**Course Web Programming Unit II**

***Contents***

*HTML Tag Reference, Global Attributes, Event Handlers, Document Structure Tags, Formatting Tags, Text Level formatting, Block Level formatting, List Tags, Hyperlink tags, Image and Image maps, Table tags, Form Tags, Frame Tags, Executable content tags. Imagemaps , Tables as a design tool, Forms : Creating Forms. Style Sheets: What are style sheets?, Why are style sheets valuable? Different approaches to style sheets, Using Multiple approaches, Linking to style information in a separate file, Setting up style information.*

**Hyper Text Markup Language [HTML]**

HTML stands for **H**yper**t**ext **M**arkup **L**anguage, and it is the most widely used language to write Web Pages.

* **Hypertext** refers to the way in which Web pages (HTML documents) are linked together. Thus the link available on a webpage are called Hypertext.
* As its name suggests, HTML is a **Markup Language** which means you use HTML to simply "mark up" a text document with tags that tell a Web browser how to structure it to display.

Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

**HTML Tags**

As told earlier, HTML is a markup language and makes use of various tags to format the content. These tags are enclosed within angle braces **<Tag Name>**. Except few tags, most of the tags have their corresponding closing tags. For example **<html>** has its closing tag **</html>** and **<body>** tag has its closing tag **</body>** tag etc.

Above example of HTML document uses following tags:

|  |  |
| --- | --- |
| **Tag** | **Description** |
| <!DOCTYPE...> | This tag defines the document type and HTML version. |
| <html> | This tag encloses the complete HTML document and mainly comprises of document header which is represented by **<head>...</head>** and document body which is represented by **<body>...</body>** tags. |
| <head> | This tag represents the document's header which can keep other HTML tags like <title>, <link> etc. |
| <title> | The **<title>** tag is used inside the <head> tag to mention the document title. |
| <body> | This tag represents the document's body which keeps other HTML tags like <h1>, <div>, <p> etc. |
| <h1> | This tag represents the heading. |
| <p> | This tag represents a paragraph. |

To learn HTML, you will need to study various tags and understand how they behave while formatting a textual document. Learning HTML is simple as users have to learn the usage of different tags in order to format the text or images to make a beautiful webpage.

World Wide Web Consortium (W3C) recommends to use lowercase tags starting from HTML 4.

**HTML Document Structure**

A typical HTML document will have following structure:

Document declaration tag

<html>

<head>

Document header related tags

</head>

<body>

Document body related tags

</body>

</html>

We will study all the header and body tags in subsequent chapters, for now let's see what is document declaration tag.

**The <!DOCTYPE> Declaration**

The <!DOCTYPE> declaration tag is used by the web browser to understand the version of the HTML used in the document. Current version of HTML is 5 and it makes use of the following declaration:

<!DOCTYPE html>

There are many other declaration types which can be used in HTML document depending on what version of HTML is being used. We will see more details on this while discussing <!DOCTYPE...> tag along with other HTML tags.

**HTML TAG REFERENCE**

Following tags have been introduced in older versions of HTML but all the tags marked with HTML-5are part of HTML-5.

