For B.E. B.C.A. M.C.A. M.C.M. B.S.C. Polythenic

HTML

by Aditya Choudhari Sir

25 Years ANNIVERSARY

CCIT

Keeping Pace with Technology

An ISO 9001 : 2008 Certified Company

website: www.ccitindia.com

Rajapeth: 0721-2563615 GadgeNagar: 0721-2552289

Table of Contents

World Wide Web

The World Wide Web is a vast collection of information that is spread across hundreds of thousands of computers around the world. When you access a document on the Web, there's a lot going on behind the scenes. Here's a very simple and brief description.

- The World Wide Web is a network of thousands of computers, all of which fall neatly into two categories: *clients* and *servers*. Through the use of special software, they form a kind of network called, not surprisingly, a *client-server network*.
- Servers store information and process requests from clients. Then they send the requested information to the clients. This information includes all kinds of data, including images, sounds, and text. Servers also send instructions to the client on how to display all this information. These instructions are sent in the form of Hypertext Markup Language (HTML).
- Clients make requests for information and then handle the chore of displaying that information to the end user. When you are using a Web browser to navigate the Web, your browsing software is acting as a client.
- The World Wide Web is a *distributed network*. That means there is no central computer for the World Wide Web. Any server on the Web can be accessed directly by any client. If a server on the World Wide Web malfunctions, it doesn't affect the performance of other servers.
- Users navigate the World Wide Web through the use of hypertext links. When you select or click on a hypertext link, you go to another area on the Internet. Almost all of the documents on the Web are interconnected through the use of hypertext links. Most of the documents on the World Wide Web are written in Hypertext Markup Language (HTML). HTML provides instructions for the client software on how the document should be displayed. HTML also contains information about how to link up to other documents on the Web.

URLs

Almost every item of information on the World Wide Web can be accessed directly. That's because every document, file, and image has a specific address. These addresses are called *Uniform Resource Locators* (URLs). URLs are used by Web browsing software to locate and access information on the World Wide Web. Think of URLs as postal addresses for the Internet.

- The first part of the URL is known as the *protocol*. This is almost always *http://*, which is short for Hypertext Transfer Protocol. Some URLs start with a different protocol, such as *ftp://* or *news://*. If you're accessing a document on your local machine instead of on the Web, the URL will begin with *file://*.
- The second part of the URL is known as the *domain name*. If you've used e-mail on the Internet, you're probably already familiar with domains. The domain represents the name of the server that you're connecting to.
- The third part of the URL is called the *directory path*. This is the specific area on the server where the item resides. Directory paths on Web servers work a lot like they do on your desktop computer. To locate a particular file on a server, you need to indicate its directory path first.
- The fourth part of the URL is called the *document file name*. This indicates the specific file being accessed. This is usually an HTML file, but it can also be an image, sound, or another file.

  Sometimes the URL contains a fifth part, known as the *anchor name*. This is a pointer to a specific part of an HTML document. It's always preceded by the pound sign (#). Anchors are especially useful for large documents.

Web Browser

Your Web browser is your gateway to the World Wide Web. A browser is the client software that allows you to access and view any document on the Web. There are a number of Web browsers that you can use to access the Web, and the number of choices available grows every month.

Different Web browsers have different features, and they all display Web pages with slight variations. Older Web browsers, which are still in widespread use, often have trouble displaying some of the newer HTML 3.2 features. If you're planning to create Web pages with HTML, you'll want to test them with a number of different Web browsers.

- To navigate to a Web page, you can type in the URL for the page here.
- Use these directional buttons to navigate backward and forward through the list of documents you have recently accessed.
- The button with the house on it always takes you back to your home page, no matter where you are.
- The status bar keeps you informed about the progress of a page as your Web browser loads it.

HTML

Without HTML, the World Wide Web wouldn't exist. HTML allows the individual elements on the Web to be brought together and presented as a collection. Text, images, multimedia, and other files can all be packaged together using HTML. This section explains the basic principles behind the interaction between HTML and the World Wide Web.

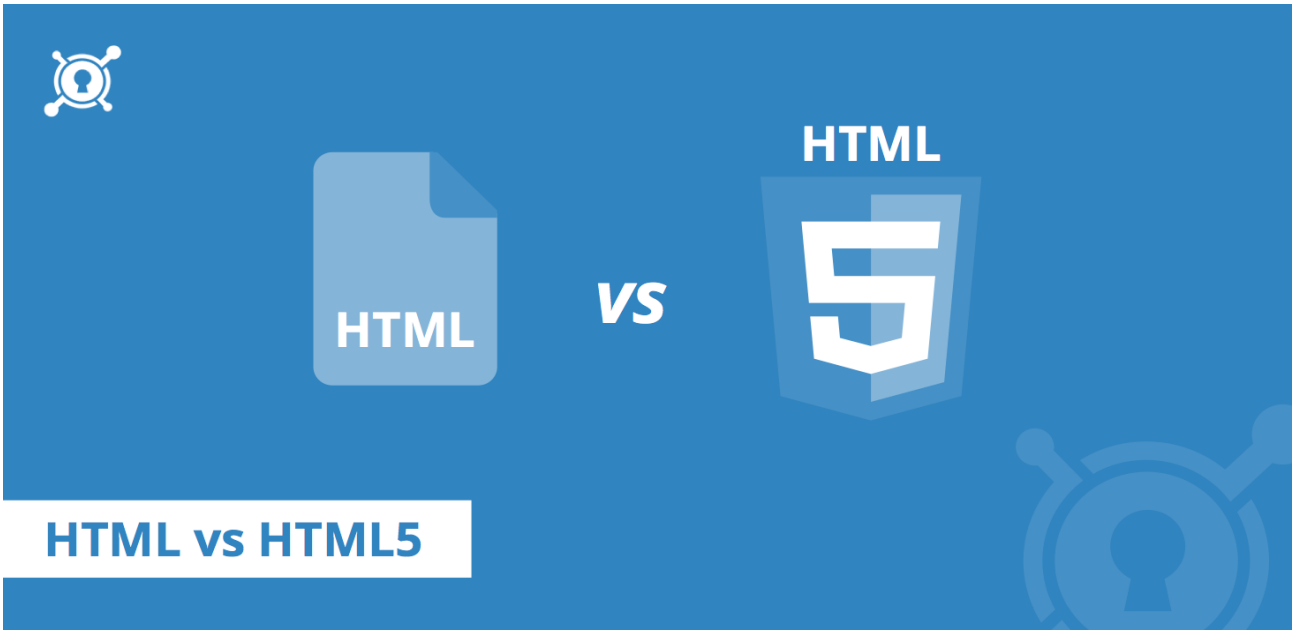
- The speed of your Web browsing software largely depends on the type of Internet connection you have. Although a modem connection at 14.4Kbps is acceptable, you should consider upgrading your hardware and contacting an Internet Service Provider who can supply a faster connection
- You can always view the HTML source code for a particular page through your browser. Once you've mastered the basics of HTML, this is a great way to learn how other authors put together their HTML documents. To view the source code of the current document in Netscape/Internet Explorer, choose Document Source from the View menu.
- The author of the Web page assembles all of the materials necessary, including text, charts, images, and sounds.
- All of the material for the Web page is linked together using HTML. HTML codes control the appearance, layout, and flow of the page. The amazing thing about HTML is that it is all done with simple text codes that anyone can understand.
- When someone connects to a Web server from his or her computer, the HTML file is transferred from server to client. Because an HTML file is simple text, this usually happens very quickly.
- The Web browsing software (the client) interprets the layout and markup commands specified in the HTML file and then displays the text exactly as the HTML author intended.Any images and charts on the page are retrieved as well. The HTML file tells the Web browser what images to download and how to display them on the page.

HTML5

HTML5 is the latest evolution of the standard that defines HTML. The term represents two different concepts. It is a new version of the language HTML, with new elements, attributes, and behaviors, and a larger set of technologies that allows the building of more diverse and powerful Web sites and applications. This set is sometimes called HTML5 & friends and often shortened to just HTML5.

Designed to be usable by all Open Web developers, this reference page links to numerous resources about HTML5 technologies, classified into several groups based on their function.

- Semantics: allowing you to describe more precisely what your content is.
- Connectivity: allowing you to communicate with the server in new and innovative ways.
- Offline and storage: allowing webpages to store data on the client-side locally and operate offline more efficiently.
- Multimedia: making video and audio first-class citizens in the Open Web.
- 2D/3D graphics and effects: allowing a much more diverse range of presentation options.
- Performance and integration: providing greater speed optimization and better usage of computer hardware.
- Device access: allowing for the usage of various input and output devices.
- Styling: letting authors write more sophisticated themes.



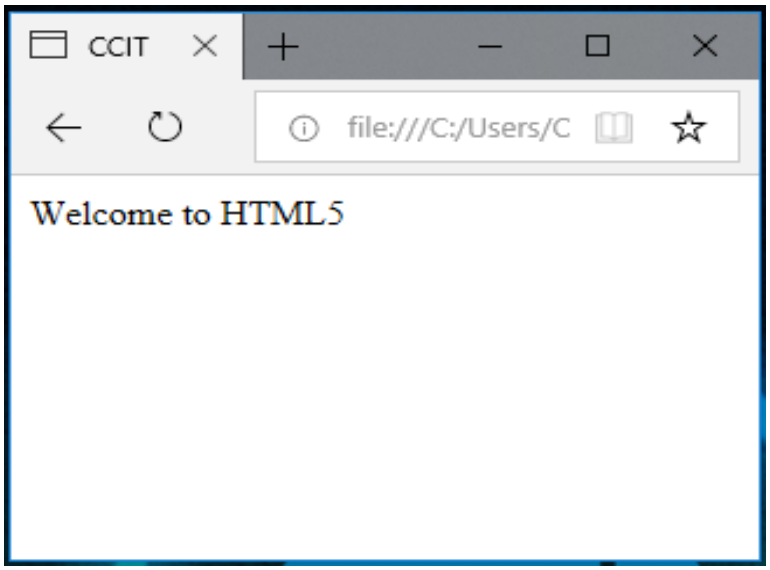
If HTML was fine for over a decade, why was it updated in 2014? The most significant difference between older versions of HTML vs HTML5 is the integration of video and audio into the language' s specifications. Additionally, HTML5 includes the following updates:

- Deprecated elements like center, font, and strike have been dropped
- Improved parsing rules allow for more flexible parsing and compatibility
- New elements including video, time, nav, section, progress, meter, aside and canvas
- New input attributes including email, URL, dates and times
- New attributes including charset, async and ping
- New APIs that offer offline caching, drag-and-drop support and more
- Support for vector graphics without the aid of programs like Silverlight or Flash
- Support for MathML to allow better display of mathematical notations
- JavaScript can now run in the background thanks to the JS Web worker API
- Global attributes such as tabindex, repeat and id can be can be applied for all elements

Format of HTML Page

The basic HTML document contains two parts: the head and the body. The head section contains important information about the document itself, such as the title. The actual text, images, and markup tags are placed in the body section.

```
<!DOCTYPE html>
<HTML>
  <head >
    ---------
  </head>
  <body>
     ----------
     ----------
  </body>
</HTML>
```



!DOCTYPE

<!DOCTYPE html>

The <!DOCTYPE> declaration must be the very first thing in your HTML document, before the <html> tag.

The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

Example

```
<!DOCTYPE html>
<HTML>

<BODY>

Welcome To CCIT

</BODY>

</HTML>
```

HTML

<HTML></HTML>

Denotes the file as an HTML document. The end-tag comes after all HTML elements in the document. This element has no attributes.

Example

```
<!DOCTYPE html>
<HTML>

<BODY>

Welcome To CCIT

</BODY>

</HTML>
```

Head

<HEAD></HEAD>

The head of an HTML document is the part that is not displayed in the web browser when the page is loaded.

Example

```
<!DOCTYPE html>
<HTML>

<HEAD>

<TITLE>"CCIT"</TITLE>

</HEAD>

</HTML>
```

TITLE

<TITLE></TITLE>

Specifies a title for the document. Most browsers uses this for the window caption. This element is valid only within the HEAD element. The end-tag is required.

Example

```
<!DOCTYPE html>
<HTML>

<HEAD>

<TITLE>"CCIT"</TITLE>

</HEAD>

</HTML>
```

BODY

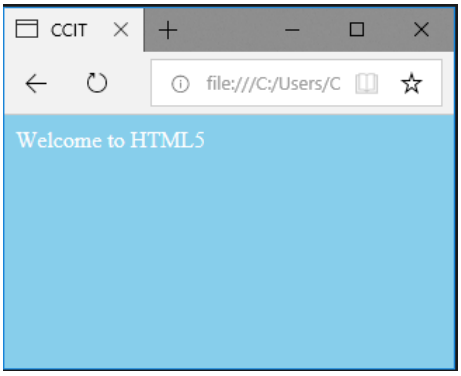
```
<BODY
ALINK=color
BACKGROUND=url
BGCOLOR=color
LEFTMARGIN=n
LINK=color
STYLE=css1 properties
TEXT=color
TOPMARGIN=n
VLINK=color>
</BODY>
```

Specifies the beginning and end of the document body. This element also allows you to set the background image, the background color, the link colors, and the top and left margins of the page. The end-tag is required.

ALINK=*color*	Specifies the color of the active hyperlink.
BACKGROUND=*url*	Specifies a background picture.
BGCOLOR=*color*	Sets the background color of the page. The *color* can be either a hexadecimal, red-green-blue color value or a predefined color name.
LEFTMARGIN=*n*	Specifies the left margin for the entire body of the page
LINK=*color*	Sets the color of hyperlinks that have not yet been visited.
STYLE=*css1 properties*	Specifies style information.
TEXT=*color*	Sets the color of text on the page.
TOPMARGIN=*n*	Specifies the margin for the top of the page
VLINK=*color*	Sets the color of hyperlinks that have already been visited.

Examples

```
<!DOCTYPE html>
<html>
<head>
    <title>CCIT</title>
</head>
<body bgcolor="skyblue"
text="white" >
    Welcome to HTML5
</body>
</html>
```



B

<B></B>

Renders text in bold. The end-tag turns off the bold formatting.

Example

```
<B>Displayed in a bold typeface.</B>
```

I

<I></I>

Renders text in italic. The end-tag turns off the italic formatting.

Example

```
<I>This text will be in italic.</I>
```

U

<U></U>

Renders underlined text. The end-tag restores the text to normal.

Example

```
<U>This text is underlined.</U>
```

BIG

<BIG></BIG>

Makes text one size larger.

Example

```
<BIG>This text is larger.</BIG>
```

S

<S></S>

Renders text in strikethrough type. The end-tag restores the formatting to normal.

Example

```
<S>This text has a line through it.</S>
```

STRIKE

<STRIKE></STRIKE>

Renders text in strikethrough type. The end-tag returns formatting to normal.

Example

```
<STRIKE>This text has a line through it.</STRIKE>
```

SUB

****

Renders text in subscript. The end-tag restores normal formatting.

Example

```
<SUB>This text is rendered as subscript.</SUB>
```

SUP

****

Renders text in superscript. The end-tag restores normal formatting.

Example

```
<SUP>This text is rendered as superscript.</SUP>
```

BR

**
**

Inserts a line break.

NOBR

<NOBR>

Turns off line breaking. Renders text without line breaks.

Example

```
<NOBR>Here's a line of text I don't want to be broken.</NOBR>
```

PRE

<PRE></PRE>

Renders text in fixed-width type. The end-tag restores the text to normal formatting.

CENTER

<CENTER></CENTER>

Centers text and images. The end-tag returns the alignment to its previous state.

MARK

<mark></mark>

To highlight a text.

