# Chapter Two: Web Development Using HTML

# 1. Basic HTML Tags

## HTML: Hypertext Markup Language

HTML, which stands for Hypertext Markup Language, is the predominant markup language used for creating web pages. A markup language is a set of markup tags, and HTML uses markup tags to describe web pages. HTML is written in the form of HTML elements consisting of HTML tags surrounded by angle brackets (e.g. <html>) within the web page content. HTML tags normally come in pairs like <b> and </b>. The first tag in a pair is the start tag, the second tag is the end tag; you can also refer to them as opening tags and closing tags.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts in languages such as JavaScript which affect the behavior of HTML webpage. HTML can also be used to include Cascading Style Sheets (CSS) to define the appearance and layout of text.

While a web developer, someone like you, uses the HTML tags to create the web page, software is required to interpret your code and display the information that you are trying to display: the software could be any web browser software. The purpose of a web browser is to read HTML documents and display them as web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

World Wide Web Consortium (W3C) is the organization that develops guidelines and specifications for many web technologies including HTML. The W3C website is found atwww.w3.org*.*

HTML document is created using a simple text editor like notepad or gedit or any other text editor. Notepad can edit HTML, and at the same time you are able to view what you edit, in a web browser.

## HTML, HEAD, TITILE and BODY Tags

The entire web page document is contained within an <html> tag. The <html> tag is called the root element because it contains all the elements in the document, and it cannot not be contained within any other element. Every web page starts with <html> tag and ends with </html>.

Let us first see how a plain html code looks like.

<html>  
<head>  
 <title>Page title here </title>  
</head>  
<body>  
 Our body content here   
</body>  
</html>

The total code is divided into two parts and both the parts are kept inside <html> tags. Our page should start with <html> and should end with </html>. The first part inside this html tags is the head and it starts with <head> and ends with </head>. The second part starts with <body> and ends with </body> tag. Inside the <body> tag we keep all our content which we want to display to our web page users. Whatever we place in <body> will be displayed by the browser to the web users.



Fig Basic HTML tags

Let’s see both pars in detail.

<HEAD>   </HEAD>  
The web page should have only one head tag. The head tag starts with <head> and ends with </head>. The text or tags that are inside the head tag will not be displayed in the browser window. Inside this tag we keep all the meta keyword tags used for search engines.

One of the important tags which is put inside <head> is <title> </title> tag. Title tags are used to give title to the browser window and it displayed at the top left side of the window.  Title tags are also important for our search engine point of view. We should keep most important keywords inside the title tag.

Example:

<html>

<head>

<title> Nuclear Energy </title>

</head>

<body>

Nuclear energy is one of the clean environment friendly energy source.

</body>

</html>

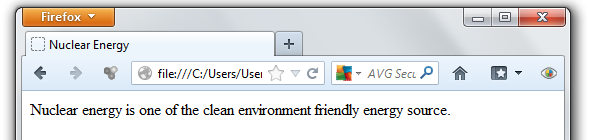


Fig Web page Title

It is also possible to put JavaScript code and Cascading Style Sheets in head section. If we are adding any JavaScript code here then that will be loaded when the browser opens the page.

**Teh <BODY> Element**

This is where we will place our content for our visitors. What we place here will be displayed to our visitors. The style and other formatting of the text and what you could do to fill the content of the web page will discuss in different sections.

A <body> element may contain anything from a couple of paragraphs under a heading to more complicated layouts containing forms and tables.

**The <title> Element**

You should specify a title for every page that you write inside the <title> element. This element is a child of the <head> element. It is used in several ways:

* It displays at the very top of a browser window.
* It is used as the default name for a bookmark in browsers.
* Its is used by search engines that use its content to help index pages.

Example:Here is the example of using title tag.

<head>

<title>HTML Basic tags</title>

</head>

**HTML Tag Attributes**

Attributes provide additional information about HTML tags.

* HTML tags can have attributes
* Attributes provide additional information about a tag
* Attributes are always specified in the start tag
* Attributes come in name/value pairs like: name="value"

The syntax for attributes is as follows:

<element attribute-name="value"> Content </element>

Or for empty elements:

<element attribute-name="value" />

Example: the background color of HTML document can be changed using “bgcolor” attribute of the <body> tag.

**<html> attribute**

**Setting HTML 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.

According to the W3C recommendation you should declare the primary language for each Web page with the lang attribute inside the <html> tag, like this:

<html lang="en">

…

</html>

ISO 639-1 defines abbreviations for languages. In HTML, they can be used in the lang attributes.

|  |  |
| --- | --- |
| **Language** | **ISO Code** |
| Chinese | Zh |
| English | En |
| French | Fr |
| German | De |
| Russian | Ru |
| Local Languages | |
| Afar | Aa |
| Amharic | Am |
| Afan Oromo | Om |
| Somali | So |
| Tigrinya | Ti |

Table ISO code of some international and local languages

**<body> Attributes**

**Background Color**

You can change background color of your web page by using <BODY> tag attribute *bgcolor*

<body bgcolor=”green”>

Color can be specified using color name or RGB value. The following also sets background color to green:

<body bgcolor=”#00FF00”>

**Background Image**

We can use a background picture for web page instead of background color. You must have an image to do this. Then you can use *background* attribute of <BODY> tag as follows:

<body background="image1.gif">

**Text Color**

We can also set the text color of the web page just like background color. We use *text* attribute of <BODY> to do this.

<body bgcolor=”yellow” text="red">

Example:

<HTML>

<HEAD>

<TITLE>Page with Back Color</TITLE>

</HEAD>

<BODY bgcolor="yellow" text="#FF0000">

Page with yellow back color and red text color

</BODY>

</HTML>

Output:

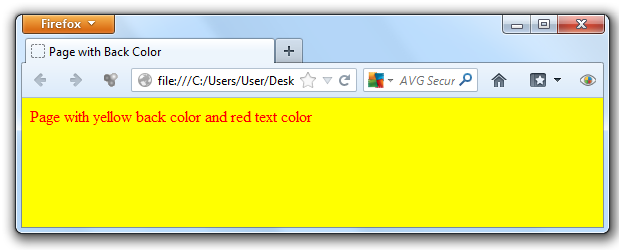


Fig Back and text colors of body

**Other <body> attributes**

* **alink:** Sets the color for active links or selected links.
* **link:** Sets a color for link text.
* **vlink:** Sets a color for *visited links* - that is, for linked text that you have already clicked on.

Example:

<html>

<head>

<title> Link Colors </title>

</head>

<body alink="#00A000" link="#00FF00" vlink="#0000FF">

<a href="first.html">first page</a><br>

<a href="first1.html"> second page</a><br>

<a href="first2.html">third page</a><br>

<a href="first3.html">last page</a><br>

</body>

</html>

**DOCTYPE**

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

A valid HTML document declares what version of HTML is used in the document. The *document type declaration* names the document type definition (DTD) in use for the document.

There are three DTDs for use in HTML 4.01:

***Strict*** - This is used mainly when the markup is very clean and there is no 'extra' markup to aid the presentation of the document. This is best used if you are using Cascading Style Sheets for presentation.

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

***Transitional*** - This should be used if you want to use presentational features of HTML in your page

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/

loose.dtd">

***Frameset*** - This should be used if you want to have frames on your page.

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/

frameset.dtd">

The Doctype should be the very first line of your html document and should be the only thing on that line.

**Meta Tag**

The <meta> tag provides metadata about the HTML document. Metadata is data (information) about data. Metadata will not be displayed on the page, but will be machine parsable.

Meta elements are typically used to specify page description, keywords, author of the document, last modified, and other metadata. The metadata can be used by browsers (how to display content or reload page), search engines (keywords), or other web services.

<meta> tags always goes inside the <head> element. Metadata is always passed as name/value pairs. The content attribute must be defined if the name or the http-equiv attribute is defined. if none of these are defined, the content attribute cannot be defined.

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| http-equiv | content-type | Provides an HTTP header for the information/value of the content attribute |
| default-style |
| Refresh |
| Name | application-name | Specifies a name for the metadata |
| Author |
| Description |
| Generator |
| Keywords |
| Content | *Text* | Gives the value associated with the http-equiv or name attribute |

**The keywords attribute**

Keywords meta tag defines keywords for search engines. It was a critical element for early search engines to index the page. Today, search engines no longer depend on keywords meta tag to index the page.

The structure is as follows:

<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript">

**The description attribute**

The description attribute provides a concise explanation of a Web page's content. The description attribute is supported by most major search engines, like Yahoo! and Bing, while Google will fall back on this tag when information about the page itself is requested.

The structure is as follows:

<meta name="description" content="Free Web tutorials on HTML, CSS, XML and JavaScript">

**The robots attribute**

The robots tag is still one of the most important tags. The robots meta tag lets you specify that a particular page should *not* be indexed by a search engine or if you do or do not want links on the page to be followed.

< meta name="robots" content="noindex, nofollow">

This means “Do not Index this page, do not follow the links on the page.” Your page will drop out of the search index and your links to other pages will not be followed. This tag is most often used when a site is in development.

Other values are:

<meta name="robots" content="index, nofollow">

<meta name="robots" content="noindex, follow">

<meta name="robots" content="index, follow">

**The refresh attribute**

We can tell the web page to refresh itself every given seconds.