|  |  |  |
| --- | --- | --- |
| **Tag** | **Description** | **Version** |
| [**<!--...-->**](https://www.tutorialspoint.com/html/html_comment_tag.htm) | Specifies a comment |  |
| [**<!DOCTYPE>**](https://www.tutorialspoint.com/html/html_doctype_tag.htm) | Specifies the document type |  |
| [**<a>**](https://www.tutorialspoint.com/html/html_a_tag.htm) | Specifies an anchor |  |
| [**<abbr>**](https://www.tutorialspoint.com/html/html_abbr_tag.htm) | Specifies an abbreviation |  |
| [**<acronym>**](https://www.tutorialspoint.com/html/html_acronym_tag.htm) | Specifies an acronym |  |
| [**<address>**](https://www.tutorialspoint.com/html/html_address_tag.htm) | Specifies an address element |  |
| [**<applet>**](https://www.tutorialspoint.com/html/html_applet_tag.htm) | Deprecated. Specifies an applet |  |
| [**<area>**](https://www.tutorialspoint.com/html/html_area_tag.htm) | Specifies an area inside an image map |  |
| [**<article>**](https://www.tutorialspoint.com/html/html_article_tag.htm) | Specifies an article | HTML-5 |
| [**<aside>**](https://www.tutorialspoint.com/html/html_aside_tag.htm) | Specifies some content loosely related to the page content. If it is removed, the remaining content still makes sense | HTML-5 |
| [**<audio>**](https://www.tutorialspoint.com/html/html_audio_tag.htm) | Specifies a sound content | HTML-5 |
| [**<b>**](https://www.tutorialspoint.com/html/html_b_tag.htm) | Specifies bold text |  |
| [**<base>**](https://www.tutorialspoint.com/html/html_base_tag.htm) | Specifies a base URL for all the links in a page |  |
| [**<basefont>**](https://www.tutorialspoint.com/html/html_basefont_tag.htm) | Deprecated. Specifies a base font |  |
| [**<bdo>**](https://www.tutorialspoint.com/html/html_bdo_tag.htm) | Specifies the direction of text display |  |
| [**<bdi>**](https://www.tutorialspoint.com/html/html_bdi_tag.htm) | Represents text that must be isolated from its surrounding for bidirectional text formatting. It allows embedding a span of text with a different, or unknown, directionality | HTML-5 |
| [**<bgsound>**](https://www.tutorialspoint.com/html/html_bgsound_tag.htm) | Specifies background music |  |
| [**<big>**](https://www.tutorialspoint.com/html/html_big_tag.htm) | Specifies big text |  |
| [**<blink>**](https://www.tutorialspoint.com/html/html_blink_tag.htm) | Specifies a text which blinks |  |
| [**<blockquote>**](https://www.tutorialspoint.com/html/html_blockquote_tag.htm) | Specifies a long quotation |  |
| [**<body>**](https://www.tutorialspoint.com/html/html_body_tag.htm) | Specifies the body element |  |
| [**<br>**](https://www.tutorialspoint.com/html/html_br_tag.htm) | Inserts a single line break |  |
| [**<button>**](https://www.tutorialspoint.com/html/html_button_tag.htm) | Specifies a push button |  |
| [**<canvas>**](https://www.tutorialspoint.com/html/html_canvas_tag.htm) | For making graphics with a script | HTML-5 |
| [**<caption>**](https://www.tutorialspoint.com/html/html_caption_tag.htm) | Specifies a table caption |  |
| [**<center>**](https://www.tutorialspoint.com/html/html_center_tag.htm) | Deprecated. Specifies centered text |  |
| [**<cite>**](https://www.tutorialspoint.com/html/html_cite_tag.htm) | Specifies a citation |  |
| [**<code>**](https://www.tutorialspoint.com/html/html_code_tag.htm) | Specifies computer code text |  |
| [**<col>**](https://www.tutorialspoint.com/html/html_col_tag.htm) | Specifies attributes for table columns |  |
| [**<colgroup>**](https://www.tutorialspoint.com/html/html_colgroup_tag.htm) | Specifies groups of table columns |  |
| [**<comment>**](https://www.tutorialspoint.com/html/html_comment_tag.htm) | Puts a comment in the document |  |
| [**<datalist>**](https://www.tutorialspoint.com/html/html_datalist_tag.htm) | A list of options for input values | HTML-5 |
| [**<dd>**](https://www.tutorialspoint.com/html/html_dd_tag.htm) | Specifies a definition description |  |
| [**<del>**](https://www.tutorialspoint.com/html/html_del_tag.htm) | Specifies deleted text |  |
| [**<dfn>**](https://www.tutorialspoint.com/html/html_dfn_tag.htm) | Specifies a definition term |  |
| [**<dialog>**](https://www.tutorialspoint.com/html/html_dialog_tag.htm) | Specifies a dialog box or window | HTML-5 |
| [**<dir>**](https://www.tutorialspoint.com/html/html_dir_tag.htm) | Deprecated. Specifies a directory list |  |
| [**<div>**](https://www.tutorialspoint.com/html/html_div_tag.htm) | Specifies a section in a document |  |
| [**<dl>**](https://www.tutorialspoint.com/html/html_dl_tag.htm) | Specifies a definition list |  |
| [**<dt>**](https://www.tutorialspoint.com/html/html_dt_tag.htm) | Specifies a definition term |  |
| [**<em>**](https://www.tutorialspoint.com/html/html_em_tag.htm) | Specifies emphasized text |  |
| [**<embed>**](https://www.tutorialspoint.com/html/html_embed_tag.htm) | Specifies a container for an external (non-HTML) application | HTML-5 |
| [**<fieldset>**](https://www.tutorialspoint.com/html/html_fieldset_tag.htm) | Specifies a fieldset |  |
| [**<figcaption>**](https://www.tutorialspoint.com/html/html_figcaption_tag.htm) | Specifies a caption for a <figure> element | HTML-5 |
| [**<figure>**](https://www.tutorialspoint.com/html/html_figure_tag.htm) | Specifies self-contained content | HTML-5 |
| [**<font>**](https://www.tutorialspoint.com/html/html_font_tag.htm) | Deprecated. Specifies text font, size, and color |  |
| [**<footer>**](https://www.tutorialspoint.com/html/html_footer_tag.htm) | Specifies a footer for a document or section | HTML-5 |
| [**<form>**](https://www.tutorialspoint.com/html/html_form_tag.htm) | Specifies a form |  |
| [**<frame>**](https://www.tutorialspoint.com/html/html_frame_tag.htm) | Specifies a sub window (a frame) |  |
| [**<frameset>**](https://www.tutorialspoint.com/html/html_frameset_tag.htm) | Specifies a set of frames |  |
| [**<h1> to <h6>**](https://www.tutorialspoint.com/html/html_hn_tag.htm) | Specifies header 1 to header 6 |  |
| [**<head>**](https://www.tutorialspoint.com/html/html_head_tag.htm) | Specifies information about the document |  |
| [**<header>**](https://www.tutorialspoint.com/html/html_header_tag.htm) | Specifies a header for a document or section | HTML-5 |
| [**<hr>**](https://www.tutorialspoint.com/html/html_hr_tag.htm) | Specifies a horizontal rule |  |
| [**<html>**](https://www.tutorialspoint.com/html/html_html_tag.htm) | Specifies an html document |  |
| [**<i>**](https://www.tutorialspoint.com/html/html_i_tag.htm) | Specifies italic text |  |
| [**<iframe>**](https://www.tutorialspoint.com/html/html_iframe_tag.htm) | Specifies an inline sub window (frame) |  |
| [**<ilayer>**](https://www.tutorialspoint.com/html/html_ilayer_tag.htm) | Specifies an inline layer |  |
| [**<img>**](https://www.tutorialspoint.com/html/html_img_tag.htm) | Specifies an image |  |
| [**<input>**](https://www.tutorialspoint.com/html/html_input_tag.htm) | Specifies an input field |  |
| [**<ins>**](https://www.tutorialspoint.com/html/html_ins_tag.htm) | Specifies inserted text |  |
| [**<isindex>**](https://www.tutorialspoint.com/html/html_isindex_tag.htm) | Deprecated. Specifies a single-line input field |  |
| [**<kbd>**](https://www.tutorialspoint.com/html/html_kbd_tag.htm) | Specifies keyboard text |  |
| [**<keygen>**](https://www.tutorialspoint.com/html/html_keygen_tag.htm) | Generate key information in a form | HTML-5 |
| [**<label>**](https://www.tutorialspoint.com/html/html_label_tag.htm) | Specifies a label for a form control |  |
| [**<layer>**](https://www.tutorialspoint.com/html/html_layer_tag.htm) | Specifies a layer |  |
| [**<legend>**](https://www.tutorialspoint.com/html/html_legend_tag.htm) | Specifies a title in a fieldset |  |
| [**<li>**](https://www.tutorialspoint.com/html/html_li_tag.htm) | Specifies a list item |  |
| [**<link>**](https://www.tutorialspoint.com/html/html_link_tag.htm) | Specifies a resource reference |  |
| [**<main>**](https://www.tutorialspoint.com/html/html_main_tag.htm) | Specifies the main or important content in the document. There is only one element in the document | HTML-5 |
| [**<map>**](https://www.tutorialspoint.com/html/html_map_tag.htm) | Specifies an image map |  |
| [**<mark>**](https://www.tutorialspoint.com/html/html_mark_tag.htm) | Specifies a text highlighted for reference purposes, that is for its relevance in another context | HTML-5 |
| [**<marquee>**](https://www.tutorialspoint.com/html/html_marquee_tag.htm) | Creates a scrolling-text marquee |  |
| [**<menu>**](https://www.tutorialspoint.com/html/html_menu_tag.htm) | Deprecated. Specifies a menu list |  |
| [**<menuitem>**](https://www.tutorialspoint.com/html/html_menuitem_tag.htm) | Specifies a command/menu item that the user can invoke from a popup menu | HTML-5 |
| [**<meta>**](https://www.tutorialspoint.com/html/html_meta_tag.htm) | Specifies meta data of an html document which is not displayed on the page |  |
| [**<meter>**](https://www.tutorialspoint.com/html/html_meter_tag.htm) | Specifies a scalar measurement within a known range (a gauge) |  |
| [**<multicol>**](https://www.tutorialspoint.com/html/html_multicol_tag.htm) | Specifies a multicolumn text flow |  |
| [**<nav>**](https://www.tutorialspoint.com/html/html_nav_tag.htm) | Specifies a section that contains only navigation links | HTML-5 |
| [**<nobr>**](https://www.tutorialspoint.com/html/html_nobr_tag.htm) | No breaks allowed in the enclosed text |  |
| [**<noembed>**](https://www.tutorialspoint.com/html/html_noembed_tag.htm) | Specifies content to be presented by browsers that do not support the <embed> tag |  |
| [**<noframes>**](https://www.tutorialspoint.com/html/html_noframes_tag.htm) | Specifies a noframe section |  |
| [**<noscript>**](https://www.tutorialspoint.com/html/html_noscript_tag.htm) | Specifies a noscript section |  |
| [**<object>**](https://www.tutorialspoint.com/html/html_object_tag.htm) | Specifies an embedded object |  |
| [**<ol>**](https://www.tutorialspoint.com/html/html_ol_tag.htm) | Specifies an ordered list |  |
| [**<optgroup>**](https://www.tutorialspoint.com/html/html_optgroup_tag.htm) | Specifies an option group |  |
| [**<option>**](https://www.tutorialspoint.com/html/html_option_tag.htm) | Specifies an option in a drop-down list |  |
| [**<output>**](https://www.tutorialspoint.com/html/html_output_tag.htm) | Specifies the result of a calculation | HTML-5 |
| [**<p>**](https://www.tutorialspoint.com/html/html_p_tag.htm) | Specifies a paragraph |  |
| [**<param>**](https://www.tutorialspoint.com/html/html_param_tag.htm) | Specifies a parameter for an object |  |
| [**<plaintext>**](https://www.tutorialspoint.com/html/html_plaintext_tag.htm) | Deprecated. Render the remainder of the document as preformatted plain text |  |
| [**<pre>**](https://www.tutorialspoint.com/html/html_pre_tag.htm) | Specifies preformatted text |  |
| [**<progress>**](https://www.tutorialspoint.com/html/html_progress_tag.htm) | Specifies a completion progress of a task | HTML-5 |
| [**<q>**](https://www.tutorialspoint.com/html/html_q_tag.htm) | Specifies a short quotation |  |
| [**<rp>**](https://www.tutorialspoint.com/html/html_rp_tag.htm) | Specifies to show browsers that do not support the ruby element | HTML-5 |
| [**<rt>**](https://www.tutorialspoint.com/html/html_rt_tag.htm) | Specifies an text ruby annotation | HTML-5 |
| [**<ruby>**](https://www.tutorialspoint.com/html/html_ruby_tag.htm) | Specifies an ruby annotation | HTML-5 |
| [**<s>**](https://www.tutorialspoint.com/html/html_strike_tag.htm) | Deprecated. Specifies strikethrough text |  |
| [**<samp>**](https://www.tutorialspoint.com/html/html_phrase_elements_tag.htm) | Specifies sample computer code |  |
| [**<script>**](https://www.tutorialspoint.com/html/html_script_tag.htm) | Specifies a script |  |
| [**<section>**](https://www.tutorialspoint.com/html/html_section_tag.htm) | Specifies a section in a document | HTML-5 |
| [**<select>**](https://www.tutorialspoint.com/html/html_select_tag.htm) | Specifies a selectable list |  |
| [**<spacer>**](https://www.tutorialspoint.com/html/html_spacer_tag.htm) | Specifies a white space |  |
| [**<small>**](https://www.tutorialspoint.com/html/html_small_tag.htm) | Specifies small text |  |
| [**<source>**](https://www.tutorialspoint.com/html/html_source_tag.htm) | Specifies a media resources for media elements, defined inside video or audio elements | HTML-5 |
| [**<span>**](https://www.tutorialspoint.com/html/html_span_tag.htm) | Specifies a section in a document |  |
| [**<strike>**](https://www.tutorialspoint.com/html/html_strike_tag.htm) | Deprecated. Specifies strikethrough text |  |
| [**<strong>**](https://www.tutorialspoint.com/html/html_strong_tag.htm) | Specifies strong text |  |
| [**<style>**](https://www.tutorialspoint.com/html/html_style_tag.htm) | Specifies a style definition |  |
| [**<sub>**](https://www.tutorialspoint.com/html/html_sub_tag.htm) | Specifies subscripted text |  |
| [**<summary>**](https://www.tutorialspoint.com/html/html_summary_tag.htm) | Specifies a summary, caption, or legend for a given <details> | HTML-5 |
| [**<sup>**](https://www.tutorialspoint.com/html/html_sup_tag.htm) | Specifies superscripted text |  |
| [**<table>**](https://www.tutorialspoint.com/html/html_table_tag.htm) | Specifies a table |  |
| [**<tbody>**](https://www.tutorialspoint.com/html/html_tbody_tag.htm) | Specifies a table body |  |
| [**<td>**](https://www.tutorialspoint.com/html/html_td_tag.htm) | Specifies a table cell |  |
| [**<textarea>**](https://www.tutorialspoint.com/html/html_textarea_tag.htm) | Specifies a text area |  |
| [**<tfoot>**](https://www.tutorialspoint.com/html/html_tfoot_tag.htm) | Specifies a table footer |  |
| [**<th>**](https://www.tutorialspoint.com/html/html_th_tag.htm) | Specifies a table heading |  |
| [**<thead>**](https://www.tutorialspoint.com/html/html_thead_tag.htm) | Specifies a table header |  |
| [**<time>**](https://www.tutorialspoint.com/html/html_time_tag.htm) | Specifies a date and time <details> | HTML-5 |
| [**<title>**](https://www.tutorialspoint.com/html/html_title_tag.htm) | Specifies the document title |  |
| [**<tr>**](https://www.tutorialspoint.com/html/html_tr_tag.htm) | Specifies a table row |  |
| [**<track>**](https://www.tutorialspoint.com/html/html_track_tag.htm) | Specifies a text tracks used in mediaplayers | HTML-5 |
| [**<tt>**](https://www.tutorialspoint.com/html/html_tt_tag.htm) | Specifies teletype text |  |
| [**<u>**](https://www.tutorialspoint.com/html/html_u_tag.htm) | Deprecated. Specifies underlined text |  |
| [**<ul>**](https://www.tutorialspoint.com/html/html_ul_tag.htm) | Specifies an unordered list |  |
| [**<var>**](https://www.tutorialspoint.com/html/html_var_tag.htm) | Specifies a variable |  |
| [**<video>**](https://www.tutorialspoint.com/html/html_video_tag.htm) | Specifies a text tracks used in mediaplayers | HTML-5 |
| [**<wbr>**](https://www.tutorialspoint.com/html/html_wbr_tag.htm) | Indicates a potential word break point within a <nobr> section |  |
| [**<xmp>**](https://www.tutorialspoint.com/html/html_xmp_tag.htm) | Deprecated. Specifies preformatted text |  |