Hn

<H*n* ALIGN=LEFT|CENTER|RIGHT> </H*n*>

Renders text in heading style. Use H1 through H6 to specify different sizes and styles of headings. The end-tag (required) restores the formatting to normal.

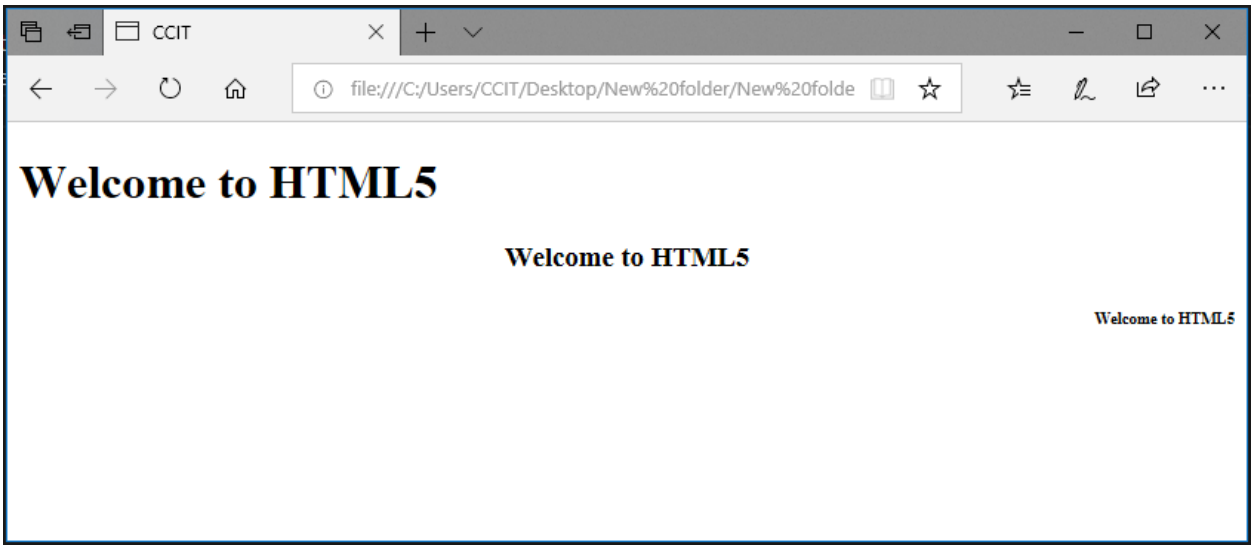
N Sets the heading level. This is an integer from 1 to 6.

ALIGN=LEFT|CENTER|RIGHT

Sets the alignment of heading text. The default is LEFT.

Example

```
<!DOCTYPE html>
<html>
<head>
       <title>CCIT</title>
</head>
<body>
       <h1 align=left>Welcome to HTML5</h1>
       <h3 align=center>Welcome to HTML5</h3>
       <h6 align=right>Welcome to HTML5</h6>
</body>
</html>
```



HR

<HR
ALIGN=LEFT|CENTER|RIGHT
COLOR=*color*
SIZE=*n*
STYLE=*css1 properties*
WIDTH=*n***>**

Draws a horizontal rule.

ALIGN=CENTER|LEFT|RIGHT
Sets the alignment of the rule. The default is CENTER.

COLOR=*color* Sets the color of the rule. The *color* can be either a hexadecimal, red-green-blue color value or a predefined color name. See Color.

SIZE=*n* Sets the height of the rule, in pixels.

STYLE=*css1 prop.* Specifies style information.

WIDTH=*n* Sets the width of the rule, either in pixels or as a percentage of window width. To specify a percentage, the *n* must end with the percent (%) sign.

Example

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
      <h1 align=center>CCIT</h1>
      <hr color="red" width="300"align="center" size=10>
</body>
</html>
```

FONT

<FONT
COLOR= *color*
FACE=*name*
SIZE=*n***></FONT>**

Sets the size, font, and color of text.

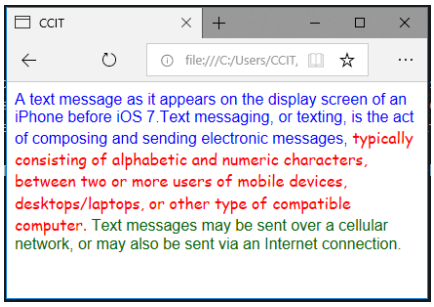
COLOR=*color* Sets font color. The *color* can be either a hexadecimal, red-green-blue color value or a predefined color name.

FACE="*name* [,*name2*[,*name3*]]"
Sets the font. A list of font names can be specified. If the first font is available on the system, it will be used; otherwise, the second will be tried, and so on. If none are available, a default font will be used.

SIZE=*n* Specifies font size between 1 and 7 (7 is largest). A plus or minus before the number indicates a size relative to the current BASEFONT setting.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<font face="arial" color=blue >
A text message as it appears on the display screen of an iPhone before
iOS 7.Text messaging, or texting, is the act of composing and sending
electronic messages,</font><font face="comic sans ms" color=red >
typically consisting of alphabetic and numeric characters, between two
or more users of mobile devices, desktops/laptops, or other type of
compatible computer. </font>
<font face="arial" color=darkgreen >Text messages may be sent over a
cellular network, or may also be sent via an Internet connection.
</font>
</body>
</html>
```



MARQUEE

<MARQUEE
ALIGN=LEFT|CENTER|RIGHT|TOP|BOTTOM
BEHAVIOR=*type*
BGCOLOR=*color*
DIRECTION=*direction*
HEIGHT=*n*
HSPACE=*n*
LOOP=*n*
SCROLLDELAY=*n*
VSPACE=*n*
WIDTH=*n*>
</MARQUEE>

Creates a scrolling text marquee. The scrolling text appears in the container.

ALIGN=LEFT|CENTER|RIGHT

Specifies how the surrounding text should align with the marquee. The default is LEFT.

LEFT Surrounding text aligns with the left of the marquee.

CENTER Surrounding text aligns with the center of the marquee.

RIGHT Surrounding text aligns with the right of the marquee.

BEHAVIOR=*type*

Specifies how the text should behave. The *type* can be one of these values:

SCROLL Start completely off one side, scroll all the way across and completely off, and then start again. This is the default.

SLIDE Start completely off one side, scroll in, and stop as soon as the text touches the other margin.

ALTERNATE Bounce back and forth within the marquee.

BGCOLOR=*color* Specifies a background color for the marquee. The *color* can be either a hexadecimal number (optionally preceded by a #) specifying a red-green-blue color value, or a predefined color name as described in Color.

DIRECTION=*direction*

Specifies in which direction the text should scroll. The *direction* can be LEFT or RIGHT. The default is LEFT, which means scrolling from right to left.

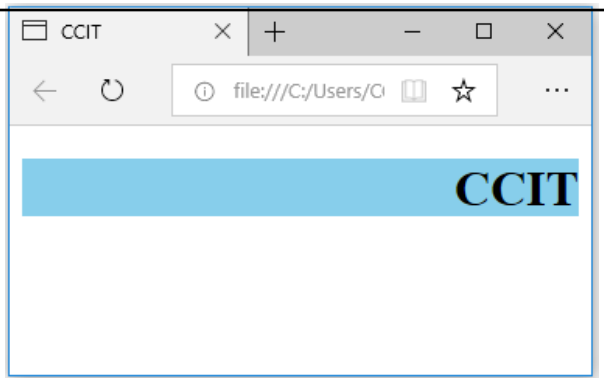
HEIGHT=*n* Specifies the height of the marquee, either in pixels or as a percentage of the screen height. To specify a percentage, the *n* must end with a percent sign (%).

LOOP=*n* Specifies how many times a marquee will loop when activated. If

WIDTH=*n* Sets the width of the marquee, either in pixels or as a percentage of the screen width. To specify a percentage, the *n* must end with a percent sign (%).

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<marquee align=Right Behavior="slide" width=300 bgcolor=skyblue
                direction=right height=40 loop=2
                size=20>CCIT</marquee>
</body>
</html>
```



HTML Comments

<!-- Comments... -->

Comments are not displayed by the browser, but they can help document your HTML.

Lists

Lists are used to display all kinds of information on line. We can create unordered lists, ordered (numbered) lists, and a special type of list known as a definition list.

The list have three type:

- Unordered lists
- Ordered lists
- Description List

Unordered lists

The simplest list in HTML is the unordered, or bulleted, list. This is ideal for listing items that have no particular hierarchy or order of importance. Web browsers usually place bullets or other markers in front of each item in an unordered list.

UL

```
<UL
STYLE=css1 properties>
</UL>
```

Specifies that the following block of text contains individual items that begin with an LI tag. These items are bulleted. The end-tag is required.

STYLE=*css1 properties*

Specifies style information.

TYPE=value

The type attribute specifies the kind of marker to use in the unordered list.

Value	Description
Disc	Sets the bullet (default)
Circle	Sets the circle
Square	Sets the square
None	The list items will not be marked

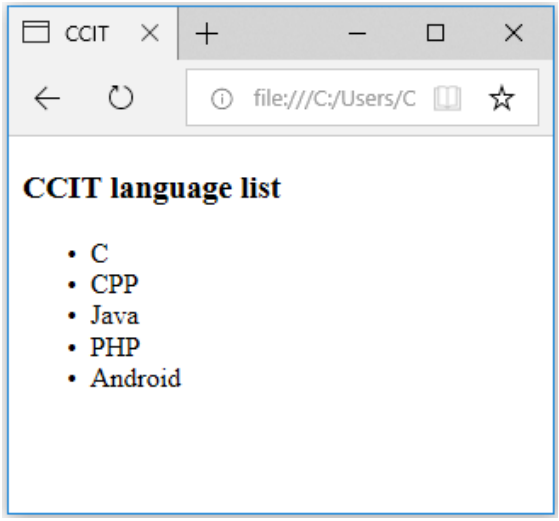
LI

<LI
VALUE=*n*></LI>

Denotes one item of a list. Each list item starts with the <li> tag.

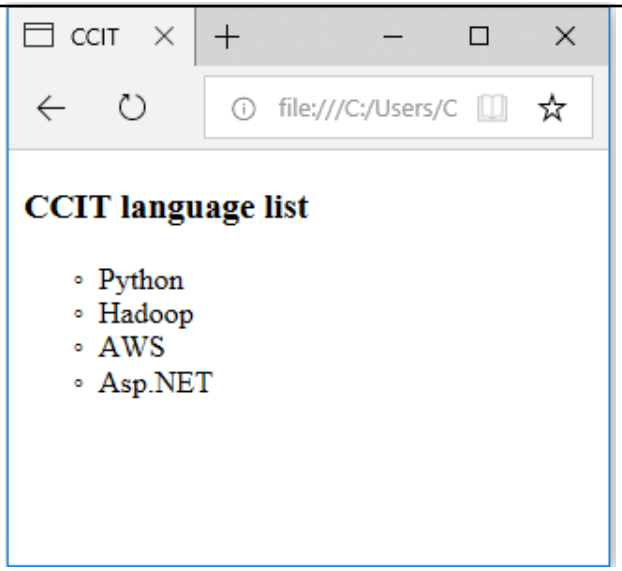
Example

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
      <h3>CCIT language list</h3>
      <ul>
            <li>C</li>
            <li>CPP</li>
            <li>Java</li>
            <li>PHP</li>
            <li>Android</li>
      </ul>
</body>
</html>
```



Example

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
      <h3>CCIT language list</h3>
      <ul type=circle>
            <li>Python</li>
            <li>Hadoop</li>
            <li>AWS</li>
            <li>Asp.NET</li>
      </ul>
</body>
</html>
```



Ordered Lists

Sometimes you need to list items in a specific order. Examples of this type of list include step-by-step instructions and "Top 10" lists. HTML provides a way to do this through ordered lists. Web browsers will place a number in front of each item, increasing the number by one for each entry down the list.

OL

<OL
STYLE=*css1 properties*
TYPE=*order-type***>**

Specifies that the following lines of text contain individual items that begin with an LI tag. These items are numbered.

STYLE=*css1 properties*

Specifies style information.

TYPE=*order-type*

Changes the style of the list. The *order-type* can be one of these values:

A Use large letters.

a Use small letters.

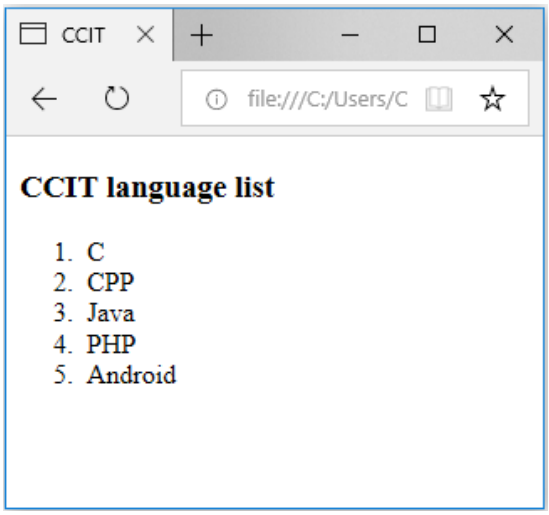
I Use large Roman numerals.

i Use small Roman numerals.

1 Use numbers. (default)

Example

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
      <h3>CCIT language list</h3>
      <ul>
            <li>C</li>
            <li>CPP</li>
            <li>Java</li>
            <li>PHP</li>
            <li>Android</li>
      </ul>
</body>
</html>
```



Description Lists

Definition lists are different from other lists in HTML, because each item in a definition list contains two parts: a term and a definition.

Syntax:

```
<DL>
  <DT>Title</DT>
  <DD> Description </DD>
</DL>
```

DL

```
<DL
STYLE=css1 properties>
</DL>
```

Specifies that the following block is a definition list, that is, an automatically formatted list with terms on the left and their definitions indented below. The end-tag is required.

See DT (directory term) and DD (directory definition) for a description of elements that appear within a directory list.

STYLE=*css1 properties*

Specifies style information.

DD

```
<DD
STYLE=css1 properties></DD>
```

Indicates that the following text is a definition of a term, and therefore should be displayed in the right-hand column of a definition list.

STYLE=*css1 properties*

Specifies style information.

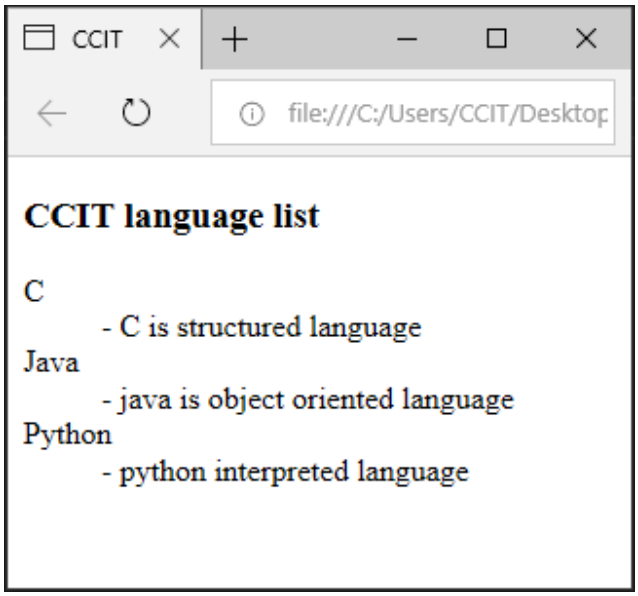
DT

```
<DT></DT>
```

Specifies a term in a definition list. Indicates that the text is a term to be defined, and should therefore be displayed in the left-hand column of a definition list.

