WHAT IS PRGRAMMING LANGUAGE?

A **programming language** is a formal computer **language** or constructed **language** designed to communicate instructions to a machine, particularly a computer. **Programming languages** can be used to create programs to control the behavior of a machine or to express algorithms.

In simple words programming language used to create enterprise applications or softwares.

WHAT IS ENTERPRISE?

Enterprise is a business or company which is producing outcome(products). To produce products business units will fallow certain manual business rules like purchases, row meterial, stocks ect.

WHAT IS APPLICATION OR SOFTWARE?

Automation of manual business rules using a programming language or group of programming languages.

TYPES OF APPLICATIONS OR SOFTWARES

We can create a application or software in fallowing flavours:

1) Desktop : The applictions which are installable in local systems are called desktop applications. Eg: skype, microsoft office, notepad ++ etc..
2) Mobile : The applications which are installable in mobiles(smart phone) or tablets downloaded from playstore for android and apple store for ios.
3) Web : The applications which are deployable in any server and can be accessible from any location using browser.

WHAT IS WEB APPLICATION?

In computing, a **web application** or **web app** is a client–server software **application** which the client (or user interface) runs in a **web** browser. Collection of electronic pages is called web.

A web application typically contains fallowing three layers:

Presentation layer : is a user interface(views) which are accessable from any web browser.

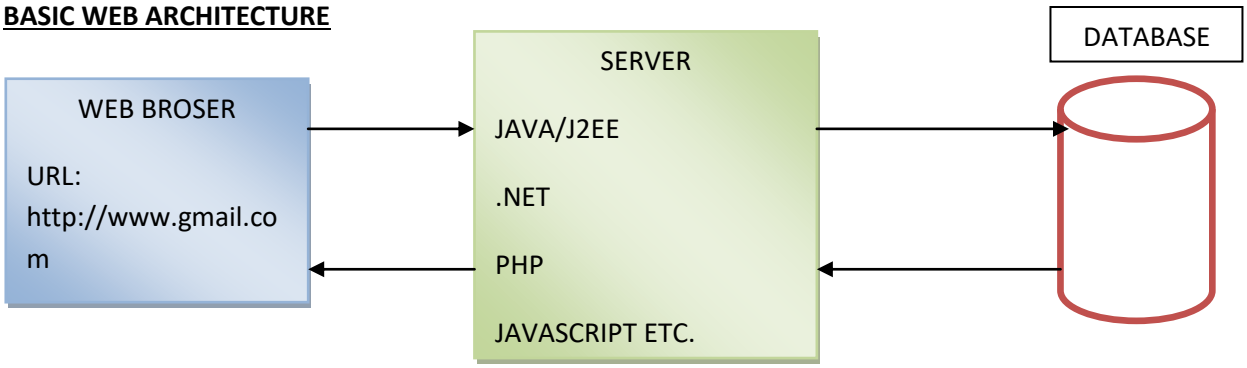
Business layer : is a server side program which are nothing but automation of business rules. Client layer will interact with business layer to persist data.

Data layer : is a database software where we can store client related data. Business layer will interact with data layer.

What is web browser?

It is a client side lightweight software. It takes http request from client to server. It renders(brings) http response from server to client. We have famous web browsers in market: Google Crome(2008), Mozilla Firefox(1998), Internet Explorer(1995), Mino Opera(1995) & Safari(2005)

BASIC WEB ARCHITECTURE



Web applications are network enable applications. We can deploy any web applications in servers adn we can access them over network using server ip address and application name. Inverntor of world wide web is Sir. **Thimothy John Berners-Lee**.

Collection of computers interlinked together is called network. Network is invented by **Leonard Kleinrock**. First network name is ARPANET(Advanced Research Projects Agency Network). First protocol in IT industry is FTP(File Trasfer Protocol).

A network can be accessable over internet. Internet is invented by **Vinton gray**. Data can be shared over network within local(domain) and internet outside of dmain. Eg: email.

Email: Electronic mail services. It is a free service to communicate with other internet users. Email is invented by **Shabeer Bhatia.** Sabeer Bhatia is an Indian entrepreneur who founded the webmail company Hotmail.com.

SMTP: Simple Mail Transfer Protocol. It takes care of deliverying emails from one server to another.

MIME: Multipurpose Internet Mail Extensions. It exchange different kinds of data.

Blog: It is daily updting website or webpage. Every post displayed in reverse cronological order.

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

Http: It is a transfer protocol to exchange hypertext documents in the world wide web.