**GLOBAL ATTRIBUTES**

= Attribute added in HTML5.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| [accesskey](http://www.w3schools.com/tags/att_global_accesskey.asp) | Specifies a shortcut key to activate/focus an element |
| [class](http://www.w3schools.com/tags/att_global_class.asp) | Specifies one or more classnames for an element (refers to a class in a style sheet) |
| [contenteditable](http://www.w3schools.com/tags/att_global_contenteditable.asp) | Specifies whether the content of an element is editable or not |
| [contextmenu](http://www.w3schools.com/tags/att_global_contextmenu.asp) | Specifies a context menu for an element. The context menu appears when a user right-clicks on the element |
| [data-\*](http://www.w3schools.com/tags/att_global_data.asp) | Used to store custom data private to the page or application |
| [dir](http://www.w3schools.com/tags/att_global_dir.asp) | Specifies the text direction for the content in an element |
| [draggable](http://www.w3schools.com/tags/att_global_draggable.asp) | Specifies whether an element is draggable or not |
| [dropzone](http://www.w3schools.com/tags/att_global_dropzone.asp) | Specifies whether the dragged data is copied, moved, or linked, when dropped |
| [hidden](http://www.w3schools.com/tags/att_global_hidden.asp) | Specifies that an element is not yet, or is no longer, relevant |
| [id](http://www.w3schools.com/tags/att_global_id.asp) | Specifies a unique id for an element |
| [lang](http://www.w3schools.com/tags/att_global_lang.asp) | Specifies the language of the element's content |
| [spellcheck](http://www.w3schools.com/tags/att_global_spellcheck.asp) | Specifies whether the element is to have its spelling and grammar checked or not |
| [style](http://www.w3schools.com/tags/att_global_style.asp) | Specifies an inline CSS style for an element |
| [tabindex](http://www.w3schools.com/tags/att_global_tabindex.asp) | Specifies the tabbing order of an element |
| [title](http://www.w3schools.com/tags/att_global_title.asp) | Specifies extra information about an element |
| [translate](http://www.w3schools.com/tags/att_global_translate.asp) | Specifies whether the content of an element should be translated or not |

**EVENT HANDLERS**

HTML 4 added the ability to let events trigger actions in a browser, like starting a JavaScript when a user clicks on an element.

Below are the global event attributes that can be added to HTML elements to define event actions.

= New event attributes in HTML5.

Window Event Attributes

Events triggered for the window object (applies to the <body> tag):

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [onafterprint](http://www.w3schools.com/tags/ev_onafterprint.asp) | *script* | Script to be run after the document is printed |
| [onbeforeprint](http://www.w3schools.com/tags/ev_onbeforeprint.asp) | *script* | Script to be run before the document is printed |
| [onbeforeunload](http://www.w3schools.com/tags/ev_onbeforeunload.asp) | *script* | Script to be run when the document is about to be unloaded |
| onerror | *script* | Script to be run when an error occurs |
| [onhashchange](http://www.w3schools.com/tags/ev_onhashchange.asp) | *script* | Script to be run when there has been changes to the anchor part of the a URL |
| [onload](http://www.w3schools.com/tags/ev_onload.asp) | *script* | Fires after the page is finished loading |
| onmessage | *script* | Script to be run when the message is triggered |
| [onoffline](http://www.w3schools.com/tags/ev_onoffline.asp) | *script* | Script to be run when the browser starts to work offline |
| [ononline](http://www.w3schools.com/tags/ev_ononline.asp) | *script* | Script to be run when the browser starts to work online |
| onpagehide | *script* | Script to be run when a user navigates away from a page |
| [onpageshow](http://www.w3schools.com/tags/ev_onpageshow.asp) | *script* | Script to be run when a user navigates to a page |
| onpopstate | *script* | Script to be run when the window's history changes |
| [onresize](http://www.w3schools.com/tags/ev_onresize.asp) | *script* | Fires when the browser window is resized |
| onstorage | *script* | Script to be run when a Web Storage area is updated |
| [onunload](http://www.w3schools.com/tags/ev_onunload.asp) | *script* | Fires once a page has unloaded (or the browser window has been closed) |

## Form Events

Events triggered by actions inside a HTML form (applies to almost all HTML elements, but is most used in form elements):

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [onblur](http://www.w3schools.com/tags/ev_onblur.asp) | *script* | Fires the moment that the element loses focus |
| [onchange](http://www.w3schools.com/tags/ev_onchange.asp) | *script* | Fires the moment when the value of the element is changed |
| [oncontextmenu](http://www.w3schools.com/tags/ev_oncontextmenu.asp) | *script* | Script to be run when a context menu is triggered |
| [onfocus](http://www.w3schools.com/tags/ev_onfocus.asp) | *script* | Fires the moment when the element gets focus |
| [oninput](http://www.w3schools.com/tags/ev_oninput.asp) | *script* | Script to be run when an element gets user input |
| [oninvalid](http://www.w3schools.com/tags/ev_oninvalid.asp) | *script* | Script to be run when an element is invalid |
| [onreset](http://www.w3schools.com/tags/ev_onreset.asp) | *script* | Fires when the Reset button in a form is clicked |
| [onsearch](http://www.w3schools.com/tags/ev_onsearch.asp) | *script* | Fires when the user writes something in a search field (for < input="search">) |
| [onselect](http://www.w3schools.com/tags/ev_onselect.asp) | *script* | Fires after some text has been selected in an element |
| [onsubmit](http://www.w3schools.com/tags/ev_onsubmit.asp) | *script* | Fires when a form is submitted |

## Keyboard Events

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [onkeydown](http://www.w3schools.com/tags/ev_onkeydown.asp) | *script* | Fires when a user is pressing a key |
| [onkeypress](http://www.w3schools.com/tags/ev_onkeypress.asp) | *script* | Fires when a user presses a key |
| [onkeyup](http://www.w3schools.com/tags/ev_onkeyup.asp) | *script* | Fires when a user releases a key |