Example:

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
      <h3>CCIT language list</h3>
      <dl>
            <dt>C</dt>
            <dd>- C is structured language</dd>
            <dt>Java</dt>
            <dd>- Java is object oriented language</dd>
            <dt>Python</dt>
            <dd>- Python interpreted language</dd>
      </dl>
</body>
</html>
```

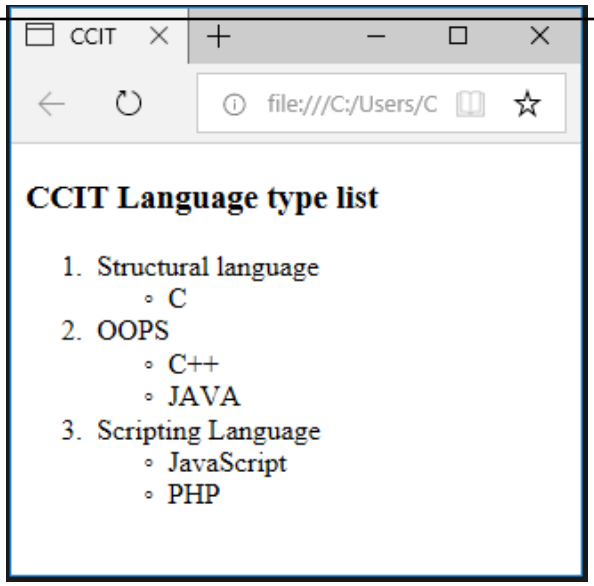


Nested HTML Lists

If a list is use in another list then such type of list is called as Nested List.

Example

```
<!DOCTYPE html>
<html>
<head>
      <title>CCIT</title>
</head>
<body>
 <h3>CCIT Language type list</h3>
  <ol>
   <li>Structural language<ul><li>C</li></ul></li>
   <li>OOPS<ul><li>C++</li><li>JAVA</li></ul></li>
   <li>Scripting Language<ul><li>JavaScript</li><li>PHP</li></ul></li>
  </ol>
</body>
</html>
```



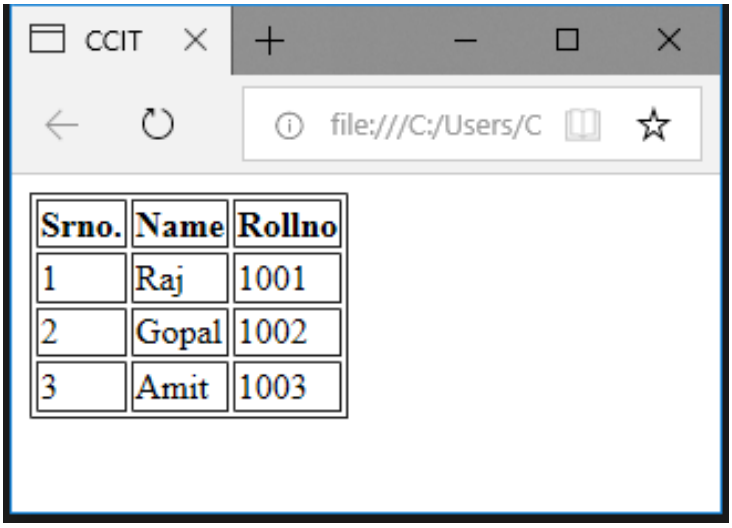
Tables

Tables give HTML authors much greater control over the display and layout of their pages. Typically, you would use tables to display any type of data that looks best in rows and columns. A good rule of thumb is if it looks good as a spreadsheet, then it belongs in a table.

Tables aren't just for numerical data. They can be used to creatively solve a number of challenges with presenting information in HTML. Tables can be used to enhance a number of existing HTML elements, such as lists and forms. You can even use tables to gain precision control over the layout of your HTML document.

Syntax:

```
<TABLE>
<TR><TH>data heading</TH><TH>data heading</TH></TR>
<TR><TD>data </TD><TD>data </TD></TR>
<TR><TD>data </TD><TD>data </TD></TR>
<TR><TD>data </TD><TD>data </TD></TR>
</TABLE>
```



TABLE

<TABLE
ALIGN=LEFT|CENTER|RIGHT|BLEEDLEFT|
BLEEDRIGHT|JUSTIFY
BACKGROUND=*url*
BGCOLOR=*color*
BORDER=*n*
BORDERCOLOR=*color*
CELLPADDING=*n*
CELLSPACING=*n*
COLS=*n*
STYLE=*css1 properties*
WIDTH=*n*>
</TABLE>

Defines a table. Use the TR, TD, and TH elements in the container to create the rows, columns, and cells. The end-tag is required.

ALIGN=LEFT|CENTER|RIGHT|	Specifies the table alignment. The default is LEFT.
BACKGROUND=*url*	Specifies a background picture.
BGCOLOR=*color*	Sets background color.
BORDER=*n*	Sets the size, in pixels, of the table border. The default is zero.
BORDERCOLOR=*color*	Sets border color. Must be used with the BORDER= attribute. The *color* is either a hexadecimal, red-green-blue color value or a predefined color name.
CELLPADDING=*n*	Sets the amount of space, in pixels, between the sides of a cell and its contents.
CELLSPACING=*n*	Sets the amount of space, in pixels, between the frame (exterior) of the table and the cells in the table.
COLS=*n*	Sets the number of columns in the table. If given, this attribute may speed up processing of tables, especially lengthy ones.
STYLE=*css1 properties*	Specifies style information.
WIDTH=*n*	Sets the width of the table in pixels or as a percentage of the window. To set a percentage, the *n* must end with a percent sign (%).

TR

```
<TR
ALIGN=CENTER|LEFT|RIGHT|JUSTIFY
BGCOLOR=color
BORDERCOLOR=color
STYLE=css1 properties
</TR>
```

Creates a row in a table.

TH

```
<TH
ALIGN=CENTER|LEFT|RIGHT|JUSTIFY
BACKGROUND=url
BGCOLOR=color
BORDERCOLOR=color
ROWSPAN=n
STYLE=css1 properties
WIDTH=n></TH>
```

Creates a row or column heading in a table. The element is similar to the TD element but emphasizes the text in the cell to distinguish it from text in TD cells. The end-tag is optional.

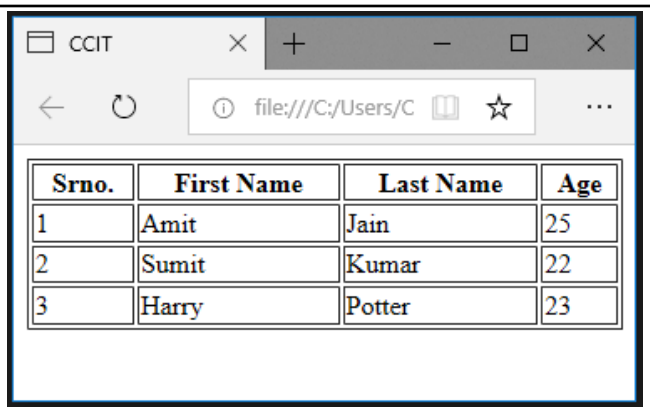
TD

```
<TD
ALIGN=CENTER|LEFT|RIGHT|JUSTIFY
BACKGROUND=url
BGCOLOR=color
BORDERCOLOR=color
COLSPAN=n
HEIGHT=n
ROWSPAN=n
STYLE=css1 properties
WIDTH=n></TD>
```

Creates a cell in a table. The end-tag is optional.

Example

```
<!DOCTYPE html>
<html>
<head>
    <title>CCIT</title>
</head>
<body>
   <table border=1 width=100%>
     <tr><th>Srno.</th><th>First Name</th><th>Last Name</th><th>Age
</th></tr>
     <tr><td>1</td><td>Amit</td><td>Jain</td><td>25</td></tr>
     <tr><td>2</td><td>Sumit</td><td>Kumar</td><td>22</td></tr>
     <tr><td>3</td><td>Harry</td><td>Potter</td><td>23</td></tr>
   </table>
</body>
</html>
```



Srno.	First Name	Last Name	Age
1	Amit	Jain	25
2	Sumit	Kumar	22
3	Harry	Potter	23

Col & Row span

This attributes are of td and th tag.

colspan This attribute contains a non-negative integer value that indicates for how many columns the cell extends. Its default value is 1. Values higher than 1000 will be considered as incorrect and will be set to the default value (1).

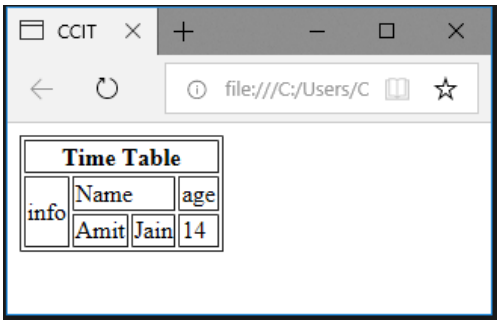
rowspan This attribute contains a non-negative integer value that indicates for how many rows the cell extends. Its default value is 1; if its value is set to 0, it extends until the end of the table section (<thead>, <tbody>, <tfoot>, even if implicitly defined), that the cell belongs to.

Syntax:

```
<TABLE>
<tr><th> table head </th></tr>
<tr><td rowspan=value> table data </td></tr>
<tr><td colspan=value> table data </td></tr>
</TABLE>
```

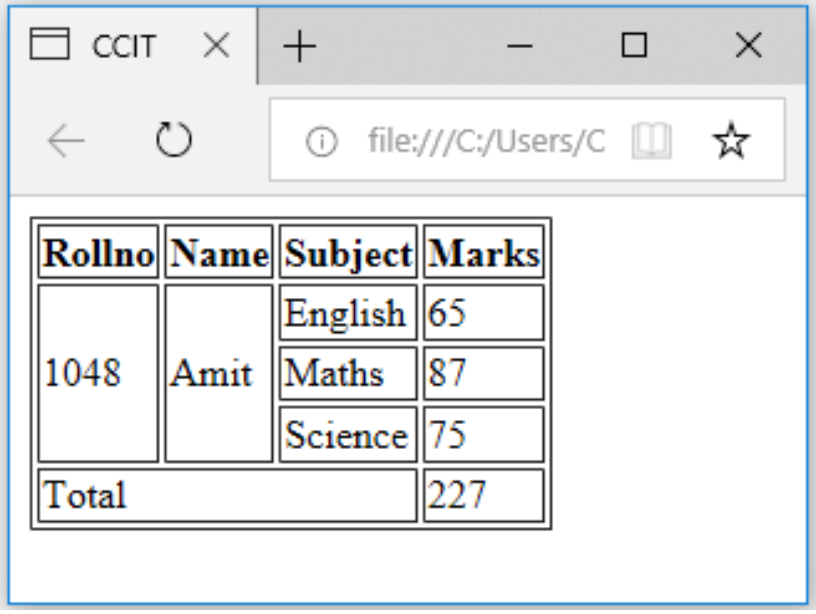
Example

```
<!DOCTYPE html>
<html>
<head>
  <title>CCIT</title>
</head>
<body>
 <table border=1 >
   <tr><th colspan="4">Time Table</th></tr>
   <tr><td rowspan="3">info</td><td colspan="2">Name</td><td>age</td>
   </tr>
   <tr><td>Amit</td><td>Jain</td><td>14</td></tr>
 </table>
</body>
</html>
```



Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<table border=1>
<tr><th>Rollno</th><th>Name</th><th>Subject</th><th>Marks</th></tr>
<tr><td                                          rowspan="3">1048</td><td
           rowspan="3">Amit</td><td>English</td><td>65</td></tr>
<tr><td>Maths</td><td>87</td></tr>
<tr><td>Science</td><td>75</td></tr>
<tr><td colspan="3">Total</td><td>227</td></tr>
</table>
</body>
</html>
```



CAPTION

<CAPTION></CATION>

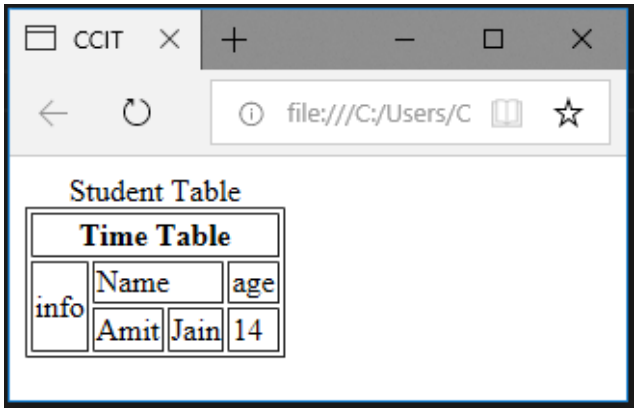
The HTML Table Caption element (<caption>) specifies the caption (or title) of a table, and if used is always the first child of a <table>. Its styling and physical position relative to the table may be changed using the CSS caption-side and text-align properties.

Syntax:

```
<TABLE>
      <caption> table caption </caption>
      <tr><th> table head </th></tr>
      <tr><td> table data </td></tr>
</TABLE>
```

Example

```
<!DOCTYPE html>
<html>
<head>
  <title>CCIT</title>
</head>
<body>
 <table border=1 >
   <caption>Student Table</caption>
   <tr><th colspan="4">Time Table</th></tr>
   <tr><td rowspan="3">info</td><td colspan="2">Name</td><td>age</td>
   </tr>
   <tr><td>Amit</td><td>Jain</td><td>14</td></tr>
 </table>
</body>
</html>
```



THEAD,TBODY,TFOOT

THEAD The HTML <thead> element defines a set of rows defining the head of the columns of the table.

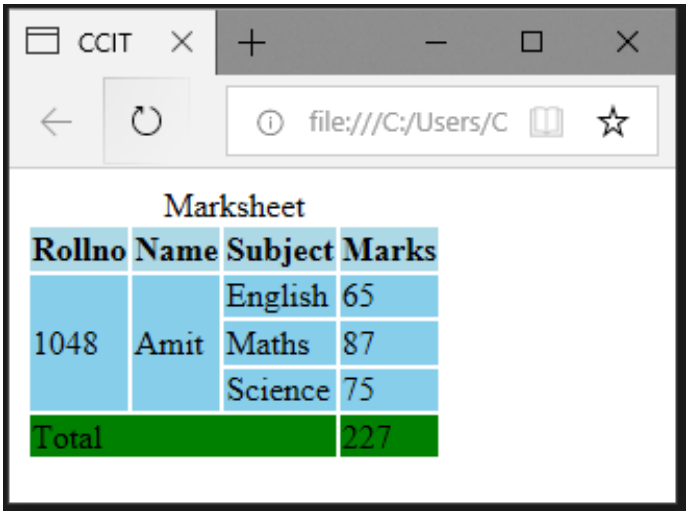
TBODY The HTML Table Body element (<tbody>) encapsulates a set of table row (<tr> elements), indicating that they comprise the body of the table (<table>).

TFOOT The **HTML** <tfoot> **element** defines a set of rows summarizing the columns of the table.

Syntax:

```
<TABLE>
<thead>
     <tr><th> table head </th></tr>
</thead>
<tbody>
     <tr><td> table data </td></tr>
     <tr><td> table data </td></tr>
</tbody>
<tfoot>
     <tr><td> table data </td></tr>
</tfoot>
</TABLE>
```

Example:



```
<!DOCTYPE html>
<html>
      <head>
            <title>CCIT</title>
      </head>
      <body>
            <table border=0>
                  <caption>Marksheet</caption>
                  <thead bgcolor="lightblue">
                        <tr>
                              <th>Rollno</th>
                              <th>Name</th>
                              <th>Subject</th>
                              <th>Marks</th>
                        </tr>
                  </thead>
                  <tbody bgcolor="skyblue">
                        <tr>
                              <td rowspan="3">1048</td>
                              <td rowspan="3">Amit</td>
                              <td>English</td>
                              <td>65</td>
                        </tr>
                        <tr>
                              <td>Maths</td>
                              <td>87</td>
                        </tr>
                        <tr>
                              <td>Science</td>
                              <td>75</td>
                        </tr>
                  </tbody>
                  <tfoot bgcolor="Green">
                        <tr>
                              <td colspan="3">Total</td>
                              <td>227</td>
                        </tr>
                  </tfoot>
            </table>
      </body>
</html>
```

COLGROUP COL

COLGROUP — The HTML <colgroup> element defines a group of columns within a table.