Http(s): Secured transfer protocol to exchange hypertext documents with the help of SSL(ciphertext).

Ciphertext is encrypted text. Plaintext is what you have before encryption, and **ciphertext** is the encrypted result. The term cipher is sometimes used as a synonym for **ciphertext**, but it more properly **means** the method of encryption rather than the result.

W3C: World Wide Web Consortium: It was founded in 1994 by Sir. Thimothy John Berners-Lee. It has 400Plus members pertnership. Well known memebers are Facebook, Google, Nokia, Samsung, IBM, Microsoft etc..

HOW MANY TYPES OF WEB APPLICATIONS WE HAVE?

A webpage is an electronic page developed on HTML. It is classified into two types.

Static webpage : A user unable to interact directly with these webpages. Eg: HTML, CSS

Dynamic webpage : End-user can able to interact directly with these webpages. Eg: HTML, CSS & Javascript

Collection of webpages or web documents are called web application(website). These are classified into tow types:

STATIC WEB APPS : The applications which can't able to handle business logic are known as static web apps. Static apps will contain only client layer. We can develop static web applications using HTML. To provide look and feel to these static pages we can use CSS. To handle client layer business logic we ca use Javascript. We can't able to maintain end user interaction(state) using static web apps.

DYNAMIC WEB APPS : The applications which can able to handle business logic are known as dynamic web apps. These type of apps contains at least 2 layers client and business. If we need to store client data then these application contains data layer too. We can develop client layer using HTML, CSS & javascript. Business layer using any one of the server programming language like .NET, JAVA/J2EE & PHP. We can store end user data using any database like mongo db, MS-SQL, MySql, Oracle etc.

What is HTML?

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

- It is not case sensitive.
- It is browsers mother language.
- It is global language.
- It is collection of tags and elements.
- We can create static webpages
- It is error free english language.
- It is a markup language(special text).

HTML history

1989 => GML: Generalised Markup Language
1991 => SGML: Standard Generalised Markup Language
1994 => HTML: Hypertext Markup Language
1998 => XHTML: Extensible Markup Language With HTML
2008 => HTML5: Advanced Markup Language For Mobiles
2014 => HTML5.1: Advanced Markup Language For Small Electronic Devices

HTML Versions

From W3C organization there are fallowing versions released.

HTML 1.0 : 1994
HTML 2.0 : 1995
HTML 3.0 : 1997
HTML 4.0 : 1999
HTML 5.0 : 2008
HTML 5.1 : 2014

What is tag?

It is special kind of text placed between left angular brace and right angular brace(<tag_name>).

Syntax : <html>example</html>

We have fallowing types of tags:

Paired tags: These tags are having opening and closing tags.

Eg: <html>html code</html>, <head>meta data</head>

Non paired tags: The tag that have only opening tags but no closing tags:

Eg:
 break, <hr> horizontal rule, <img> image

Structure of HTML

As per W3C standard HTML document has the following detaile structure.

```
<html>
        <head>
                <title>application title</title>
        </head>
        <body>
                User interface implementation
        </body>
</html>
```

In html document the fallowing elements are critical or common elements:
Html, head,title & 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
3) Save with .htm or .html extension
4) Right click on the saved file open with any web browser
5) If you need to update any text or logic, right click on the saved file open with notepad format
6) Do require changes and save it. Refresh the webpage

HTML5 comments

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

HTML5 supportscomment notation for single and multiline. Comments begins with <!—text -->

Note: Comments are applicable in the title part.

Parts in html documents

Generally every html document contains the following parts.

- Version information
- Head section
- Body section

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 language of your webpage. It contains two letter country code and language code.

Eg:

English(US) <html lang=”en-us”>

English(UK) <html lang=”en-uk”>

English(IN) <html lang=”en-in”>

Note: If language not declared default language is US.

Head section:

This section supports the following list of tags

Title: It defines title of the webpage it is paired tag.

Syntax: <title>my first application</title>

Link: It defines the relationship between a document and on external resource it is a non paired tag.

Syntax: <link>

Example

A browser supports favicons and displays in the browser url bar. The following image extentions are prefered: .png(pointable network graphics), .gif(graphical interchange format), .ico(icon format)

These images must be 16x16 or 32x32 pixels.