<meta http-equiv="refresh" content="30">

We can tell the page to redirect/refresh within the given seconds.

<meta http-equiv="refresh" content="x\_seconds; url=http://www.yourhost.com/pagetosendto.html">

**The content-type attribute**

The content-type attribute specifies the default charset for plain text using meta:

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" >

**The expires attribute**

The expires attribute specifies when the content should be refreshed from the webserver:

<meta http-equiv="Expires" content="Tue, 20 Aug 1996 14:25:27 GMT">

This will result in the HTTP header:

Expires: Tue, 20 Aug 1996 14:25:27 GMT

This can be used by caches to determine when to fetch a fresh copy of the associated document.

**The author attribute**

The author attribute defines the author of the page. The structure is:

<meta name="author" content="John Li">

## Creating Links

A hyperlink (or link) is a word, group of words, or image that you can click on to jump to a new document or a new section within the current document. Web pages can contain links that take you directly to other pages or specific parts of the given page. These links are known as hyperlinks. Hyperlinks allow visitors to navigate between Web sites by clicking on words, phrases, and images. Thus you can create hyperlinks using text or images available on your web page.

Links are specified in HTML using the <a> tag. A link is specified using the <a> element. This element is called anchor tagas well. Anything between the opening <a> tag and the closing </a> tag becomes part of the link and a user can click that part to reach to the linked document.

The <a> tag can be used in two ways:

* To create a link to another document, by using the href attribute
* To create a bookmark inside a document, by using the name attribute

**The href Attribute**: A link in HTML is always composed of two parts, the clickable part (the link text) and the URL (the destination site, page or resource). The URL is specified using href attribute. Here is an example:

<a href=” http://mail.yahoo.com”> Yahoo mail</a>

Here:

* Clickable part (link text): Yahoo mail
* URL: http://mail.yahoo.com

**The target attribute**

It is used to specify where to display the contents of a selected hyperlink. If set to:

* ***\_blank*** then a new window will be opened to display the loaded page
* ***\_top*** or ***\_parent*** then same window will be used to display the loaded document.
* ***\_self*** then loads the new page in current window. By default its \_self.

Example: a link that opens on a new window

<a href="first.html" target="\_blank"> First Page</a><br>

**The name Attribute**

The name attribute specifies the name of the anchor. The name attribute is used to create a bookmark inside an HTML document.Bookmarks are not displayed in any special way. They are invisible to the reader.

Example: A named anchor inside an HTML document:

<a name="example">Sample Examples</a>

Then create a link to the "Sample Examples" inside the same document:

<a href="#example">Go to Sample Example</a>

**The accesskey attribute:**

The accesskey attribute provides a keyboard shortcut that can be used to activate the link. For example, you could make the T key an access key so that when the user presses either the Alt or Ctrl key on his keyboard (depending on his operating system) along with the T key, the link gets activated.

Example:

<a href="first.html" target="\_blank" accesskey="m">First page</a>

**The title attribute**

The title attribute is used to type a short description of the link. If you place the cursor over the link, you will see the text in the title attribute.

Example:

<a href="http://www.html.net/" title="Visit html.net and learn HTML">Learn HTML </a>

**Linking to email**

It is also possible to make a link to an e-mail address. It is done in almost the same way as when you link to a document. The only difference between a link to an e-mail and a link to a file is that instead of typing the address of a document, you type *mailto:* followed by an e-mail address.

Example:

<a href="mailto:home@gmail.com">Send an e-mail to home at gmail</a>

When the link is clicked, the default e-mail program opens with a new blank message addressed to the specified e-mail address. Remember that this function will only work if there is an e-mail program installed on your computer.

**HTML Comments**

Comments are piece of code which is ignored by any web browser. It is good practice to comment your code, especially in complex documents, to indicate sections of a document, and any other notes to anyone looking at the code. Comments help you and others understand your code.

HTML Comment lines are indicated by the special beginning tag <!-- and ending tag --> placed at the beginning and end of every line to be treated as a comment. You can comment multiple lines by the special beginning tag <!-- and ending tag --> placed before the first line and end of the last line to be treated as a comment.

For example: Given line is a valid comment in HTML

<!-- This is commented out -->

## Working with Colors

In HTML colors can be used to enhance the way your web page looks like. It can be used to change the background of your webpage, the color of your text or the content of a table. While choosing the right type of color is your responsibility, HTML provides you with many options to mingle with. HTML colors are defined using a hexadecimal notation (HEX) for the combination of Red, Green, and Blue color values (RGB). The lowest value that can be given to one of the light sources is 0 (in HEX: 00). The highest value is 255 (in HEX: FF). HEX values are specified as 3 pairs of two-digit numbers, starting with a # sign.

For example:

* Red = #FF0000
* Green = #00FF00
* Blue = #0000FF
* Cyan (blue and green) = #00FFFF
* Magenta (red and blue) = #FF00FF
* Yellow (red and green) = #FFFF00

Both <body bgcolor=”red”> and <body bgcolor=”#FF0000”> will change the background color in to red.

## HTML Text Formatting Tags

There are options if one wants to create a text in HTML. Let’s see some of the tags that can be used to create text information.

**HTML Headings**

In HTML, you can create different heading levels in your document to help you organize the document into sections, just as you might do when writing a book. A heading element briefly describes the topic of the section it introduces. Headings are defined with the <h1> to <h6> tags. <h1> defines the largest heading and <h6> the smallest heading.

Example:

<body>

<h1>this is heading level 1.</h1>

<h2>this is heading level 2.</h2>

<h3>this is heading level 3.</h3>

<h4>this is heading level 4.</h4>

<h5>this is heading level 5.</h5>

<h6>this is heading level 6.</h6>

</body>

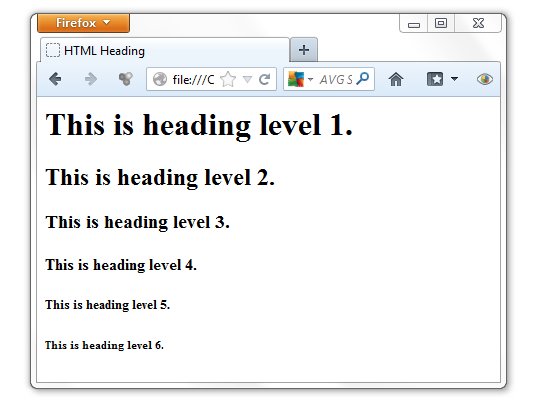


Fig HTML heading levels

Heading tags are block tags, and you must specify opening and closing tags.

**HTML Paragraphs**

Authors traditionally divide their thoughts and arguments into sequences of paragraphs. Along with using headings to structure your Web page, you can add structure by grouping common text into paragraphs.

Paragraphs are defined with the <p> tag.

Example:

<p>This is a paragraph</p>

<p>This is another paragraph</p>

Both of the above tags will enforce a new line whenever you write the tag. The <p> element is a block element. It cannot contain block-level elements including <p> itself.

**The align attribute**

You can use align attribute to align your paragraphs. Paragraphs can be left aligned, center aligned, right aligned or justified. You can do this by using align attribute. Align attribute can be used with other tags like headers, table, etc.

Example:

<p align="center">This is center aligned.</p>

<p align="justify">This is justified. This works when you have multiple lines in your paragraph and you want to justify all the lines so that they can look more nice.</p>

**HTML Fonts**

The <font> tag is used to add font type, size, and color to the text on your site. The font tag is has three attributes called size, color, and face to customize your fonts. To change any of the font attributes at any time within your page, simply use the <font> tag. The text that follows will remain changed until you close with the </font> tag. You can change any or all of the font attributes at the one time, by including all the required changes within the one <font> tag.

**Font Size:**

You can set the size of your font with size attribute. The range of accepted values is from 1(smallest) to 7(largest). The default size of a font is 3.

Example:

<font size="1">Font size="1"</font>

<font size="2">Font size="2"</font>

<font size="3">Font size="3"</font>

<font size="4">Font size="4"</font>

<font size="5">Font size="5"</font>

<font size="6">Font size="6"</font>

<font size="7">Font size="7"</font>

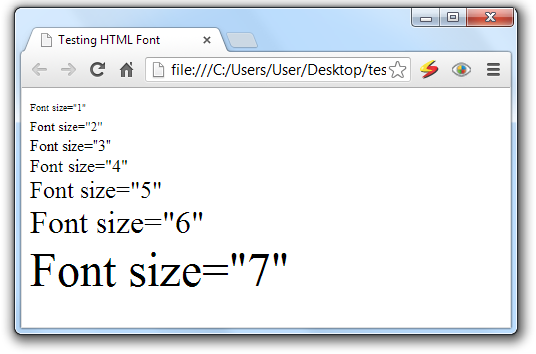


Fig Font size

**Specify Relative Font Size**

You can also specify relative font sizes instead of exact font size. This can be done like:

<font size="+n">

<font size="-n">

This specifies how many sizes larger or how many sizes smaller than the preset font size should be.

