canvas width="200px" height = "100px" style='border: 2px solid
   #FF00FF' id = "mycan">
 </canvas>
   <'script type="text/javascript">
   var c = document.getElementById("mycan");
   var ctx = c.getContext("2d");
   ctx.fillStyle="#FFFF00";
   ctx.fillRect(25,10,150,80);
    </script>
    </body>
    </html>
```

canvas with multiple Rectangles



```
<body>
<DOCTYPE html>
<html lang='en-IN'>
<head>
<meta charset='utf-8'>
<title> canvas Element </title>
</head>
<body>
<canvas id =" canvasEx" > </canvas>
<script type="text/javascript">
'var canvas = document.getElementById('canvasEx');
var ctx= canvas.getContext('2d');
ctx.fillStyle ='#FFCC00';
ctx.fillRect(0,5,80,110);
ctx.fillStyle='#FF9900';
ctx.fillRect(25,0,120,50);
ctx.fillStyle="#FF00CC';
ctx.fillRect (100,35,160,60);
</script>
</body>
```

Canvas paths : we can design different paths on a canvas with the help of following methods.

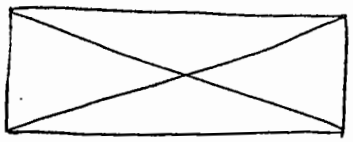
1] The beginPath() defines a new drawing path.

2] To moveTo()- method creates a new subpath for the given point.

3] The lineTO() method draws a line from the context point to the given point.

4] The stroke() method assigns a color to the line and make it visible. Unless otherwise specified the default stroke color is black.

```
<body>
<canvas>
<p style='color:red'> oops your Browser not supporting
                      canvas feature update and try </p>
<canvas width="200px" height="100px" style='border:2px
    solid #FF00FF' id="mycan">
</canvas>
<script type="text/javascript">
var c= document.getElementById("mycan");
var ctx= c.getContext("2d");
  ctx.moveTO(0,0);
  ctx.lineTO(200,100);
 ctx.moveTO(200,0);
  ctx.lineTO(0,100);
  ctx.stroke();
</script>
</body>
```

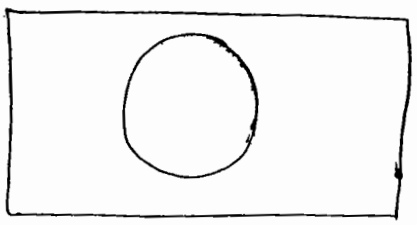


Canvas Circles

We can able to design circle on a canvas with the help of following syntax:

```
arc(x,y,r,start,stop)
```

Here, x represents width y represents height r represents radius Start represent starting position stop represent ending position.



```
<body>
 <canvas  style='color: red'  width= 200px  height= 200px
          id = "canvas">
<p> oops your Browser unable to support canvas feature </p>
 </canvas>
  <script type= 'text/javascript'>
   var c = document. getElement ById ("canvas");
    var ctx = c. getContext ("2d");
    ctx. beginPath();
    ctx. arc (95, 40, 0, 2 * Math. PI);
      ctx. stroke();
     </script>
    </body>
```

ex:

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

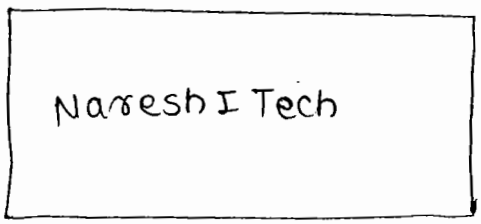
→ 3.141

Canvas Text : To draw text on a canvas the following properties and methods frequently we are using

1] font - defines the font properties for text

2] FillText (text, x, y) - Draws "filled" text on the canvas.

3] StrokeText (text, x, y) - Draws text on the canvas.



```
<body>
<canvas id="mycanvas" width="200px" height="100px"
        style="border:2px solid #FF0000;">
</canvas>
<script type="text/javascript">
Var c = document.getElementById("mycanvas");
Var ctx = c.getContext("2d");
ctx.font = "30px tahoma";
ctx.fillText("NareshiTech", 20, 55);
</script>
</body>
```

★ Canvas Gradients

Gradients can be used to fill rectangles, circles, lines and text etc. Shapes on the canvas not limited to solid colors. There are two different types of gradients.

1] Linear Gradient
2] Radial Gradient

1] Linear Gradient : It is used to create linear color shades from left to right

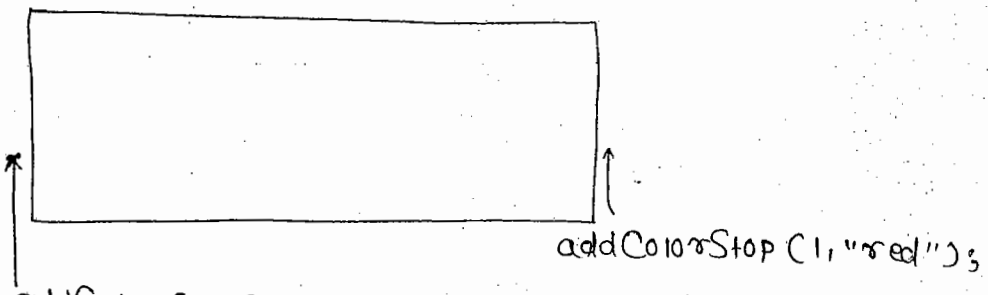
Syntax: createLinearGradient(x,y,x1,y1)

x,y represents starting width and height

x1,y1 represents ending width and height

addcolorstop() method

It specifies the color stops and its position along the gradient. Gradient positions can be anywhere between 0 to 1. To use the gradient set the fillstyle or ~~stroke~~ strokestyle property to the gradient.



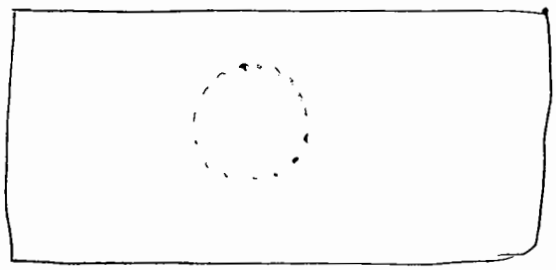
example :

```
<body>
<canvas id = "mycanvas">
</canvas>
<script type = "text/javascript">
var c = document.getElementById ("mycanvas");
var ctx = c.getContext ("2d");
// create gradient
var grd = ctx.createLinearGradient (0,0,250,0);
grd.addColorStop(0, "green");
grd.addcolorstop(1, "red");
// fill with gradient
ctx.fillStyle = grd;
ctx.fillRect(10,10,200,80);
</script>
</body>
```

Radial Gradients :

These gradients are popularly known as circular gradients it has the following detailed syntax :

createradiaGradient (x,y,r,x1,y1,r1);



```
<body>
<canvas - id = "mycanvas")
</canvas>
<script type="text/javascript">
var c = document.getElementById("mycanvas");
var ctx = c.getContext("2d");
var grad = ctx.createRadialGradient(110,50,5,90,60,100);
grad.addColorStop(0, "white");
grad.addColorStop(1, "blue");
ctx.fillStyle = grad;
ctx.fillRect(10,10,200,80);
</script>
</body>
```

DuckDuckGo

Inkscape.org

HTML5 SVG : (Scalable Vector Graphics).

It is a graphics format in which the shapes are specified in XML. SVG is a language for describing two-dimensional vector graphics in XML.

1] JPEG (Joint Photographic expert group).
2] PNG (Portable Network Graphics)
3] GIF (Graphic interchange format).

SVG versions

SVG 1.0 → 2001
SVG 1.1 → 2003
SVG 1.1 (second edition) → 2011

SVG features :

1] W3C standard
2] Image scaling
3] Smaller file size
4] SVG tools are already available
5] SVG images can be searched, indexed, scripted, and compressed.
6] SVG images can be printed with high quality at any resolution

7] compatibility
8] Accessibility
9] Search Engine optimization.

SVG Syntax :

```
<SVG xmlns="http://www.w3.org/2000/svg">
    ----
    ----
    ----
    ----
</SVG>
```

Namespace

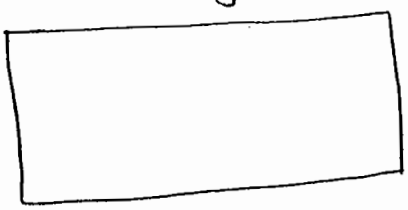
SVG Shapes : SVG supports the following list of shapes.

Shape	keyword
1] Rectangle	<rect>
2] circle	<circle>
3] Ellipse	<Ellipse>
4] Line	<line>
5] polyline	<polyline>

ex1 : Create SVG Rectangle

```
<!doctype html>
<html lang="en-IN">
 <head>
  <title>
    My SVG Graphics
  </title>
<meta charset="utf-8">
  </head>
  <body>
<h2> SVG Rectangle ..... </h2>
<SVG xmlns="http://www.w3c.org/2000/svg">
<rect width="200px" height="100px" fill="lightblue"
      Stroke="green">
 </svg>
</body>
</html>
```

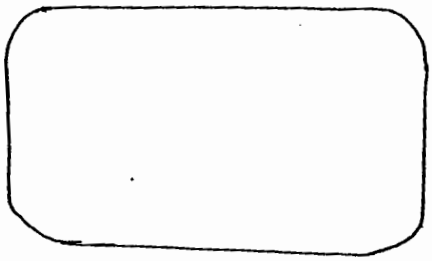
SVG Rectangle



ex: with style attribute

```
<body>
 <h2 style='color: Red'> SVG Rectangle </h2>
 <svg xmlns = "http://www.w3c.org/2000/svg">
  <rect width = "200px" height = "100px" style = "fill : rgb(0,255,0);
   stroke-width -1; stroke : rgb(255,0,0) "> x=20px, y=20px
   </svg>
   </body>
```

SVG Rectangle with Radius



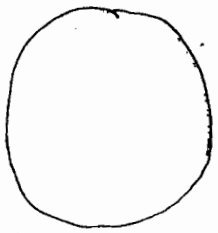
```
<body>
<h2 style='color : Red'> SVG Rectangle </h2>
<SVG xmlns = "http://www.w3c.org/2000/svg">
<rect x="50" y="20" rx="20" ry="20" width="200" height="100"
 style = "fill : pink ; stroke : green ; stroke : width : 5" />
  </SVG>
  </body>
```

SVG Circles

With the help of circle keyword we can able to design circles on the webpage.