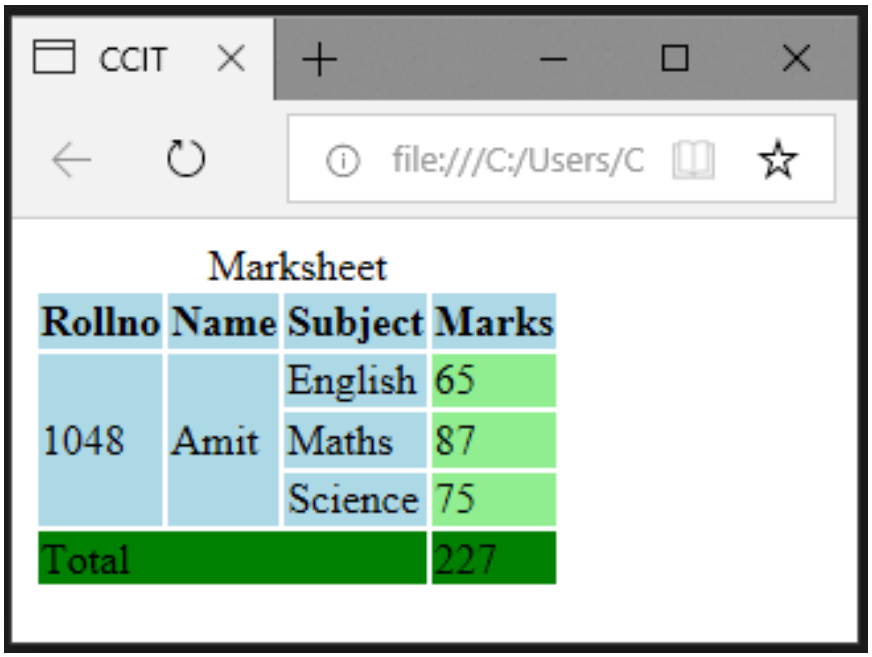
COL — The HTML <col> element defines a column within a table and is used for defining common semantics on all common cells. It is generally found within a <colgroup> element.

Attribute:

span	Specifies the number of columns a column group should span.

```
Syntax:
<TABLE>
                <colgroup>
                <col span="value" >
                </colgroup>
                <tr><th> table head </th></tr>
                <tr><td> table data </td></tr>
                </TABLE>
```

Example:



```
<!DOCTYPE html>
<html>
      <head>
            <title>CCIT</title>
      </head>
      <body>
            <table border=0>
            <caption>Marksheet</caption>
            <colgroup>
                  <col span="3" bgcolor="lightblue">
                  <col span="1" bgcolor=lightgreen>
            </colgroup>
            <thead bgcolor="lightblue">
                  <tr>
                        <th>Rollno</th>
                        <th>Name</th>
                        <th>Subject</th>
                        <th>Marks</th>
                  </tr>
            </thead>
            <tbody >
                  <tr>
                        <td rowspan="3">1048</td>
                        <td rowspan="3">Amit</td>
                        <td>English</td>
                        <td>65</td>
                  </tr>
                  <tr>
                        <td>Maths</td>
                        <td>87</td>
                  </tr>
                  <tr>
                        <td>Science</td>
                        <td>75</td>
                  </tr>
            </tbody>
            <tfoot bgcolor="Green">
                  <tr>
                        <td colspan="3">Total</td>
                        <td>227</td>
                  </tr>
            </tfoot>
            </table>
      </body>
</html>
```

Graphics

Graphical images are an integral part of any HTML document . The two most common image file formats in use on the World Wide Web are GIF (.GIF) and JPEG (.JPG) files. The GIF format is directly supported by every graphical Web browser, while JPEG is still gaining acceptance as a standard image format on the World Wide Web. Although both GIF and JPEG files can be used in your HTML documents, there are a few important differences between the two formats.

- Remember that JPEG images are 24-bit color, and require the appropriate video hardware to view properly. If the user's system can only support 256 colors, the images will be automatically adjusted to 256 colors by the Web browser or external viewer through a process known as color dithering. This will always degrade the quality of the image, and may lead to results you did not anticipate. Therefore, it's best to use 24-bit JPEG images only when absolutely necessary.
- Images take a considerable amount of time to load in an HTML document, especially when the reader has a slow modem connection to the Internet. Try to keep your images as compact as possible. Crop the images wherever possible to show only the relevant portions, thereby reducing the image size. Color depth also plays a huge role in overall image size. Consider decreasing the number of colors to 16 or 256 if it won't adversely affect the image.
- The GIF format, developed by CompuServe, is a cross-platform format, which means it can be viewed on almost any type of computer system, making it ideal for use on the World Wide Web. The one significant limitation of the GIF format is that images are limited to 256 colors.
- The JPEG format, developed by the Joint Photographic Experts Group, is also a cross-platform format, although it is not directly supported by all Web browsers. JPEG images can use the full spectrum of 16.7 million colors.
- JPEG images are compressed files. JPEG compression results in some image quality loss; however, the difference is usually not noticeable to the human eye.

- If you're placing line art, company logos, or icons in your HTML document, you should save these images in the GIF89a format. By doing so, you'll be able to take advantage of the interlacing and transparency features

IMG

<IMG
ALIGN= LEFT|RIGHT|CENTER
BORDER=*n*
HEIGHT=*n*
SRC=*url*
STYLE=*css1 properties*
WIDTH=*n*>

Inserts an image.

ALIGN= CENTER|LEFT|RIGHT

Sets the alignment of the image or of the surrounding text. The default is TOP.

CENTER Surrounding text is aligned with the center of the image.

LEFT The picture is drawn as a left-flush "floating image," and text flows around it.

RIGHT The picture is drawn as a right-flush "floating image," and text flows around it.

BORDER=*n* Specifies the size of a border to be drawn around the image. If the image is a hyperlink, the border is drawn in the appropriate hyperlink color. If the image is not a hyperlink, the border is invisible.

HEIGHT=*n* Specifies the height of an image

SRC=*url* Specifies the address of the picture to insert.

STYLE=*css1 properties*

Specifies style information.

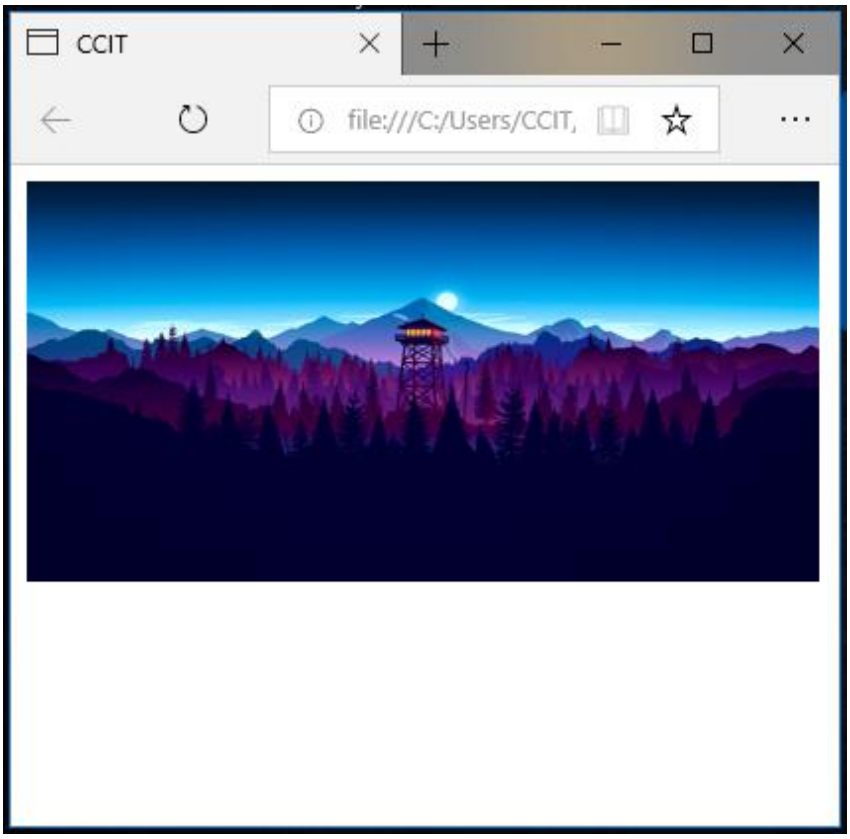
WIDTH=*n* Specifies the Width of an image

Syntax:

<IMG src="file_path">

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<img src="img4.jpg" height="200" width="400" alt="Can not show">
</body>
</html>
```



AUDIO

Audio The HTML <audio> element is used to embed sound content in documents. It may contain one or more audio sources, represented using the src attribute or the <source> element: the browser will choose the most suitable one.

Source The <source> tag is used to specify multiple media resources for media elements, such as <video> and <audio>.
The <source> tag allows you to specify alternative video/audio files which the browser may choose from, based on its media type or codec support.

Attribute:

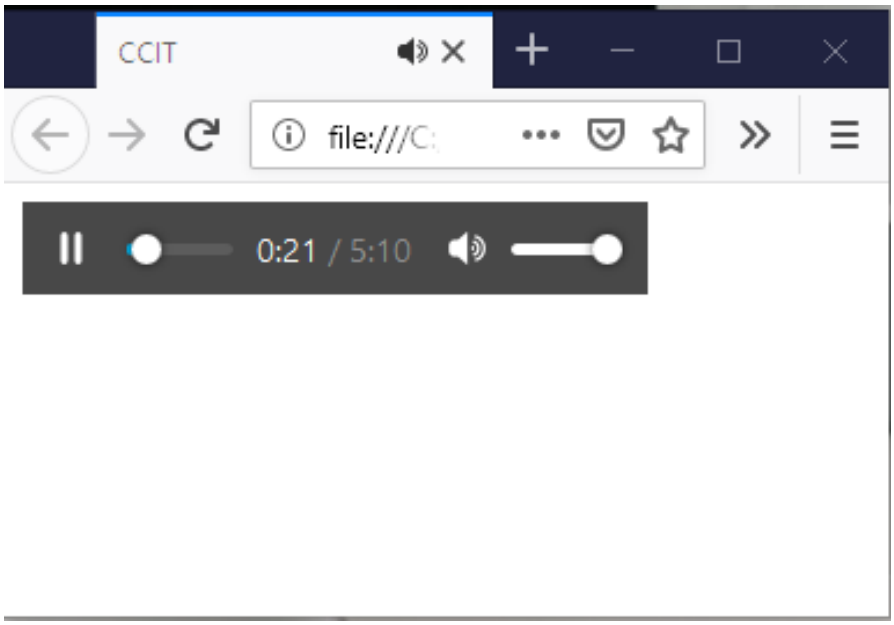
Attribute	Description
controls	It defines the audio controls which is displayed with play/pause buttons.
autoplay	It specifies that the audio will start playing as soon as it is ready.
loop	It specifies that the audio file will start over again, every time when it is completed.
muted	It is used to mute the audio output.
src	It specifies the source URL of the audio file.

Syntax:

```
<audio controls>
     <source src="File path" >
</audio>
```

Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
           <audio controls autoplay loop>
           <source src="audio.mp3" >
      </audio>
    </body>
</html>
```



VIDEO

VIDEO The HTML Video element (<video>) embeds a media player which supports video playback into the document. You can use <video> for audio content as well, but the <audio> element may provide a more appropriate user experience.

Source The <source> tag is used to specify multiple media resources for media elements, such as <video> and <audio>.

The <source> tag allows you to specify alternative video/audio files which the browser may choose from, based on its media type or codec support.

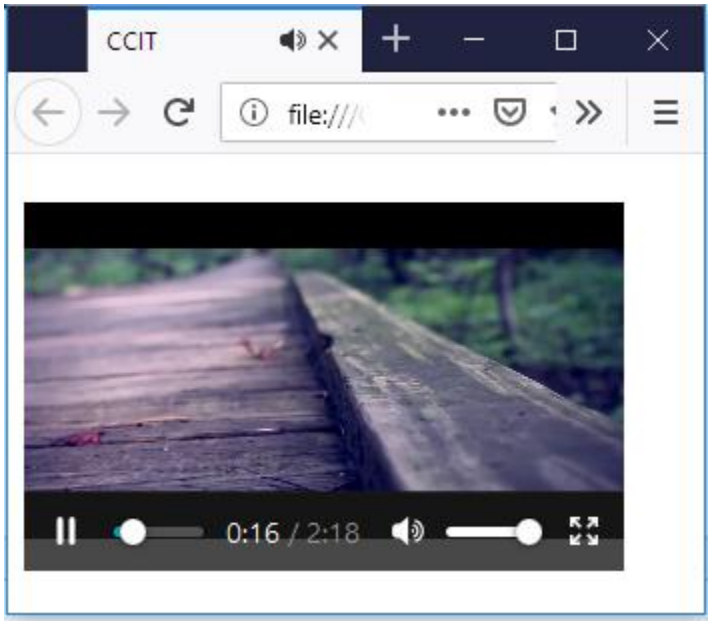
Attribute:

Attribute	Description
Controls	It defines the video controls which is displayed with play/pause buttons.
Height	It is used to set the height of the video player.
Width	It is used to set the width of the video player.
Autoplay	It specifies that the video will start playing as soon as it is ready.
Loop	It specifies that the video file will start over again, every time when it is completed.
Muted	It is used to mute the video output.
Src	It specifies the source URL of the video file.

```
Syntax:
     Syntax:
          <VIDEO controls>
               <source src="File path" >
          </VIDEO>
```

Example

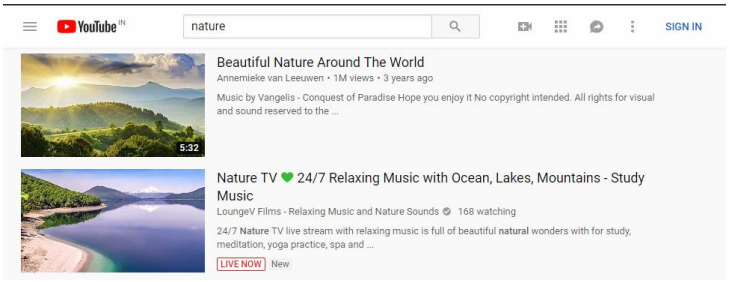
```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
           <VIDEO controls autoplay loop height=200 width=300>
                <source src="Nature.mp4" >
           </VIDEO>
    </body>
</html>
```



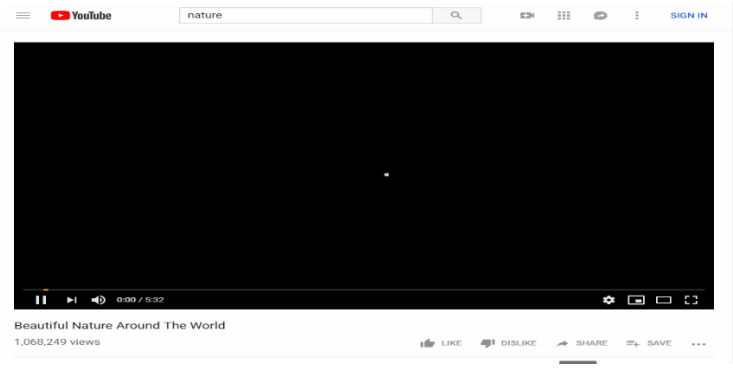
YouTube Videos

You can add YouTube videos to your web site by using either the iframe HTML tags.
To add YouTube Video in website inside follow this steps:

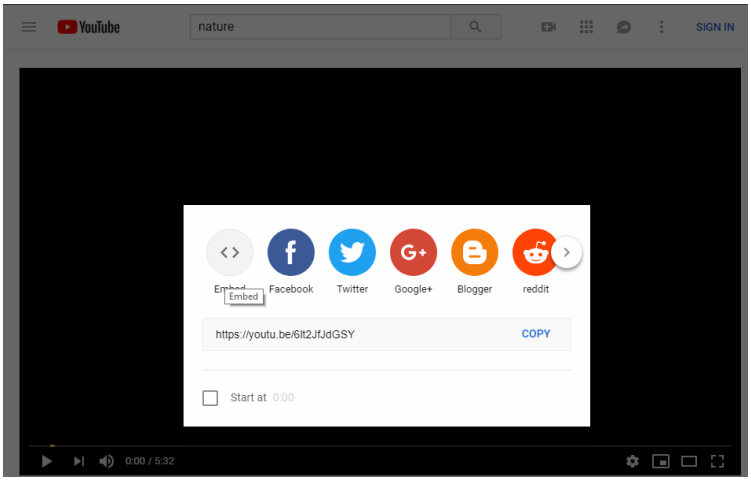
1. Use the youtube site to find the video you want



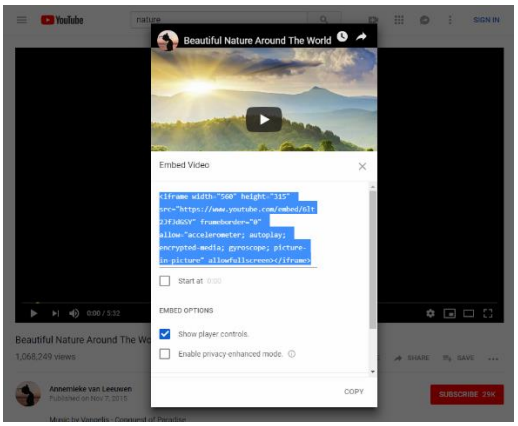
2. Click the 'Share' button below the video



3. Click the 'Embed' button next to the link they show you



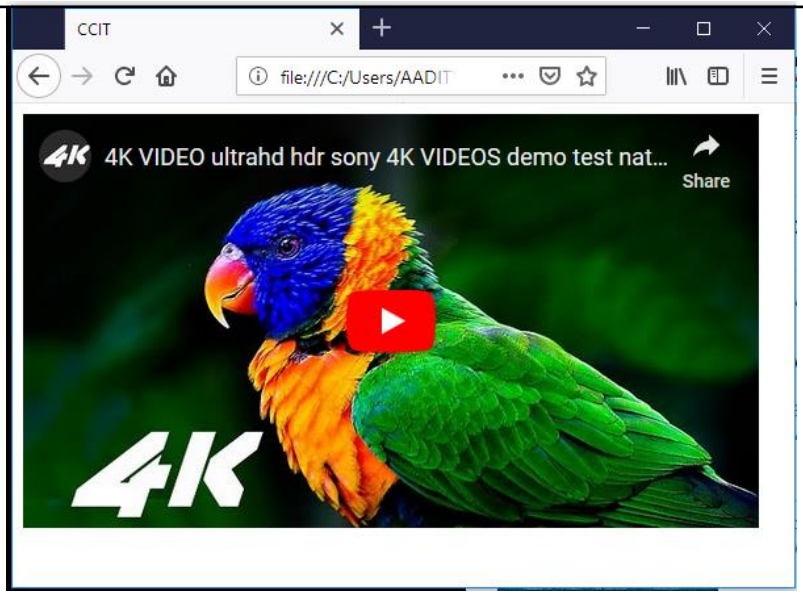
4. Copy the iframe code given and paste it into the html of your web page.



Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <iframe width="560" height="315" src="https://www.youtube-

      nocookie.com/embed/Bey4XXJAqS8"  allowfullscreen>
      </iframe>
    </body>
</html>
```



Hypertext Links

The single greatest feature of the World Wide Web is its diverse collection of documents, which number in the millions. All of these documents are brought together through the use of hypertext links. Users navigate the Web by clicking on the links that HTML authors provide. Hypertext links are a crucial part of HTML-which, after all, is short for Hypertext Markup Language.

Hyperlinks connect two different documents. You can link to one of your own documents or to any other document on the World Wide Web. You can even link to a different section in the same document.

To make a link to another document, you need to use a special type of HTML tag known as an anchor tag, also commonly known as a link tag. Locate the place in your HTML document where you want to insert the hypertext link. Type <A HREF=", followed by the URL of the document you want to link to. Then close the tag by typing ">.

Relative Path Names

If you're linking to different documents on the same Web server (usually your own), you don't always need to use the full URL. You can use relative path names.

- The simplest relative path name is no path name at all. If you're linking to another document that's in the same directory, all you have to do is type in the file name of the new document in place of the full URL. For example, to link to a document named Demo.html, type

  <A HREF="Demo.html">.

- To link to documents or files in a subdirectory, all you need to specify is the path and file name relative to the current document. For example, to link to a document called Demo.html in a subdirectory named Links, you would type

  <A HREF="Links/Demo.html">.

- You can also navigate up the directory tree of your server by using two periods (..) to move up one level. For example, to link from the Demo.html file in the previous example back to the main document, you would type

  <A HREF="../Demo.html">.

A

<A
HREF=*reference*
NAME=*name*
onClick=*function*
onMouseOver=*function*
REL=SAME|NEXT|PARENT|PREVIOUS
STYLE=*style*
TARGET=*window*
TITLE=*title*>
</A>

Stands for anchor. The end-tag is required.

HREF=*reference*

Specifies either a destination address or a destination file. A destination address must be in URL format. A destination file must name a file and be in the format of the given file system. If no path or domain name is specified, the file is searched for in the same location as the current document.

NAME=*name*

Specifies a named reference within an HTML document.

STYLE=*style*

Specifies style information.

TARGET=*window*

Specifies to load the link into the targeted window. This attribute can be used with a frameset where a frame has been named in the FRAME element. The *window* can be one of these values:

window Specifies to load the link into the targeted window. The *window* must begin with an alphanumeric character to be valid, except for the following four target windows:

Blank Load the link into a new blank window. This window is not named.

Parent Load the link into the immediate parent of the document the link is in.

Self Load the link into the same window the link was clicked in.

Top Load the link into the full body of the window.

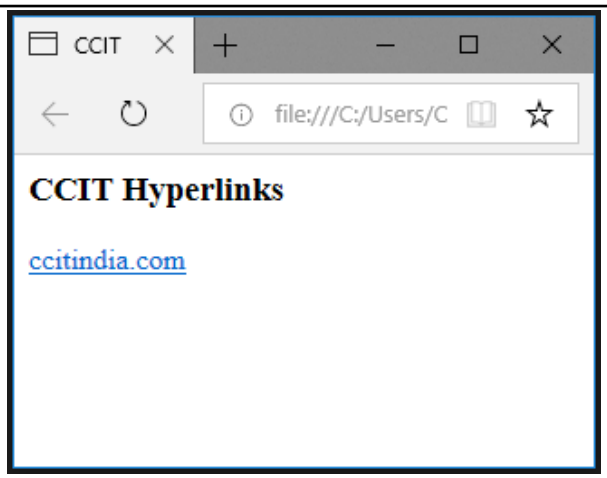
The properties of elements that can follow A are applied to the data characters or elements in the container. The anchor element is used to link text or other elements using the HREF= attribute. The anchor element is used to specify text or graphics as a named reference, to which hyperlinks can link, using the NAME= attribute. Anchors cannot be nested.

Syntax

```
<A href="url"> Symbol… </A>
```

Example

```
<!DOCTYPE html>
<html>
      <head>
            <title>CCIT</title>
      </head>
      <body>
            <h3>CCIT Hyperlinks</h3>
            <a href="http://www.ccitindia.com" >ccitindia.com</a>
      </body>
</html>
```



Links target Attribute

_blank	This value will open link in new tab.
_top	This value will open link in same tab.

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<h3>CCIT local Hyperlinks</h3>
<a href="http://www.ccitindia.com" target="_blank" >Link page</a>
</body>
</html>
```

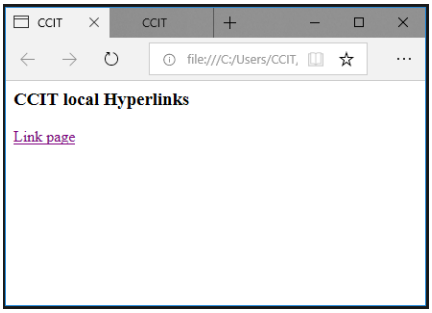


IMAGE AS LINK

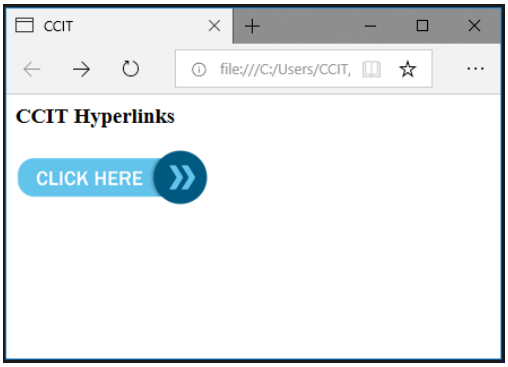
We can use image as hyperlink.

Syntax:

```
<a href="url…" >
      <img src="url…">
</a>
```

Example

```
<!DOCTYPE html>
<html>
      <head>
            <title>CCIT</title>
      </head>
      <body>
            <h3>CCIT Hyperlinks</h3>
            <a href="http://www.ccitindia.com" target="_blank" >
                  <img src="click.png" height="50px">
            </a>
      </body>
</html>
```



DOWNLOAD LINK

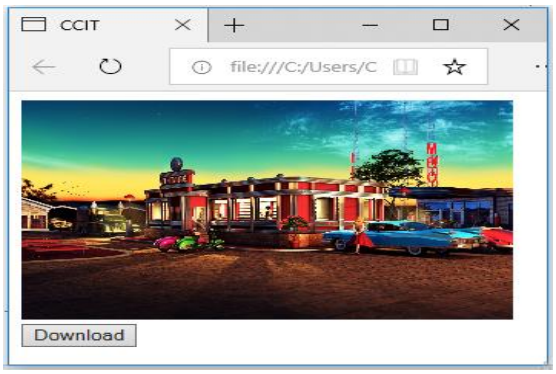
To make a link to download some file than use download attribute.

Syntax:

```
<a href="file url" download >
      content...
</a>
```

Example:

```
<!DOCTYPE html>
<html>
      <head>
            <title>CCIT</title>
      </head>
      <body>
      <img src="img1.jpg" height="200">
      <br>
      <a href="click.png" download  >
            <Button>Download</Button>
      </a>
      </body>
</html>
```



LINKS AS BOOKMARKS

HTML bookmarks are used to allow readers to jump to specific parts of a Web page. For bookmark a section of a page use id attribute.

Syntax:

```
<element id="id_name"></element>
```

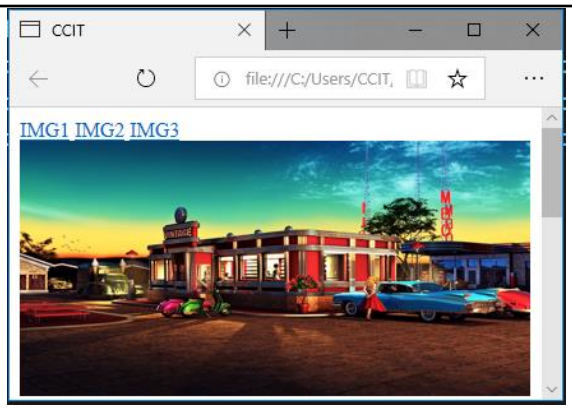
To set hyper link use #element_id

Syntax:

```
<a href="#element_id" >
     Content...
</a>
```

Example

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body>
<h3>CCIT Hyperlinks</h3>
  <a href="#img1">IMG1</a>
  <a href="#img2">IMG2</a>
  <a href="#img3">IMG3</a>
  <img id=img1 src="img1.jpg" height="500">
  <img id=img2 src="img3.jpg" height="500">
  <img id=img3 src="img4.jpg" height="500">
</body>
</html>
```



FRAMES

Frames gives you the power to divide the reader's browser window into multiple panes. You can display different HTML documents in each. More importantly, you can control the display of one frame from another.

The first thing to understand about frames is that they use an entirely new kind of HTML document, called a frame document. Frame documents control the layout and appearance of the frames. Frame documents don't contain any other HTML content. Once you've built your frame document, you can fill the frames with regular HTML documents. But before we get too far ahead of ourselves, let's concentrate on creating a very simple set of frames.

In this section, we'll create an empty frame document. Actually, the frame document is not empty. It will only appear empty when viewed with the browser, because we won't be putting any regular HTML documents inside.

FRAMESET

<FRAMESET
COLS=*col-widths*
FRAMEBORDER=1|0
FRAMESPACING=*spacing*
ROWS=*row-heights***>**
</FRAMESET>

Hosts the FRAME, FRAMESET, and NOFRAMES elements. FRAMESETs can be nested within each other to have layouts within a frame. Each frameset can exist at the same level as a frame.

The frameset element holds one or more frame elements. Each frame element can hold a separate document.

Syntax:

```
<FRAMESET>
  ---------
  ---------
</FRAMESET>
```

Attribute:

COLS=*col-widths*	Specifies the number and size of columns in a frameset Col-widths: *pixels*,%,*
FRAMEBORDER=1|0	Provides the option to display border for a frame.
ROWS=*row-heights*	Specifies the number and size of rows in a frameset rows-height: *pixels*,%*

FRAME

<FRAME
FRAMEBORDER=1|0
NAME=*name*
NORESIZE
SCROLLING=yes|no
SRC=*address***>**

Defines a single frame in a frameset. There is no matching end-tag.

FRAMEBORDER=1|0	Renders a 3-D edge border around the frame. 1 (default) inserts a border. 0 displays no border.
NAME=*name*	Provides a target name for the frame.
NORESIZE	Prevents the user from resizing the frame.
SCROLLING=yes|no	Creates a scrolling frame.
SRC=*address*	Displays the source file for the frame.

Syntax:

```
<FRAMESET>
      <FRAME SRC=url>
      <FRAME SRC=url>
</FRAMESET>
```

Example

Main.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<frameset rows="20%,300px,*">
  <frame src=Frame1.html>
  <frame src=Frame2.html>
  <frame src=Frame3.html>
</frameset>
</html>
```

Frame1.html

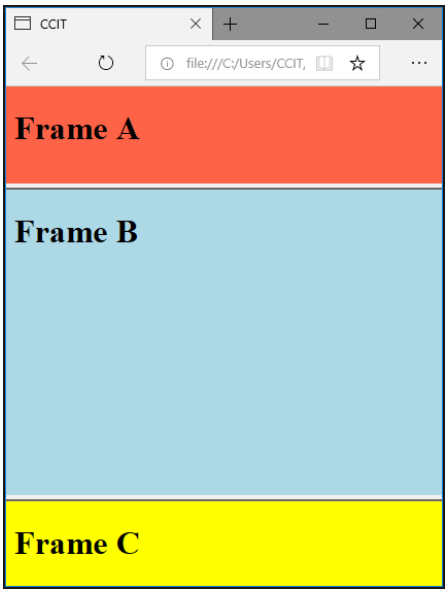
```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=tomato >
<H1>Frame A</H1>
</body>
</html>
```