Example:

<font size="-1">Font size="-1"</font>

<font size="+1">Font size="+1"</font>

<font size="+2">Font size="+2"</font>

<font size="+3">Font size="+3"</font>

<font size="+4">Font size="+4"</font>

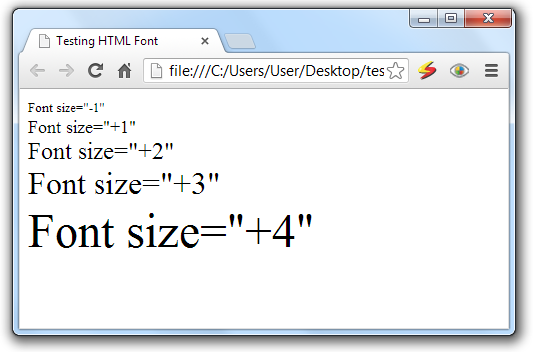


Fig Relative font size

**Font Face:**

You can set any font you like using *face* attribute but be aware that if the user viewing the page doesn't have the font installed, they will not be able to see it. Instead they will default to Times New Roman of your font with size attribute.

Example:

<font face="Times New Roman" size="5"> Times New Roman </font>

<font face="Verdana" size="5"> Verdana </font>

<font face="Comic sans MS" size="5"> Comic Sans MS </font>

<font face="WildWest" size="5"> WildWest </font>

A visitor will only be able to see your font if they have that font installed on their computer. So, it is possible to specify two or more font face alternatives by listing the font face names, separated by a comma.

Example:

<font face="arial, helvetica">

<font face="Lucida Calligraphy, Comic Sans MS, Lucida Console>

**Font Color:**

You can set any font color you like using color attribute. You can specify the color that you want by either the color name or hexadecimal code for that color. Check a complete list of HTML Color Name with Codes.

Example:

<font color="#0000FF">This text blue color</font>

<font color="red">This text is red</font>

**The <basefont> Element:**

The <basefont> element is used to set a default font size, color, and typeface for any parts of the document that are not otherwise contained within a <font> element. You can then use the <font> elements to override the <basefont> settings.

The attributes that the <basefont> element takes are exactly the same as for the <font> element. You can also set the size of fonts relative to the size of the <basefont> by giving them a value of +1 for a size larger or -2 for two sizes smaller

Example:

<basefont face="arial, verdana, sans-serif" size="2" color="#ff0000">

<p>This is the page's default font.</p>

<h2>Example of the &lt;basefont&gt; Element</h2>

<p><font size="+2" color="darkgray">Here is some darkgray text two sizes larger</font></p>

<p><font face="courier" size="-1" color="#000000">Here is a courier font, a size smaller, in black</font></p>

**NB:** The <basefont> tag is only supported by Internet Explorer, and should be avoided!

**Formatting Tags**

In HTML, there are many tags that you can use to enhance and change the look of the text. You can make text bold, italicized, or underlined; these are just some of the presentational options available to indicate how text can appear in HTML.

**Bold Text - The <b> Element:**

Anything that appears in a <b>...</b> element is displayed in bold. The <b> tag makes text to be displayed in bold face.

Example:

<p>The use nuclear energy needs <b> safety caution </b> because of the associated danger.</p>

This will produce following result:

The use nuclear energy needs **safety caution** because of the associated danger.

**Italic Text - The <i> Element:**

Anything that appears in a <i>...</i> element is displayed in italicized face.

Example:

<p>The following word uses a <i>italicized</i> typeface.</p>

This will produce following result:

The following word uses a *italicized* typeface.

**Underlined Text - The <u> Element:**

Anything that appears in a <u>...</u> element is displayed with underline.

Example:

<p>The following word uses a <u>underlined</u> typeface.</p>

This will produce following result:

The following word uses a underlined typeface.

**Strike Text - The <strike> Element:**

Anything that appears in a <strike>...</strike> element is displayed with strikethrough, which is a thin line through the text.

Example

<p>The following word uses a <strike>strikethrough</strike> typeface.</p>

This will produce following result:

The following word uses a ~~strikethrough~~ typeface.

**Centering Content - The <center> Element:**

You can use <center> tag to put any content in the center of the page or any table cell.

Example:

<p>This is not in the center.</p>

<center> <p>This is in the center.</p> </center>

**Tags and Their Attributes**

The <p>, <center>, <b>, <i>, <u>, <pre>, <strike> tags support the following standard attributes:

|  |  |  |
| --- | --- | --- |
| **attribute** | **Value** | **Description** |
| Dir | rtl ltr | Specifies the text direction for the content in an element |
| Lang | language\_code | Specifies a language code for the content in an element |
| Title | Text | Specifies extra information about an element |

**Create Line Breaks - The <br> Element:**

The <br> tag inserts a single line break. You can also use <br /> which does the same.

Whenever you use the <br> element, anything following it starts on a new line. This tag is an example of an emptyelement, where you do not need opening and closing tags, as there is nothing to go in between them.

The <br /> element has a space between the characters br and the forward slash. If you omit this space, older browsers will have trouble rendering the line break, while if you miss the forward slash character and just use <br> it is not valid XHTML.

Example

Hello<br />

You come most carefully upon your hour.<br>

Thanks<br />

Mahnaz

This will produce following result:

Hello

You come most carefully upon your hour.

Thanks

Mahnaz

**Preserve Formatting - The <pre> Element:**

Sometimes you want your text to follow the exact format of how it is written in the HTML document. In those cases, you can use the preformatted tag (<pre>).

The <pre> tag defines preformatted text. Text in a pre element is displayed in a fixed-width font (usually Courier), and it preserves both spaces and line breaks.

Example:

<pre>

function testFunction( int a, int b ){

int sum = a + b;

alert (sum);

return;

}

</pre>

This produces the following output:

function testFunction( int a, int b ){

int sum = a + b;

alert (sum);

return;

}

**Horizontal Rules - The <hr> Element**

<hr> stands horizontal rules are used to visually break up sections of a document. The <hr> tag creates a horizontal line in an HTML page. The hr element can be used to separate content in an HTML page.

For example you may want to give a line between two paragraphs as follows:

<p>This is paragraph one and should be on top</p>

<hr>

<p>This is paragraph two and should be at bottom</p>

This produces the following output:

This is paragraph one and should be on top

This is paragraph two and should be at bottom

|  |  |  |
| --- | --- | --- |
| Attribute | Value | Description |
| Align | left center right | Deprecated. Use styles instead. Specifies the alignment of a hr element |
| noshade | noshade | Deprecated. Use styles instead. Specifies that a hr element should render in one solid color (noshaded), instead of a shaded color |
| Size | *pixels* | Deprecated. Use styles instead. Specifies the height of the hr element |
| Width | *pixels %* | Deprecated. Use styles instead. Specifies the width of the hr element |

Example:

<p>This is paragraph one and should be on top</p>

<hr size=”5” width=”50” align=”center”>

**Subscript and Superscript Text:**

The <sub> tag defines subscript text. Subscript text appears half a character below the baseline. Subscript text can be used for chemical formulas, like H2O.

The <sup> tag defines superscript text. Superscript text appears half a character above the baseline. Superscript text can be used for mathematical expressions like x2+y or footnotes, like HTTP[1].

Example:

<p> The chemical formula for water is H<sub>2</sub>O </p>

<p> Solve 5X<sup>2</sup>+6X+10 </p>

**Big and small Text:**

The <big> tag displays texts in big. The content of the <big> element is displayed one font size larger than the rest of the text surrounding it.

The <small> tag renders a smaller text. The content of the <small> element is displayed one font size smaller than the rest of the text surrounding it.

Example:

<html>

<head>

<title>Formatting Tags</title>

</head>

<body>

<font size="5">

<p> The chemical formula for water is H<sub>2</sub>O </p>

<p> Solve 5X<sup>2</sup>+6X+10 </p>

<p>The following word uses a <small>small</small> typeface.</p>

<p>The following word uses a <big>big</big> typeface.</p>

</font>

</body>

</html>

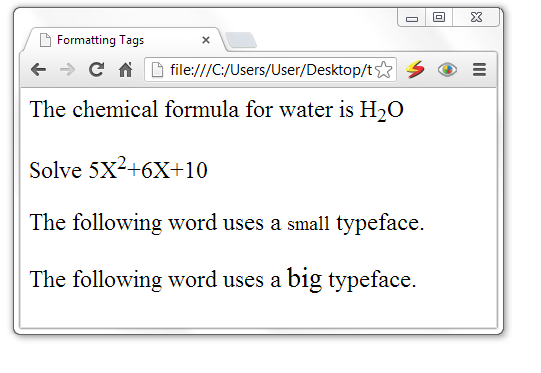


Fig superscript, subscript, big and small texts

**Bidirectional Text - <bdo> Tag**

bdo stands for bidirectional override. The <bdo> tag allows you to specify the text direction and override the bidirectional algorithm. This is especially useful for languages like Hebrew and Arabic where text is written from right to left.