```
  <body>
   <h2> SVG Rect circles </h2>
<SVG xmlns = "http://www.w3c.org/2000/svg">
<circle cx = "100px" cy = "60px" r="50px"
    style = "fill. pink ; stroke : green ; stroke -width : 2px" >
  </SVG>
   </body>
```

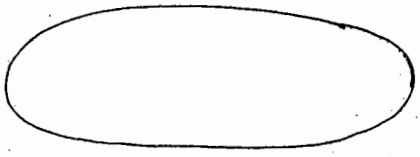
SVG circle



→ SVG Ellipse

This element is used to create an elipse. It is closely related to the circle.

SVG Ellipse

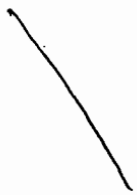


```
<body>
 <h2> SVG Ellipse </h2>
<svg xmlns="http://www.w3c.org/2000/svg">
<ellipse cx="150px" cy="60px" rx="100px" ry="50px"
  style="fill: yellow; stroke: green; stroke-width: 2px" />
</svg>
</body>
```

* Line

This element is used to create line.

SVG Line



```
<body>
 <h2> SVG line </h2>
 <svg>
  <line x1="0" y1="0" x2="100" y2="100" style="stroke:
   rgb(255,0,0); stroke:width: 2" />
 </svg>
  </body>
```

5] polyline

Difference between canvas & SVG.

canvas

1] Resolution dependent
2] support for event handlers
3] poor text rendering capabilities
4] you can save the resulting image as :Png or .jpg
5] suited for game application.

SVG :

1] Resolution Independent
2] No support for event handlers.
3] Best suited for applications with large rendering areas (Googlemaps).
4] slow rendering if complex.
5] Not suited for Game application.

HTML5 MathML (mathematical markup language).

MathML is an application of XML for describing mathematical notations and capturing structure in the content.

MathML versions

1] MathML 1.0 [1998]
2] MathML 2.0 [2003]
3] MathML 3.0 [2010]

MathML Syntax :

```
<math xmlns="http://www.w3.org/1998/math/MathML">
  ---
  ---
  ---
</math>
```

★ Features of MathML

1] Linking
2] Directionality
3] Line breaking
4] Including images
5] Elementary math layouts
6] MathML is part of HTML5.

MathML Tags

→ MathML mainly focus displaying of mathematical equations, every math elements starts with m.

1] <mi>x</mi> — identifiers
2] <o> + </mo> — operators
3] <mn> 2 </mn> — numbers
4] <mtext> non zero </mtext> — text
5] <mrow> — a horizontal row of item.
6] <mfrac> — fractions.

1] msup :

The MathML <msup> element is used to attach a superscript to an expression.

Syntax : <msup> base superscript </msup>

Ex:

```
<body>
<math xmlns = "http://www.w3c.org/1998/math/mathML">
<msup>
<mi> x </mi>
<mn> 2 </mn>
</msup>
</math>
</body>
```

2] Msub :

The mathML <msub> element is used to attach a subscript to an expression.

Syntax : <msub> ..... </msub>

example :

```
<body>
  <math xmlns = "http://www.w3.org/1998/Math/MathML">
    <msub>
      <mi> x </mi>
      <mn> 2 </mn>
    </msub>
  </math>
</body>
```

3] msubsup :

This element is used to display a subscript and superscript together.

Syntax : <msubsup> ..... </msubsup>

Ex :

```
<body>
  <math xmlns = "http://www.w3.org/1998/Math/MathML">
    <msubsup>
      <mo> &#x2222B; <!-- Integral --> </mo>
      <mn> 0 </mn>
      <mn> 1 </mn>
    </msubsup>
  </math>
</body>
```

4] mroot : This element is used to display roots.

Syntax: <mroot> ..... </mroot>

```
<body>
 <math>
 <mroot>                    ∛x
 <mi> x </mi>
 <mn> 3 </mn>
 </mroot>
 </math>
 </body>
```

5] msqrt : This element is used to display square roots.

Syntax: <msqrt> ..... </msqrt>

Ex:

```
<body>
 <math>
 <msqrt>
 <mi> x </mi>                √x
 </msqrt>
 </math>
 </body>
```

6] <menclose : This element renders its content inside an enclosing notation specified by the notation attribute.

Syntax: <menclose> ..... </menclose>

notation attribute supports the following list of values

1] box
2] roundedbox
3] circle
4] left
5] right
6] top
7] bottom
8] updiagonalstrike
9] downdiagonal strike
10] verticalstrike
11] horizontalstrike etc.

```
<body>
 <math>
 <menclose notation="box">
   <mi> x </mi>
    <mo> + </mo>
   <mi> y </mi>
 </menclose>
 </math>
 </body>
```

mfrac: This element is used to display fractions. syntax:

```
<mfrac> numerator denominator </mfrac>
```

$$\frac{a}{b} / \frac{c}{d}$$

bevelled = "true"

↑ attribute

```
<body>
 <math>
 <mfrac bevelled ="true">
   <mfrac>
   <mi> a </mi>
   <mi> b </mi>
 </mfrac>
   <mfrac>
  <mi> c </mi>
   <mi> d </mi>
   </mfrac>
 </mfrac>
 </math>
 </body>
```

```
<body>
 <math>
<mfrac buelled ="true">
<mfrac>
<mi> a </mi>
<mo> + </mo>
 <mi>b </mi>
</mfrac>
<mfrac>
  <mi> c </mi>
  <mo> + </mo>
   <mi> d </mi>
   </mfrac>
   </mfrac>
   </math>
   </body>
```

a+b/c+d

Web hosting (How to start a new URL)

godaddy

It is a type of service in internet environment that allows individuals, organizations to maintain own website in the world wide web.

Remember the following points.

1) A common internet service is web hosting.
2) web hosting means storing your web site on a public server.
3) web hosting normally includes email services.
4) web hosting often includes domain name registration.

Web hosting Types

These are classified into the following three types.

1) Free hosting
2) Shared hosting
3) Dedicated hosting

1) Free hosting : free web hosting is best suited for small sites with low traffic, like personal site.

2) Shared (virtual) Hosting

with shared hosting your web sites gets its own domain name, and is hosted on a powerfull server along with other web sites.

3) Dedicated Hosting :

Dedicated hosting is the most expensive option. This option is best suited for large web sites with high traffic, and web sites that use special software.

Web hosting providers and their services.

To make your website visible to the world you should stored in the public web server, they are prouding following web services

1) connection speed
   ~~Most IPs~~
2) powerful hardware
3) Security and stability
4) 24 hour support

5] Daily backup
6] Email capabilities

* Web Hosting Technologies

The following technologies frequently we are using in the web hosting environment

1] Windows Hosting: Windows hosting means hosting of web services that runs on the windows operating system.

2] Linux Hosting: Linux hosting means hosting of web services that runs on the Linux operating system.

3] Unix Hosting: Unix hosting means hosting of web services that runs on the unix operating system. Less expensive than windows.

Web Hosting Database Technologies:

There are many different database systems.

1] SQL Server: Microsofts SQL server is a popular database software for database driven web sites with high traffic

2] Oracle: Oracle is also a popular database software for database driven web sites with high traffic.

3] MySQL: (Open source): My SQL is also a popular database software for web sites. My SQL is a very powerful robust and full featured SQL database system.

What is Domain Name

It is a unic name for your website. Your domain name should be easy to remember and easy to type. While you are selecting domain name follow the characterstics.

1] Short
2] Meaningfull
3] Clear
4] Exposure

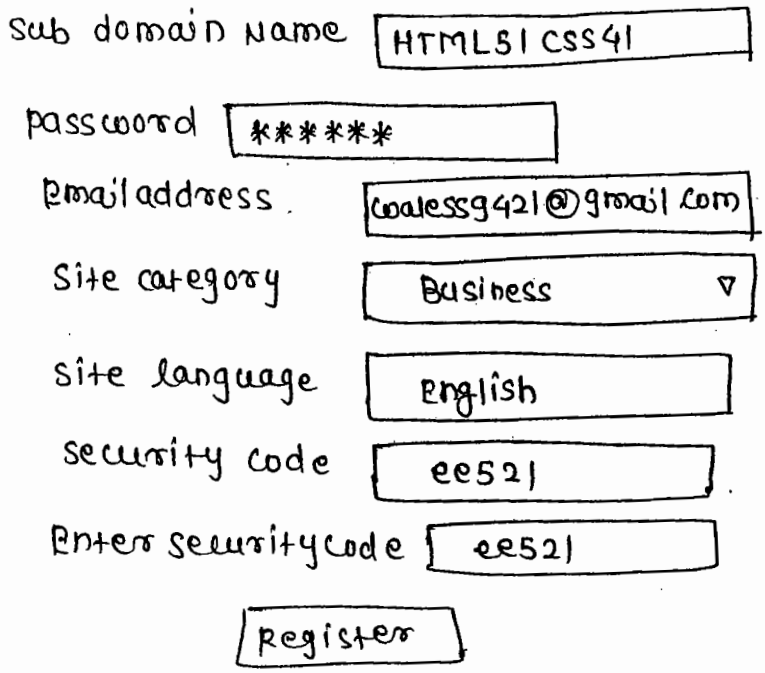
How to register a Domain

1] go to byethost.com (https://byethost.com)

2] click on free hosting

3] click here to sign up for free hosting

4] fill the registration form.



Note: Internet explorer not recomended

It displays an activation message, this message redirect to specific mail id.

→ go to your registered mail inbox.

→ Activate byethost link.

→ After few second it displays account order confirmation.

→ go back to required email inbox check for new mail, it contains cpanel details as follows.

cpanel username : b3_16554880

cpanel password : 12345678

your URL : http://html5icss41.byethost3.com

FTP Server : ftp:byethost3.com

FTP Login : b3_16554880

FTP password : 12345678

MysqlUsername : b3_16554880

mysql password : 12345678

→ Click on cpanel URL

→ It displays the following login screen.

username

b3-16554880

password

12345678

Login

→ Click on login it displays the following list of components.

1] preferences
2] Domains
3] Email
4] Databases
5] Logs
6] Files
7] security etc.

→ Click on online file manager → files

It displays file manager details click on htdocs

How to upload a file

Go to file manager click on upload button click on choose a file select the required file. click on confirm. click on back button.

It displays uploaded file (displayed. nit. html).

If you open this file you need to type http://html15css14.byethost3.com/hit.html

To avoid the file name follow the navigation.

go to file manager select required file click on rename enter index.html click on confirmation. click on backbutton.

→ It is displaying as index.html

we can open directly as follows.

→ html15css14.byethost3.com

Q] How to modify the webpage in the live.

go to file manager select the required html file click on edit it displays online editor with our existing source code.

do require modifications click on save. and refresh webpage.

Q) How to upload multiple resources

→ we can upload multiple files to our server location through the following navigation.

Step1 : select the required list of files.
Right click. click on add to archive. select zip format
Click on ok.

Step2 : go to byethost file manager click on upload select choose file select the required zip file click on confirm. click on back button

Q) How to unzip a folder in the server?

⇒ select the required file click on unzip confirm the message. click on back button

Control panel components

Control panel contains the following list of components

1) preferences : It contains the following list of preferences.

2) i) Account settings ii) change password iii) upgrade to update contact email iv) change language v) upgrade plans

2) Domains : It contains the following list of domains.
i) subdomains ii) Add on domains iii)

3) Email : It contains list of email accounts
i) webmail ii) personal mail iii) email accounts related to all categories.

4) Databases : It contains the details of the database like my-sql
It is default database for all free websites.

5) Files : It contains every file information including the following components.
i) backups ii) online file manager iii) Disc space usage iv) FTP accounts

Q] How to create security layer https. getbootstrap

→ we can create secured socket layer through SSL manager.

Domain Naming system

It contains actual url with the following components.

i] redirects ii] records

Software / Services

It contains the following list of components

i] cloud computing details ii] account statistic iii] error pages

iv] password protect directories etc.

Introduction to cascading style sheet

CSS is a simple design language to make web pages more presentable

Cascading : Represents inherit.

Style : Represents attribute or property

Sheets : That represent .css file.

Features of CSS :

CSS has the following list of unic features

1] Easy manage
2] Global change
3] save lot of time
4] Easy maintainance
5] Inline style sheets
6] Internal style sheets
7] External style sheets
8] performance
9] Superior styles
10] Multidevice compatibility
11] Global webstandards etc.

Why CSS?

CSS required because of the following reasons.

1] It is collection of style sheet mechanism
2] It is used to add more affects.
3] we can embed CSS in html, java script, PHP, ASP etc.
4] The extension of css file is .css.
5] use same style on multiple pages.
6] Reduce download size etc.

CSS versions History

CSS has the following list of versions

1] CSS 1.0 [1996]
2] CSS 2.0 [1998]
3] CSS 3.0 [2008]
4] CSS 4.0 [2014]

CSS Structure

→ As per w3c standard CSS has the following detailed structure

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

ex1:

```
<html>
<head>
 <style type="text/css">
 div
  {
   color: blue;
   background-color : #FFFFF0;
   text-decoration : underline;
   font-family : Arial unicode ms;
  }
 </style>
 </head>
<body>
<div> welcome to cascading style sheet.....! </div>
<div> welcome to cascading style sheet.....! </div>
</body>
</html>
```

Css comments

→ comments are non executable statements / non ignore statements.
Using these comment notations we can declare customized statements or user defined statements in the web environment or in the style sheet

→ In css every comment begins with "/*" and ends with "*/"

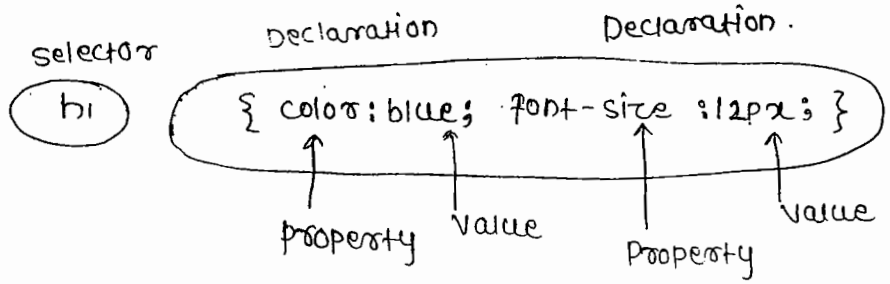
example :

```
<html>
<head>
<style type="text/css">
 div
  {
   /* color: blue;
      background-color: pink; */
  }
 </style>
</head>
<body>
<div> welcome in cascading style sheet </div>
<div> welcome in cascading style sheet </div>
</body>
</html>
```

Css Basic syntax :

It is made up of three parts

1] selector
2] property
3] value



Syntax: made up of three parts

Selector { property: value }

→ The selector is normally the html tag

```
<head>
<style type="text/css">
  div ← selector
  {
    color : blue ;
  }
  </style>
  </head>
<body>
  <div> welcome to cascading style sheet </div>
  </body>
```

property

value

Types of stylesheets

In css stylesheets are classified into the following three types.

1) Inline style sheets.
2) Internal / embeded style sheets.
3) External style sheets.

1) **Inline style sheets** : If we specify styles inside the tag in the body part. These styles able to apply only ~~that~~ for that perticular line

ex:

```
<head>
<title style='color:red'>
  Hello
</title>
</head>
<body>
<s style='color:red'> It is removed text from the web page </s>
<p style='color:blue ; background-color : lightgreen '> It is removed
text from the web page.... </p>
<b style=' text-decoration : underline'> It is removed text from the
web page </b>
</body>
```

style is not applicable in head section only applicable in body section.

2] Internal style sheets : These are popularly known as embeded style sheets. If we specify the style in our html file itself then they are called as internal styles. These styles cannot be used in other files. but we should implement again and again in the same file.

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

ex:
```
<head>
<style type="text/css">
 p
 {
  color: #FF00FF ;
  font-size: 20px;
  font-weight: bolder;
 }
</style>
</head>
<body>
 <p> welcome to Internal style sheets.... </p>
 <p> welcome to Internal style sheets.... </p>
 </body>
```

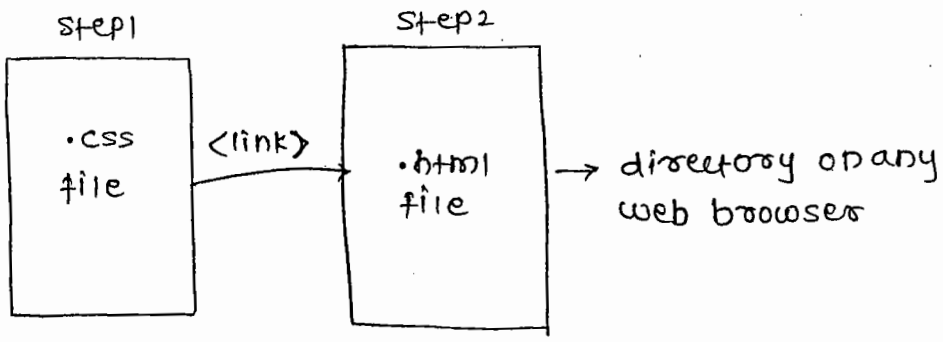
3] External style sheets:

If we declare the styles outside 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 = "stylesheet" href ="#" type = "text/css">
</head>
```

20-8-15

How to implement external style sheets.



step1: creating .css file

```
div
{
  color: #FF00FF; font-size: 25px;
  font-family: Arial Unicode MS;
  text-decoration: underline;
  font-weight: bold;
}
```

save with one.css extension @ any location.

step2: creating required html file

ex:

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

→ save with .html extension and run on any major web Browser.

ex2: with file path

```
<head>
<link rel ="stylesheet" href = "file:///C:\Users\subbaraj\
        Desktop\nit.css" >
</head>
```

```
<body>
<div> welcome to External style sheet ..... </div>
</body>
</html>
```

Working with css selectors

→ selector means styles reusability. In css there are different types of selectors.

1] Tag selector
2] Type selector
3] ID selector
4] Class selector
5] Grouping selector
6] Universal selector
7] Decendent selector
8] Customized selector etc.

1] **Tag selectors** : These are popularly known as type selectors it matches every instance of the element type in the document tree

Syntax:
```
div
{
 styles
 styles
 styles
}
```

ex:
```
<head>
<style type='text/css'>
 h1
 {
  color: #FF0000;
  background-color : #FFFF00;
  font-size : 20px;
  text-decoration : underline;
  font-family: comic sans ms';
 }
</style>
</head>
```

```
<body>
<h1> welcome to Type or Tag selector </h1>
</body>
```

2] ID selectors:

It is used to specify style for a single, unique element. The id selector uses the id attribute of the HTML element, and is defined with a "#"

Syntax: 1]

```
#div
  {
   styles
   styles
   styles
  }
```

Syntax: 2]

```
div#div
 {
   styles
   styles
   styles
 }
```

ex: 1]

```
<body>
#div <style type="text| css>
#h1
{
  color: #FF0000; font-size: 20px;
  text decoration: underline;
  font-family : comic sans ms;
 }
</style>
 </head>
<body>
<p id='h1'> welcome to ID selector </p>
</body>
```

ex: with prefix

```
<body>
<style type="text/css">
b#h1
{
  font-color: FF0000; font-size: 20px;
  text-decoration: underline;
  font-family: comic san cs;
}
</style>
</head>
<body>
<b id="h1"> welcome to ID selector </b>
</body>
```

ex:

```
<head>
<style type='text/css'>
#diu
{
  color: blue; font-family: tahoma;
}
</style>
<script type='text/css javascript' language="javascript">
function myvalc)
{
  document.getElementById("diu1").innerHTML= "HTML5";
}

<\script>
</head>
<body>
<p> click the button to change the value based on ID... </P>
</tbody>
```

```
<diu id = "diu"> AngularJs </diu>
<diu id = 'diu1"> jQuery </diu>
<button onclick = " my valc)"> Click me </button>
</body>
```

3] * Selectors Grouping

We can group selector using comma separator

1] Example 1 :

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

is equivalent to

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

ex 2 :

```
<body>
<head>
<style type = "text/css">
#div , p  <-- Grouping selectors (id, tag)
{
    Color: blue ; font-size : 20px ;
    font-family : tahoma ;
}
</style>
</head>
<body>
<diu id = "diu"> Java script is client side or scripting language
</diu>
<p> welcome to selector grouping </p>
</body>
```

3] class selectors

It is used to specify a style for a group of elements. The class selector uses the HTML classes attribute, and is defined with a "." (period).

Syntax:

```
.div
{
  styles
  styles
  styles
}
```

ex1:

```
<head>
<style type='text/css'>
.div
{
  color: blue; font-size: 20px;
  font-family: tahoma;
}
</style>
</head>
<body>
<div class="div"> welcome to class selector </div>
<div class="div">
</body>
```

ID and class grouping

```
<head>
<style type='text/css'>
div.div, #h1
{
  color: blue; font-size: 20px;
}
</style>
</head>
<body>
<div class="div"> welcome to class </div>
<p id="H1"> welcome to div ID selector </p>
</body>
```

Class selector with java script

```
<head>
 <style='text/css'>
  .div
   {
    Color: blue ; font-size: 20px ;
   }
  </style>
 <script type='text/javascript'>
 function myval()
  {
   document.getElementById("div").innerHTML = "HTML5";
  }
 </script>
  </head>
  <body>
 <p> click the button to display class related result </p>
 <div id= "div" class="div"> Angularjs </div>
 <div id= "div1" class= "div"> Java script </div>
<button onclick = "myvalue()"> click me </button>
 </body>
```

Working with css units or measurements

css supports a number of measurement including absolute and relative unix. These units are:

1] EM
2] Ex
3] px
4] in
5] cm
6] mm etc.

1] EM : It is a unit of measurement in the field of typography
It is developed based on "m"

Syntax: em;

ex:
```
<head>
<style type='text/css'>
 div
 {
   font-size: 4em;
 }
 </style>
</head>
<body>
<div> working with css units ....!! </div>
</body>
```

2] ex: is another measurement develoed based on lower case 'x'.

Syntax: ex;

```
<head>
<style type='text/css'>
div
{
  font-size: 4ex;
}
</style>
</head>
<body>
<div> working with css units.... </div>
</body>
```

3] px: It defines a measurement in screen pixels.

syntax : px;

```
<head>
<style type='text/css'>
<h div
    {
      font-size: 4 px;
    }
<div> coorking
```

4] in: It defines a measurement in inches.

syntax : in;

```
<head>
<style type='text/css'>
 div
 {
   font-size! 1 in ;
 }<body>
<div> coorking with css units !! </div>
</body>
```

5] cm: It defines a measurement in centimeters.

syntax: cm;

```
<head>
<style type='text/css'>
 div
  {
    font-size : 1cm ;
  }
 </style>
 </head>
<body>
<div> coorking with css units !! </div>
 </body>
```

6) ~~mm~~ pt: It defines a measurement in points.

Syntax: pt;

```
<head>
<style type='text/css'>
  div
  {
    font-size = 1pt;
  }
</style>
</head>
<body>
<div> working

</body>
```

pc: It defines measurement in picas.

Syntax: pc;

```
<head>
<style
 div
  {
    font-size = 1pc;
  }
<      >
<      >
</body>
 <div> working
```

% It defines measurement in percentage

Syntax: % ;

```
<
<
 div
 {
  font-size = 100% ;
 }
<     >
<     >
<body>
 <div>
 </body>
```

CSS Background property

CSS supports the following list of background properties.

1] background-color
2] background-image
3] background-repeat
4] background-attachment
5] background-position

1] background-color

It is used to set the background color of an element

Syntax: background-color: colorName | colorcode | Hexa code ;
or
background: Color Name | Color code | Hexa code;

Ex:

```
<head>
<style type='text/css'>
 div
  {
   background-color: #FF00FF ;
  }
</style>
</head>
<body>
 <div> working with css background... </div>
</body>
```

2) Background-image : It is used to set the background image of an element.

Syntax: background-image : url ('imgpath ');
or
background : url ('imgpath');

```
<head>
< head>
<style
body
{
 background- image : url("html5.png");
 }
<\style>
<\head>
<body>
<div> working with css background ~. <\div>
</body>
```

3) Background repeat: It is used to control the repeatition of an image in the background.