Frame2.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=lightblue >
<H1>Frame B</H1>
</body>
</html>
```

Frame3.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=Yellow >
<H1>Frame C</H1>
</body>
</html>
```



Main.html

```
<!DOCTYPE html>
<html>
<head>
  <title>CCIT</title>
</head>
<frameset rows="30%,*">
  <frame src=Frame1.html >
  <frameset cols="50%,50%">
    <frame src=Frame2.html >
    <frame src=Frame3.html >
  </frameset>
</frameset>
</html>
```

Frame1.html

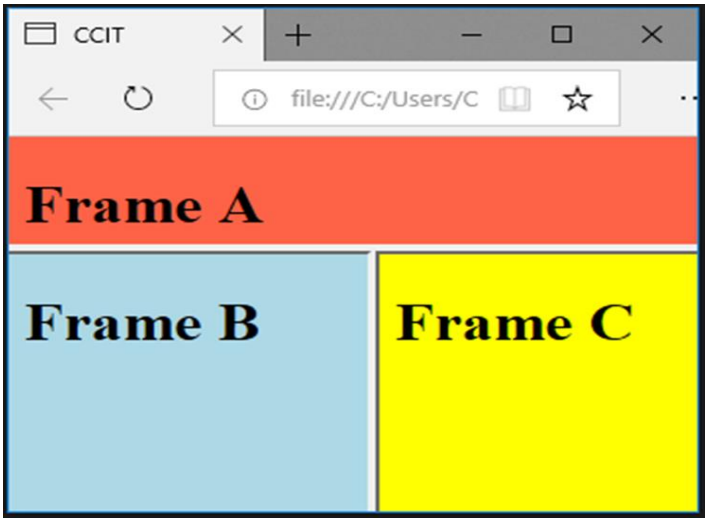
```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=tomato >
<H1>Frame A</H1>
</body>
</html>
```

Frame2.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=lightblue >
<H1>Frame B</H1>
</body>
</html>
```

Frame3.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=Yellow >
<H1>Frame C</H1>
</body>
</html>
```



Main.html

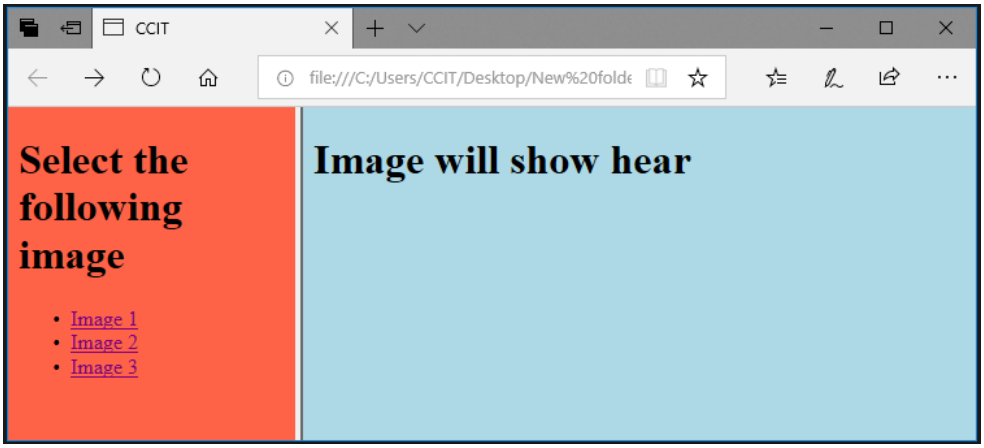
```
<!DOCTYPE html>
<html>
<head>
  <title>CCIT</title>
</head>
<frameset cols="30%,*">
  <frame src=Frame1.html >
  <frame src=Frame2.html name="frm2">
</frameset>
</html>
```

Frame1.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=tomato >
<H1>Select the following image</H1>
<ul>
<li><a href="img1.jpg"
target="frame2">Image 1</a></li>
<li><a href="img3.jpg"
target="frame2">Image 2</a></li>
<li><a href="img4.jpg"
target="frame2">Image 3</a></li>
</ul>
</body>
</html>
```

Frame1.html

```
<!DOCTYPE html>
<html>
<head>
<title>CCIT</title>
</head>
<body bgcolor=lightblue >
<H1>Image will show hear</H1>
</body>
</html>
```



IFRAME

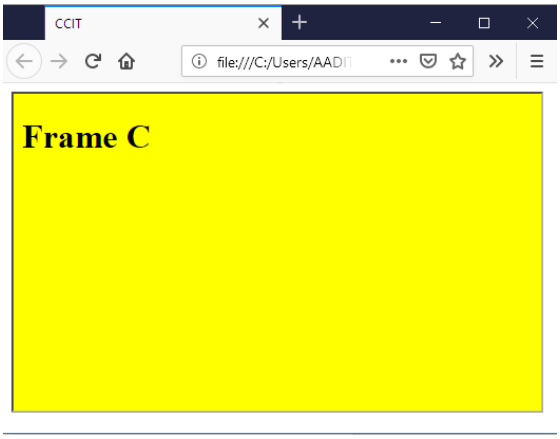
```
<IFRAME
ALIGN=LEFT|CENTER|RIGHT|TOP|BOTTOM
FRAMEBORDER=1|0
MARGINHEIGHT=mheight
MARGINWIDTH=mwidth
HEIGHT=height
WIDTH=width
NAME=name
NORESIZE
SCROLLING=yes|no
SRC=address>
```

Defines a inline frame or a floating frame.

FRAMEBORDER=1|0	Renders a 3-D edge border around the frame. 1 (default) inserts a border. 0 displays no border.
MARGINHEIGHT=*height*	Controls the margin height for the frame, in pixels.
MARGINWIDTH=*width*	Controls the margin width for the frame, in pixels.
HEIGHT=*height*	Controls the height for the frame, in pixels.
WIDTH=*width*	Controls the width for the frame, in pixels.
NAME=*name*	Provides a target name for the frame.
NORESIZE	Prevents the user from resizing the frame.
SCROLLING=yes|no	Creates a scrolling frame.
SRC=*address*	Displays the source file for the frame.

Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <iframe src="FrameC.html"
height=300 width="500">
    </iframe>
  </body>
</html>
```



SVG

Scalable Vector Graphics (SVG) is an XML-based vector image format for two-dimensional graphics with support for interactivity and animation. The SVG specification is an open standard developed by the World Wide Web Consortium (W3C) since 1999.

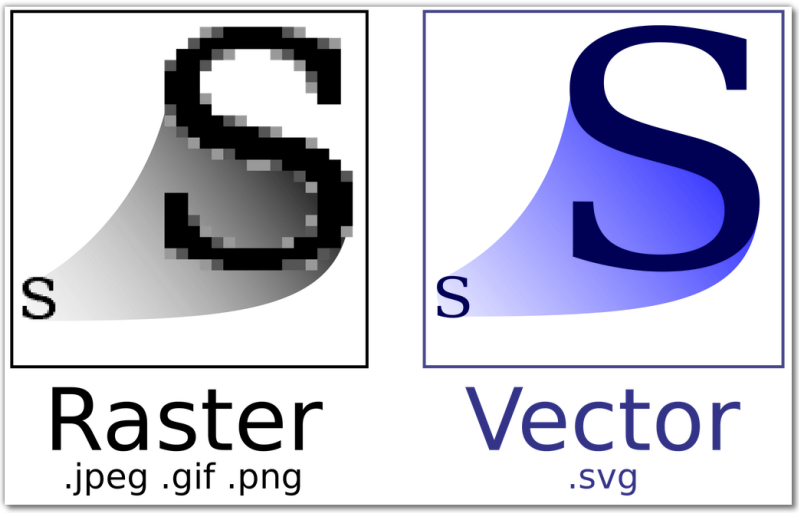
SVG images and their behaviors are defined in XML text files. This means that they can be searched, indexed, scripted, and compressed. As XML files, SVG images can be created and edited with any text editor, as well as with drawing software.

SVG drawings and images are created using a wide array of elements which are dedicated to the construction, drawing, and layout of vector images and diagrams. Here you'll find reference documentation for each of the SVG elements.

SVG Advantages

Advantages of using SVG over other image formats (like JPEG and GIF) are:

- SVG images can be created and edited with any text editor.
- SVG images can be searched, indexed, scripted, and compressed.
- SVG images are scalable.
- SVG images can be printed with high quality at any resolution.
- SVG images are zoom able.
- SVG graphics do NOT lose any quality if they are zoomed or resized.
- SVG is an open standard.
- SVG files are pure XML.



SVG

```
<SVG
HEIGHT=height
WIDTH=width
></SVG>
```

In HTML5, you can embed SVG elements directly into your HTML pages. By using SVG tag.

Attribute:

Height	It sets the height of svg container.
Width	It sets the width of svg container.

Syntax:

```
<SVG>
  -------
</SVG>
```

RECTANGLE

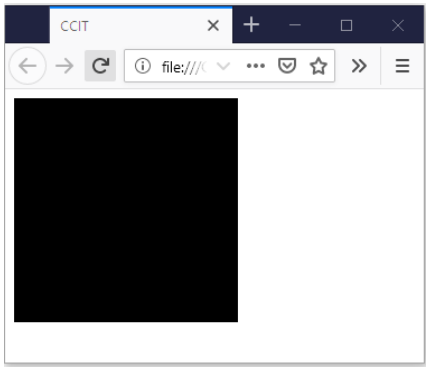
The <rect> element is a basic SVG shape that creates rectangles, defined by their corner's position, their width, and their height. The rectangles may have their corners rounded.

Attribute:

Height	It sets the height of rectangle.
width	It sets the width of rectangle.
X	The x position of corner of the rectangle.
Y	The y position of corner of the rectangle.
RX	The x radius of the corners of the rectangle
RY	The y radius of the corners of the rectangle

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
            <rect height=200 width=200></rect>
        </SVG>
    </body>
</html>
```



FILL & STROKE ATTRIBUTE

SVG offers a wide range of stroke properties.

Fill=*color*

Fill sets the color inside the object.

Stroke=*color*

stroke sets the color of the line drawn around the object.

stroke-width=*size*

The stroke-width property defines the width of this stroke.

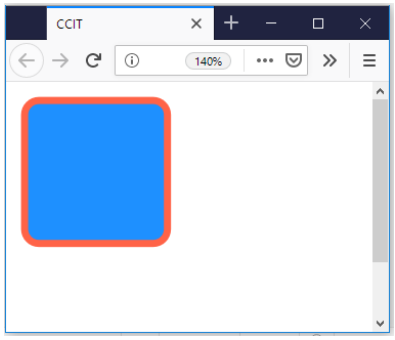
stroke-linejoin=*value*

The joint where the two meet is controlled by the stroke-linejoin attribute.

Value: *round|bevel*

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
<rect height=100 width=100 fill=dodgerblue rx=10
ry=10 x=5 y=5 stroke=tomato stroke-width=5 ></rect>
        </SVG>
    </body>
</html>
```



CIRCLE

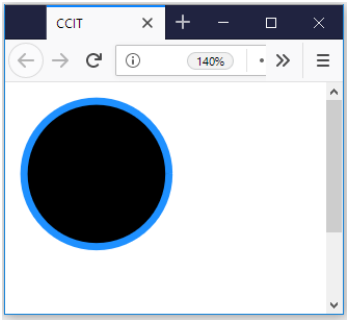
The <circle> SVG element is an SVG basic shape, used to create circles based on a center point and a radius.

Attribute:

CX	The x position of the center of the circle.
CY	The y position of the center of the circle.
R	The radius of the circle.

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
<circle cx=55 cy=55 r=50 stroke=dodgerblue stroke-
width=5 rx=10 ry=10 ></circle>
        </SVG>
    </body>
</html>
```



ELLIPSE

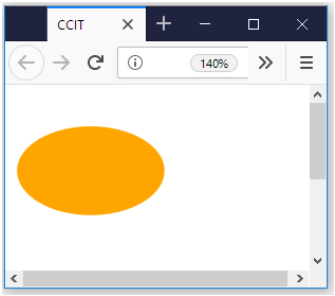
The <ellipse> element is an SVG basic shape, used to create ellipses based on a center coordinate, and both their x and y radius.

Attribute:

CX	The x position of the ellipse.
CY	The y position of the ellipse.
RX	The radius of the ellipse on the x axis.
RY	The radius of the ellipse on the y axis.

Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
            <ellipse rx=50 ry=30 cx=50 cy=50
fill=orange></ellipse>
        </SVG>
    </body>
</html>
```



LINE

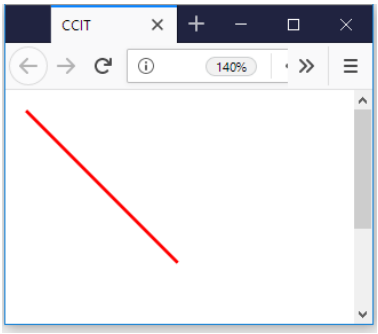
The <line> element is an SVG basic shape used to create a line connecting two points.

Attribute:

X1	Defines the x-axis coordinate of the line starting point.
Y1	Defines the y-axis coordinate of the line starting point.
X2	Defines the x-axis coordinate of the line ending point.
Y2	Defines the y-axis coordinate of the line ending point.

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
   <line x1=5 y1=5 x2=100 y2=100 stroke=red stroke-
width=2> </line>
        </SVG>
    </body>
</html>
```



TEXT

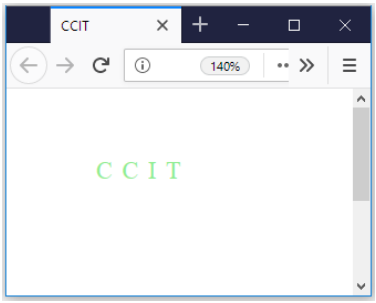
The SVG <text> element defines a graphics element consisting of text.

Attribute:

X	Set X-axis Coordinate.
Y	Set Y-axis Coordinate.
textLength	This attribute lets specify the width into which the text will be drawn.

Example:

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
            <text x=50 y=50 textlength=55
fill=lightgreen>CCIT</text>
        </SVG>
    </body>
</html>
```



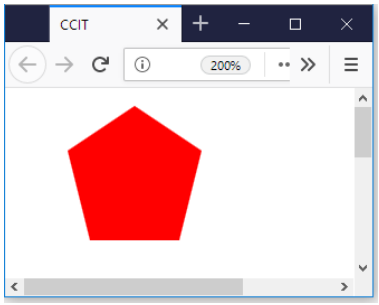
POLYGON

The <polygon> element defines a closed shape consisting of a set of connected straight line segments. To draw polygon must content at least 3 coordinate.

Attribute:

Points	This attribute defines the list of points (pairs of x,y absolute coordinates) required to draw the polygon.

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
            <polygon points="50,0 80,20 70,60
30,60 20,20" fill=red></polygon>
        </SVG>
    </body>
</html>
```



POLYLINE

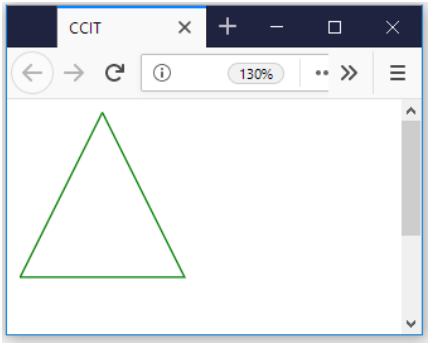
The <polyline> SVG element is an SVG basic shape that creates straight lines connecting several points.

Attribute:

Points	This attribute defines the list of points (pairs of x,y absolute coordinates) required to draw the polygon.

Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
        <SVG height=200 width=200>
            <polygon points="50,0 80,20 70,60
30,60 20,20" fill=red></polygon>
        </SVG>
    </body>
</html>
```



PATH

The <path> SVG element is the generic element to define a shape. All the basic shapes can be created with a path element.