## Mouse Events

Events triggered by a mouse, or similar user actions:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [onclick](http://www.w3schools.com/tags/ev_onclick.asp) | *script* | Fires on a mouse click on the element |
| [ondblclick](http://www.w3schools.com/tags/ev_ondblclick.asp) | *script* | Fires on a mouse double-click on the element |
| [ondrag](http://www.w3schools.com/tags/ev_ondrag.asp) | *script* | Script to be run when an element is dragged |
| [ondragend](http://www.w3schools.com/tags/ev_ondragend.asp) | *script* | Script to be run at the end of a drag operation |
| [ondragenter](http://www.w3schools.com/tags/ev_ondragenter.asp) | *script* | Script to be run when an element has been dragged to a valid drop target |
| [ondragleave](http://www.w3schools.com/tags/ev_ondragleave.asp) | *script* | Script to be run when an element leaves a valid drop target |
| [ondragover](http://www.w3schools.com/tags/ev_ondragover.asp) | *script* | Script to be run when an element is being dragged over a valid drop target |
| [ondragstart](http://www.w3schools.com/tags/ev_ondragstart.asp) | *script* | Script to be run at the start of a drag operation |
| [ondrop](http://www.w3schools.com/tags/ev_ondrop.asp) | *script* | Script to be run when dragged element is being dropped |
| [onmousedown](http://www.w3schools.com/tags/ev_onmousedown.asp) | *script* | Fires when a mouse button is pressed down on an element |
| [onmousemove](http://www.w3schools.com/tags/ev_onmousemove.asp) | *script* | Fires when the mouse pointer is moving while it is over an element |
| [onmouseout](http://www.w3schools.com/tags/ev_onmouseout.asp) | *script* | Fires when the mouse pointer moves out of an element |
| [onmouseover](http://www.w3schools.com/tags/ev_onmouseover.asp) | *script* | Fires when the mouse pointer moves over an element |
| [onmouseup](http://www.w3schools.com/tags/ev_onmouseup.asp) | *script* | Fires when a mouse button is released over an element |
| onmousewheel | *script* | Deprecated. Use the [onwheel](http://www.w3schools.com/tags/ev_onwheel.asp) attribute instead |
| [onscroll](http://www.w3schools.com/tags/ev_onscroll.asp) | *script* | Script to be run when an element's scrollbar is being scrolled |
| [onwheel](http://www.w3schools.com/tags/ev_onwheel.asp) | *script* | Fires when the mouse wheel rolls up or down over an element |

## Clipboard Events

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [oncopy](http://www.w3schools.com/tags/ev_oncopy.asp) | *script* | Fires when the user copies the content of an element |
| [oncut](http://www.w3schools.com/tags/ev_oncut.asp) | *script* | Fires when the user cuts the content of an element |
| [onpaste](http://www.w3schools.com/tags/ev_onpaste.asp) | *script* | Fires when the user pastes some content in an element |

## Media Events

Events triggered by medias like videos, images and audio (applies to all HTML elements, but is most common in media elements, like <audio>, <embed>, <img>, <object>, and <video>).

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| onabort | *script* | Script to be run on abort |
| oncanplay | *script* | Script to be run when a file is ready to start playing (when it has buffered enough to begin) |
| oncanplaythrough | *script* | Script to be run when a file can be played all the way to the end without pausing for buffering |
| oncuechange | *script* | Script to be run when the cue changes in a <track> element |
| ondurationchange | *script* | Script to be run when the length of the media changes |
| onemptied | *script* | Script to be run when something bad happens and the file is suddenly unavailable (like unexpectedly disconnects) |
| onended | *script* | Script to be run when the media has reach the end (a useful event for messages like "thanks for listening") |
| onerror | *script* | Script to be run when an error occurs when the file is being loaded |
| onloadeddata | *script* | Script to be run when media data is loaded |
| onloadedmetadata | *script* | Script to be run when meta data (like dimensions and duration) are loaded |
| onloadstart | *script* | Script to be run just as the file begins to load before anything is actually loaded |
| onpause | *script* | Script to be run when the media is paused either by the user or programmatically |
| onplay | *script* | Script to be run when the media is ready to start playing |
| onplaying | *script* | Script to be run when the media actually has started playing |
| onprogress | *script* | Script to be run when the browser is in the process of getting the media data |
| onratechange | *script* | Script to be run each time the playback rate changes (like when a user switches to a slow motion or fast forward mode) |
| onseeked | *script* | Script to be run when the seeking attribute is set to false indicating that seeking has ended |
| onseeking | *script* | Script to be run when the seeking attribute is set to true indicating that seeking is active |
| onstalled | *script* | Script to be run when the browser is unable to fetch the media data for whatever reason |
| onsuspend | *script* | Script to be run when fetching the media data is stopped before it is completely loaded for whatever reason |
| ontimeupdate | *script* | Script to be run when the playing position has changed (like when the user fast forwards to a different point in the media) |
| onvolumechange | *script* | Script to be run each time the volume is changed which (includes setting the volume to "mute") |
| onwaiting | *script* | Script to be run when the media has paused but is expected to resume (like when the media pauses to buffer more data) |

## Misc Events

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| [onerror](http://www.w3schools.com/tags/ev_onerror.asp) | *script* | Fires when an error occurs while loading an external file |
| [onshow](http://www.w3schools.com/tags/ev_onshow.asp) | *script* | Fires when a <menu> element is shown as a context menu |
| [ontoggle](http://www.w3schools.com/tags/ev_ontoggle.asp) | *script* | Fires when the user opens or closes the <details> element |

**DOCUMENT STRUCTURE TAGS**

**Basic HTML Document**

In its simplest form, following is an example of an HTML document:

<!DOCTYPE html>

<html>

<head>

<title>This is document title</title>

</head>

<body>

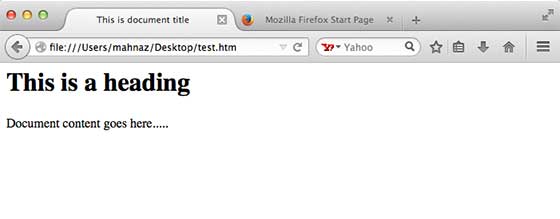
<h1>This is a heading</h1>

<p>Document content goes here.....</p>

</body>

</html>

Either you can use **Try it** option available at the top right corner of the code box to check the result of this HTML code, or let's save it in an HTML file **test.htm** using your favorite text editor. Finally open it using a web browser like Internet Explorer or Google Chrome, or Firefox etc. It must show the following output:



**FORMATTING TAGS**

If you use a word processor, you must be familiar with the ability to make text bold, italicized, or underlined; these are just three of the ten options available to indicate how text can appear in HTML and XHTML.

Bold Text

Anything that appears within **<b>...</b>** element, is displayed in bold as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Bold Text Example</title>

</head>

<body>

<p>The following word uses a <b>bold</b> typeface.</p>

</body>

</html>

This will produce following result:

The following word uses a **bold** typeface.

Italic Text

Anything that appears within **<i>...</i>** element is displayed in italicized as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Italic Text Example</title>

</head>

<body>

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

</body>

</html>

This will produce following result:

The following word uses a *italicized* typeface.

Underlined Text

Anything that appears within **<u>...</u>** element, is displayed with underline as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Underlined Text Example</title>

</head>

<body>

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

</body>

</html>

This will produce following result:

The following word uses a underlined typeface.

Strike Text

Anything that appears within **<strike>...</strike>** element is displayed with strikethrough, which is a thin line through the text as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Strike Text Example</title>

</head>

<body>

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

</body>

</html>

This will produce following result:

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

Monospaced Font

The content of a **<tt>...</tt>** element is written in monospaced font. Most of the fonts are known as variable-width fonts because different letters are of different widths (for example, the letter 'm' is wider than the letter 'i'). In a monospaced font, however, each letter has the same width.

Example

<!DOCTYPE html>

<html>

<head>

<title>Monospaced Font Example</title>

</head>

<body>

<p>The following word uses a <tt>monospaced</tt> typeface.</p>

</body>

</html>

This will produce following result:

The following word uses a monospaced typeface.

Superscript Text

The content of a **<sup>...</sup>** element is written in superscript; the font size used is the same size as the characters surrounding it but is displayed half a character's height above the other characters.

Example

<!DOCTYPE html>

<html>

<head>

<title>Superscript Text Example</title>

</head>

<body>

<p>The following word uses a <sup>superscript</sup> typeface.</p>

</body>

</html>

This will produce following result:

The following word uses a superscript typeface.