- <link rel="icon" href="favicon.png" sizes="16x16" type="image/png"> (1 size)
- <link rel="icon" href="favicon.png" sizes="16x16 32x32" type="image/png"> (2 sizes)
- <link rel="icon" href="icon.svg" sizes="any" type="image/svg+xml"> (any size)

Example

```
<!DOCTYPE html>
<html>
```

```
    <head>
        <title>my html5 logo</title>
        <link href="html5.png" rel="icon" size="16x16" type="image/png">
    </head>
</html>
```

HTML5 meta element: meta means after. Metadata means data about data. Meta elements are classified into different categories. The following are frequently used:
Meta charset: UTF stands for unicode transformation format. It is used to develop a webpage of different languages like arebic, chinese ect..

Syntex:

HTML4: <meta http-equiv="content-type" content="text/html; charset=utf-8">
HTML5: <meta charset="utf-8">

Unicode: It provides a unique number for every charactor. As per W3C standard unicode indicates the following information.

- No matter what the platform
- No matter what the program
- No matter what the language

Example

```
<head>
    <title>Meta charset example</title>
    <meta charset="utf-8">
</head>
<body>
    <h1>Hello how are you?</h1>
    <h1>మీరు ఏలా ఉన్నారు ?</h1>
</body>
```

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

Syntax: <meta name="keywords" content="required list of keyworkds with , seperator">

Eg: <meta name="keywords" content="live cricket scores, live cricket stream, cricket">

Meta description: It defines the description of your page. In case of empty descriptions, search engines will generate a description from the content of the page.

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

Eg: <meta name="description" content="Ui technologies website">

Meta title: It defines the title of meta content.

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

Eg: <meta name="title" content="Welcome to naresh IT Hyderabad">

Meta author: It defines the author of the webpage.

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

Eg: <meta name="author" content="vikram, naidu">

Style tag

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

Syntax:

```
<head>
        <style type=”text/css”>
                        /*html document syle code*/
        </style>
</head>
```

Script tag

It is used to define client side script such as javascript.

Syntex:

```
<head>
        <script type=”text/javascript”>
                //javascript code
        </script>
</head>
```

Body section

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

Systex: <body><!-- body content --></body>

Attributes:

Background: Specifies a background image for a document

Bgcolor: Specifies background color for a document

Text: Specifies the color of a text for a document

Hexadecimal color code system

As per W3C standard hexadecimal color system, HTML5 colors are define using hexadecimal notation. It is six digit representation of a color. It is the combination of RGB.

With HTML, RGB values can also be specified using hexadecimal color values in the form: #RRGGBB, where RR (red), GG (green) and BB (blue) are hexadecimal values between 00 and FF (same as decimal 0-255).

For example, #FF0000 is displayed as red, because red is set to its highest value (FF) and the others are set to the lowest value (00).

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: <any attribute="parameter">

Eg: <body background-color="blue">

Types of attributes

In HTML5 attributes are cassified into following types.

Element specific: These attributes are exclusivel for a specific element of tag.

Eg: <body> => bgcolor, background, text

<img> => src, width, height, alt

Global: These attributes we can use with any html element.
Eg: style, id, class ect..

Event: These attributes are related to javascript.
Eg: onclick, oninput, onchange, onfocus ect..

Optional: These attributes are used to provide additional infomation and not mandatory in HTML5.
Eg:
HTML4: <link rel="stylesheet" type="text/css" href="css file url">
HTML5: <link rel="stylesheet" href="css file url">

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

example: © ® ™ ← → ♠ ♥

HTML4 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) <isindex>
11) <noframes>
12) <s>
13) <strike>
14) <tt>
15) <u>
16) <xmp>

1) **<acronym>**: it defines abbrevation or full form from initial letters. It is paired tag.
syntax: <acronym> </acronym>

```
<html>
    <body>
```

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

<abbr>(New): It indicates an abbrevation or an acronym. It is paired tag.

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

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

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

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

```
    syntax: <object> .....</object>
Eg:
<body>
    <object data="windows xp shutdown.wall">
        <param name="autoplay" value="true"/>
    </object>
</body>
```

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

```
    syntax: <basefont>
Note: It supports only internet explorer webbrowser.
<head>
    <basefont color="blue" size="5" face="tahoma"/>
</head>
<body> <p> This is a paragraph </p> </body>

Example2 with css:
<head>
    <style type='text/css'>
        div{
            font-size : 20px;
            font-family: tahoma;
            color:blue;
            font-weight:bold;
        }
    </style>
</head>
<body>
    <div> welcome to css font properties .....</div>
</body>
```

4)<big>: Itis used to make text bigger. It is paired tag

syntax: <big>.....</big>

```
<body>
        <big> welcome to big tag </big>
</body>
```