Syntax: background-repeat : no-repeat ;
or
background : no-repeat;

ex: <head>
<style

property value

value	Description
repeat	The background image will be repeated both vertically and horizontally.
repeat-x	The background image will be repeated only horizontally.
repeat-y	The background image will be repeated only vertically
no-repeat	The background image will not be repeated.

ex:
```
<head>
<style
body
{
  to background-image : url("html5.png");
    background-repeat : no-repeat;
}
</style>
</head>
<body> working with css Backgrounds ....!! </body>
</body>
```

Background attachment : It is used to controll scrolling of an image in the background.

Syntax : background-attachment : fixed;
or
background : fixed;

Background - position : It is used to control position of an image in the background.

Syntax : background - position : center;
or
background : center;

```
<head>
<style>
body
{
  background- image : url ("html5.png");
  background- repeat : no repeat;
  background- attachment : fixed;
  background- position : center;
}
</style>
</head>
<body>
<div> working with css Background </div>
</body>
```

CSS font family

CSS supports following list of font properties:

1) The font-family property is used to change the face of a font.
2) The font-style property is used to make a font italic or oblique
3) The font-varient property is used to create a small-caps effect.
4) The font-weight property is used to display bold or light a font appears.
5) The font-size property is used to increase or decrease the size of a font.

ex:

```
<head>
<style type="text/css">
div
{
 font-family: tahoma;
 font-size: 10px;
 font-style: oblique;
 font-weight: bolder;
 font-varient: small-caps;
}
</style>
<body>
<div> working with css ~~background~~ ~~prop~~ font family </div>
</body>
```

⊛ Css Text properties

Css supports the following list of text properties:

1) color:
2) Direction
3) Text-decoration
4) text-indent
5) text-align
6)

1) color: using this property we can change the text color.

Syntax: color: colorName / color code

Example:

```
<head>
<style type='text/css'>
div
{
  color: blue;
}
</style>
</head>
<body>
<div> welcome to CSS Text properties...</div>
</body>
```

2) Direction: It is used to display the direction of the text.

The Text default direction is left to right.

Syntax: direction: value(s).

Example:

```
<head>
<style type='text/css'>
div
{
  direction: ltr;
}
p
{
  direction: rtl;
}
</style>
<body>
<div> welcome to CSS direction property </div>
<p> welcome to CSS Text property</p>
</body>
```

3] Text-decoration : It supports different decoration, like, underline, overline, line through.....

Syntax: text-decoration : value ;

Ex:
```
<head>
<style type='text/css'>
 div
 {
   text-decoration : underline;
 }
 p
 {
   text-decoration : overline;
 }
 b
 {
   text-decoration: line through;
 }
<\style>
 <\head>
 <body>
<div> It is underline format <\div>
 <p> It is underline format<\p>
 <b> It is underline format <\b>
 <\body>
```

4] Text-Indent: This property is used to display indent at the starting of a paragraph.

Syntax: Text-indent : pixels ;

5] Text-align: This property is used to align the text in a specific direction.

ex: Syntax : text-align : left/right/center/justify ;

ex: <head>
<style type='text/css'>
div p
{
text-indent: 50px;
text-align: justify;
}
</style>
</head>
<body>
 welcome to indent property
</body>

6] Letter-spacing: This property is used to apply space between letters.

Syntax: letter spacing pixels;

ex: <head>
<style type='text/css'>
p
{
letter-spacing: 10 ~~pixels~~ px;
}
</style>
</head>
<body>
 welcome to css properties
</body>

7] word-spacing: This property is used to add or substract space between the words.

<head>
<style-type='text/css'>
p
{
word-spacing: -5px;
}

```
</style>
</head>
<body>
<p> welcome to css property </p>
</body>
```

8] Text-transform : This property is used to capitalize the text, convert text upper case or lower case format.

Syntax : Text-transform : value;

ex:
```
<head>
<style = 'text/css'>
p
{
  text-transform : capitalize;
}
</style>
</head>
<body>
<p> welcome to css properties </p>
</body>
```

9] white-space : This property is used to control the flow and formating of text.

```
<head>
<style type= 'text/css'>
 p
 {
   White-space: nowrap;
 }
</style>
 </head>
 <body>
 <p> Line never break ; Line never break </p>
 </body>
```

10] Vertical Align: This property sets the vertical alignment of an element. It supports the following list of values:

1] baseline
2] sub
3] super
4] top
5] text-top
6] middle
7] bottom
8] text-bottom
9] 0px
10] 10px

```
<head>
  <style type = 'text/css'>
   p
    {
      vertical-align : sub ;
    }
   </style>
   <body> Vertical alignment of an <p> element </p>
   </body>
```

CSS Borders :

CSS supports the following list of border properties.

1] Border-color: The border color specifies the color of a border.

2] 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 specifies the width of a border.

ex:
```
<head>
<style type='text/css'>
div
{
border-color: green;
border-style: dashed;
border-width: 2px;
}
</style>
</head>
<body>
<div> welcome </div>
</body>
```

Working with Advanced style sheets (CSS3)

CSS3 is new style sheet, it is divided into several modules. Each module having new capabilities and extended features.

1] @font-face
2] opacity
3] RGBA (Red Green Blue Alpha or Amber)
4] Border-Radius
5] Box-Shadow
6] Text-Shadow
7] Gradient
8] Multiple background images
9] Transform
10] Transition
11] multi column layout
12] Styling forms with Attribute selector.
13] Wrapping up
14] Box-sizing and Box-model
15] CSS3 selector

*] CSS3 Browser support

→ In CSS3 we are using the following prefixes for advanced properties.

1] IE requires the prefix -ms- (Microsoft seem)
2] Firefox requires the prefix -moz-
3] Chrome and Safari requires the prefix -webkit-
4] Opera requires the prefix -o-.

Working with CSS3 multiple column properties

In CSS3 we can create multiple columns layout for E-news papers. The following list of properties frequently we are using.

1] column-count
2] column-gap
3] column-rule
4] column-fill
5] column-rule-color
6] column-rule-style
7] column-rule-width
8] column-span
9] column-width
10] columns

1] column-count : This property is used to specify number of columns and element divided ~~should~~ into:

Syntax: column-count: number| auto;

Value	Description
number	The optimal number of columns into which the content of the element will be flowed.
auto	The number of columns will be determined by other properties.

ex1:

```
<head>
<style type='text/css'>
 div
 {
  column-count: 3;
 -moz- column-count: 3; /* mozilla firefox */
 -webkit- column-count: 3; /* chrome/safari */
 -ms- column-count: 3; /* IE */
 -o- opera- column-count: 3; /* opera */
 }
 </style>
 </head>
 <body>
<div>Some text.... !! </div>
 </body>
```

2] column-gap

This property specifies the gap between the columns.

syntax: column-gap: length | normal;

value	Description
length	a specified length that will set the gap between the column.
normal	specifies a normal gap between the column.

ex:

```
<head>
<style type='text/css'>
 div
 { text-align: justify;
   column-gap: 80px;
  -moz- column-gap: 30px;
   column-count: 3;
  -moz- column-count: 3;
 }
```

```
</style>
 </head>
</body>
 <div> some Text.... !! </div>
 </body>
```

3) column-rule-property : It is a shorthand property for setting all the column-rule-* properties. The column rule property sets the width, style, and color of the rule between columns.

Syntax: column-rule: column-rule-width column-rule-style column-rule-color;

Value	Description
column-rule-width	sets the width of the rule between columns
column-rule-style	sets the style of the rule between columns.
column-rule-color	sets the color of the rule between columns.

ex:

```
<head>
<style type='text/css'>
 div
 {
   text align: justify;
   column-count: 3;
 -moz-column-count: 3;
   column-gap: 30px;
 -moz-column-gap: 30px;
 -column-rule-width: 2px;
 -moz-column-rule-width: 2px;
  Column-rule-style: solid;
 -moz-column-rule-style: solid;
  Column-rule-color: #FF0000;
 -moz-column-rule-color: #FF0000;
 }
 </style>
 </head>
```

```
<body>
<div> some Text .... !! </div>
</body>
```

4] column-fill : It specifies how to fill columns, balanced or not.

Syntax: column-fill : balance| auto;

value	Description
balance	columns are balanced. Browsers should minimize the variation in column length.
auto	columns are filled sequentially, and they will have different lengths.

```
<head>
<style type='text/css'>
```

Note: currently no major web browser support.

5] column-span : It specifies how many columns an element should span across.

Syntax: column-span : 1|all ;

value	Description
1	The element should span across 1 column.
all	The element should span across all columns.

ex:
```
<head>
<style type= 'text/css'>
 div newspaper div
 {
   text-align: justify;
 -webkit- column- count: 3;
   column- count: 3
 }
 h2
 {
 - moz- column-span : 1 ;
   column-span: 1 ;
 }
```

```
</style>
</head>
<body>
<div>
<div><h2> Heading </h2>
Some Text </div>
</body>
```

6] column-width : It specifies the width of the column.

syntax : column-width: auto | length;

Value	Description
auto	The column width will be determined by the browser.
length	A length that specifies the width of the column.

Note: If you are increasing the page size number of columns will decrease and vice versa.

ex:
```
<head>
<style type='text/css'>
div
{
-moz-column-width: 100px;
column-width: 100px;
}
</style>
</head>
<body>
<div> Some Text ..... </div>
</body>
```

7] columns : This property is shorthand property. for setting column width and column count.

Syntax: columns: column-width column-count;

Value	Description
column-width	The width of the column
column-count	The number of column.

ex:
```
<head>
<style type='text/css'>
 div
  {
    columns: 100px 3;
   -moz- columns: 100px 3;
  }
<!style>
 <!head>
  <body>
  <div> Some Text <!div>
  <!body>
```

(*) Working with CSS3 background property

CSS3 supports the following list of advanced background properties

1] background size:

property	Description
background-size	specifies the size of the background images.
background-origin	specifies the positioning area of the background images.
background-clip	specifies the painting area of the background images.

1] Background size : This property specifies the size of the background images. Before CSS3 the background image size was

determined by the actual size of an image.

Syntax: background-size: length| percentage| cover| contain;

ex1:

```
<head>
<style type='text/css'>
body
{
  background: url(img_flwr.gif);
  background-size: 480px 480px;
 -moz- background-size: 480px 480px;
  background-repeat: no repeat;
}
</style>
</head>
<body>
 <p>
  CSS-image
 </p>
<p> original image: <img src="water.gif"> alt=
 "Flowers" width="200" height="200" /></p>
</body>
```

Background clip: It specifies the painting area of the background.

Syntax: background-clip: border-box| padding-box| content-box;

ex:

```
<head>
<style='text/css'>
div
 {
  width: 300px; height: 300px;
  padding: 50px; background-color: lightblue;
  background-clip: border-box;
   border: 4px dashed #FF0000;
   text-align: justify;
  }
```

```
</style>
</head>
<body>
<div> some Text.... !! </div>
</body>
```

3] Background origin : This property specifies what the background position property relative to

Syntax : background-origin : padding-box| border-box| content box;

Ex:
```
<head>
<style>
div
{
  border: 2px solid red; padding: 30 px;
  background image: url('chrome.png');
  background-repeat: no repeat;
  background-position: left;
  background-origin: content-box;
}
</style>
</head>
<body>
  <div> some Text..... !! </div>
  </body>
```

Working with CSS3 borders

CSS3 supports the following list of advanced properties:

Property	Description
border-image	A shorthand property for setting all the border image-* properties.
border-radius	A shorthand property for setting all the four border-*-radius properties.
box-shadow	Attaches one or more drop-shadows to the box.

* **Border image:** It is a shorthand property for setting a border image source, border image slice, border-image width, border-image outset and border image repeat properties.