Subscript Text

The content of a **<sub>...</sub>** element is written in subscript; the font size used is the same as the characters surrounding it, but is displayed half a character's height beneath the other characters.

Example

<!DOCTYPE html>

<html>

<head>

<title>Subscript Text Example</title>

</head>

<body>

<p>The following word uses a <sub>subscript</sub> typeface.</p>

</body>

</html>

This will produce following result:

The following word uses a subscript typeface.

Inserted Text

Anything that appears within **<ins>...</ins>** element is displayed as inserted text.

Example

<!DOCTYPE html>

<html>

<head>

<title>Inserted Text Example</title>

</head>

<body>

<p>I want to drink <del>cola</del> <ins>wine</ins></p>

</body>

</html>

This will produce following result:

I want to drink wine

Deleted Text

Anything that appears within **<del>...</del>** element, is displayed as deleted text.

Example

<!DOCTYPE html>

<html>

<head>

<title>Deleted Text Example</title>

</head>

<body>

<p>I want to drink <del>cola</del> <ins>wine</ins></p>

</body>

</html>

This will produce following result:

I want to drink wine

Larger Text

The content of the **<big>...</big>** element is displayed one font size larger than the rest of the text surrounding it as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Larger Text Example</title>

</head>

<body>

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

</body>

</html>

This will produce following result:

The following word uses a big typeface.

Smaller Text

The content of the **<small>...</small>** element is displayed one font size smaller than the rest of the text surrounding it as shown below:

Example

<!DOCTYPE html>

<html>

<head>

<title>Smaller Text Example</title>

</head>

<body>

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

</body>

</html>

This will produce following result:

The following word uses a small typeface.

Grouping Content

The **<div>** and **<span>** elements allow you to group together several elements to create sections or subsections of a page.

For example, you might want to put all of the footnotes on a page within a <div> element to indicate that all of the elements within that <div> element relate to the footnotes. You might then attach a style to this <div> element so that they appear using a special set of style rules.

Example

<!DOCTYPE html>

<html>

<head>

<title>Div Tag Example</title>

</head>

<body>

<div id="menu" align="middle" >

<a href="/index.htm">HOME</a> |

<a href="/about/contact\_us.htm">CONTACT</a> |

<a href="/about/index.htm">ABOUT</a>

</div>

<div id="content" align="left" bgcolor="white">

<h5>Content Articles</h5>

<p>Actual content goes here.....</p>

</div>

</body>

</html>

This will produce following result:

The <span> element, on the other hand, can be used to group inline elements only. So, if you have a part of a sentence or paragraph which you want to group together, you could use the <span> element as follows

Example

<!DOCTYPE html>

<html>

<head>

<title>Span Tag Example</title>

</head>

<body>

<p>This is the example of <span style="color:green">span tag</span> and the <span style="color:red">div tag</span> alongwith CSS</p>

</body>

</html>

This will produce following result:

This is the example of span tag and the div tag alongwith CSS

**LIST TAGS**

HTML offers web authors three ways for specifying lists of information. All lists must contain one or more list elements. Lists may contain:

* **<ul>** - An unordered list. This will list items using plain bullets.
* **<ol>** - An ordered list. This will use different schemes of numbers to list your items.
* **<dl>** - A definition list. This arranges your items in the same way as they are arranged in a dictionary.

HTML Unordered Lists

An unordered list is a collection of related items that have no special order or sequence. This list is created by using HTML **<ul>** tag. Each item in the list is marked with a bullet.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Unordered List</title>

</head>

<body>

<ul>

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ul>

</body>

</html>

This will produce following result:

* Beetroot
* Ginger
* Potato
* Radish

The type Attribute

You can use **type** attribute for <ul> tag to specify the type of bullet you like. By default it is a disc. Following are the possible options:

<ul type="square">

<ul type="disc">

<ul type="circle">

Example

Following is an example where we used <ul type="square">

<!DOCTYPE html>

<html>

<head>

<title>HTML Unordered List</title>

</head>

<body>

<ul type="square">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ul>

</body>

</html>

This will produce following result:

* Beetroot
* Ginger
* Potato
* Radish

Example

Following is an example where we used <ul type="disc"> :

<!DOCTYPE html>

<html>

<head>

<title>HTML Unordered List</title>

</head>

<body>

<ul type="disc">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ul>

</body>

</html>

This will produce following result:

* Beetroot
* Ginger
* Potato
* Radish

Example

Following is an example where we used <ul type="circle"> :

<!DOCTYPE html>

<html>

<head>

<title>HTML Unordered List</title>

</head>

<body>

<ul type="circle">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ul>

</body>

</html>

This will produce following result:

* Beetroot
* Ginger
* Potato
* Radish

HTML Ordered Lists

If you are required to put your items in a numbered list instead of bulleted then HTML ordered list will be used. This list is created by using **<ol>** tag. The numbering starts at one and is incremented by one for each successive ordered list element tagged with <li>.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol>

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

The type Attribute

You can use **type** attribute for <ol> tag to specify the type of numbering you like. By default it is a number. Following are the possible options:

<ol type="1"> - Default-Case Numerals.

<ol type="I"> - Upper-Case Numerals.

<ol type="i"> - Lower-Case Numerals.

<ol type="a"> - Lower-Case Letters.

<ol type="A"> - Upper-Case Letters.

Example

Following is an example where we used <ol type="1">

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="1">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

Example

Following is an example where we used <ol type="I">

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="I">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

Example

Following is an example where we used <ol type="i">

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="i">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

Example

Following is an example where we used <ol type="A">

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="A">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

Example

Following is an example where we used <ol type="a">

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="a">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

The start Attribute

You can use **start** attribute for <ol> tag to specify the starting point of numbering you need. Following are the possible options:

<ol type="1" start="4"> - Numerals starts with 4.

<ol type="I" start="4"> - Numerals starts with IV.

<ol type="i" start="4"> - Numerals starts with iv.

<ol type="a" start="4"> - Letters starts with d.

<ol type="A" start="4"> - Letters starts with D.

Example

Following is an example where we used <ol type="i" start="4" >

<!DOCTYPE html>

<html>

<head>

<title>HTML Ordered List</title>

</head>

<body>

<ol type="i" start="4">

<li>Beetroot</li>

<li>Ginger</li>

<li>Potato</li>

<li>Radish</li>

</ol>

</body>

</html>

This will produce following result:

1. Beetroot
2. Ginger
3. Potato
4. Radish

HTML Definition Lists

HTML and XHTML support a list style which is called **definition lists** where entries are listed like in a dictionary or encyclopedia. The definition list is the ideal way to present a glossary, list of terms, or other name/value list.

Definition List makes use of following three tags.

* <dl> - Defines the start of the list
* <dt> - A term
* <dd> - Term definition
* </dl> - Defines the end of the list

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Definition List</title>

</head>

<body>

<dl>

<dt><b>HTML</b></dt>

<dd>This stands for Hyper Text Markup Language</dd>

<dt><b>HTTP</b></dt>

<dd>This stands for Hyper Text Transfer Protocol</dd>

</dl>

</body>

</html>

This will produce following result:

**HTML**

This stands for Hyper Text Markup Language

**HTTP**

This stands for Hyper Text Transfer Protocol

**HYPERLINK TAGS**

Links are found in nearly all web pages. Links allow users to click their way from page to page.

## HTML Links - Hyperlinks

HTML links are hyperlinks.

You can click on a link and jump to another document.

When you move the mouse over a link, the mouse arrow will turn into a little hand.

**Note:** A link does not have to be text. It can be an image or any other HTML element.

## HTML Links - Syntax

In HTML, links are defined with the **<a>** tag:

<a href="*url*">*link text*</a>

### Example

<a href="http://www.w3schools.com/html/">Visit our HTML tutorial</a>

The **href** attribute specifies the destination address

The **link text** is the visible part (Visit our HTML tutorial).

Clicking on the link text will send you to the specified address.

**Note:** Without a forward slash on subfolder addresses, you might generate two requests to the server. Many servers will automatically add a forward slash to the address, and then create a new request.

## Local Links

The example above used an absolute URL (A full web address).

A local link (link to the same web site) is specified with a relative URL (without http://www....).

### Example

<a href="html\_images.asp">HTML Images</a>

## HTML Link Colors

By default, a link will appear like this (in all browsers):

* An unvisited link is underlined and blue
* A visited link is underlined and purple
* An active link is underlined and red

You can change the default colors, by using styles:

### Example

< style>  
a:link    {color:green; background-color:transparent; text-decoration:none}  
a:visited {color:pink; background-color:transparent; text-decoration:none}  
a:hover   {color:red; background-color:transparent; text-decoration:underline}  
a:active  {color:yellow; background-color:transparent; text-decoration:underline}  
</style>

## HTML Links - The target Attribute

The **target** attribute specifies where to open the linked document.