Attribute:

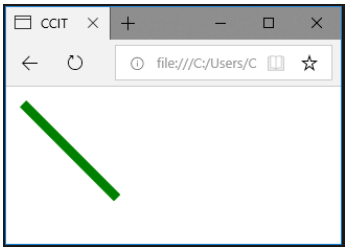
D	The "d" attribute contains a series of commands and parameters used by those commands.

LINE COMMANDS OF PATH

M x,y	The M command appears at the beginning of paths to specify where the drawing should start.
L x,y	A line from the current position to a new position.

Example

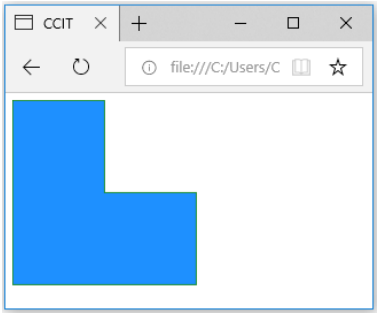
```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <svg height=200 width=200>
        <path d="M 10,10 L 110,110"
stroke=green stroke-width=10 ></path>
    </svg>
  </body>
</html>
```



H x	draws a horizontal line.
V y	draws a vertical line.
Z	This command draws a straight line from the current position back to the first point of the path.

Example

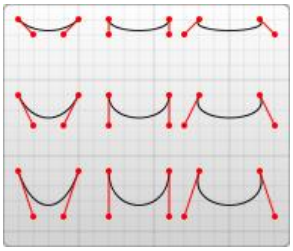
```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <svg height=200 width=200>
        <path d="M 0,0 h 100 v 100 h 100 v 100
h -200 z "stroke=green fill=dodgerblue></path>
    </svg>
  </body>
</html>
```



Curve commands of path tags

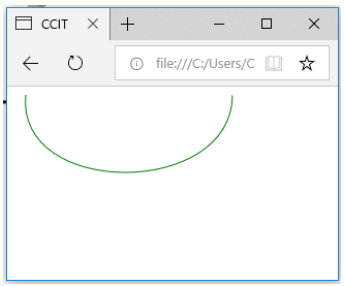
The cubic curve, C, is the slightly more complex curve. Cubic Beziers take in two control points for each point.

Syntax: **C x1 y1, x2 y2, x y**



Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <svg height=200 width=200>
        <path d="M 10,0 C 0 100,210 100,210 0
" stroke=green fill=none></path>
    </svg>
  </body>
</html>
```



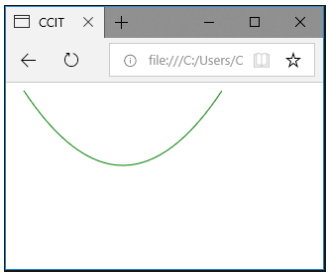
Curve commands of path tags

The quadratic curve called with Q, is actually a simpler curve than the cubic one.

Syntax: **Q x1 y1, x y**

Example

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <svg height=200 width=200>
        <path d="M 10,0 Q 110 150,210 0  "
stroke=green fill=none></path>
    </svg>
  </body>
</html>
```

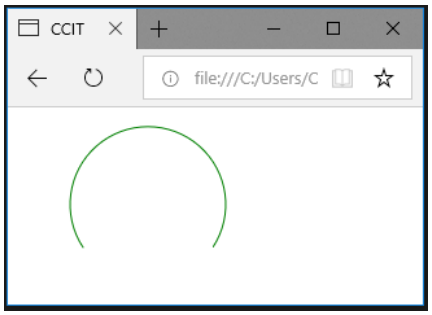


Arcs commands of path tags

The other type of curved line you can create using SVG is the arc, called with A. Arcs are sections of circles or ellipses.

Syntax: `A rx ry x-axis-rotation large-arc-flag sweep-flag x y`

```
<!DOCTYPE html>
<html>
    <head>
        <title>CCIT</title>
    </head>
    <body>
      <svg height=200 width=200>
        <path d="M 50,100 a 60 60 0 1 1 100 00 "
stroke=green fill=none  ></path>
    </svg>
  </body>
</html>
```



FORMS

The biggest tool for allowing your readers to communicate with you via the Web is the *HTML form*. Forms are special collections of markup tags that work with Web servers to produce a means of obtaining whatever information you need from visitors to your Web site. In this chapter, we'll discover how to create a basic form in HTML, as well as how to use all the available types of input fields at your disposal.

HTML Forms are one of the main points of interaction between a user and a web site or application. They allow users to send data to the web site. Most of the time that data is sent to the web server, but the web page can also intercept it to use it on its own.

An HTML Form is made of one or more widgets. Those widgets can be text fields (single line or multiline), select boxes, buttons, checkboxes, or radio buttons. Most of the time those widgets are paired with a label that describes their purpose — properly implemented labels are able to clearly instruct both sighted and blind users on what to enter into a form input.

The main difference between a HTML form and a regular HTML document is that most of the time, the data collected by the form is sent to a web server.

FORM

```
<FORM
ACTION=url
METHOD=GET|POST>
</FORM>
```

Denotes a form. The end-tag is required.

ACTION=*url* Specifies the address to be used to carry out the action of the form. If none is specified, the base URL of the document is used.

METHOD=GET|POST

Indicates how the form data should be sent to the server. The default is GET.

GET Appends the arguments to the action URL and opens it as if it were an anchor.

POST Sends the data via an HTTP post transaction.

LABEL

```
<label
For=id>
</label>
```

The label tag defines a label for an element.

Attribute:

For	Specifies which form element a label is bound to.

INPUT

The HTML <input> element is used to create interactive controls for web-based forms in order to accept data from the user; a wide variety of types of input data and control widgets.

Syntax:

```
<form>
   <input type=value >
</form>
```

Attribute:

Attribute	Value	Description
`checked`		If type = "radio" or type = "checkbox" it will already be selected when the page loads.
`disabled`		Disables the input control.
`maxlength`	number	Defines the maximum number of characters allowed in a text field
`Name`	Text	Assigns a name to the input control.
`readonly`		Sets the input control to read-only. It won't allow the user to change the value.
`Size`	number	Specifies the width of the control.

Src	URL	Defines the URL of the image to display. Used only for type = "image".
Value	Text	Specifies the intial value for the control.If type = "checkbox" or type = "radio" this attribute is required.

Value of type attributes can be:

button	Defines a clickable button
text	Default. Defines a single-line text field (default width is 20 characters)
Password	Defines a password field (characters are masked)
Checkbox	Defines a checkbox
radio	Defines a radio button
Submit	Defines a submit button
reset	Defines a reset button (resets all form values to default values)
file	Defines a file-select field and a "Browse..." button (for file uploads)
hidden	Defines a hidden input field
image	Defines an image as the submit button
month	Defines a month and year control
Number	Defines a field for entering a number
range	Defines a control for entering a number whose exact value is not important (like a slider control). Default range is from 0 to 100.
color	Defines a color picker
date	Defines a date control (year, month and day (no time))
Email	Defines a field for an e-mail address
search	Defines a text field for entering a search string
tel	Defines a field for entering a telephone number
Time	Defines a control for entering a time (no time zone)
url	Defines a field for entering a URL
week	Defines a week and year control (no time zone)

ID Attribute

The id attribute specifies a unique id for an HTML element .Specifies a unique id for the element.

Naming rules:

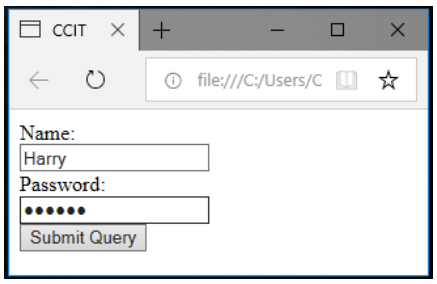
- Must contain at least one character.
- Must not contain any space characters.

Syntax:

```
<elements id="value"></element>
```

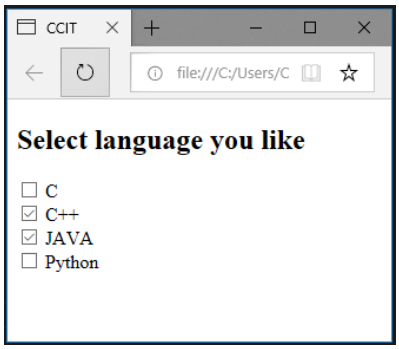
Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <Label for=name >Name: </Label><br>
      <input type="text" id="name" ><br>
      <Label for=pass >Password: </Label><br>
      <input type="password" id="pass" ><br>
      <input type=submit>
    </form>
  </body>
</html>
```



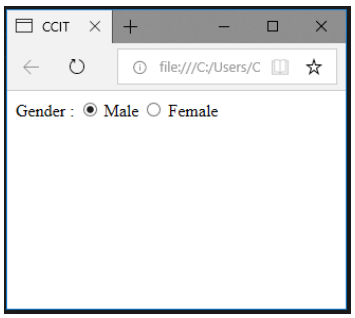
Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <input type="checkbox" id=c>
      <Label for=c >C </Label><br>
      <input type="checkbox" id=cpp>
      <Label for=cpp >C++ </Label><br>
      <input type="checkbox" id=java>
      <Label for=java >JAVA </Label><br>
      <input type="checkbox" id=py>
      <Label for=py >Python </Label>
    </form>
  </body>
</html>
```



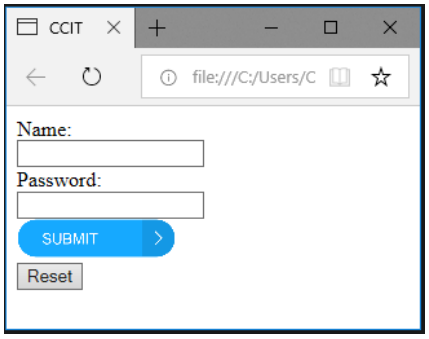
Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      Gender :
      <input type="radio" id=m name=gen>
      <Label for=m >Male </Label>
      <input type="radio" id=f name=gen>
      <Label for=f >Female </Label>
    </form>
  </body>
</html>
```



Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <Label for=name >Name: </Label><br>
      <input type="text" id="name" ><br>
      <Label for=pass >Password: </Label><br>
      <input type="password" id="pass" ><br>
      <input type=image src="Submit.png"
height="30px"><br>
      <input type="reset">
    </form>
  </body>
</html>
```



TEXTAREA

The HTML <textarea> element represents a multi-line plain-text editing control.

Syntax:

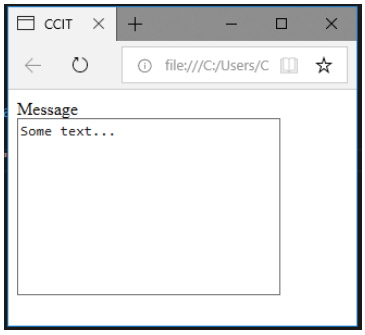
```
<textarea> Write something here </textarea>
```

Attribute:

cols	The visible width of the text control, in average character widths. If it is specified, it must be a positive integer. If it is not specified, the default value is 20.
name	The name of the control.
rows	The rows attribute specifies the visible number of lines in a text area.

Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <label for="txta">Message</label><br>
      <textarea rows="10" cols="30" id=txta >some
text...</textarea><br>
    </form>
  </body>
</html>
```



SELECT

<SELECT
NAME=*name*
SIZE=*n*>
</SELECT>

Denotes a list box or drop-down list. The end-tag encloses any OPTION elements that may appear within the SELECT element.

NAME=*name* Specifies a name for the list box or drop-down list.

SIZE=*n* Specifies the height of the list control.

Syntax:

```
<SELECT>
  -----------
  -----------
</SELECT>
```

OPTION

<OPTION
SELECTED
VALUE=*value*>

Denotes one choice in a list box. In a SELECT block, denotes one of the choices that will appear in the list.

SELECTED Indicates that this item is the default.

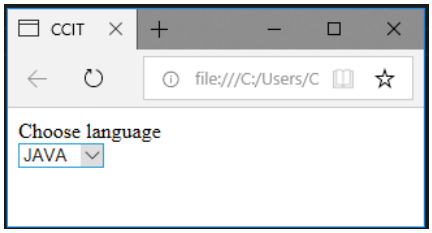
VALUE=*value* Indicates the value that will be returned if this item is chosen.

Syntax:

```
<SELECT >
      <OPTION> Text… </OPTION>
      <OPTION> Text… </OPTION>
</SELECT>
```

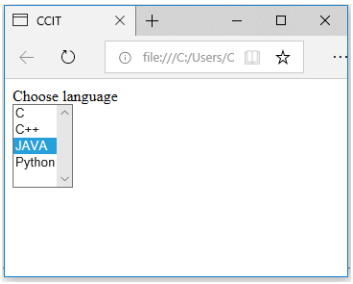
Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <label for="sel">Choose language</label><br>
      <select id=sel size=5>
        <option value=1>C</option>
        <option value=2>C++</option>
        <option value=3>JAVA</option>
        <option value=4>Python</option>
      </select>
    </form>
  </body>
</html>
```



Example:

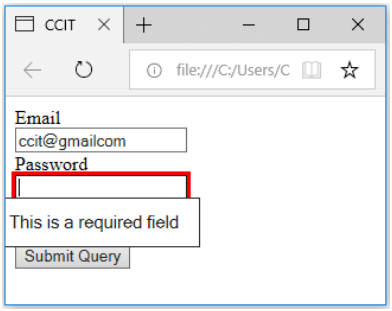
```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <label for="sel">Choose language</label><br>
      <select id=sel size=5>
        <option value=1>C</option>
        <option value=2>C++</option>
        <option value=3>JAVA</option>
      </select>
    </form>
  </body>
</html>
```



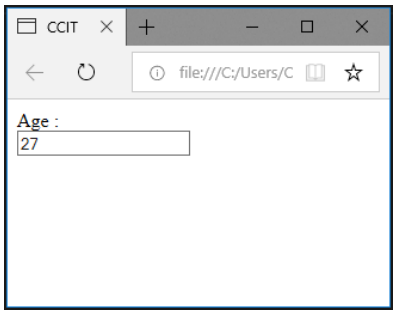
HTML5 Attribute:

Attribute	Value	Description
autofocus		Specifies that the element should automatically get focus when the page loads.
autocomplete	off | on	Specifies whether the <form> or the <input> element should have autocomplete enabled
required		To Make a Input Text Compulsory.
max	Value	The max attribute specifies the maximum value of the element.
min	Value	The min attribute specifies the minimum value of an element.
placeholder	Text	Specifies a short hint that describes the expected value of the element
pattern	regular expression	Specifies a regular expression that an <input> element's value is checked against
multiple		Specifies that a user can enter more than one value
step	Value	input field with a specified legal number intervals
list	Id	Input field with pre-defined values in a <datalist>

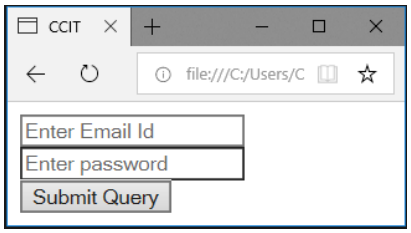
```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <label for=#email >Email</label><br>
      <input id=email type="text" required><br>
      <label for=#pwd >Password</label><br>
      <input id=pwd type="password" required><br>
      <input type="submit">
    </form>
  </body>
</html>
```



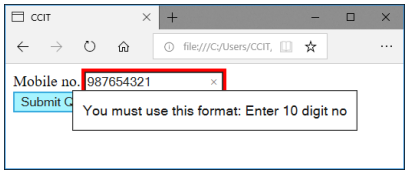

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      <Label for=age >Age : </Label>
      <br>
      <input type="number" id=age max=60 min=18>
      <br>
    </form>
  </body>
</html>
```



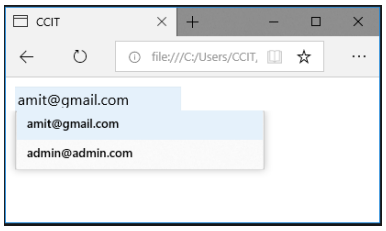
```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
<input type="email" placeholder="Enter Email Id" >
<br>
<input type="password"placeholder="Enter password"
> <br>
<input type="submit">
    </form>
  </body>
</html>
```



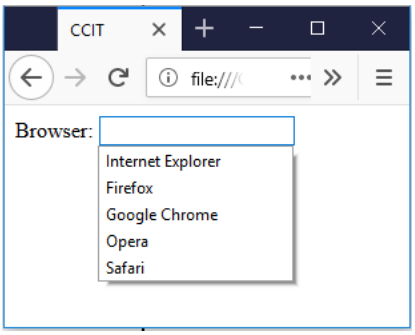

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
<label>Mobile no.</label>
<input type="tel" pattern="[0-9]{10}" title="Enter
10 digit no" ><br>
<input type="submit">
    </form>
  </body>
</html>
```




```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
<input type="email" name=email autocomplete="on"
><br>
<input type="submit">
    </form>
  </body>
</html>
```



```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <form>
      Browser: <input list="browsers" >
      <datalist id="browsers">
        <option value="Internet Explorer">
        <option value="Firefox">
        <option value="Google Chrome">
        <option value="Opera">
        <option value="Safari">
      </datalist>
    </form>
  </body>
</html>
```



PROGRESS

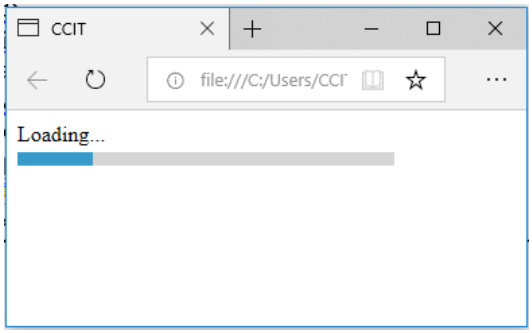
The HTML <progress> element displays an indicator showing the completion progress of a task, typically displayed as a progress bar.