Syntax: border-image: source slice^width outset repeat;

ex:

```
<head>
<style>
 div
 {
   border: 15px solid transparent;
   width: 250px; padding: 10px 20 px;
 }
#round
{
  border-image: url( border.png ) 30 30 round;
}
</style>
</head>
 <body>
        <img src=
<div id='round'>  some Text.... </div>
 </body>
```

Box shadow : The box shadow property attaches one or more drop shadows to the box.

Syntax : box-shadow: h-shadow, v-shadow, blur shadow color inset;

Ex :
```
<head>
<style>
div
{
 width: 300 px ; height: 80 px;
 background-color: #FFFF00;
 box-shadow: 15px 15px 15 px #FF0000 ;
 }
<\style>
<\head>
<body>
<div> <\div>
<\body>
```

* Border-Radius : It ~~specifies~~ is a shorthand property for setting the four border-*-radius properties.

Syntax : border-radius : 1-4 length % / px;

ex :
```
<head>
<style>
div
{
 width: 300px ; height: 10px;
 border : 2px solid #FF0000 ;
  padding : 5px;
  border-radius : 5px;
 }
<\style>
<\head>
<body>
```

 welcome to Borders....

</body>

⊛ CSS3 Text properties

CSS3 supports the following list of text properties :

1) hanging-punctuation
2) punctuation-trim
3) Text-emphasis
4) Text-justify
5) Text-outline
6) Text-overflow
7) Text-shadow
8) Text-wrap
9) word-break
10) word-wrap

1) Word-wrap property :

This property allows long words to be able to be broken and wrap onto the next line.

Java script syntax:

```
object.style.wordwrap = "break-word"
```

Syntax :

```
word-wrap: normal | break-word ;
```

value	Description
normal	Break words only at allowed break points.
break-word	Allows unbreakable words to be broken.

ex:

```
<head>
<style>
 div
  {
    width: 10em;
    border: 2px solid #FF0000;
    word-wrap : break-word;
  }
```

```
</style>
</head>
<body>
<div> HTML5 is New Hypertext markup language for latest
web apps. It is new for mobile apps for responsive webapp </div>
</body>
```

2] word-break : It specifies the line breaking rules for non-CJK scripts.

Syntax : word-break : normal|break-all| hyphenate;

Java script syntax :

object.style.wordBreak = "hyphenate".

Value	
normal	Breaks non-CJK scripts according to their own rules.
break-all	Lines may break between any two characters for non-CJK script.
hyphenate	words may be broken at an appropriate hyphenation point.

3] Text wrap property

This property specifies line breaking rules for text

Syntax : text-wrap: normal|none|unrestricted|suppress;

Java script syntax : object.style.textwrap = "none"

Note: This property currently major web browsers are not supported

4] Text shadow : This property applies shadow to text. you specify the horizontal shadow, the vertical shadow, the blur distance, and the color of the shadow.

Syntax : text-shadow : h-shadow v-shadow blur color;

Java script syntax :

object. style . text shadow = "2px 2px #ff0000"

ex:

```
<head>
<style type='text/css'>
div
{
  font-size: 30px;
  text-shadow: 5px 5px 5px #FF0000;
}
p
{
  font-size: 30px;
  text shadow: 5px 5px #FF0000;
}
</style>
</head>
<body>
<div> welcome to text shadow property.... </div>
<p> welcome to text shadow property.... </p>
</body>
```

Text overflow property :

This property specifies what should happen when text overflows the containing element.

syntax : text-overflow : clip | ellipses | string ;

Java script syntax : object.style.textoverflow = "ellipses" ;

value	description
clip	clips the text
ellipses	Render an ellipsis ("....") to represent clipped text
String	Render the given string to represent clipped text

```
<head>
 <style>
 div.test
 {
   white-space :nowrap;
   width :12em ; overflow: hidden;
   border : 1px solid #ff0000 ;
   text overflow : ellipsis ;
 }
 </style>
 </head>
  <body>
   <div . class = "test"> This will some long text that will not fit in
            box.</div>
   </body>
```

Text-justify : This property specifies the justification method to use when text-align is set to "justify". This property specifies how justified text should be aligned and spaced.

syntax : Text-justify : auto |inter-word| inter-ideograph|
inter-cluster | distribute| kashida |trim;

Java script syntax : object.style.textjustify = "inter-word";

ex:
```
<head>
<style
div
{
  text-align: justify;
  text-justify : inter-word;
 }
</style>
</head>
<body>
  <div> some text.... Resize the Browser window..... </div>
  </body>
```

CSS3 Font properties

CSS3 supports the following list of advanced font properties.

1) @ font-face
2) font-size-adjust
3) font-stretch

1) @ font-face : The CSS3 font-face rule: your "own" fonts are defined in the CSS3 @font face. Before CSS3 webdesigners had to use fonts that are already installed in the users computers.

www.1001freefonts.com

1) .ttf - True Type Fonts
2) .otf - opene type fonts
3) EOT - Embeded open type
4) WOFF → web open font format
5) SVG → Scalable

Note : Firefox, Chrome, Safari and opera support font of type .ttf and .~~ttf~~ ,

Note : Enternet Explorer support the new @ font-face rule but it only supports fonts of type .eot.

Ex1 :
```
<head>
<style
@font-face
  {
    font-family : myFirstFont;
    src: url ('sansation_Bold_Italic.ttf')
  }
div
{
  font- family : my first font ;
}
</style>
</head>
<body>
  <div> some text </div>
  </body>
```

2] font_size_adjust

This property gives you better control of the font size.

Syntax: font-size-adjust : number| none| inherit;

Value	Description
number	Defines the aspect value to use.
none	Default value. No font size adjustment
Inherit	Inherits the font size adjustment from parent elements.

ex:
```
<head>
<style
div
{
  font-size-adjust : 20px;
}
</style>
</head>
<body>
<div> some text..... </div>
</body>
```

3] font-stretch : This property make or allow text wider and narrower.

Syntax: font-stretch: wider| narrower| ultra-condensed| extra-condensed| condensed| semi-condensed| normal| semi-expanded| extra-expanded| ultra-expanded| inherit.

Note: none of the major supports the font-stretch property.

CSS3 Transforms:

Transform is a powerfull property to move, scale, turn, spin and stretch element. It supports the following list of properties.

1] transform
2] transform-origin
3] transform-style

Transform support the following five methods.

1] translate()
2] rotate()
3] scale()
4] skew()
5] matrix()

1] translate() : with the help of this method you can move your object depending on its parameter. Two type of parameter you can pass in this method one is from left (x-axis) and the another is from top (y-axis).

Syntax: transform : translate(x,y);

Example:

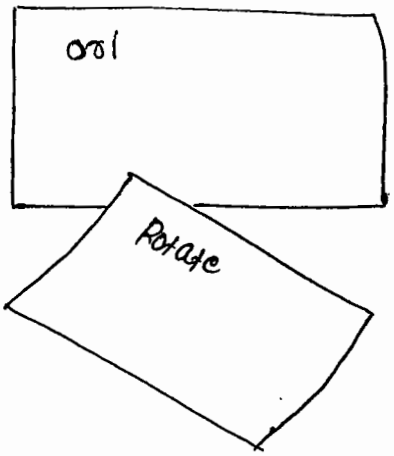
```
<head>
<style type='text/css'>
div
{
  width :90px ; height:60px;
  background-color: #FF9900;
  border: 2px solid #FF00FF;
}
div#div1
{
 transform : translate (20px, 30px);
}
div #div2
 {
  transform : translate (40px, 60px);
 }
</style>
</head>
<body>
<div> </div>
<div id="div1"> </div>
<div id = "div2"> </div>
</body>
```

2] Rotate ( ): with the help of this method you can rotate your object depending on its value. Two types of value you can pass in this method one is positive and the another one is negative

Syntax: transform: rotate (x, y);

x → represent clock wise rotation

y → represent counter clock wise rotation. (Anticlock).



ex:
```
<head>
<style>
 div
 {
  transform: width: 90px ; height : 60 px;
    background-color : #FF9900;
    border: 2px solid #FF00FF;
 }
 div #div1
 {
  transform : rotate (30deg);
 }
</style>
</head>
 <body>
<div> ori </div>
<div id='div1'> rotate </div>
</body>
```

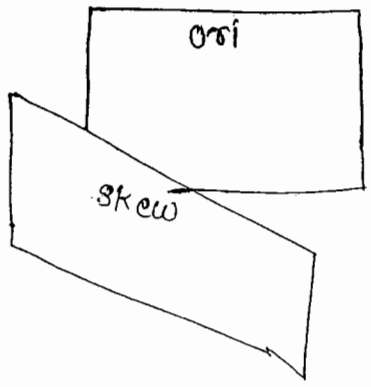
3] scale : with the help of this method you can increase or decrease your object size depending on its value passes in the parameter Two types of value you can pass in the parameter, one is for width (x-axis) and the another one is for height (y-axis).

Ex: scale(x,y);

ex:
```
<head>
<style>
 div
 {
   width: 90 px; height: 60 px;
    background color: #FF9900;
    border: 2px solid #FF00FF;
 }
div#div1
{
  margin: 100 px;
} transform : scale (2,3);
</style>
 </head>
<body>
<div> Rori </div>
<div id='div1' > scale </div>
 </body>
```

4] skew(): with the help of this method you can change the angle of your object depending on its value passed in the parameter. Two types of value you can pass in the parameter, one is for horizontal (x-axis) and the another one for vertical (y-axis).

Syntax : transform: skew(x,y);



ex:
```
<head>
<style>
div
{
 height: 60px ; width: 90 px;
  background-color: #FF9900;
  border: 2px solid #FF00FF;
}
div#div1
{
 transform: skew (20deg, 30deg);
}
</style>
</head>
<body>
 <div> ori </div>
 <div id='div1' > skew </div>
 </body>
```

5] Matrix() :

CSS3 Transitions

A Transition is such a property of CSS3 which is used to animate the object without using flash or any other animation application. With this feature of CSS3 you can change the shape and size of your object with animated effects.

Transition properties : It supports the following list of properties

1] Transition	A shorthand property for setting the four transition property into a single property.
2] transition-property	specifies the name of the css property to which the transition is applied
3] Transition-Duration	defines the length of time that a transition takes.
4] transition-timing-function	describes how the speed during the transition will be calculated.

```
    transition-duration : 5s ; float: right;
  }
 </style>
 </head>
 <body>
  <div>    </div>
 <div1>    </div1>
 </body>
```

Transition Delay : It specifies when the transition effect will start. The transition delay value is ~~defined~~ defined in seconds(s) or milliseconds (ms).

Syntax : transition-delay: time;

Value	Description
time	specifies the number of seconds for milliseconds to wait before the transition effect will start.

ex:
```
<head>
 <style>
  div
  {
   width: 90 px; height: 60 px;
   background-color: #FF0000;
   transition: width; border-radius: 20px;
   transition-duration: 2s ; transition-delay: 1s;
  }
  div : hover
  {
    width: 380 px;
  }
 </style>
 </head>
 <body>
  <div> </div>
  </body>
```

Working with CSS3 Animation.

CSS3 supports the following list of Advanced Animations. We can create animations, which can replace animated images, flash animation and javascript in many webpages.

Animation is a property to change an object from one style to another style in a animated way.

Animation properties

CSS3 supports the following list of animation properties

1] @keyframes
2] animation
3] animation name
4] animation duration
5] animation delay
6] animation direction etc.

1] **@keyframes rules** : The @keyframe rule is where the animation is created. specify a css style inside the @keyframes rule and the animation will gradually change from the current style to the new style. we can create user defined animations.