The target attribute can have one of the following values:

* \_blank - Opens the linked document in a new window or tab
* \_self - Opens the linked document in the same window/tab as it was clicked (this is default)
* \_parent - Opens the linked document in the parent frame
* \_top - Opens the linked document in the full body of the window
* framename - Opens the linked document in a named frame

This example will open the linked document in a new browser window/tab:

### Example

<a href="http://www.w3schools.com/" target="\_blank">Visit W3Schools!</a>

**Tip:** If your webpage is locked in a frame, you can use target="\_top" to break out of the frame:

### Example

<a href="http://www.w3schools.com/html/" target="\_top">HTML5 tutorial!</a>

## HTML Links - Image as Link

It is common to use images as links:

### Example

<a href="default.asp">  
  <img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0;">  
</a>

**Note:** border:0; is added to prevent IE9 (and earlier) from displaying a border around the image (when the image is a link).

## HTML Links - Create a Bookmark

HTML bookmarks are used to allow readers to jump to specific parts of a Web page.

Bookmarks can be useful if your webpage is very long.

To make a bookmark, you must first create the bookmark, and then add a link to it.

When the link is clicked, the page will scroll to the location with the bookmark.

## Example

First, create a bookmark with the id attribute:

<h2 id="tips">Useful Tips Section</h2>

Then, add a link to the bookmark ("Useful Tips Section"), from within the same page:

<a href="#tips">Visit the Useful Tips Section</a>

Or, add a link to the bookmark ("Useful Tips Section"), from another page:

### Example

<a href="html\_tips.html#tips">Visit the Useful Tips Section</a>

**IMAGE AND IMAGE MAPS**

Images are very important to beautify as well as to depict many complex concepts in simple way on your web page. This tutorial will take you through simple steps to use images in your web pages.

Insert Image

You can insert any image in your web page by using **<img>** tag. Following is the simple syntax to use this tag.

<img src="Image URL" ... attributes-list/>

The <img> tag is an empty tag, which means that it can contain only list of attributes and it has no closing tag.

Example

To try following example, let's keep our HTML file test.htm and image file test.png in the same directory:

<!DOCTYPE html>

<html>

<head>

<title>Using Image in Webpage</title>

</head>

<body>

<p>Simple Image Insert</p>

<img src="/html/images/test.png" alt="Test Image" />

</body>

</html>

You can use PNG, JPEG or GIF image file based on your comfort but make sure you specify correct image file name in **src** attribute. Image name is always case sensitive.

The **alt** attribute is a mandatory attribute which specifies an alternate text for an image, if the image cannot be displayed.

Set Image Location

Usually we keep our all the images in a separate directory. So let's keep HTML file test.htm in our home directory and create a subdirectory **images** inside the home directory where we will keep our image test.png.

Example

Assuming our image location is "/html/image/test.png", try the following example:

<!DOCTYPE html>

<html>

<head>

<title>Using Image in Webpage</title>

</head>

<body>

<p>Simple Image Insert</p>

<img src="/html/images/test.png" alt="Test Image" />

</body>

</html>

This will produce following result:

Set Image Width/Height

You can set image width and height based on your requirement using **width** and **height** attributes. You can specify width and height of the image in terms of either pixels or percentage of its actual size.

Example

<!DOCTYPE html>

<html>

<head>

<title>Set Image Width and Height</title>

</head>

<body>

<p>Setting image width and height</p>

<img src="/html/images/test.png" alt="Test Image" width="150" height="100"/>

</body>

</html>

Set Image Border

By default image will have a border around it, you can specify border thickness in terms of pixels using **border** attribute. A thickness of 0 means, no border around the picture.

Example

<!DOCTYPE html>

<html>

<head>

<title>Set Image Border</title>

</head>

<body>

<p>Setting image Border</p>

<img src="/html/images/test.png" alt="Test Image" border="3"/>

</body>

</html>

Set Image Alignment

By default image will align at the left side of the page, but you can use **align** attribute to set it in the center or right.

Example

<!DOCTYPE html>

<html>

<head>

<title>Set Image Alignment</title>

</head>

<body>

<p>Setting image Alignment</p>

<img src="/html/images/test.png" alt="Test Image" border="3" align="right"/>

</body>

</html>

Description

The HTML <map> tag is used for defining an image map along with <img> tag.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML map Tag</title>

</head>

<body>

<img src="/images/html.gif" alt="HTML Map" border="0" usemap="#html"/>

<!-- Create Mappings -->

<map name="html">

<area shape="circle"

coords="154,150,59" href="about/about\_team.htm" alt="Team"

target="\_self" />

</map>

</body>

</html>

This will produce following result, find the image map on bottom right:



Global Attributes

This tag supports all the global attributes described in - [**HTML Attribute Reference**](https://www.tutorialspoint.com/html/html_attributes_reference.htm)

Specific Attributes

The HTML <map> tag also supports following additional attributes:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| name | unique\_name | Defines a unique name for the map tag. |

Event Attributes

This tag supports all the event attributes described in - [**HTML Events Reference**](https://www.tutorialspoint.com/html/html_events_reference.htm)

Browser Support

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chrome** | **Firefox** | **IE** | **Opera** | **Safari** | **Android** |
| Yes | Yes | Yes | Yes | Yes | Yes |

**TABLE TAGS**

The HTML tables allow web authors to arrange data like text, images, links, other tables, etc. into rows and columns of cells.

The HTML tables are created using the **<table>** tag in which the **<tr>** tag is used to create table rows and **<td>** tag is used to create data cells.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Tables</title>

</head>

<body>

<table border="1">

<tr>

<td>Row 1, Column 1</td>

<td>Row 1, Column 2</td>

</tr>

<tr>

<td>Row 2, Column 1</td>

<td>Row 2, Column 2</td>

</tr>

</table>

</body>

</html>

This will produce following result:

|  |  |
| --- | --- |
| Row 1, Column 1 | Row 1, Column 2 |
| Row 2, Column 1 | Row 2, Column 2 |

Here **border** is an attribute of <table> tag and it is used to put a border across all the cells. If you do not need a border then you can use border="0".

Table Heading

Table heading can be defined using **<th>** tag. This tag will be put to replace <td> tag, which is used to represent actual data cell. Normally you will put your top row as table heading as shown below, otherwise you can use <th> element in any row.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Header</title>

</head>

<body>

<table border="1">

<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>

</body>

</html>

This will produce following result:

|  |  |
| --- | --- |
| **Name** | **Salary** |
| Ramesh Raman | 5000 |
| Shabbir Hussein | 7000 |

Cellpadding and Cellspacing Attributes

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

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Cellpadding</title>

</head>

<body>

<table border="1" cellpadding="5" cellspacing="5">

<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>

</body>

</html>

This will produce following result:

|  |  |
| --- | --- |
| **Name** | **Salary** |
| Ramesh Raman | 5000 |
| Shabbir Hussein | 7000 |

Colspan and Rowspan Attributes

You will use **colspan** attribute if you want to merge two or more columns into a single column. Similar way you will use **rowspan** if you want to merge two or more rows.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Colspan/Rowspan</title>

</head>

<body>

<table border="1">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

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

</body>

</html>

This will produce following result:

|  |  |  |
| --- | --- | --- |
| **Column 1** | **Column 2** | **Column 3** |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Tables Backgrounds

You can set table background using one of the following two ways:

* **bgcolor** attribute - You can set background color for whole table or just for one cell.
* **background** attribute - You can set background image for whole table or just for one cell.

You can also set border color also using **bordercolor** attribute.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Background</title>

</head>

<body>

<table border="1" bordercolor="green" bgcolor="yellow">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

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

</body>

</html>

This will produce following result:

|  |  |  |
| --- | --- | --- |
| **Column 1** | **Column 2** | **Column 3** |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Here is an example of using **background** attribute. Here we will use an image available in /images directory.

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Background</title>

</head>

<body>

<table border="1" bordercolor="green" background="/images/test.png">

<tr>

<th>Column 1</th>

<th>Column 2</th>

<th>Column 3</th>

</tr>

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

</body>

</html>

This will produce following result. Here background image did not apply to table's header.

|  |  |  |
| --- | --- | --- |
| **Column 1** | **Column 2** | **Column 3** |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Table Height and Width

You can set a table width and height using **width** and **height** attrubutes. You can specify table width or height in terms of pixels or in terms of percentage of available screen area.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Width/Height</title>

</head>

<body>

<table border="1" width="400" height="150">

<tr>

<td>Row 1, Column 1</td>

<td>Row 1, Column 2</td>

</tr>

<tr>

<td>Row 2, Column 1</td>

<td>Row 2, Column 2</td>

</tr>

</table>

</body>

</html>

This will produce following result:

|  |  |
| --- | --- |
| Row 1, Column 1 | Row 1, Column 2 |
| Row 2, Column 1 | Row 2, Column 2 |

Table Caption

The **caption** tag will serve as a title or explanation for the table and it shows up at the top of the table. This tag is deprecated in newer version of HTML/XHTML.