Example:

<bdo dir="rtl">Here is some Hebrew text!</bdo>

This produces the following output:



**Other Formatting Tags**

The following are some more tags.

|  |  |
| --- | --- |
| Tag | Description |
| <em> | Renders as emphasized text |
| <strong> | Renders as strong (highlighted) text |
| <dfn> | Defines a definition term |
| <code> | Defines computer code text |
| <samp> | Defines sample output from computer code |
| <kbd> | Defines keyboard text |
| <var> | Defines a variable part of a text |
| <cite> | Defines a citation |
| <abbr> | Defines an abbreviation |
| <acronym> | Defines an acronym |
| <q> | Defines a short quotation. |
| <blockquote> | To quote a passage from another source |

**Example:**

<html>

<head>

<title>HTML Tags</title>

</head>

<body>

<abbr title="World Health Organization">WHO</abbr> reports that malaria epidemic is increasing.

Two <acronym title="North Atlantic Treaty Organization">NATO</acronym> troops are killed in Afghanistan.

<em>Emphasized text</em> <br>

<strong>Strong text</strong> <br>

<dfn>Definition term</dfn> <br>

<code>Computer code text</code> <br>

<samp>Sample computer code text</samp> <br>

<kbd>Keyboard text</kbd> <br>

<var>Variable</var> <br>

<cite>Citation</cite> <br>

</body>

</html

This produces the following output

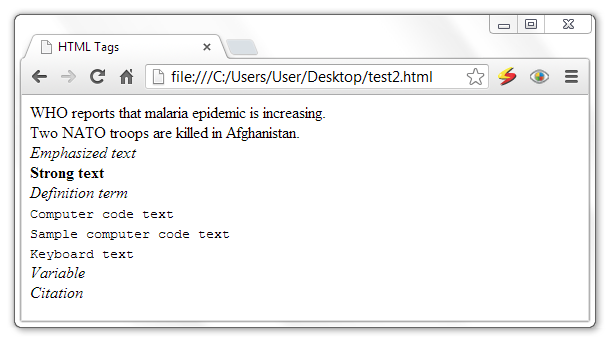


Fig strong, emphasized, abbreviation, acronym, citation

**Special Characters**

Characters within HTML documents that are not part of a tag are rendered as-is by the browser. However, some characters have special meaning and are not directly rendered, while other characters can't be typed into the source document from a conventional keyboard. Special characters need either a special name or a numeric character encoding for inclusion in an HTML document.

|  |  |  |  |
| --- | --- | --- | --- |
| **Character** | **Code** | **Short name** | **Character name** |
| " | &#34; | &quot; | quotation mark |
| ' | &#39; | &apos; | apostrophe |
| & | &#38; | &amp; | ampersand |
| < | &#60; | &lt; | less-than |
| > | &#62; | &gt; | greater-than |
|  | &#160; | &nbsp; | non-breaking space |
| ¡ | &#161; | &iexcl; | inverted exclamation mark |
| ¢ | &#162; | &cent; | cent |
| £ | &#163; | &pound; | pound |
| ¤ | &#164; | &curren; | currency |
| ¥ | &#165; | &yen; | yen |
| ¦ | &#166; | &brvbar; | broken vertical bar |
| § | &#167; | &sect; | section |
| ¨ | &#168; | &uml; | spacing diaeresis |
| © | &#169; | &copy; | copyright |
| ª | &#170; | &ordf; | feminine ordinal indicator |
| « | &#171; | &laquo; | angle quotation mark (left) |
| ¬ | &#172; | &not; | negation |
| ­­ | &#173; | &shy; | soft hyphen |
| ® | &#174; | &reg; | registered trademark |
| ¯ | &#175; | &macr; | spacing macron |
| ° | &#176; | &deg; | degree |
| ± | &#177; | &plusmn; | plus-or-minus |
| ² | &#178; | &sup2; | superscript 2 |
| ³ | &#179; | &sup3; | superscript 3 |
| ´ | &#180; | &acute; | spacing acute |
| µ | &#181; | &micro; | micro |
| ¶ | &#182; | &para; | paragraph |
| · | &#183; | &middot; | middle dot |
| ¸ | &#184; | &cedil; | spacing cedilla |
| ¹ | &#185; | &sup1; | superscript 1 |
| º | &#186; | &ordm; | masculine ordinal indicator |
| » | &#187; | &raquo; | angle quotation mark (right) |
| ¼ | &#188; | &frac14; | fraction 1/4 |
| ½ | &#189; | &frac12; | fraction 1/2 |
| ¾ | &#190; | &frac34; | fraction 3/4 |
| ¿ | &#191; | &iquest; | inverted question mark |
| × | &#215; | &times; | multiplication |
| ÷ | &#247; | &divide; | division |

Table Special characters

## Working with Graphics and Images

Images are very important to beautify as well as to depicts many concepts on your web page. It is often said that an single image is worth than thousands of words. So as a Web Developer you should have clear understanding on how to use images in your web pages.

In HTML, images are defined with the <img> tag.  The <img> tag has attributes, but not closing tag. To display an image on a page, you need to use the src attribute. The value of the src attribute is the URL of the image you want to display.

Syntax for inserting an image:

<img src="url" alt="Alternative Text"/>

**The alt attribute**

The required alt attribute specifies an alternate text for an image, if the image cannot be displayed. The alt attribute provides alternative information for an image if a user for some reason cannot view it (because of slow connection, an error in the src attribute).

Example:

<img src="boat.gif" alt="Big Boat" />

**The width and height attributes**

To define the height and width of the image, rather than letting the browser compute the size, use the height and width attributes.

The attributes:

* **width:** sets width of the image. This can have a value like 10 (pixels) or 20% (percentage of the available window size) etc.
* **height:** sets height of the image. This can have a value like 10(pixels) or 20% (percentage of the available window size) etc.

Example:

<img src=”sunset.jpg” height=”50” width=”100”>

**The border attribute**

The border attribute sets a border around the image. This will have a value like 1 or 2 etc.

**The align attribute**

The align attribute sets horizontal alignment of the image and takes value either left, right or center.

Example:

<img src="coffee.gif" alt="Highland coffee" width="100" height="100" border="2" align="right" title="HTML Tutorial" />

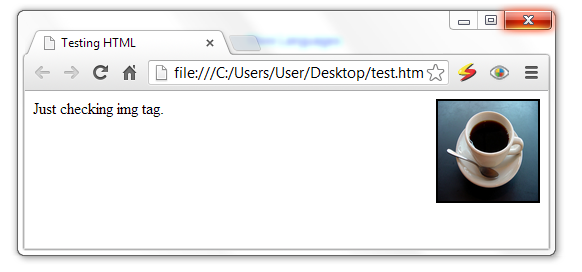


Fig Image insertion

**The valign attribute**

The valign attribute sets vertical alignment of the image and takes value either top, bottom or center.

**The hspace and vspace attributes**

* **hspace:** sets horizontal space around the image. This will have a value like 10 or 20% etc.
* **vspace:** sets vertical space around the image. This will have a value like 10 or 20% etc.

**The title attribute**

The title attribute specifies a text title. The browser displays the title when the mouse passes over the link.

Example:

<html>

<head>

<title>Testing HTML</title>

</head>

<body>

Just checking img tag.

<img src="coffee.png" alt="Highland coffee" align="left" hspace="50" vspace="100" />

</body>

</html>

## Image maps

Image maps enable you to define multiple links within a single image. For example, if you have a picture of African countries, you can create links in the image that opens each country’s picture/page for detailed information. Clickable regions within image maps can be basic shapes like rectangles and circles or complex polygonal shapes.

Regardless of the complexity of the image and the defined regions, the method of creating an image map remains the same. The following diagram is the image used in this section to show how a basic image map is created. It contains three geometric shapes that will be turned into clickable hot-spots.



Fig Shapes for image map

The image is added to the web page in the usual way, but with the addition of a usemap attribute, whose value must be preceded by a hash sign (#).

Example:

<img src="image-map-image.gif" alt="Shapes" width="398" height="398" usemap="#shapes" />

The value of the usemap attribute must correlate with the id values of the associated map element. The name attribute is required for backward compatibility, whereas the id attribute is mandatory.

<map id="shapes" name="shapes">

</map>

To create the map used by img tag, we use the following tags and attributes:

* **<map>** tagmaps information for an imagemap
  + **name="***text***" -** The legacy method for giving the map a name
  + **id="***text***" -** The current method for giving the map a name
* **<area>** tagcontains information for a clickable area in an imagemap. The <area> element specifies the shape and the coordinates that define the boundaries of each clickable hotspot.
  + **shape="rect**|**circle**|**poly" -** shape of the linked area
  + **coords="***numbers***" -** pixel coordinates for the linked area
  + **href="***url***" -** target file for the link

Each of the area elements have a shape attribute that corresponds to the intended active link area:

* **rect** - defines a rectangular area. The coords (coordinates) attribute contains two pairs that define the top-left and bottom-right corners of the rectangle in terms of pixel values.
* **circle** - is used to define a circular area. Out of the three values within the cords attribute, the first two define the horizontal and vertical position of the circle’s center, and the third defines the radius.
* **poly** - enables you to define as many coordinate pairs as you wish, which allows you to define active areas for complex and irregular shapes—in the previous code block, there are three pairs, each of which defines a corner of the triangle.

The map element acts as a container for specifications regarding the map’s active areas, which are added as area elements.