How to Implement Animation :

An animation is an effect that lets an element gradually change from one style to another style : you can change as many styles you want, as many times you want.

Specify when the change will happen in percent or the keywords "from" and "to" which is the same as 0% and 100% . 0% is the begining of the animation . 100% when the animation is complete.

CSS3 Animation property

The Animation property is a shorthand property for six of the animation properties: animation-name, animation-duration, animation timing-function, animation-delay, animation-iteration count, and animation direction.

ex: <head>
<style>
div
{
width: 90px; height: 60px;
background-color: #FF9900;
transition: width; float: right;
transition-duration: 5s;
}
div:hover
{
width: 350px;
}
</style>
</head>
<body>

</body>

Multiple Transitions

```
<head>
<style>
 div
 {
  width: 90px; height: 60px;
  background color : #FF0000;
  transition: width; border: 5px; -radius: 20px;
   transition-duration: 2s;
 }
 div:hover
 {
  width: 350px;
 }
 div1
 {
    width: 90px; height: 60px;
      background-color: #0000FF;
    transition: width; border-radius: 5px;
```

Transition property : The transition property is a shorthand property for the four transition properties: transition property, transition duration, transition-timing-function, and transition-delay.

Syntax : transition: property duration timing-function delay;

Ex1 :

```
<head>
<style>
div
{
 width: 90px; height : 60px;
  background-color : #FF0000;
  transition: width 2s ;
 }
div hover
 {
   width: 350 px;
 }
</style>
</head>
<body>
 <div>  </div>
</body>
```

Transition Duration : It specifies how many seconds (s) or milliseconds (ms) a transition effect takes to complete.

Syntax : transition-duration: time;

Value	Description
time	specifies how many seconds or milliseconds a transition effect takes to complete.

Transition timing function : It specifies the speed curve of the transition effect. This property allows a transition effect to change speed over its duration.

Syntax: Transition-timing-function; linear| ease| ease-in| ease-out | ease-in-out | cubic-bezier (n,n,n,n);

Syntax: animation: name duration timing-function delay
-iteration-count direction;

ex1:

```
<head>
<style type='text/css'>
 div
 {
   width: 80px ; height : 70px;
    background-color: #FF0000;
     position: relative;
     animation: mymove 15 4 ;
      border-radius: 20px;
    }
-moz-@keyframes mymove  /* firefox */
    {
      from {left : 0px;}
      to { left :300px;}
    }
</style>
</head>
<body>
 <div>  </div>
</body>
```

ex2:

```
- moz- @ keyframes mymove
    {
```

CSS3 Animation duration property

The animation duration property defines how many seconds or milliseconds an animation takes to complete one cycle.

Syntax: animation-duration: time;

Value	Description
time	specifies the length an animation takes to finish. Default value is 0, meaning there will be no animation.

Animation iteration count property

The animation iteration count property defines how many times an animation should be played.

Syntax : animation-iteration-count: value;

value	Description
n	A number that defines how many times an animation should be played.
infinite	specifies that the animation should be played infinite times.

ex:

```
<head>
<style
div
{
  width: 80px; height: 90px;
  background-color : #FF0000;
   position: relative;
   animation: mymove;
   animation-duration: 5s;
   animation-iteration-count : 6;
   border-radius: 20px;
}
@keyframes mymove
{
  from { left : 0px ;}
    to {left : 300px;}
}
</style>
</head>
```

ex: with different direction

```
<head>
<style
div
{
  width: 60 px; height : 70 px;
   background color : #FF0000 ;
   position: relative ;
   animation: mymove ;
   animation-duration: 1s ;
   animation- iteration- count : infinite ;
   border-radius : 20 px ;
}
 @keyframes mymove
 {
   from { top: 0px ;}
   to {top: 250 px ;}
   }
 </style>
 </head>
 <body>
  <div> </div>
  </body>
```

Animation Direction

The animation direction property defines whether or not the animation should play in reverse on alternate cycles. If the animation direction value is "alternate" the animation will be played as normal every odd time (1,3,5 etc...) and backwards every even time (2,4,6. etc....).

Syntax: animation direction : value ;

value	Description
normal	Default value. The animation should be played as normal value.
alternate	The animation should be played in reverse on alternate cycle.

ex:

```
<head>
<style
div
{
  height : 70 px; width : 60 px;
  background-color : #FF0000;
   position: relative;
  animation: mymove;
   animation-duration: 5s;
  animation- iteration- count: infinite;
   border-radius: 20px;
   animation- direction: alternate;
}
-moz- @keyframe mymove /* firefox*/
{
0% { background: Red ; left : 0px ; top : 0px ; }
25% { background: yellow ; left : 200px ; top: 0px ; }
50% { background: Green ; left : 0px ; top : 200px ; }
75% { background: bule ; left : 0px ; top : 200px ; }
100% { background: Red ; left : 0px ; top : 0px ; }
}
</style>
</head>
<body>
<div> </div>
</body>
```

Working with Aduanced selectors :

In CSS3 selector means styles reusability. There are the following list of aduanced selectors:

1. [attribute^= values] selector

The [attribute^= value] selector matches every element whose attribute value begins with a specified value.

Note: The [attribute^=value] selector is supported in all major browser.

Ex:
```
<head>
 <style
 div [class^= "test"]
 {
   background: #ff0000; font family: tahoma;
 }
<\style>
 <\head>
<body>
 <div class = "first_test" > The first div element <\div>
 <div class = "test" > The second div element <\div>
 <div p class = "test" > This is some text in a <\p>
 <\body>
```

ex 2:
```
<head>
 <style
  di [class ^= "test"]
     {
       background : #ff0000 ; font family: tahoma;
     }
  <\style>
  <\head>
 <body>
 <div class = "first -test" >
```

2] CSS3 [attribute $= value] selector

The [attribute $=value] selector matches every element whose attribute value ends with a specified value.

Note: The [attribute $= value] selector is supported in all major browsers.

ex:
```
<head>
<style>
div[class $= 'test']
{
  background: #ffff11;
}
</style>
</head>
<body>
<div class = "first_test"> The first div element </div>
<div class= "second"> The second div element </div>
<div class = "test"> The Third div element </div>
<p class = "test"> The fourth div element </div>
</body>
```

3] CSS3 [attribute * = value] selector

The [attribute * = value] selector matches every element whose attribute value containing a specified value.

Note: The [attribute *= value] selector is supported in all major web browsers.

ex:
```
<head>
<style>
[class* = "test"]
{
  background = #ffff0;
}
</style>
</head>
```

```
<body>
<div class = "first_test"> The first div element </div>
<div class = "second "> The second div element </div>
<div class = "test"> The third div element </div>
<d p class = "test"> The fourth div element </div>
</body>
```

CSS3 - first of type selector

The first of type selector matches every element that is the first child, of a particular type of its parent.

Note: The first of type selector is supported in all major web browser except IE8 and earlier.

ex:
```
<head>
<style>
P: first-of-type
  {
    background : #00ffff;
  }
</style>
</head>
<body>
<h1> This is heading </h1>
<p> This is first paragraph </p>
<p> The second paragraph </p>
<p> The Third paragraph </p>
</body>
```

CSS 3- only-of-type selector

The only of type selector is matches every element that is the only child of its type of its parent.

ex:
```
<head>
<style>
{
p: only-of-type
{
  background: #ff00ff;
  font-family: tahoma;
}
</style>
</head>
<body>
<p> This is paragraph </p>
<div><p> This is paragraph </p>
<p> This is paragraph </p> </div>
</body>
```

CSS3 - nth-child selector

The nth child selector matches every element that is the nth child regardless of type of its parent.

Note: The nth child selector is supported in all major web browsers. except IE8 and earlier.

ex:
```
<head>
<style>
p: nth-child selector (4)
{
  background : #ff cc00 ; font family: tahoma;
}
</style>
</head>
<body>
<h1> This is heading </h1>
<p> The first paragraph </p>
<p> The second paragraph </p>
<p> The Third paragraph </p>
<p> The fourth paragraph </p>
</body>
```

Css3 User Interface

User interface is powerfull property in css3 it is used to implement user interface without modifiging a code section. you can resize div element and set the user interface property as follows:

css3 Resizing

It specifies whether or not an element should be resizable by the user.

resize: is a such property of user interface, by which you can resize your div layout on your browser. Three features of resize you use.

a) resize: both
b) resize: vertical
c) resize: horizontal

Syntax: resize: none| both| horizontal | vertical.

Value	Description
none	User cannot resize the element user can adjust both the height and the width.
horizontal	User can adjust the width of the element
Vertical	User can adjust the height of the element.

ex:
```
<head>
<style>
div
{
  border: 2px solid;
  padding: 10px 40px;
  width: 300 px;
  resize: vertical;
  overflow: auto;
}
</style>
</head>
<body>
  <div> Some Text.... </div>
</body>
```

HTML5 Web storage

It has the following aliase name:

1] client side storage
2] web storage
3] off line storage
4] Local storage

It is very close to cookies concept. The main intesion of web storage is, store the data at client side without disturbing the performance of the website.

In HTML5 web storage is classified into the following two types:

1] session storage
2] Local storage

1] session storage : It is also use browsers data locally but it is for limited period when we close the browser it will automatically delete the stored data and we cant see the browsers stored data again. HTML5 introduces the session storage attribute which would be used by the sites to add data to the session storage.

ex:
```
<head>
<script type='text/ javascript'>
  var sno=100;
  alert(" The Entered Number is :" +sno);
</script>
</head>
<body>
<a href = "page2.html" > page2 </a>
</body>
```

o/p is 100, page2 (link).

ex:
```
<head>
<script>
alert("hi");
alert(sno);
</script>
</head>
```

o/p: hi, NO msg

ex3 : with session storage

```
<head>
<script>
sessionStorage.sno=100;
alert(SessionStorage.sno).
</script>
</head>
<body>
<a href="page2.html"> page2 </a>
</body>
```

o/p : 100 , page2.

ex:
```
<head>
<script type='text/javascript'>
alert("hi");
alert(session storage.sno);
</script>
</head>
```

o/p is Hi, 100.

Local storage : which is used to store the data locally. It stores the data with no expiration date, means if browser is closed then it will not delete the data and we can view the data any time and even after year.

ex1 :
```
<head>
<script>
localstorage.sno =100
alert (local storage.sno);
</script>
</head>
<body>
<a href ="page2.html" > page2 </a>
</body>
```

o/p : 100 , page2 (link)

ex2:

```
<head>
<script type = "text/javascript>
  alert ("hi");
  alert (local storage.sno)
  </script>
  </head>
```

o/p is hi, 100.

Advanced Example for session storage:

```
<body>
<script type= 'text/javascript'>
 if(session storage.hits)
  {
   Sessionstorage.hits = Number(session storage.hits)+1 ;
  }
else
 {
  sessionStorage.hits = 1;
 }
 document.write ("Total Hits" + session Storage.hits);
  </script>
  <p> Refresh the page to increase the number of Hits </p>
  </body>
```

ex2:

```
<body>
  <script>
  if (session Storage.hits)
  {   local
   local
   sessionStorage.hits = Number (local session Storage.hits)+1;
  }
 else
  {
   session localstorage.hits = 1;
  }
  document.write ("Total hits" + local storage.hits );
  </script>
  <p> Refresh the page to increase the number of Hits </p>
  </body>
```

In web storage storing sensitive data on a local machine could be dangerous and could leave a security whole. A session storage data deleted automatically when you close the browser where as local storage unable to that time we should clear the local storage with the help of local storage.clear() method.

Define web SQL ?

web SQL Database is web page API for storing data in databases that can be queried as a using a variant of SQL. The API is supported by Google Chrome, opera, safari and the Android Browser.