Example 3 with css:

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

5) ** :** It display the text in page center.It is paired tag.

syntax: .....

ex:

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

Example (css) :

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

6)**<dir>:** It is used to list titles. It is paired tag.

syntax: <dir> ..... </dir>

example 1:

```
<body>
        <dir>
                <li> HTML5 </li>
                <li> css </li>
        </dir>
</body>
```

example 2:

```
<body>
	<ul type="square">
		<li>HTML5</li>
		<li>css</li>
	</ul>
</body>
```

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

syntax: ....

eg 1:

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

eg 2(css):

```
<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>
```

8)**<frameset>**: Using this tag we can divided the web page as mulpiple frames.It is paired tag.

syntax: <frameset> .... </frameset>

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

syntax: <frame>

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

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

Eg1:

```
<frameset rows="50%,50%" cols="40%,*">
	<frame src="http://www.NareshIt.com">
	<frame src="http://www.NareshIt.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	**paremeters**
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

Eg:

```
<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>

eg:

```
<body>
        <s> It is remove 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 paired tag.

syntax: <tt>.....</tt>

Eg:

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

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

syntax: <u> .... </u>

eg1:

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

eg2:

```
<head>
	<style>
		div{
			text-decoration:underline;
		}
	</style>
</head>
<body>
	<div> welcome to css</div>
</body>
```

<IsIndex>: It is used to display single line text input control.It is nonpaired 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>

eg:

```
<body>
	<xmp>
		1 2 3 4
		5
	</xmp>
	<pre>
		1 2 3 4
		5
	</pre>
</body>
```

Features of HTML5

It is the latest version developed by w3c and **Web Hypertext Application Technology Working Group(**WHATWG). It is has the following list of unique features.

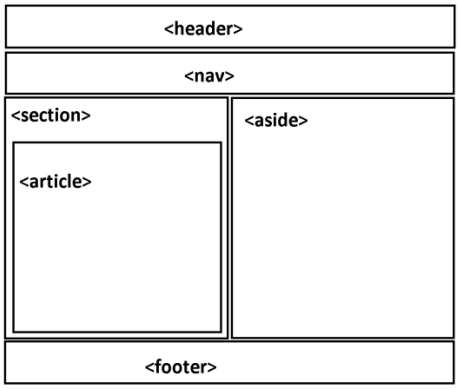
- HTML5 symantic elements
- HTML inline elements
- advanced forms
- Advanced form input types
- Latest form attributes
- Updated input attributes
- New form elements
- HTML5 multimedia
- HTML5 web RTC(real time communication)
- HTML5 graphics (2D and 3D)
- HTML5 canvas
- HTML5 SVG (scalable vector graphics)
- MathML (Mathematical markup language)
- HTML5 Geolocation
- HTML5 drag and drop
- HTML5 web storage
- server sent events
- HTML5 web workers
- HTML5 application cache
- HTML5 speech input
- performance and integration

HTML4 page layouts

In html4 page layouts are as follows

HTML5 page layouts

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



HTML5 structure

HTML5 has the following detailed document structure

```
<!doctype html>
<html lang="en-us">
        <head>
                <meta charset="UTF-8"/>
                <title> page title </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>

semantics types:

1) **General semantics**:

section: It defines section in a document such as chapters,header,footers any other section of the document.It is a paired tag. No element specific attributes.

syntax: <section> ..... </section>

header: It is used to display subheadings,version information, navigational controls etc.It is a paired tag. No element specific attributes.

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

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

footer: It is used to define footer of an html document or section.It is a paired tag. No element specific attributes.

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

article: It is used to represent an article.It ia an independant content on the webpage.It is paired tag. No element specific attributes.

syntax: <article> ... </article>

Architecture for semantics

```
<section>                    <section>
<header>...</header>         <header></header>
<article>                    <article>
<p> ... </p>                 <header></header>
<p> ... </p>                 <p></p>
</article>                   <p></p>
<footer>...</footer>         <footer></footer>
</section>                   </article>
                             <footer></footer>
                             </section>
```

eg:

```
<!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="auther" 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 &copy;</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. No element specifc attributes.

syntax: <hgroup> ... </hgroup>

eg1:

```
<body>
        <hgroup>
                <h1> power of html5...</h1>
                <h2> Introduction to web...</h2>
        </hgroup>
        <p>  </p>
        <p>  </p>
</body>
```

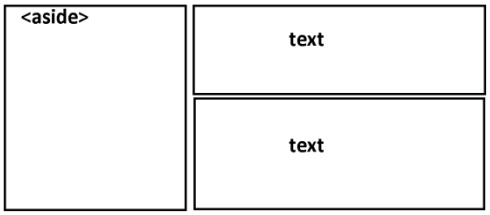
aside: It is used to represent content that is related to surrounding content within an article or web page.It is paired tag.

syntax: <aside> ... </aside>

eg1:

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

eg2:




```
<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>
```

eg: aside with images

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

```
                <img src="goodmorning.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="simly.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>

Eg:

```
<body>
        <button onlick="alert('welcome to events')">click me</button>
        <command onlick="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	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. No element specific attributes.

syntax:<summary> ... </summary>

Note: The <summary> tag is currently only supported in chrome & opera.

The <summary> element should be first chld element of <details> element.
widget/gadget for interactive (show/hide) power of html.

Eg:

```
<body>
        <details open="open">
                <summary> power of html5 </summary>
                <p>file system APIS</p>
                <p>geolocation</p>
                <p>semantics</p>
        </details>
</body>
```

eg2:

```
<body>
        <details>
                <summary style='color:#000FF'>power of html5!! </summary>
                <ul type='square' style='clor: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. No element specific attributes.

syntax: <figure> ... </figure>

figcaption: It defines a caption for figure element. This element represents a caption or a legend for a figure.It is paired tag. No element specific attributes.

syntax:<figcaption> ... <figcaption>

Eg:

```
<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 paired tag. No element specific attributes.

syntax: <nav> ... </nav>

Eg:

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

wbr: It is used to word break opportunity. It is a nonpaired tag. No element specific attributes.

syntax: <wbr>

Eg:

```
<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 paired tag.

syntax: <bdo> ... </bdo>

attributes:

attribute	value	Description
dir	ltr	Specifies the text direction
	rtl	specifies the text direction.

Eg:

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

</body>

bdi: Biderectional isolation. This can be useful when displayin right to left text.It is a paired tag. No element specific attributes.

syntax:<bdi> ... </bdi>

```
<body>
        <ul>          |
                <li>user<bdi> suresh </bdi></li>
                <li>user<bdi> daniel </bdi></li>
        </ul>
</body>
```

*Chinese Semantics:

ruby: It used for specific ruby annotations. Which is used in past asian typography. It is a paired tag. No element specific attributes.

syntax: <ruby> ... </ruby>

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. No element specific attributes.

syntax: <rt> ... </rt>

rp: The HTML <rp> is used in ruby annotations for the benefit of browsers tht don't support ruby annotations. It is a paired tag. No element specific attributes.

syntax: <rp> ... </rp>

Eg:

```
<html>
        <body>
                <p>漢 <rt> 厂马` </rt></p>
        </body>
</html>
```

<u>HTML5 Inline Elments</u>

HTML5 support the following inline elements

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

mark: It is used for indicating text as marked or highlighted for refrence purposes.It is a paired tag. No element specific attributes.

syntax: <mark> ...</mark>

Eg:

```
<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'>
</body>
```

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	specifies one or more forms.
high	number	specifies the range that considered to be high.
low	number	specifies 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.
optimun	number	what value is the optimum value for gauge.
value	number	specifies the current value for the gauge.

Eg:

```
<body>
        <meter value="2" max="10">
                <p style='color:red'>oops your browser not supporting meter tag</p>
        </meter>
        <br/>
        <meter value="8" 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>
        <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.

Syntax: <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">
                <p 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

eg:

```
<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>

Attributes:

form attributes	parameters
Attributes	any name
method	get, post
name	formName
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 field set
<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
3) textarea

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

FieldName	Keyword(type)	syntax
text box	text	<input type="text">

passwordbox	password	<input type="passsword">
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>

```
<body>
        <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
colos	number
readonly	true,false
disabled	disabled
checked	checked
multiple	true,false

Eg: 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>
```

Eg: Textarea

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

```
</body>
```

Eg: Select List(combobox)

```
<body>
        <form>
                <select>
                        <option> select any item</option>
                        <option> HTML5 </option>
                        <option> Jquery </option>
                        <option> Bootstrap </option>
                        <option> css </option>
                </select>
        </form>
</body>
```

Eg: 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 || o Google chrome

```
<body>
        <p> What is your favourite web browser </p>
        <form>
                <input type="radio" name="browser" value="IE||"/>
                        Internet Explorer||
                <input type="radio" name="browser" value="GC"/>
                Google Chrome
        </form>
</body>
```