<img src="image-map-image.gif" alt="Shapes" width="398" height="398" usemap="#shapes" />

<map id="shapes" name="shapes">

<area title="Access the squares page" shape="rect" coords="29,27,173,171" href="square.html">

<area title="Access the circles page" shape="circle" coords="295,175,81" href="circle.html">

<area title="Access the triangles page" shape="poly" coords="177,231,269,369,84,369" href="triangle.html" alt="A triangle">

</map>

Creating image maps is a tedious process, and it is advisable to use visual web design tools. For example, you can GIMP image editing software to create image maps in easier way.

# 2. Tables and Lists

## 2.1 Creating Tables

Tables are defined with the <table> tag. A table is divided into rows with the <tr> tag, and each row is divided into data cells with the <td> tag. td stands for "table data," and holds the content of a data cell. A <td> tag can contain text, links, images, lists, forms, other tables, etc.

**Table headers**

Headers in a table are defined with the <th> tag. The text in a <th> element will be bold and centered.

**Example:**

<table border="1">  
<tr>  
<th>Header 1</th> <th>Header 2</th>  
</tr>  
<tr><td>row 1, cell 1</td> <td>row 1, cell 2</td></tr>

<tr>  
<td>row 2, cell 1</td> <td>row 2, cell 2</td>  
</tr>  
</table>

HTML code above looks in a browser:

|  |  |
| --- | --- |
| **Header 1** | **Header 2** |
| row 1, cell 1 | row 1, cell 2 |
| row 2, cell 1 | row 2, cell 2 |

If you do not specify a border attribute, the table will be displayed without borders. Sometimes this can be useful, but most of the time, we want the borders to show.

**Attributes of table:**

The attributes of a table will be applied on the whole table element which include one or more rows (<tr>), header cells (<th>) or data cells (<td>).

|  |  |  |
| --- | --- | --- |
| Attribute | Value | Description |
| Align | Left  Center  right | Specifies the alignment of a table according to surrounding text |
| Bgcolor | rgb(x,x,x)  #xxxxxx  colorname | Specifies the background color for a table |
| Background | Image url | Sets background image of the table |
| Border | pixels | Specifies the width of the borders around a table |
| Bordercolor | rgb(x,x,x)  #xxxxxx  colorname | Specifies the color used for the border |
| Cellpadding | pixels | Specifies the space between the cell wall and the cell content |
| Cellspacing | pixels | Specifies the space between cells |
| Width | Pixels  % | Specifies the width of a table |
| Height | Pixels  % | Specifies the height of a table |

Table 2.1 attributes of table

Example:

<table border="5" bordercolor="green" bgcolor="gray">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

<tr><td rowspan="2">Row 1 Cell 1</td>

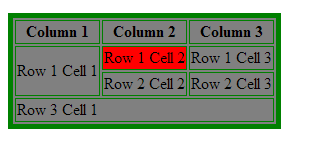
<td bgcolor="red">Row 1 Cell 2</td><td>Row 1 Cell 3</td></tr>

<tr><td>Row 2 Cell 2</td><td>Row 2 Cell 3</td></tr>

<tr><td colspan="3">Row 3 Cell 1</td></tr>

</table>

This produces the following output:

****

**Table Cellpadding and Cellspacing:**

There are two attribiutes called *cellpadding* and *cellspacing* which you will use to adjust the white space in your table cell. Cellspacing defines the width of the border, while cellpadding represents the distance between cell borders and the content within.

Example:

<table border="1" cellpadding="15" cellspacing="10">

<tr> <th>Name</th> <th>Salary</th> </tr>

<tr> <td>Ramesh Raman</td> <td>5000</td> </tr>

<tr> <td>Shabbir Hussein</td> <td>7000</td> </tr>

</table>

This produces the following output:

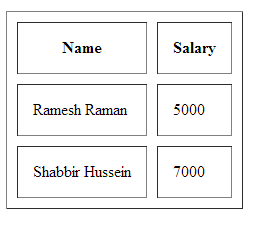
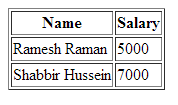
 

Fig Table with cellpadding 15 and cellspacing 10 and table with no values set

**Attributes of rows and cells:** this attributes will be applicable only to the header cell or the data cell if it is used with <th> or <td> tag while it will affect the whole content of the row if it is used by the <tr> tag.

|  |  |  |
| --- | --- | --- |
| Attribute | Value | Description |
| Align | Left | right | center | justify | Aligns the content in a cell |
| bgcolor | rgb(x,x,x)  #xxxxxx  colorname | Specifies a background color for a cell |
| colspan | number | Specifies the number of columns a cell should span |
| rowspan | number | Sets the number of rows a cell should span |
| height | Pixels  % (percent) | Sets the height of a cell |
| Width | Pixels  %(percent) | Specifies the width of a cell |
| nowrap | nowrap | Specifies that the content inside a cell should not wrap |
| Valign | top|middle|bottom | Vertical aligns the content in a table row |

Table 2.2 attributes of rows and cells

**Spanning rows and cells**

It’s sometimes necessary for data to span multiple rows or columns. This is achieved via the rowspan and colspan attributes, respectively.

<table border="1" cellpadding="2">

<tr>

<td>A cell</td>

<td>Another cell</td>

<td>Yet another cell!</td>

</tr>

<tr>

<td rowspan="2">A cell that spans two rows</td>

<td colspan="2">A cell that spans two columns</td>

</tr>

<tr>

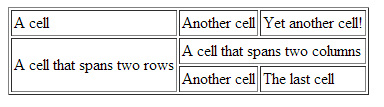
<td>Another cell</td>

<td>The last cell</td>

</tr>

</table>

This produces the following output:



**Vertical alignment of cell content**

If you set your table’s width to a small value, or if you have a lot of content in one cell and relatively little in an adjacent one, something else becomes apparent: web browsers vertically align content in the middle of cells.

You can use the valign attribute to set where the text is displayed vertically. The attribute can be added to a row or cell start tag, and set to the desired value.



**Using a Header, Body, and Footer:**

Tables can be divided into three portions: a header, a body, and a foot. The head and foot are rather similar to headers and footers in a word-processed document that remain the same for every page, while the body is the main content of the table.

The three elements for separating the head, body, and foot of a table are:

* **<thead> -** to create a separate table header.
* **<tbody> -** to indicate the main body of the table.
* **<tfoot> -** to create a separate table footer.

A table may contain several <tbody> elements to indicate different *pages* or groups of data. But it is notable that <thead> and <tfoot> tags should appear before <tbody>.

Example:

<table border="1" width="100%">

<thead> <tr>

<td colspan="4">This is the head of the table</td>

</tr>

</thead>

<tfoot>

<tr> <td colspan="4">This is the foot of the table</td> </tr>

</tfoot>

<tbody>

<tr>

<td>Cell 1</td> <td>Cell 2</td>

<td>Cell 3</td> <td>Cell 4</td>

</tr>

<tr>

...more rows here containing four cells...

</tr>

</tbody>

<tbody>

<tr>

<td>Cell 1</td> <td>Cell 2</td>

<td>Cell 3</td> <td>Cell 4</td>

</tr>

<tr>

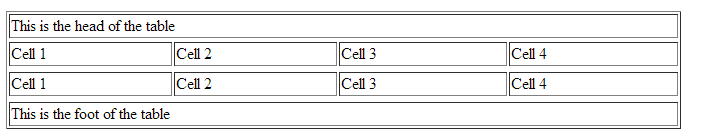
...more rows here containing four cells...

</tr>

</tbody>

</table>

This produces the following output:



**Practice 2.1**

1. Create a table that displays an exam schedule for your department?

## 2.2 Using Ordered and Unordered List

The most common HTML lists are ordered and unordered lists.

|  |  |
| --- | --- |
| An ordered list:   1. The first list item 2. The second list item 3. The third list item | An unordered list:   * List item * List item * List item |

**Unordered List**

An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.

The list items are marked with bullets (typically small black circles). The “type” attribute can be used to specifies the style of the bullet points of the list items, its value includes disc, square and circle.

Example:

<ul>  
<li>Banana</li>  
<li>Orange</li>  
</ul>

How the HTML code above looks in a browser:

* Banana
* Orange

**<ul> attributes:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| compact | Compact | Specifies that the list should render smaller than normal |
| Type | Disc  square  circle | Specifies the style of the bullet points of the list items |

**Ordered List**

An ordered list starts with the <ol> tag. Each list item starts with the <li> tag. The list items are marked with numbers.

Example:

Types of fruits are:

<ol>  
<li>Banana</li>  
<li>Orange</li>

<li>Apple</li>  
</ol>

How the HTML code above looks in a browser:

Types of fruits are:

1. Banana
2. Orange
3. Apple

**The start attribute:**

If you want a numbered list to start at a number other than “1,” you can use the **start** attribute to specify another starting number.

Example

<ol start="17">

<li>Highlight the text with the text tool.</li>

<li>Select the Character tab.</li>

<li>Choose a typeface from the pop-up menu.</li>

</ol>

The resulting list items would be numbered 17, 18, and 19, consecutively.

**The <ol> attributes:**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| compact | compact | Specifies that the list should render smaller than normal |
| start | *number* | Specifies the start point for the list items |
| type | 1 I i  A a | Specifies which kind of bullet points will be used |

Example:

Types of fruits are:

<ol type=”I”>

<li>Banana</li>

<li>Orange</li>

<li>Apple</li>  
</ol>

This produces the following attribute:

Types of fruits are:

1. Banana
2. Orange
3. Apple

**Nesting Lists**

One list can be put inside another to create nested list.