Indexed DB : It is basically a persistant data store in the browser- a database on the client side. Like a regular relational databases it maintains indexes over the records it stores and developers use the IndexedDB Java script API to locate records by key or by looking up an index.

HTML5 Geolocation (Deeper Integration with OS)

HTML5 Geolocation API allows users to share their location with web applications by capturing the approximate longitude and latitude coordinates of the user with their permission.

Define latitude : latitude is a geographic coordinate that specifies the north-south position of a point on the Earths surface.

Longitude : It is a geographic co-ordinate that specifies the east west position of a point on the Earths surface.

Types of maps

maps are classified into following basic types

1) ROADMAP (normal, default 2D map)

2) SATELITE (photographic map)

3) HYBRID (photographic map + Roads & city names)

4) TERRAIN (map with mountains, rivers etc.)

Google maps controls

Google map supports the following list of controls.

1] zoom
2] Pan
3] map Type
4] Street View
5] Scale
6] Rotate

@] write a script to verify the Browser support.

```
<head>
 <script>
  function my support()
  {
    if(navigator.geolocation)
     {
       alert("your Browser supports");
     }
   else
     {
       alert("your Browser unable to support");
     }
    }
  </script>
   </head>
  <body>
   <p> click the button to display the Browser </p>
  <button onclick =" my support ()"> support_Browser </button>
     </body>
```

Geolocation Methods

getbootstrap.com

Method	Description
getCurrentPosition()	It retrieves the current geographic location of the user.
WatchPosition()	It retrieves periodic updates about the current geographic location of the device.
clearWatch()	This method cancels an ongoing watch position call.

7-9-15

Working with HTML5 Drag and Drop

Define Drag and Drop

It is powerful user interface concept which makes it easy to copy, reorder and deletion of items with the help of mouse clicks. It allows the user to click and hold the mouse button down over an element, drag it to another location, release the mouse button.

Creating draggable content

Making an object draggable is simple. set draggable = true attribute on the element you want to make moveable.

Example :

```
<body>
    <img src="html5.png" width="100px" height="100px"
      draggable="true">
</body>
```

Drag and drop Events

Drag and Drop supports the following list of events.

Events	
dragstar	drop
dragenter	dragend
dragover	
dragleave	
drag	

allow Drop : By default all elements are not dragged or dropped. To allow a drop we must prevent the default handling of the element

Ex:
```
function allowDrop(ev)
{
 ev.preventDefault();
}
```

What to Drag : The dataTransfer.setData() method sets the data type and the value of the dragged data.

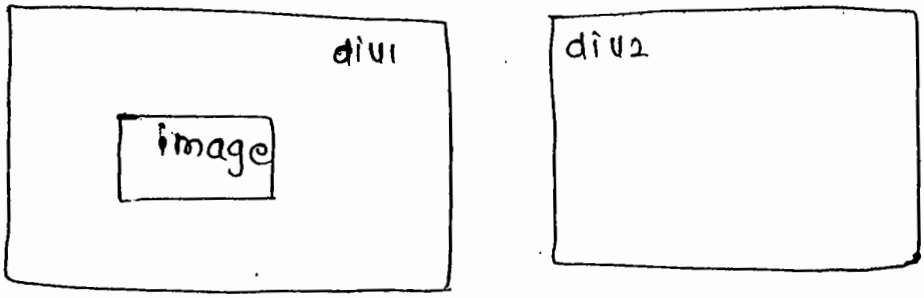
Ex:
```
function drag(ev)
{
 ev.dataTransfer.setData("text", ev.target.id);
}
```

Do the Drop : when the drag data is dropped the drop event occures. This event fires as follows:

```
function drop(ev)
{
 ev.preventDefault();
 var data = ev.dataTransfer.getData("text");
 ev.target.appendChild
 (document.getElementById(data));
}
```

```
<head>
<style>
 #div1
  {
    width: 350px; height: 70px;
     border: 1px solid #FF00FF;
  }
 </style>
<script type="text/javascript">
 function allowDrop(ev)
 {
   ev.preventDefault();
  }
 function drag(ev)
 {
  ev.dataTransfer.setData("text/html", ev.target.id);
 }
 function drop(ev)
 {
   ev.preventDefault();
   var data= ev.dataTransfer.getData("text/html");
   ev.target.appendchild(document.getElementById(data));
  }
 </script>
  </head>
  <body>
    <div id = "div1" ondrop ="drop(event)" ondragover="allowDrop
        (event)"> </div>
     <br>
    <img id = "drag1" src = "water.gif" draggable ="true"
        ondragstart = "drag(event)" width ="336" height ="69" >
   </body>
```

drag and drop between diuisions



ex:

```
<head>
 <style>
#diu1,#diu2
 {
   width : 100px; height : 95 px;
   border: 1px solid #FF00FF;
 }
</style>
<script type = "text/javascript">
 function allow Drop(ev)
 {
   ev. prevent Default ( );
 }
 function Drag (ev)
 {
   ev. dataTransfer. setData ( "text/html" , ev. target. id );
 }
function drop (ev)
 {
```

```
<body>
<div id="div1" ondrop="drop(event)" ondragover="allowdrop
    (event)">
 <img id="drag1" src="html5.png" draggable="true"
   ondragstart="drag(event)" width="100px" height="50px">
  </div>
<div id="div2" ondrop="drop(event)" ondragover="allowdrop
    (event)"></div>
</body>
```

Task 1 → Gmail login

Task 2 → Facebook Registration

Task 3 → Naukri registration

Task 4 → Gmail Registration page

Tas 5 → Google & Bing search page

Ksubbaraju.tm@gmail.com

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
1] Weakly typed programming or loosely typed programming or lightweight	1] Strictly typed programming
2] Easy to understand	2] Complex to understand.
3] Simple to develop	3] Trouble to develop
4] no headers files required	4] headers files mandatory.
5] no libraries require	5] libraries compulsory
6] no special compiler require	6] special compiler mandatory.
7] Client side validation	7] server / client side validation / verification.
8] poor graphics	8] rich graphics.
9] 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 :-
```
<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 (exp1, exp2, exp3, ...)

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 javoscript 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 : corriage 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 & documents.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
```

```
documents.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 javascript");
document.write("<br>");
document.write('Livescript is java script');
</script>
</head>
```

* Javascript semicolon (;)

In javascript 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')
</scipt>
</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 :-

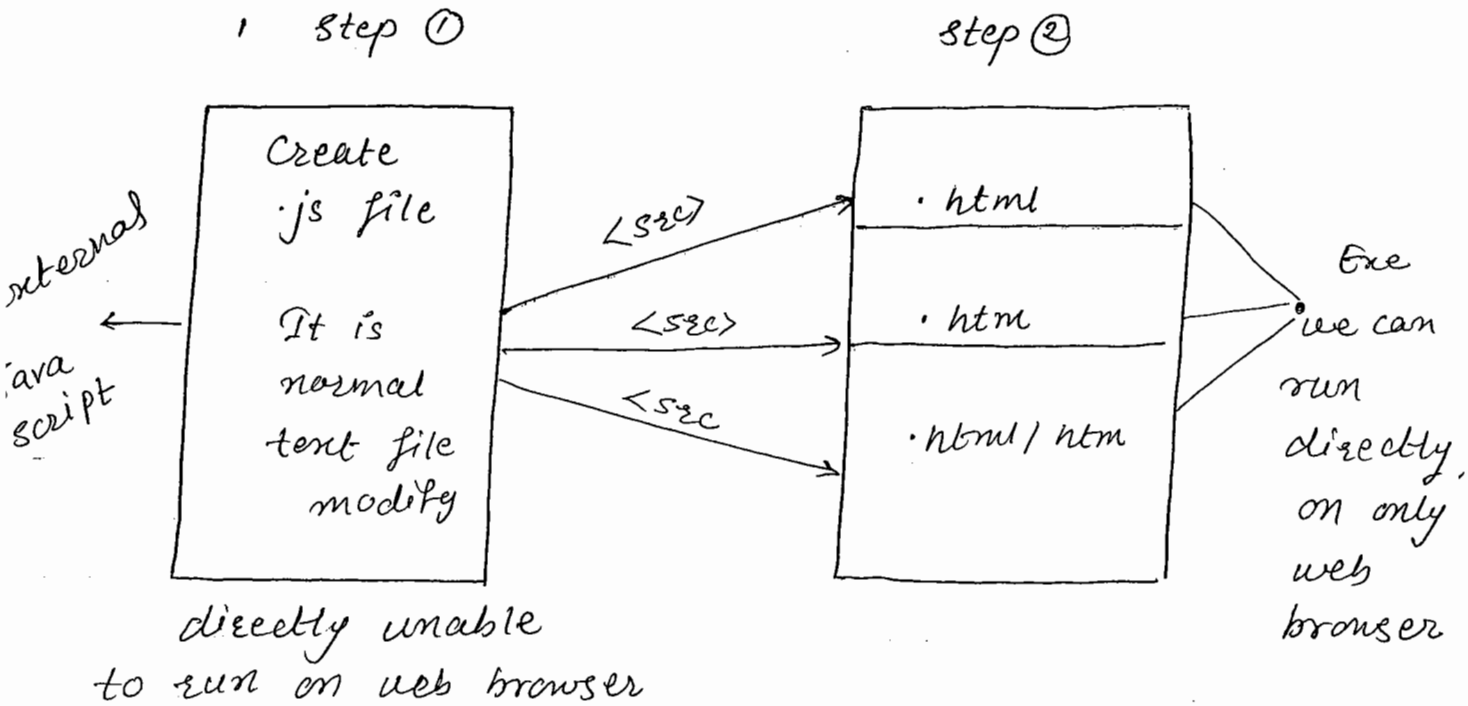
Javascript can also be placed in a external files, these files contains javascript code, this code we can apply on diff. webpages. External javascript files extensions is '.JS'

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 :-



directly unable to run on web browser

* 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.

1) 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 ("Invalide Entery");
</script>
</body>
```

* How to display mulliple line on the alert.

We cannot the use
 tag here because alert is a method of the windows object, that cannot be interpret HTML tag.

Instead we use the new line escape character.

```
<head>
<script type = "text/javascript'>
 alert ("Javascript \nis a \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 </but
</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 Canele");
 alert (" User Seleeted Option is : " +x);
 </script>
 </head>
```

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

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 if (myval <= 100)
{
alert ("User Entered Value is Small or Equal");
{ </script>
</head>
```

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 myConfirm()
  {
```

```
    confirm ("Click OK or Cancel");
}
    function MyPrompt()
{
    prompt ("Enter any Value:");
}
```

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

Step 2:- Preparing html file.
required.

```
<html>
<head>
<script type = "text/javascript" src = "myscript.js"
</script>
</head>
<body>
<p> Click the button to display alert Message..</p
<button onclick = "MyAlert()"> Alert </button>
<P> Click the button to display Confirm
Message..</P>
<button onclick = "MyConfirm()"> Confirm </button
<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) keyboards should not be used as variables
(6) In Java script every variable should starts with var keywords

In software environment 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 declarabion.

Note : Explicit detlarabion is allways recommanded 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] Steing
- 2] Number
- 3] Boolean
- 4] Undefined
- 5] Null

2] Non-premitive datatypes :-

These are popularly known as referenee 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 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 that 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 declared 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 constraints 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] Comparison operators
3] Logical operators etc

* Arithmetical operators

Using these operators we can perform the arithmabical 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

* Comparison 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 statements 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 statement, 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 Statement.
}
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 condition 1 is true.
 }
 else if (condition 2)
 {
  code to be executed if condition 2 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, CSS3, 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, lables 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 repetatedly 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 = 'text/javascript'>
for (i=1 ; i<=10 ; i++)
  {
   document.write ("The value is : " +i);
  document.write ("The value is : "+i);
  }
  </script>
  </head>
```

ex,

```
<head>
<script type = 'text/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 statements 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 statements

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:
" + MyReturn (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 or

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 background color of page, It is a property inside document object.