Example

<!DOCTYPE html>

<html>

<head>

<title>HTML Table Caption</title>

</head>

<body>

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

<caption>This is the caption</caption>

<tr>

<td>row 1, column 1</td><td>row 1, columnn 2</td>

</tr>

<tr>

<td>row 2, column 1</td><td>row 2, columnn 2</td>

</tr>

</table>

</body>

</html>

This will produce following result:

|  |  |
| --- | --- |
| This is the caption | |
| row 1, column 1 | row 1, columnn 2 |
| row 2, column 1 | row 2, columnn 2 |

Table 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 holder 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

<!DOCTYPE html>

<html>

<head>

<title>HTML Table</title>

</head>

<body>

<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>

</tbody>

</table>

</body>

</html>

This will produce following result:

| This is the head of the table | | | |
| --- | --- | --- | --- |
| This is the foot of the table | | | |
| Cell 1 | Cell 2 | Cell 3 | Cell 4 |

Nested Tables

You can use one table inside another table. Not only tables you can use almost all the tags inside table data tag <td>.

Example

Following is the example of using another table and other tags inside a table cell.

<!DOCTYPE html>

<html>

<head>

<title>HTML Table</title>

</head>

<body>

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

<tr>

<td>

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

<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>

</td>

</tr>

</table>

</body>

</html>

This will produce following result:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Name** | **Salary** | | Ramesh Raman | 5000 | | Shabbir Hussein | 7000 | |

**FORM TAGS**

HTML Forms are required when you want to collect some data from the site visitor. For example during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML **<form>** tag is used to create an HTML form and it has following syntax:

<form action="Script URL" method="GET|POST">

form elements like input, textarea etc.

</form>

Form Attributes

Apart from common attributes, following is a list of the most frequently used form attributes:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Action | Backend script ready to process your passed data. |
| Method | Method to be used to upload data. The most frequently used are GET and POST methods. |
| Target | Specify the target window or frame where the result of the script will be displayed. It takes values like \_blank, \_self, \_parent etc. |
| Enctype | You can use the enctype attribute to specify how the browser encodes the data before it sends it to the server. Possible values are:   * **application/x-www-form-urlencoded** - This is the standard method most forms use in simple scenarios. * **mutlipart/form-data** - This is used when you want to upload binary data in the form of files like image, word file etc. |

**Note:** You can refer to [**Perl & CGI**](https://www.tutorialspoint.com/perl/perl_cgi_programming.htm) for a detail on how form data upload works.

HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form:

* Text Input Controls
* Checkboxes Controls
* Radio Box Controls
* Select Box Controls
* File Select boxes
* Hidden Controls
* Clickable Buttons
* Submit and Reset Button

Text Input Controls

There are three types of text input used on forms:

* **Single-line text input controls -** This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML **<input>** tag.
* **Password input controls -** This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTMl <input> tag.
* **Multi-line text input controls -** This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML **<textarea>** tag.

Single-line text input controls

This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.

Example

Here is a basic example of a single-line text input used to take first name and last name:

<!DOCTYPE html>

<html>

<head>

<title>Text Input Control</title>

</head>

<body>

<form >

First name: <input type="text" name="first\_name" />

<br>

Last name: <input type="text" name="last\_name" />

</form>

</body>

</html>

This will produce following result:

Top of Form

First name:   
Last name: 

Bottom of Form

Attributes

Following is the list of attributes for <input> tag for creating text field.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for text input control it will be set to **text**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | This can be used to provide an initial value inside the control. |
| size | Allows to specify the width of the text-input control in terms of characters. |
| maxlength | Allows to specify the maximum number of characters a user can enter into the text box. |

Password input controls

This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input> tag but type attribute is set to **password**.

Example

Here is a basic example of a single-line password input used to take user password:

<!DOCTYPE html>

<html>

<head>

<title>Password Input Control</title>

</head>

<body>

<form >

User ID : <input type="text" name="user\_id" />

<br>

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

</form>

</body>

</html>

This will produce following result:

Top of Form

User ID :     
Password: 

Bottom of Form

Attributes

Following is the list of attributes for <input> tag for creating password field.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for password input control it will be set to **password**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | This can be used to provide an initial value inside the control. |
| size | Allows to specify the width of the text-input control in terms of characters. |
| maxlength | Allows to specify the maximum number of characters a user can enter into the text box. |

Multiple-Line Text Input Controls

This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

Example

Here is a basic example of a multi-line text input used to take item description:

<!DOCTYPE html>

<html>

<head>

<title>Multiple-Line Input Control</title>

</head>

<body>

<form>

Description : <br />

<textarea rows="5" cols="50" name="description">

Enter description here...

</textarea>

</form>

</body>

</html>

This will produce following result:

Top of Form

Description :   


Bottom of Form

Attributes

Following is the list of attributes for <textarea> tag.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| rows | Indicates the number of rows of text area box. |
| cols | Indicates the number of columns of text area box |

Checkbox Control

Checkboxes are used when more than one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to **checkbox**.

Example

Here is an example HTML code for a form with two checkboxes:

<!DOCTYPE html>

<html>

<head>

<title>Checkbox Control</title>

</head>

<body>

<form>

<input type="checkbox" name="maths" value="on"> Maths

<input type="checkbox" name="physics" value="on"> Physics

</form>

</body>

</html>

This will produce following result:

Top of Form

Maths Physics

Bottom of Form

Attributes

Following is the list of attributes for <checkbox> tag.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for checkbox input control it will be set to **checkbox**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | The value that will be used if the checkbox is selected. |
| checked | Set to *checked* if you want to select it by default. |

Radio Button Control

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to **radio**.

Example

Here is example HTML code for a form with two radio buttons:

<!DOCTYPE html>

<html>

<head>

<title>Radio Box Control</title>

</head>

<body>

<form>

<input type="radio" name="subject" value="maths"> Maths

<input type="radio" name="subject" value="physics"> Physics

</form>

</body>

</html>

This will produce following result:

Top of Form

Maths Physics

Bottom of Form

Attributes

Following is the list of attributes for radio button.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| type | Indicates the type of input control and for checkbox input control it will be set to **radio**. |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| value | The value that will be used if the radio box is selected. |
| checked | Set to *checked* if you want to select it by default. |

Select Box Control

A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

Example

Here is example HTML code for a form with one drop down box

<!DOCTYPE html>

<html>

<head>

<title>Select Box Control</title>

</head>

<body>

<form>

<select name="dropdown">

<option value="Maths" selected>Maths</option>

<option value="Physics">Physics</option>

</select>

</form>

</body>

</html>

This will produce following result:

Top of Form



Bottom of Form

Attributes

Following is the list of important attributes of <select> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| size | This can be used to present a scrolling list box. |
| multiple | If set to "multiple" then allows a user to select multiple items from the menu. |

Following is the list of important attributes of <option> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| value | The value that will be used if an option in the select box box is selected. |
| selected | Specifies that this option should be the initially selected value when the page loads. |
| label | An alternative way of labeling options |

File Upload Box

If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the <input> element but type attribute is set to **file**.

Example

Here is example HTML code for a form with one file upload box:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<input type="file" name="fileupload" accept="image/\*" />

</form>

</body>

</html>

This will produce following result:

Top of Form

Bottom of Form

Attributes

Following is the list of important attributes of file upload box:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| name | Used to give a name to the control which is sent to the server to be recognized and get the value. |
| accept | Specifies the types of files that the server accepts. |

Button Controls

There are various ways in HTML to create clickable buttons. You can also create a clickable button using <input> tag by setting its type attribute to **button**. The type attribute can take the following values:

|  |  |
| --- | --- |
| **Type** | **Description** |
| submit | This creates a button that automatically submits a form. |
| reset | This creates a button that automatically resets form controls to their initial values. |
| button | This creates a button that is used to trigger a client-side script when the user clicks that button. |
| image | This creates a clickable button but we can use an image as background of the button. |

Example

Here is example HTML code for a form with three types of buttons:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<input type="submit" name="submit" value="Submit" />

<input type="reset" name="reset" value="Reset" />

<input type="button" name="ok" value="OK" />

<input type="image" name="imagebutton" src="/html/images/logo.png" />

</form>

</body>

</html>

This will produce following result:

Top of Form



Bottom of Form

Hidden Form Controls

Hidden form controls are used to hide data inside the page which later on can be pushed to the server. This control hides inside the code and does not appear on the actual page. For example, following hidden form is being used to keep current page number. When a user will click next page then the value of hidden control will be sent to the web server and there it will decide which page has be displayed next based on the passed current page.