Attributes:

`max`	This attribute describes how much work the task indicated by the progress element requires. The default value is 1.
`value`	This attribute specifies how much of the task that has been completed. It must be a valid floating point number between 0 and max, or between 0 and 1 if max is omitted.

Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
      Loading...<br>
      <progress value=20 max=100 ></progress>
  </body>
</html>
```



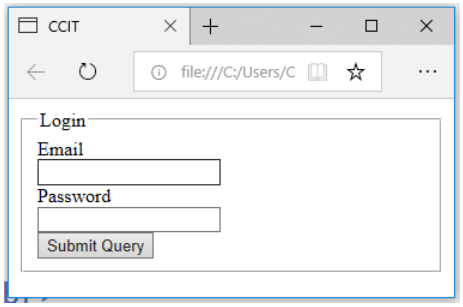
FIELDSET

The fieldset tag is used to group related elements in a form.
The legend tag defines a caption for the fieldset element.

Syntax:

```
<fieldset>
     <legend> content... </legend>
     -------------
     -------------
<fieldset>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
   <fieldset>
      <legend>Login</legend>
      <form>
         <label for="#email">Email</label><br>
         <input id="email" type="email"><br>
         <label for="#pwd">Password</label><br>
         <input id="pwd" type="password"><br>
         <input type="submit">
      </form>
   </fieldset>
  </body>
</html>
```



Head

The head of an HTML document is the part that is not displayed in the web browser when the page is loaded. It contains information such as the page <title>, links to CSS if you choose to style your HTML content with CSS, links to custom favicons, and other metadata data about the HTML, such as the author, and important keywords that describe the document.

Syntax:

```
<head>
  ----------
  ----------
</head>
```

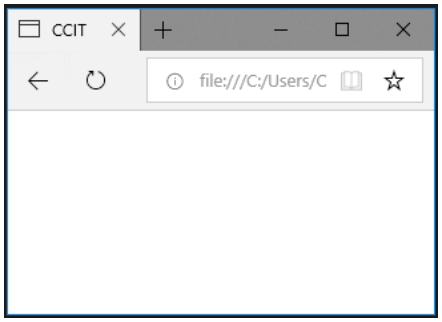
TITLE

The HTML Title element (<title>) defines the document's title that is shown in a browser's title bar or a page's tab. It only contains text and tags within the element are ignored.

Syntax:

```
<head>
     <title>text…</title>
</head>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
  </body>
</html>
```



LINK

The HTML External Resource Link element (<link>) specifies relationships between the current document and an external resource. This element is most commonly used to

link to stylesheets, but is also used to establish site icons (both "favicon" style icons and mobile home screen/app icons) among other things.

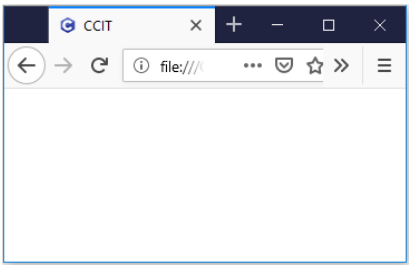
TITLE WEBSITE LOGO

To set webpage browser tab icon use link.

```
Syntax:    <head>
             <link rel=icon href=url>
           </head>
Attribute:
```

rel	Specifies the relationship between the current document and the linked document

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
    <link rel="icon" href="c.png">
  </head>
  <body>
  </body>
</html>
```



CSS LINK

The link element is used to link to external style sheets.

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
  </body>
</html>
```

META

The Meta element is used to specify which character set is used, page description, keywords', author, and other metadata.

Syntax: `<meta attribute=value>`

Attribute:

Name	Specifies a name for the metadata
Content	specifies the actual meta content.

Character set

Character set defined different alphanumeric characters that could be used on the internet.

Such as : Numbers (0-9), English letters (A-Z), and some special characters like ! $ + - ( ) @ < > .

Syntax: `<meta charset="UTF-8">`

Character encoding standards

ASCII	ASCII was the first character encoding standard . There are 128 different alphanumeric characters.
ANSI	ANSI was the original Windows character set It support 256 different alphanumeric characters.
UTF-8	UTF-8 (Unicode) covers almost all of the characters and symbols in the world.

Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
    <meta charset="UTF-8">
  </head>
  <body>
  </body>
</html>
```

Meta elements are typically used to specify page description, keywords, author of the website.

Description

Define a description of your web page.

Syntax: **<meta name="description" content="content…">**

Keywords

Define keywords for search engines.

Syntax: **<meta name="keywords" content="Keywords,keywords,...">**

Author

Define the author of a page.

Syntax: **<meta name="author" content="name">**

Refresh

Refresh document every 30 seconds.

Syntax: **<meta http-equiv="refresh" content="value">**

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <meta name="description" content="Welcome to HTML5">
    <meta name="keywords" content="HTML5,CSS3,JavaScript">
    <meta name="author" content="Harry Potter">
    <meta http-equiv="refresh" content="30">
    <title>CCIT</title>
  </head>
  <body>
    <h1>Welcome to HTML5 </h1>
  </body>
</html>
```

VIEWPORT

HTML5 introduced a method to let web designers take control over the viewport, through the <meta> tag.

The viewport is the user's visible area of a web page. It varies with the device, and will be smaller on a mobile phone than on a computer screen.

Syntax:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width,initial-
scale=1.0">
  </head>
  <body>
    <h1>Welcome to HTML 5 </h1>
    <div>
HTML5 introduced a method to let web designers take control over the
viewport, through the meta tag.
The viewport is the user's visible area of a web page. It varies with
the device, and will be smaller on a mobile phone than on a computer
screen.
    </div>
  </body>
</html>
```

Without Viewport

With Viewport

Website Language

The HTML Lang attribute can be used to declare the language of a Web page or a portion of a Web page. This is meant to assist search engines and browsers.

Syntax:
```
<html lang="language">
   ----------
</html>
```

Value	Language
En	English
Hi	Hindi
Mr	Marathi
Ta	Tamil
Ja	Japanese
Sa	Sanskrit

Website Country Code

In HTML they can be used as an addition to the language value in the lang attribute.

The first two characters of a language code defines the language.

The last two defines the country.

Syntax:
```
<html lang="language-country">
   ----------
</html>
```

Value	Country
IN	INDIA
US	UNITED STATES
GB	UNITED KINGDOM
JP	JAPAN

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
     <title>CCIT</title>
  </head>
  <body>
    <h1>Welcome to HTML 5 </h1>
  </body>
</html>
```

HTML Entity

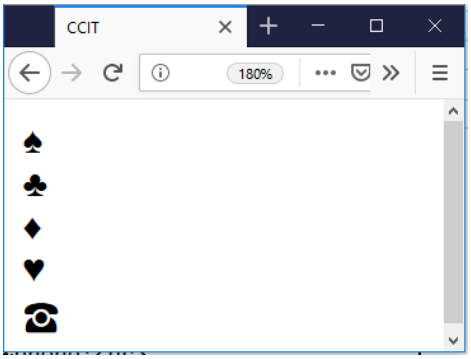
HTML entities are used to display reserved characters in HTML. Characters that are not present on your keyboard can also be replaced by entities.

An HTML entity is a piece of text ("string") that begins with an ampersand (&) and ends with a semicolon (;) . Entities are frequently used to display reserved characters (which would otherwise be interpreted as HTML code), and invisible characters (like non-breaking spaces). You can also use them in place of other characters that are difficult to type with a standard keyboard.

Syntax: `&entity_name;`
or
`&#entity_number;`

Description	Entity Name	Entity Number
non-breaking space		
less than	<	<
greater than	>	>
Cent	¢	¢
Pound	£	£
Yen	¥	¥
Euro	€	€
Rupee		₹
Copyright	©	©
Pie	π	π
Spade	♠	♠
Club	♣	♣
Heart	♥	♥
Diamond	♦	♦
TELEPHONE	☎	☎
MALE SIGN	♂	♂
FEMALE SIGN	♀	♀
CHECK MARK	✓	✓
CROSS MARK	✗	✗

```
<!DOCTYPE html>
<html>
  <head>
      <title>CCIT</title>
  </head>
  <body>
&spades;<br> &clubs;<br> &diamondsuit;<br>
&hearts;<br> &phone;<br>
  </body>
</html>
```



HTML Semantic

A semantic element clearly describes its meaning to both the browser and the developer. Semantic HTML elements clearly describe its meaning in a human and machine readable way. Elements such as <header>, <footer> and <article> are all considered semantic because they accurately describe the purpose of the element and the type of content that is inside them.

non-semantic elements: and - Tells nothing about its content.

semantic elements: <form>, <table>, and <article> - Clearly defines its content.

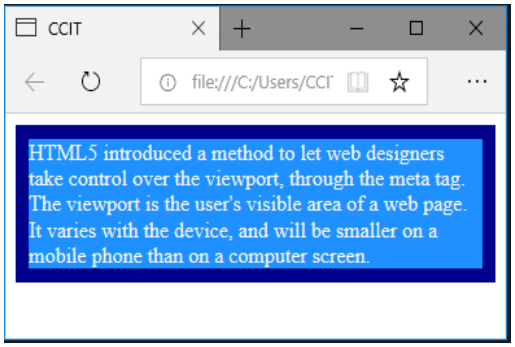
DIV

The HTML Content Division element ( ) is the generic container for flow content. It has no effect on the content or layout until styled using CSS. The tag defines a division or a section in an HTML document. The element is often used as a container for other HTML elements to style them with CSS.

Syntax: ** Content... **

Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <div style=”border: solid 10px darkblue;
background-color:dodgerblue; color:
whitesmoke;”>
        HTML5 introduced a method to let web
designers take control over the viewport, through
the meta tag.
        The viewport is the user's visible area of
a web page. It varies with the device, and will be
smaller on a mobile phone than on a computer
screen.
    </div>
  </body>
</html>
```



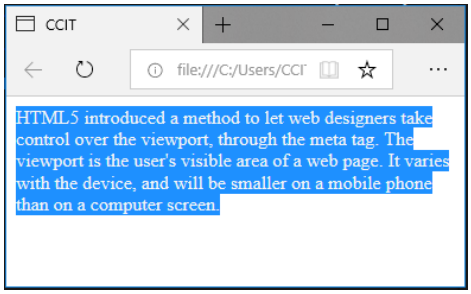
SPAN

The HTML element is a generic inline container for phrasing content, which does not inherently represent anything. The tag is used to group inline-elements in a document. The tag provides no visual change by itself.

Syntax: `<span> content... </span>`

Example:

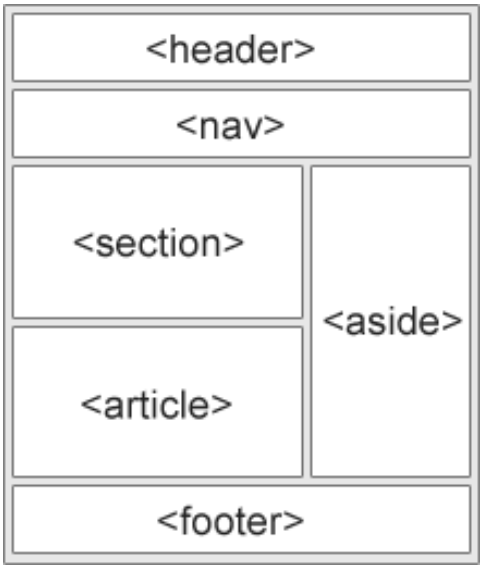
```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <span style=”background-color:dodgerblue;
color: white;”>
        HTML5 introduced a method to let web
designers take control over the viewport, through
the meta tag.
        The viewport is the user's visible area of
a web page. It varies with the device, and will be
smaller on a mobile phone than on a computer
screen.
    </span>
  </body>
</html>
```



Semantic Elements

HTML5 offers new semantic elements to define different parts of a web page.

```
<article>
<aside>
<details>
<figcaption>
<figure>
<footer>
<header>
<main>
<mark>
<nav>
<section>
<summary>
<time>
```



HEADER

The <header> element specifies a header for a document or section. The <header> element should be used as a container for introductory content.

Syntax: `<header> content... </header>`

NAV

The <nav> element defines a set of navigation links.

Syntax: `<nav> content... </nav>`

SECTION

The <section> element defines a section in a document.

Syntax: `<section> content... </section>`

ARTICLE

The<article> element specifies independent, self-contained content.

Syntax: `<article> content... </article>`

FIGURE and FIGCAPTION

The HTML <figure> element represents self-contained content, frequently with a caption (<figcaption>), and is typically referenced as a single unit.

Syntax:

```
<figure>
      <img src=url>
      <figcaption> text... </figcaption>
</figure>
```

FOOTER

The <footer> element specifies a footer for a document or section. A <footer> element should contain information about its containing element.

Syntax: `<footer> content... </footer>`

Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CCIT</title>
  </head>
  <body>
    <header><h1>CCIT</h1></header>
    <Nav>
      <table border="1" width=100%>
        <tr><td>Home</td><td>News</td><td>About</td><td>Contact</td></tr>
      </table>
    </Nav>
    <section>
      <h3>Welcome To CCIT Education HUB</h3>
      <p>Our IT training programs and IT Certifications help professionals
to acquire skills in cutting-edge technologies that are being deployed in
todays organizations and get an edge over their colleagues.</p>
      <figure>
        <img src="ccit.png" height=100>
        <figcaption>CCIT</figcaption>
      </figure>
    </section>
    <footer style="background-color:black;color:white;">
      <p align=center>Copyright &copy; 2019 All rights reserved to CCIT
</p>
    </footer>
  </body>
</html>
```