Checkboxes: What is your favourite web browser

0 Internal Explorer || 0 Google chrome

```
<body>
        <p> What is your favourite web browser </p>
        <form>
                <input type="checkbox" name="browser"/>Internet Explorer ||
```

```
                <input type="checkbox" name="browser"/> 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 field to indicate that they are related. It is paired tag.

syntax: <fieldset> ... </fieldset>

Eg:

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

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

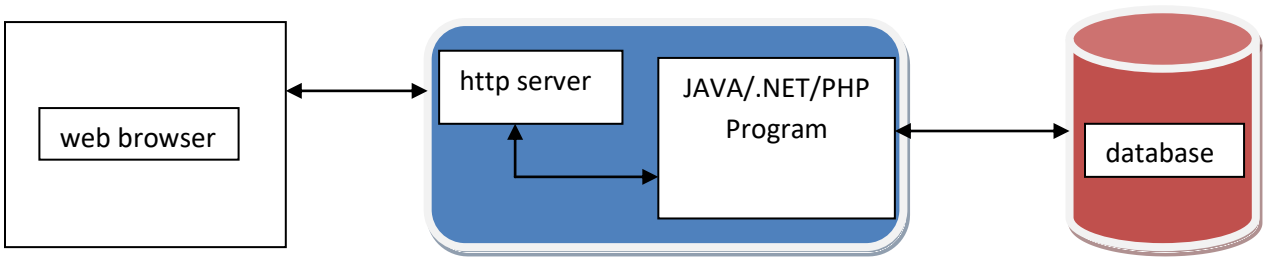
syntax: <legend> .... </legend>

Attributes:

Attribute	Parameters
Align	right,center,left

```
<body>
        <form>
                <fieldset>
                        <legend> favourite web browser </legend>
                        <input type="radio" name="browser">  IEs|| <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

Get method: In this method we don't have security for our data and only limited data can be sent to the server page. It is carrying row data between client server. This is the default method of the form.

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

eg:

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

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">

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">

eg:

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

Working with Advanced forms

HTML5 supports the following advanced 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 browser 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

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='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>
```

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

syntax: input type='datetime'

Eg:

```
<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="nextpage">
	</form>
</body>
```

datetime local: This input type allows the user to select dateandtime without time zone.

syntax: input type='datetime-local'

Eg:

```
<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>
```

month: This input type allows the user to select month and year.

syntax: input type='month'

Eg:

```
<body>
        <form action="nit.html" name="myform" id="foorm1">
                <label style='color:blue'>select valid month & year ...! </label><br/>
                <input type="month" name="my"><br/>
                <input type="submit" value="next page">
        </form>
</body>
```

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

syntax: input type='time'

Eg:

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

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

syntax: input type='week'

Eg:

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

General form controls

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

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

syntax: input type='color'

Eg:

```
<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>
```

email: This i/p type allows user to enter vallid email address.

syntax: input type='email'

Eg:

```
<body>
	<form action="nit.html" name="myform" id="form1">
		<label style='color:blue'>Enter valid emailId </label>
		<input type="email" name="e1"><be/>
		<input type="submit" value="next page">
	</form>
</body>
```

number: This input type allows to display numerical spinners

syntax:input type='number'

Attributes:

max: Specifies the maximum value allowd.

min: Specified the minimum value allowd.

step: Specifies the legal number intervals

value: Specifies the default value.

Eg:

```
 <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>
```

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

syntax: input type='range'

Attribute

max

min

step

value

Eg:

```
<body>
```

```
        <form action="nit.html" name="myform" id="form1">
                <label style='color:blue'> 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>
```

search: In html5 we can define textbox as searchbox 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" your input string will be clear and you can start to type a new string.

syntax: input type='search'

Eg:

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

Tel: This i/p type accept only phone numbers.It doesn't confirm any specific pattern.

syntax: input type='tel'

Eg:

```
<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>
```

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

syntax: input type='url'

Eg:

```
<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

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.