Example:

<ol>

<li>Super computer</li>

<li>Mainframe computer</li>

<li>Mini computer</li>

<li>Micro computer</li>

<ul>

<li>Desktop computer</li>

<li>Laptop Computer</li>

<li>Palmtop computer</li>

</ul>

</ol>

**Practice 2.2**

1. Write a code that displays the courses that you are currently taking (use unordered list)?
2. Write a code that displays your favorite books in order?

# 3. Frames

## 3.1 Working with Frames

Frames can divide the screen into separate windows. Each of these windows can contain an HTML document. A file that specifies how the screen is divided into frames is called a frameset. The frameset page lacks a <body> element (although it still requires head and title elements) and instead uses a <frameset> element, which sets the attributes for how the frames are positioned. The frameset element houses frame elements, which define the location and attribute of each frame.

When a frameset page is loaded, the browser automatically loads each of the pages associated with the frames. Each HTML document is called a frame, and each frame is independent of the others.

The disadvantages of using frames are:

* The web developer must keep track of more HTML documents
* It is difficult to print the entire page

The frameset element holds two or more frame elements. Each frame element holds a separate document. The frameset element states only how many columns or rows there will be in the frameset and it uses <frameset> tag. The <frame> tag defines one particular window (frame) within a frameset.

In the example below we have a frameset with two columns. The first column is set to 25% of the width of the browser window. The second column is set to 75% of the width of the browser window. The document "frame\_a.htm" is put into the first column, and the document "frame\_b.htm" is put into the second column:

<frameset cols="25%,75%">  
   <frame src="frame\_a.htm" />  
   <frame src="frame\_b.htm" />  
</frameset>

**Note**:

* The frameset column size can also be set in pixels (cols="200,500"), and one of the columns can be set to use the remaining space, with an asterisk (cols="25%,\*").
* You cannot use the body element together with the frameset element. However, if you add a <noframes> tag containing some text for browsers that do not support frames, you will have to enclose the text in a body element.

**Attributes of frameset**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| Cols | *pixels*  %  \* | Specifies the number and size of columns in a frameset |
| Rows | *pixels*  %  \* | Specifies the number and size of rows in a frameset |

**Example:**

<frameset cols="30%,\*">

<frame src="frame-one.html">

<frame src="frame-two.html">

</frameset>

This produces the following output:

****

Fig Frameset

**Attributes of frame**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| frameborder | 0 or 1 | Specifies whether or not to display a border around a frame |
| marginheight | pixels | Specifies the top and bottom margins of a frame |
| marginwidth | pixels | Specifies the left and right margins of a frame |
| noresize | noresize | Specifies that a frame cannot be resized |
| scrolling | yes  no  auto | Specifies whether or not to display scrollbars in a frame |
| Src | URL | Specifies the URL of the document to show in a frame |

You can also nest framesets, to create a combination of columns and rows:

<frameset rows="120,\*">

<frame noresize=”noresize” src="frame-one.html">

<frameset cols="150,\*">

<frame src="frame-two.html">

<frame src="frame-three.html">

</frameset>

</frameset>



Fig Nested frame

**Practice 2.3**

## What is the advantage of using frames

1. Create a web page that is divided in to three equal rows and the first row is divided in to two columns.
2. Create a web page that is divided in to 3:2:1 proportion of columns.

## 3.2 Nested Frames

Framesets may be nested to any level. In the following example, the outer FRAMESET divides the available space into three equal columns. The inner FRAMESET then divides the second area into two rows of unequal height.

Example:

<FRAMESET cols="33%, 33%, 34%">

<frame src=”one.html”>

<FRAMESET rows="40%, 50%">

<frame src=”two.html”>

<frame src=”three.html”>

</FRAMESET>

<frame src=”four.html”>

</FRAMESET>

## 3.3 Working with iframes (internal frames)

The <iframe> tag defines an inline frame that contains another document. <iframe>s enable you to update a page section without reloading the rest of it. In addition, <iframe>s can be handy for enabling users to update a portion of a site’s design without touching the rest of the design.

Example:

<html>

<head>

<title>Internal Frame</title>

</head>

<body>

<p> Maths is the base of many sciences including engineering, computer science, physics, etc. </p>

<br> <br>

<iframe src ="test.html" width="100%" height="300">

<p>Your browser does not support iframes.</p>

</iframe>

<br> <br>

<table border="1">

<tr> <th>Name</th> <th>Salary</th> </tr>

<tr> <td>Chan Li</td> <td>5000</td> </tr>

<tr> <td>Anna Andrey</td> <td>7000</td> </tr>

</table></body></html>

# 4. HTML Forms



## Building HTML Forms

HTML forms are used to pass data to a server. Users generally "fill" a form by modifying its controls (entering text, selecting list items, etc.) before submitting the form for further processing by server (e.g., to a Web server, to a mail server, etc.). Forms are a vital tool for the webmaster to receive information from the web surfer, such as: their name, email address, credit card, etc. A form will take input from the viewer and depending on your needs; you may store that data into a file, place an order, gather user statistics, register the person to your web forum, or maybe subscribe them to your weekly newsletter.

A form will take input from the site visitor and then will post to your back-end application such as CGI, ASP, PHP script etc. Then your back-end application will do required processing on that data in whatever way you like.

The <form> tag is used to create an HTML form:

<form>  
input elements  
</form>

The <FORM> tag takes the three attributes shown in the following table. The ACTION attribute is required in every <FORM> tag.

**Attributes of the <FORM> Tag**

|  |  |
| --- | --- |
| **Attribute** | **Purpose** |
| ACTION | Specifies the URL of the processing script |
| ENCTYPE | Supplies the MIME type of a file used as form input |
| METHOD=GET|POST | Tells the browser how it should send the form data to the server |

* **ACTION: -** ACTION is set equal to the URL of the processing script so that the browser knows where to send the form data once it is entered. Without it, the browser would have no idea where the form data should go.
* **METHOD=GET|POST: -** METHOD specifies the HTTP method to use when passing the data to the script and can be set to values of GET or POST. When you're using the GET method, the browser appends the form data to the end of the URL of the processing script. The POST method sends the form data to the server in a separate HTTP transaction
* METHOD is not a mandatory attribute of the <FORM> tag. In the absence of a specified method, the browser uses the GET method.

**N.B**. Some servers may have operating environment limitations that prevent them from processing an URL that exceeds a certain number of characters-typically 1 kilobyte of data. This limitation can be a problem when you're using the GET method to pass a large amount of form data. Because the GET method appends the data to the end of the processing script URL, you run a greater risk of passing an URL that's too big for the server to handle. If URL size limitations are a concern on your server, you should use the POST method to pass form data.

* **ENCTYPE: -** The ENCTYPE attribute was introduced by Netscape for the purpose of providing a file name to be uploaded as form input. You set ENCTYPE equal to the MIME type expected for the file being uploaded. ENCTYPE does not create the input field for the file name; rather, it just gives the browser a heads-up as to what kind of file it is sending. When prompting for a file to upload, you'll need to use an <INPUT> tag with TYPE set equal to FILE.

E.g.

<FORM ACTION="process\_it.cgi" METHOD=POST ENCTYPE="text/html">

Enter the name of the HTML file to validate:

<INPUT TYPE="FILE" NAME="html\_file">

<INPUT TYPE="SUBMIT" VALUE="Validate it!">

</FORM>

The most important element inside form element is the input element. The input element is used to accept user information. An input element can vary in many ways, depending on the type attribute. An input element can be of type text field, checkbox, password, radio button, submit button, and more. The most used input types are described in the next subsections.

The form itself is not visible but form elements are visible.

## Working with Text Fields and Passwords

**Text Field**

You can create text field by using:

<input type="text">

This defines a one-line input field that a user can enter text into.

Text fields have two important attributes: name and value. The name attribute gives name to the text field for identification purpose and to make it easily accessible. The value attribute sets content of the text field.

<form>  
First name: <input type="text" name="firstname" /><br />  
Last name: <input type="text" name="lastname" />  
</form>

How the HTML code above looks in a browser:

Top of Form

First name:    
Last name: 

The default width of a text field is 20 characters.

The following is the list of attributes for <input type=”text”> tag.

* **name:** The **name** attribute is required for identifying the input field name.
* **value :** The **value** attribute specifies default text that appears in the field when the form is loaded. When you reset a form, it returns to this value.
* **size:** By default, browsers display a text-entry box that is 20 characters wide, but you can change the number of characters using the **size** attribute.
* **maxlength:** By default, users can type an unlimited number of characters in a text field regardless of its size. You can set a maximum character limit using the **maxlength** attribute if the forms processing program you are using requires it.

Example:

<input type="text" name="username" size="8" maxlength="8">

**Practice 2.4**

1. Create a web page that has a form for course registration; use your registration slip as a guide.
2. Create a web page that has a form for signing in to an email.

**Password**

A password field works just like a text entry field, except the characters are obscured from view using asterisk (\*) or bullet (•) characters, or another character determined by the browser.