Syntax :-

document . bgcolor = " 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 frequenbly 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
mouseover	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

11 On Click events

This events fires when the user clicked on a element

Syntax in 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> , < br>, <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</button>
```

2] On double click

This events fires when the user double click on the element

```
In HTML
<element ondblclick = "SomeJavaScriptCode">

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 = "Click Me"
      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 ("ImageLoadedSuecessfully");
 }
  </script>
  </head>
  <body>
  <P> Refresh the page and Observe...</P>
  <img src = "html5.png" width = 150px height
  = 150 px onload = "ImgLoad()" alt = "sorryNoImg
  </body>
```

Ex 43 On error

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

Syntax In HTML:

```
<element onerror = "some JavaScript Code">
```

In JavaScript:

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

Example

```
<head>
<script type = 'text / javascript'>
  function ImgErr()
  {
    alert (" Image Fail to Load");
  }
  </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 cursor to an element

Syntax:

In HTML:

```
<element onmouseover = " some JavaScript Code ">
```

In JavaScript:

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

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 ① :-

```
<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] 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 :- object.onblur = "Some Javascript Code"

Non Supported HTML events :- onblur event unable to support the following HTML elements

<base>, <bdo>,
, <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 gets a focus it is most often used in,
→ It is most often used in input, 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>,
, <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 JavascriptCode"
```

Example :-

```
<head>
<script type = 'text/javascript'>
function myPage()
{
 alert ("Sorry Page Resized");
}
 </script>
</head>
<body onresize = "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 divideed into. a) Syntax error
b) Runtime error
c) Logical error.

Java Script supports following three types of errors :-

Syntax Error :- It 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");
alerts ("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 to 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; ]
  }
   finally
  {
   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 objeet.

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 ("Goodmorning")+
    "<br/>");
  document.write (isfinite (423) + "<br/>");
  </script>                          ↓ true
  </body>
```

isNaN() 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'>
document.write (isNan ("Good morning") "</br>");
document.write (isNan ("2009/10/15") "</br>");
document.write (isNan (455) "</br>");
</script>
</body>
```

true
true
false

parseInt() function

It parse a string and returns an integer

Syntax parseInt (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>
```

→ 200

```
<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;
var 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'>
var str = "Nareshitechnologies";
document.write("The length of the string is :" +str.
length);
document.write("<br/>");
```

```
document.write("The string in Uppercase: "+str.toUpperCase())
document.write("<br/>");
document.write("The string in Lower case: "+str.toLowerCase());
</script>
</body>
```

Common javascript 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...!! </P>
<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
availHeight	Returns the height of the screen (excluding the windows task bar)
availWidth	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>
```

7] 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 baekword history ... !! <IP>
<Button onclick = "HisBack()"> His_Back </button>
<P> Click the button to transfer te 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 objeet. 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 spaee

output :- URL = http://2: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 documents 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 properby.

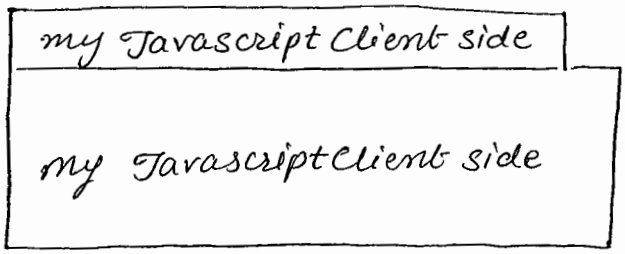
1] Document title property :-

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 :-



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 Elements 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.getElementById ("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").innerHTML = 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 = "myRev()"> 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();

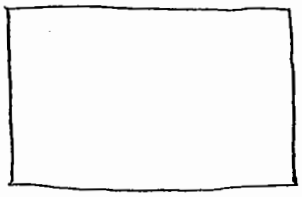
shift() :- This method remove the array elements from left to right that means 1st to last.

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.innerHTML = 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', 'nature2.pg']
   i=0;
  function fun1()
   {
```

```
i++
if (i==5)
{
  alert ("No more images")
}
  else
  {
    document.getElementById ("img1").src = "img/"
    + arr[i];
  }
  }
  </script>
  <body>
  <image src = "img/fish.jpg" width = "300" height
   = "250" id = "img1">
  <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 myBoolean = new Boolean
```

Boolean object properties

Note :. Except 1 remaining all values returns false.

Example 1

```
<head>
<script type = 'text/javascript'>
  var 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'>
var 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. var 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'>
var 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.getElementsById("yr");
x.innerHTML = 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 (yy) format-

e.g :-

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

```
var currentTime = new Date()
var month = currentTime.getMonth() + 1
var day = currentTime.getDate()
var year = currentTime.getFullYear()
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 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("funI()", 1000);
 function funI()
 {
   var d = new Date
    str = d.getHours()+":"+d.getMinutes()+":"+d.
      getSeconds() document.getElementById('spI').
        innerHTML = str;
 }
  </script>
 </head>
 <body>
 <span id = "spI" 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 charaters.

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()
 {
```

```
var str = "Naresh i Technology";
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 = "myfun()"> 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.toUpperCase());
document.write("<br/>");
document.write(str.toLowerCase());
document.write("<br/>");
document.write(str.charAt(5));
</script>
</body>
```

* 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 radius 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.

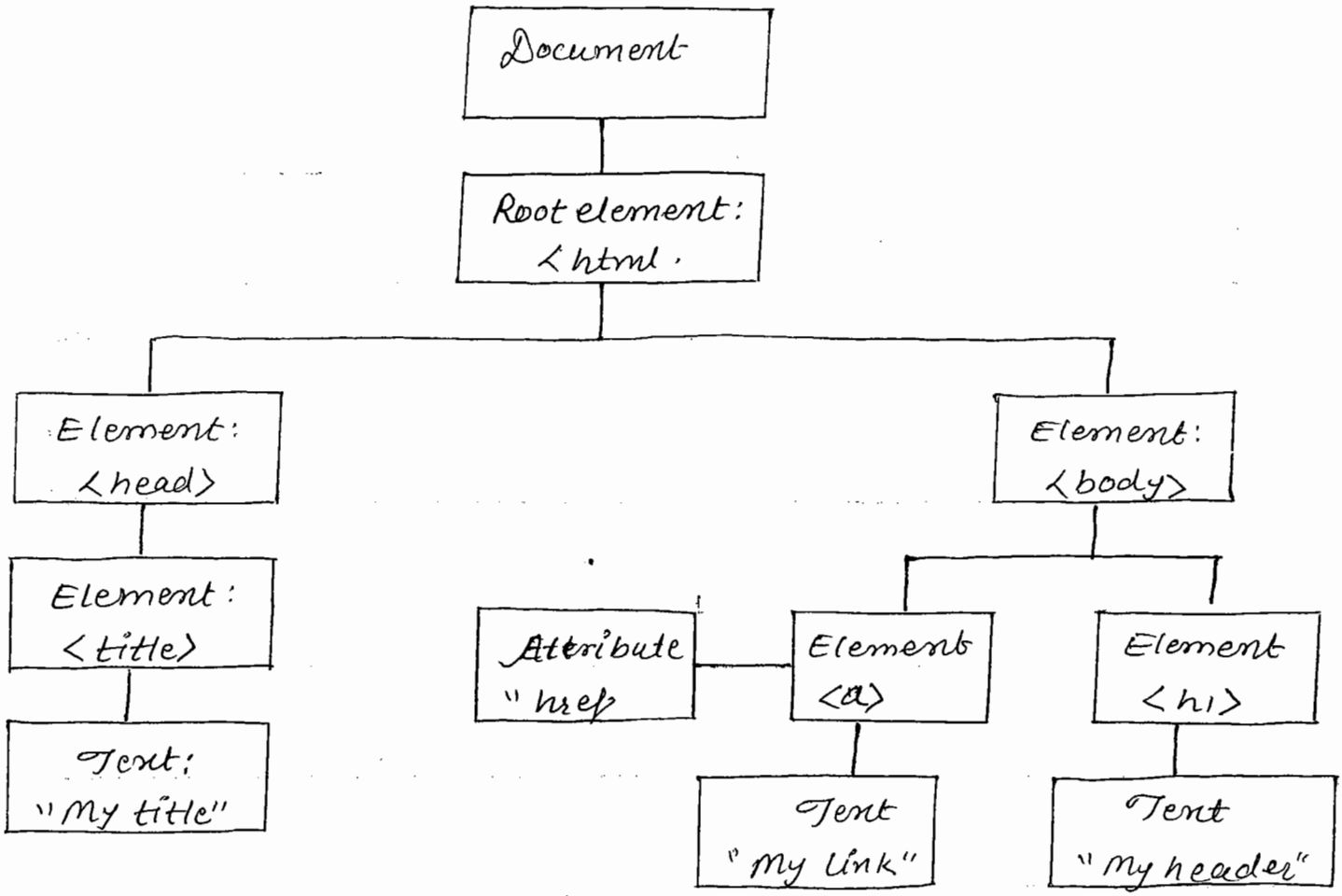
* Metacharacters — A metacharacter is simply an alphabetical character preceded by a backslash.

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 neutral 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. Every 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.getElement-ById ("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 }').
  innerHTML = '{ 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 :- Textbox, 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.getElementById('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 ...<IP>
  <input type = 'text' value = " javascript " id =
  "txt 1"> <br/>
 <button onclick = "mytext()"> Click me </button>
  <body>
```

Example 2.

```
  <head>
 <
 fun
 {
   var value = document.getElementById('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=100px
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 contains 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 field that required it. This would need just loop through each field in the form and check for data.

(2)

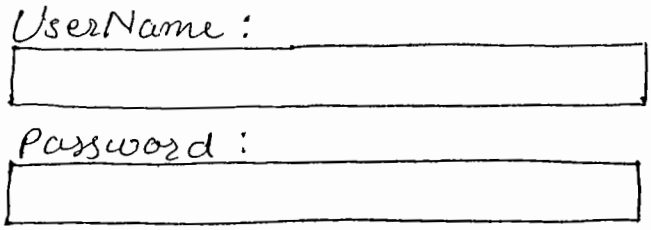
The data that is entered must be checked for correct form and value. This would need to put more logic to test correctness of data.

Validating TextBox

| From validate |

```
<head>
<script type = 'text/javascript'>
 function notEmpty()
 {
   var myTextField = document.getElementById
   ('myText');
   if (myTextField.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 :-




```
<head>
<script type = 'text/javascript'>
 function funchklen (len , cid)
 {
   if (len<6)
    {
     document.getElementById (cid).style. border
     color = "red"
    }
    else
    {
     document.getElementById
    (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 = "txt 2"
  onblur = " funchklen (this.value.length , 'txt 2')">
   </body>
```

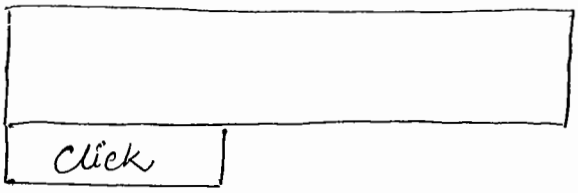
Validating Radio Buttons :-

You are ?

○ male ◉ female

[submit]

```
<head>
<script type = "text/javascript">
  function validate() {
   var r1 = document.getElementById('male').
     checked;
   var r2 = document.getElementById('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 = 'password' 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.