Eg:

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

novalidate: It is a boolean attribute. When present it specifies that the form data(input) should not be validate when submitted.

syntax: <form novalidate="novalidate">

Eg:

```
<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
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) spell check
18) contenteditable
19) accesskey

placeholder: It specifies a short hint that describes the expected value of an input field. The hint is displayed in the input field when it is empty and disappears when enter some content.

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

Attribute	values
value	description
text	specifies a short hint that describes th expected value of the input filed.

Eg:

```
<body>
	<form action="nit.html">
		<fieldset>
			<legend align="center">user login form....!</legend>
			<input type="text" name="fname" placeholder="firstname">
			<br/>
			<input type="text" name="lname" placeholder="lastname">
			<br/>
			<input type="password" name="pwd" placeholder="password">
			<br/>
			<input type="submit" value="login">
		</fieldset>
	</form>
</body>
```

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

syntax: <input autofocus="autofocus"/>

(or)

<input autofocus>

<input autofocus="">

note: autofocus attribute works on any input level.

ex:

```
<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>
```

Required: 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="">

Eg:

```
<body>
```

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

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.

Eg:

```
<body>
        <form action="nit.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>
```

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

syntax: <input form="id/name">

Eg:

```
<body>
        <form action="nit.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>
```

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>
```

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"

Eg:

```
<body>
        <form action="demo_post_enctype.asp" method="post">
                Name: <input type="text" name="fname" value="Ståle Refsnes"><br>
                <button type="submit">Submit with character encoding</button>
                <button type="submit" formenctype="text/plain">Submit without character encoding</button>
        </form>
</body>
```

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

Syntax: <form method='get/post'>...</form>

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

Eg:

```
<body>
        <form action="nit.html" method="get">
                first name: <br/>
                <input type="text" name="fname"><br/>
                password: <br/>
                <input type="password" name="pwd"><br/>
                <button type="submit" value="submit">
        </form>
</body>
```

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";

Eg:

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

```
</body>
```

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">

Eg:

```
<body>
	<form="nit.html">
		firstname:<br/>
		<input type="text" name="uname"><br/>
		lastname:<br/>
		<input type="text" name="lname"><br/>
		<input type="submit" value="@sameTW">
		<input type="submit" value="@newTW" formtarget="_blank">
	</form>
</body>
```

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">

Eg:

```
<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>
```

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

syntax: <input list="name">

Eg:

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

Multiple Attributes: 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" maultiple="on/off">

Eg:

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

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

syntax: <input type pattern=pegExp>

Eg:

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

Spell check attribute: It specifies whether the element is to have it's spelling and grammer checked or not.

syntax: <element spellcheck="true/false">

Attribute value	
true	The element is to have it's spelling and grammer check.
false	The element is not to be checked(default)

Eg:

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

contenteditable attribute: Using this attribute we can modify the content dirctly on the web page.

syntax: <element contenteditable="true/false">

Attribute value	values
true	The element is able to modify
false	The element content is unable to modify(default).

Eg:

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

accesskey attribute: Accesskeys allows easier navigation by assigning a keyboard shortcut to a link(which will useally gain focus when user presses 'alt' or 'ctr' the accesskey).

syntax:<element accesskey="character">

Attribute	values
value	
character	specifie the shortcut key to activate/focus the element.

Eg:

```
<body>
    <a href="http://www.nareshit.com" accesskey="n">nareshit</a>
    <a href="http://www.nareshit.com" accesskey="s">nit</a>
</body>
```

HTML5 new form elements

HTML5 support the following list of new elements:

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

<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 paired tag.

syntax: <datalist> ... </datalist>

attribute: data => specifies a xml file that can be used to prefill the datalist.

Eg:

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

<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 nonpaired tag.

syntax: <keygen>

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.

Eg: with challenge attribute

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

```
                user name:<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>

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.

Eg:

```
<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="ab"></output>
        </form>
</body>
```

Working with HTML5 multimedia

Multimedia comes in many different formats in web environment modern wenpages having pictures,music,sound,video,records films,animation etc.

Multimedia video formats:

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

- AVI(.avi) ---> AVI (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 formate 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 customised controls,audio files can be prefetched or downloaded on demand.It is a paired tag.

synatx: <audio> ... </audio>

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

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.

Eg:

```
<body>
        <audio src="song.mp3" controls="controls" loop="3" autoplay="autoplay">
                <p> oops your browser not supporting audio feature
        </audio>
</body>
```

Eg:

```
<body>
        <audio src="song.mp3" controls="controls" autoplay="autoplay">
                <p style='color:red'>oops your browser not supporting</p>
        </audio>
</body>
```

HTML5 audio tag with JS controls:

play audio pause audio i volume d volume

Eg:

```
<body>
        <h2 style='color:blue'>HTML5 audio tag with js controls</h2>
        <audio src="http://sound26.mp3pk.com" id="myaud"></audio>
        <div>
                <button onclick="document.getElementById('myaud').play()">playaudio</button>
                <button onclick="document.getElementById('myaud').pause()">
                        pauseaudio
                </button>
                <button onclick="document.getElementById('myaud').volume+=0.1">
                        ivolume
                </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.

synatx: <video> ... </video>

Note: Any text between the<video> and </video> tags will be displayed in browser that don't 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	autometadata none	specifies it and how author thinks the video should be loaded when the page loads

.