<input type="password" /> defines a password field:

<form>  
Password: <input type="password" name="pwd">  
</form>

How the HTML code above looks in a browser:

Top of Form

Password: 

Bottom of Form

**Text Area**

At times, you may want your users to be able enter multiple line of text. For these instances, use the <textarea> element that is replaced by a multi-line, scrollable text entry box when displayed by the browser.

Example:

<textarea name="comment"> Tell us what you feel about our tutorial with 50 words or less. </textarea>

**Text Area attributes:**

Text area has the following attributes:

* **name –** name is used to identify the text area
* **rows -** specifies the number of lines of text area should display. Scrollbars will be provided if the user types more text than fits in the allotted space.
* **cols -** specifies the width of the text area measured in number of characters.

Example:

<textarea name="comment" rows="5" cols="100"> Tell us what you feel about our tutorial with 50 words or less. </textarea>

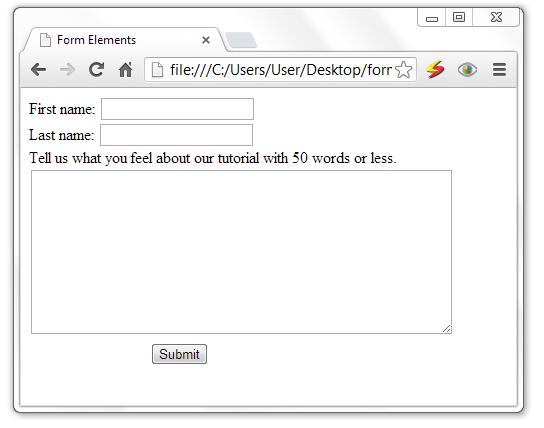


Fig Text field and text area

## Using Buttons, Checkboxes and Selection Lists

**Radio Buttons**

Radio buttons are a popular form of interaction. You may have seen them on quizzes, questionnaires, and other web sites that give the user a multiple choice question. Below are a couple attributes you should know that relate to the radio button.

* value: specifies what will be sent if the user chooses this radio button. Only one value will be sent for a given group of radio buttons.
* name: defines which set of radio buttons that it is a part of.

<input type="radio"> defines a radio button. Radio buttons let a user select ONLY ONE

of a limited number of choices:

<form>  
<input type="radio" name="sex" value="male" /> Male<br />  
<input type="radio" name="sex" value="female" /> Female

</form>

* Bottom of Form
* Bottom of Form

Radio button attributes are:

* *name*: sets name of the radio button
* *value*: sets the value of the radio button. This is the data sent to server when the user submits the form.
* *checked*: sets whether the radio button is checked by default or not. It accepts the value checked.

Example:

<form method=”post” action=”register.php”>  
 <input type="radio" name="sex" value="male" checked=”checked”> Male <br>  
 <input type="radio" name="sex" value="female"> Female

</form>

**Practice 2.5**

1. Create a web page that let students choose their department; use the list of department in your faculty as a guideline.
2. Use a checkbox to create a web page that will enable students to select a subjects they enjoy most (more than one selection is possible).

**Checkboxes**

Check boxes allow for multiple items to be selected for a certain group of choices. The check box's name and value attributes behave the same as a radio button.

<input type="checkbox" /> defines a checkbox. Checkboxes let a user select ONE or MORE options of a limited number of choices.

<form>  
<input type="checkbox" name="vehicle" value="Bike" /> I have a bike<br />  
<input type="checkbox" name="vehicle" value="Car" /> I have a car   
</form>

Checkbox attributes are:

* *name*: sets name of the checkbox
* *value*: sets the value of the checkbox. This is the data sent to server when the user submits the form.
* *checked*: sets whether the checkbox is checked by default or not. It accepts the value checked.

Example:

What type of food do you like?

<ul>

<li><input type="checkbox" name="genre" value=" spaghetti " checked="checked"> Spaghetti</li>

<li><input type="checkbox" name="genre" value="pizza" checked="checked"> Pizza</li>

<li><input type="checkbox" name="genre" value="sandwich">Sandwich </li>

<li><input type="checkbox" name="genre" value="Burger">Burger</li>

</ul>

**Selection lists**

Drop down menus are created with the <select> and <option> tags. <select> is the list itself and each <option> is an available choice for the user.

Educational level: <br>

<select name="degree">

<option>Choose One</option>

<option>Some High School</option>

<option>High School Degree</option>

<option>Some College</option>

<option>Bachelor's Degree</option>

<option>Doctorate</option>

</select>

**Attributes of select**

|  |  |  |
| --- | --- | --- |
| Attribute | Value | Description |
| disabled | disabled | Specifies that a drop-down list should be disabled |
| multiple | multiple | Specifies that multiple options can be selected |
| Name | *text* | Specifies the name of a drop-down list |
| Size | *number* | Specifies the number of visible options in a drop-down list |

**Attribute of option**

|  |  |  |
| --- | --- | --- |
| Attribute | value | Description |
| selected | selected | Specifies whether the option is selected or not when the form loads |

**Scrolling Lists**

To make the menu display as a scrolling list, simply specify the number of lines you’d like to be visible using the size attribute.

Example:

Select the fruits you like:

<select name="fruits" size="6" multiple="multiple">

<option>Orange</option>

<option>Apple</option>

<option selected="selected">Banana</option>

<option selected="selected">Mango</option>

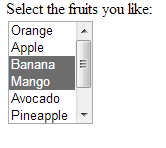
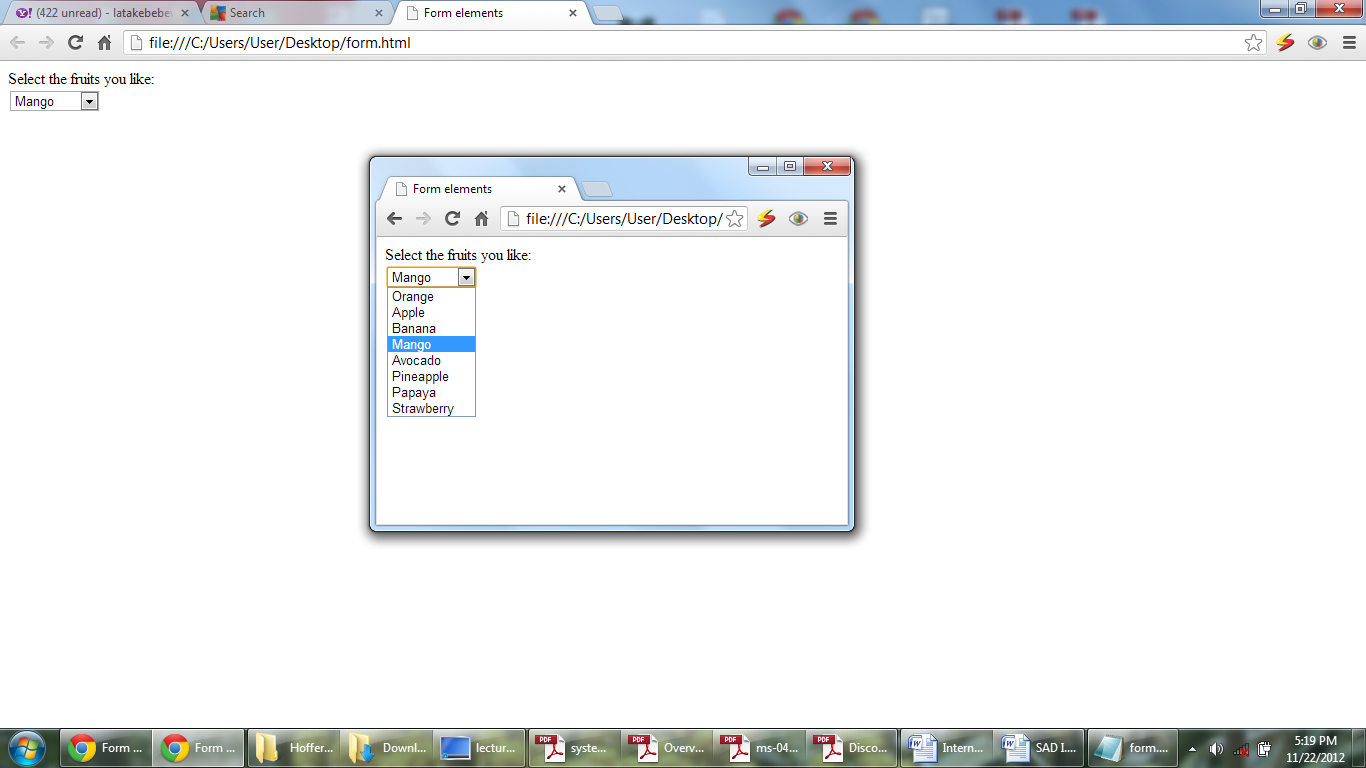
<option> Avocado</option>

<option>Pineapple</option>

<option>Papaya</option>

<option>Strawberry</option></select>

This produces the following output:

**Grouping menu options**

You can use the optgroupelement to create conceptual groups of options. The required label attribute in the optgroupelement provides the heading for the group.