Eg:

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

Ex: 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("myvid").volume+=0.1">
                        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 subtitle, 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 VTF-8 format.

How to create web VTT files?
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

Ex: 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 is a paired tag. 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 enchncement

syntax: <canvas> .... </canvas>

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

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

Specific 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_attribute	css code specify inline the html tag is presented

Event attribute:

attribute	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

How to create a canvas?

```
<!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'>
                        <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='utt-8'>
                <title>canvas element</title>
        </head>
        <body>
                <canvas width='200px' height="100px" style='border:2px solid #FF00FF'
                id="MyCan">
                </canvas>
                <script tyep="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:

```
<!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='#FFgg00';
			ctx.fillRect(25,0,120,50);
			ctx.fillStyle='#FF00CC';
			ctx.fillRect(100,35,160,60);
		</script>
	</body>
</html>
```

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.

Eg:

```
<body>
	<canvas>
		<p style='color:red'>
			oops your browser not supporting canvas feature update and try
		</p>
	</canvas>
	<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.storke();
	</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 represents ending position.

Eg:

```
<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.getElementById("canvas");
		var ctx=c.getContext("2d");
		ctx.beginPath();
		ctx.arc(95, 50, 40, 0, 2 * Math.PI);
		ctx.stroke();
	</script>
</body>
```

Eg:

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

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.

Eg:

```
<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**: 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 any where between 0 to 1. To us the gradient set the fillStyle or stroke style property to the gradient.

Eg:

```
<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>
```

2) Radial Gradients: These gradients are popularly known as circular gradients. It has the following detailed syntax

```
createRadioGradient(x, y, r, x1, y1, r1);
```

Eg:

```
<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>
```

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 image 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>
```

SVG shapes: Svg supports the following list of shapes.

shape	keyword
Rectangle	<rect>
circle	<circle>
Ellipse	<Ellipse>
Line	<line>
polyline	<polyline>

Eg: 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>
```

Eg: 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:2;
```

```
                stroke:rgb(255,0,0)"> x=20px, y=20px
        </svg>
</body>
```

Eg: SVG Rectangle with radios

```
<body>
        <h2 style='color:red'>SVG Rectangle</h2>
        <svg xmls="https://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>
```

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

```
<body>
        <h2> SVG circle </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>
```

Eg: Ellipse element is used to create an ellipse. It is closely related to the circle.

```
<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>
```

Eg: Line element is used to create 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>
```

Difference between canvas & svg:

canvas:

- Resolution dependent
- support for event handlers
- poor text rendering capabilities
- you can save the resulting image as .png or .jpg
- suited for game application.

SVG:

- Resloution independent
- No support for event handlers
- Best suited for applications with large rendering areas (GoogleMaps)
- Slow rendering if complex
- 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>*</mi> ---> identifiers
2) <mo>+</mo> ---> operators
3) <mn>2</mn> ---> numbers
4) <mtext>nonzero</mtext> ---> text
5) <mrow> ---> a horizontal row of item
6) <mfrac> ---> fractions

Msup: The mathML <msup> element is used to attach a superscript to an expression.

syntax: <msup> base superscript </msup>

eg:

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

```
</body>
```
Msub: The MathML <msub> element is used to attach a subscript to an expression.

syntax: <msub> ..... </msub>

eg:

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

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

syntax: <msubsup> ..... </msubsup>

eg:

```
<body>
        <math xmlns="http://www.w3.org/1998/math 0/mathml">
                <msubsup>
                        <mo>&#x222B; <!---Integera1---></mo>
                        <mn>0</mn>
                        <mn>1</mn>
                </msubsup>
        </math>
</body>
```

mroot: This element is used to display roots.

syntax: <mroot> .... </mroot>

eg:

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

msqrt: This element is used to display square roots.

syntax: <msqrt> ... </msqrt>

eg:

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

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) downdiagonalstrike
10) verticalstrike
11) horizontalstrike etc.

Eg:

```
<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> numberator denominator </mfrac>

Eg:

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

Eg:

```
<body>
```

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