<select name="icecream" multiple="multiple">

**<optgroup label="traditional">**

<option>vanilla</option>

<option>chocolate</option>

**</optgroup>**

**<optgroup label="fancy">**

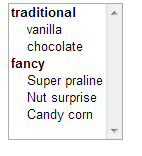
<option>Super praline</option>

<option>Nut surprise</option>

<option>Candy corn</option>

**</optgroup>**</select>

This produces the following output:



**Practice 2.6**

1. Can you think of safe working condition and procedure as technician? Explain its importance?

**Button**

There are a number of different kinds of buttons that can be added to web forms. Some are: submit button, reset button, and client side button.

**Submit Button**

When clicked, the submit button immediately sends the collected data in the form to the server for processing.

Submit button is defined as follows:

<input type="submit">

Submit button has the following attributes:

* value: this sets the text displayed on the button as a label.
* name: used to give name to the submit button

Example:

<input type=”submit” name=”info” value=”Send”>

**Reset Button**

The reset button returns the form controls to the state they were in when the form loaded. This clears the text users typed into text fields, and removes selections made.

Reset button can be defined as follows:

<input type="reset">

**Client Side Button**

This is a button that is used to trigger a client-side script when the user clicks on that button. This is used to execute scripting language such as JavaScript. It has no predefined function on its own, as submit and reset buttons do.

Client side button can be defined as follows:

<input type="button">

Client side button has the following attributes:

* value: this sets the text displayed on the button as a label.
* name: used to give name to the button

Example:

<input type=”button” name=”adder” value=”Add”>

**Image Button**

This type of inputcontrol allows you to replace the submit button with an image of your choice. The image will appear flat, not like a 3-D button.

Image button can be defined as follows:

<input type="image">

Image button attributes:

* src: sets the image to be used as the submit button
* value: text displayed on the button
* name: name of the submit button

Example:

<input type="image" src="didessa.png" value="Submit">

**The button element**

The button element is a flexible element for creating custom buttons similar to those created with the input element. The content of the button element (text and/or images) is what gets displayed on the button.

In this example, a button element is used as a submit button. The button includes a label and a small image.

<button type="submit" name="submit">

<img src="thumbs-up.gif" alt=""> Ready to go.

</button>

**Hidden controls**

There may be times when you need to send information to the form processing application that does not come from the user. In these instances, you can use a hidden form control that sends data when the form is submitted, but is not visible when the form is displayed in a browser.

<input type="hidden">

Hidden controls are added using the input element with the type set to hidden. Its sole purpose is to pass a name/value pair to the server when the form is submitted.

Example:

<input type="hidden"name="page

" value="http://www.example.com/littlechair\_thankyou.html">

**File selection control**

Web forms can collect more than just data. They can also be used to transmit external documents from a user’s hard drive. For example, a printing company could use a web form to receive artwork for a business card order. A magazine could use a form on their site to collect digital photos for a photo contest.

The file selection control makes it possible for users to select a document from the hard drive to be submitted with the form data. It is added to the form using our old friend the input element with its type set to file.

Example:

<input type="file">

The browser displays a “file” input as a text field with a button that allows the user to navigate the hard drive and select the file for upload.

Example:

<input type="file" name="photo" size="28" id="form-photo">

It is important to note that when a form contains a file selection input element, you must specify the encoding type (**enctype**) of the form as **multipart/form-data** and use the POST method.

Example:

<form action="/client.php" method="post" enctype="multipart/form-data">

<p> Send a photo to be used as your online icon:<br>

<input type="file" name="photo" size="28" id="form-photo" /></p>

</form>

Example: a form that combines the above input fields

<html>

<head>

<title>Form Elements</title>

</head>

<body>

<form name="test" method="post">

First name: <input type="text" name="fname"> <br>

Last name: <input type="text" name="lname"> <br>

Sex: <input type="radio" name="sex" value="male" checked> Male

<input type="radio" name="sex" value="female"> Female

<br>

Educational level: <br>

<select name="education" size="6">

<option>Primary School</option>

<option>Secondary School</option>

<option>College Diploma</option>

<option>First Degree</option>

<option>Masters Degree</option>

<option>PhD</option>

</select>

<br>

Which fields are you interested in? <br>

<input type="checkbox" name="" value="Electronics"> Electronics <br>

<input type="checkbox" name="" value="Software Engineering"> Software Engineering <br>

<input type="checkbox" name="" value="Computer Engineering"> Computer Engineering <br>

<input type="checkbox" name="" value="Networking"> Networking

<br> <br>

Tell us what you feel about our tutorial with 50 words or less. <br>

<textarea name="comment" rows="10" cols="50"> </textarea> <br>

<input type="submit" name="submit">

</form>

</body>

</html>

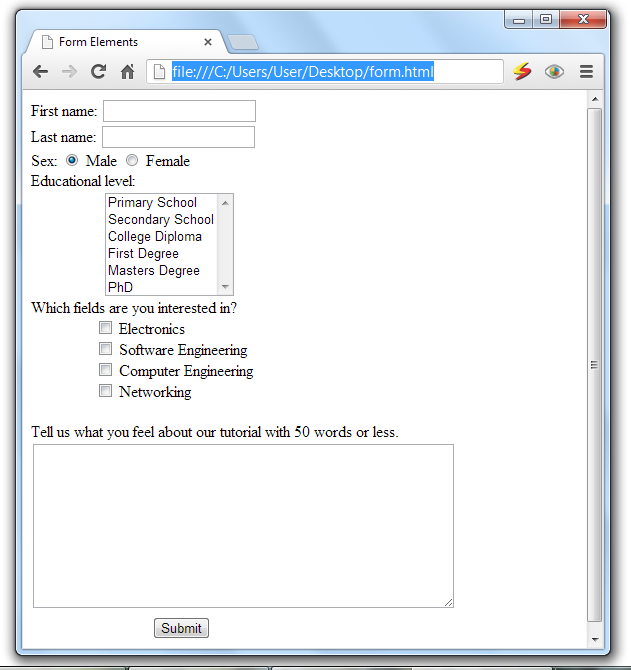
****

Fig Example form with input fields

## Linking HTML Forms with PHP Scripts

To create a link between PHP scripts and HTML forms, we need to use the functionalities of buttons, and the attribute of the <form tag>, specifically “action” and “method”.

**Submit buttons:**

When activated, a submit button submits a form. A form may contain more than one submit button.

**Action and method:**

A submit button is used to send form data to a server. The data is sent to the page specified in the form's action attribute (how to write scripts will be discussed in the coming chapters). The file defined in the action attribute usually does something with the received input:

<form name="input" action="action.php" method="get">  
Username: <input type="text" name="user" />  
<input type="submit" value="Submit" />

## Automating Processing: Info Forms and Emails

If you would like to provide your web site visitors with a simple way to contact you from your web site, but really don't want to display your email address, this HTML form code may be what you're looking for. You can create a simple form, as displayed below, to enable your visitors to send you comments, questions, product support requests, or whatever you'd like.

<form action=mailto:test@yahoo.com method="POST" enctype="text/plain">

Your Name: <input type="text" size="20" name="VisitorName"><br><br>

Your Comment: <textarea name="VisitorComment" rows="4" cols="20">

</textarea><br><br>

<input type="submit" value="Email This Form">

</form>

## Inserting Multimedia

You can add music or video into your web page. The easiest way to add video or sound to your web site is to use < audio> and <video> tag and you can also use <embed> tag. The src attribute of <embed> tag defines what video/audio file to embed into the page. This tag causes the browser itself to include controls for the multimedia automatically.

Example:

<embed src="example.mpeg" autostart="false" />

The following is the list of important attributes for <embed> element.

* **align** - determines how to align the object. It takes either *center, left or right*.
* **autostart** - Indicates if the media should start playing automatically. Netscape default is true, Internet Explorer is false.
* **loop** - Specifies if the sound should be played continuously (set loop to true), a certain number of times (a positive value) or not at all (false). This is supported by Netscape only.
* **playcount** - Specifies the number of times to play the sound. This is alternate option for *loop* if you are usiong IE.
* **hidden** - Defines if the object shows on the page. A false value means no and true means yes.
* **height** – set height of the object.
* **width** – set width of the object.
* **pluginspage** - Specifies the URL to get the plugin software.
* **name** - A name used to reference the object.
* **src** - URL of the object to be embedded. This can be any recognizable by the user's browser. It could be .mid, .wav, .mp3, .avi and so on).
* **volume** - Controls volume of the sound. Can be from 0 (off) to 100 (full volume). This attribute is supported by Netscape only.
* **Controller** – whether to show controllers like play, stop, pause, etc.

Example:

<embed src="http://www.computerhope.com/issues/ibm-linux.mov"

Pluginspage="http://www.apple.com/quicktime/" width="320" height="250"

CONTROLLER="true" LOOP="false" AUTOPLAY="false" name="IBM Video">

</embed>

File types that are supported by the embed tag are Flash movies (.swf), mpeg(.mpg or .mpeg), AVI's (.avi), and MOV's (.mov).

* .swf files - are the file types created by Adobe Flash program.
* .wmv files - are Microsoft's Window's Media Video file types.
* .mov files - are Apple's Quick Time Movie format.
* .mpeg files - are movie files created by the Moving Pictures Expert Group.

Macromedia's .swf and .mpeg formats may be the best options for use with the web because the high compression rate of these file types reduces file size and expedites the download/buffering periods for your page visitors.