Example

Here is example HTML code to show the usage of hidden control:

<!DOCTYPE html>

<html>

<head>

<title>File Upload Box</title>

</head>

<body>

<form>

<p>This is page 10</p>

<input type="hidden" name="pagename" value="10" />

<input type="submit" name="submit" value="Submit" />

<input type="reset" name="reset" value="Reset" />

</form>

</body>

</html>

This will produce following result:

Top of Form

This is page 10



Bottom of Form

**FRAME TAGS**

HTML frames are used to divide your browser window into multiple sections where each section can load a separate HTML document. A collection of frames in the browser window is known as a frameset. The window is divided into frames in a similar way the tables are organized: into rows and columns.

Disadvantages of Frames

There are few drawbacks with using frames, so it's never recommended to use frames in your webpages:

* Some smaller devices cannot cope with frames often because their screen is not big enough to be divided up.
* Sometimes your page will be displayed differently on different computers due to different screen resolution.
* The browser's *back button* might not work as the user hopes.
* There are still few browsers that do not support frame technology.

Creating Frames

To use frames on a page we use <frameset> tag instead of <body> tag. The <frameset> tag defines how to divide the window into frames. The **rows** attribute of <frameset> tag defines horizontal frames and **cols** attribute defines vertical frames. Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.

Example

Following is the example to create three horizontal frames:

<!DOCTYPE html>

<html>

<head>

<title>HTML Frames</title>

</head>

<frameset rows="10%,80%,10%">

<frame name="top" src="/html/top\_frame.htm" />

<frame name="main" src="/html/main\_frame.htm" />

<frame name="bottom" src="/html/bottom\_frame.htm" />

<noframes>

<body>

Your browser does not support frames.

</body>

</noframes>

</frameset>

</html>

This will produce following result:



Example

Let's put above example as follows, here we replaced rows attribute by cols and changed their width. This will create all the three frames vertically:

<!DOCTYPE html>

<html>

<head>

<title>HTML Frames</title>

</head>

<frameset cols="25%,50%,25%">

<frame name="left" src="/html/top\_frame.htm" />

<frame name="center" src="/html/main\_frame.htm" />

<frame name="right" src="/html/bottom\_frame.htm" />

<noframes>

<body>

Your browser does not support frames.

</body>

</noframes>

</frameset>

</html>

This will produce following result:



The <frameset> Tag Attributes

Following are important attributes of the <frameset> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| cols | specifies how many columns are contained in the frameset and the size of each column. You can specify the width of each column in one of four ways:   * Absolute values in pixels. For example to create three vertical frames, use *cols="100, 500,100"*. * A percentage of the browser window. For example to create three vertical frames, use *cols="10%, 80%,10%"*. * Using a wildcard symbol. For example to create three vertical frames, use *cols="10%, \*,10%"*. In this case wildcard takes remainder of the window. * As relative widths of the browser window. For example to create three vertical frames, use *cols="3\*,2\*,1\*"*. This is an alternative to percentages. You can use relative widths of the browser window. Here the window is divided into sixths: the first column takes up half of the window, the second takes one third, and the third takes one sixth. |
| rows | This attribute works just like the cols attribute and takes the same values, but it is used to specify the rows in the frameset. For example to create two horizontal frames, use *rows="10%, 90%"*. You can specify the height of each row in the same way as explained above for columns. |
| border | This attribute specifies the width of the border of each frame in pixels. For example border="5". A value of zero means no border. |
| frameborder | This attribute specifies whether a three-dimensional border should be displayed between frames. This attrubute takes value either 1 (yes) or 0 (no). For example frameborder="0" specifies no border. |
| framespacing | This attribute specifies the amount of space between frames in a frameset. This can take any integer value. For example framespacing="10" means there should be 10 pixels spacing between each frames. |

The <frame> Tag Attributes

Following are important attributes of <frame> tag:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| src | This attribute is used to give the file name that should be loaded in the frame. Its value can be any URL. For example, src="/html/top\_frame.htm" will load an HTML file available in html directory. |
| name | This attribute allows you to give a name to a frame. It is used to indicate which frame a document should be loaded into. This is especially important when you want to create links in one frame that load pages into an another frame, in which case the second frame needs a name to identify itself as the target of the link. |
| frameborder | This attribute specifies whether or not the borders of that frame are shown; it overrides the value given in the frameborder attribute on the <frameset> tag if one is given, and this can take values either 1 (yes) or 0 (no). |
| marginwidth | This attribute allows you to specify the width of the space between the left and right of the frame's borders and the frame's content. The value is given in pixels. For example marginwidth="10". |
| marginheight | This attribute allows you to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example marginheight="10". |
| noresize | By default you can resize any frame by clicking and dragging on the borders of a frame. The noresize attribute prevents a user from being able to resize the frame. For example noresize="noresize". |
| scrolling | This attribute controls the appearance of the scrollbars that appear on the frame. This takes values either "yes", "no" or "auto". For example scrolling="no" means it should not have scroll bars. |
| longdesc | This attribute allows you to provide a link to another page containing a long description of the contents of the frame. For example longdesc="framedescription.htm" |

Browser Support for Frames

If a user is using any old browser or any browser which does not support frames then <noframes> element should be displayed to the user.

So you must place a <body> element inside the <noframes> element because the <frameset> element is supposed to replace the <body> element, but if a browser does not understand <frameset> element then it should understand what is inside the <body> element which is contained in a <noframes> element.

You can put some nice message for your user having old browsers. For example *Sorry!! your browser does not support frames.* as shown in the above example.

Frame's name and target attributes

One of the most popular uses of frames is to place navigation bars in one frame and then load main pages into a separate frame.

Let's see following example where a test.htm file has following code:

<!DOCTYPE html>

<html>

<head>

<title>HTML Target Frames</title>

</head>

<frameset cols="200, \*">

<frame src="/html/menu.htm" name="menu\_page" />

<frame src="/html/main.htm" name="main\_page" />

<noframes>

<body>

Your browser does not support frames.

</body>

</noframes>

</frameset>

</html>

Here we have created two columns to fill with two frames. The first frame is 200 pixels wide and will contain the navigation menubar implemented by **menu.htm** file. The second column fills in remaining space and will contain the main part of the page and it is implemented by **main.htm** file. For all the three links available in menubar, we have mentioned target frame as **main\_page**, so whenever you click any of the links in menubar, available link will open in main\_page.

Following is the content of menu.htm file

<!DOCTYPE html>

<html>

<body bgcolor="#4a7d49">

<a href="https://www.google.com" target="main\_page">Google</a>

<br /><br />

<a href="https://www.microsoft.com" target="main\_page">Microsoft</a>

<br /><br />

<a href="https://news.bbc.co.uk" target="main\_page">BBC News</a>

</body>

</html>

Following is the content of main.htm file:

<!DOCTYPE html>

<html>

<body bgcolor="#b5dcb3">

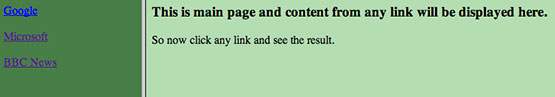
<h3>This is main page and content from any link will be displayed here.</h3>

<p>So now click any link and see the result.</p>

</body>

</html>

When we load **test.htm** file, it produces following result:



Now you can try to click links available in the left panel and see the result. The *target* attribute can also take one of the following values:

|  |  |
| --- | --- |
| **Option** | **Description** |
| \_self | Loads the page into the current frame. |
| \_blank | Loads a page into a new browser window.opening a new window. |
| \_parent | Loads the page into the parent window, which in the case of a single frameset is the main browser window. |
| \_top | Loads the page into the browser window, replacing any current frames. |
| targetframe | Loads the page into a named targetframe. |

**EXECUTABLE CONTENT TAGS**

# Executable Content

One of the most exciting recent developments in web technologies is the ability to deliver applications directly to the user's browser. These typically small programs, which are known as applets, perform simple tasks on the client computer, from responding to user mouse- or keyboard-actions to spicing up your web page displays with multimedia-enabling software.

You may embed applets into your documents using a special programming language known as JavaScript. Or you can load and execute Java-based, platform-independent applets over the Internet. During execution, these programs may generate dynamic content, interact with the user, validate form data, or even create windows and run entire applications independent of your pages. The possibilities are endless, and they go far beyond the document model that was originally envisioned for HTML.

In this chapter, we show you, with simple examples, how to embed and include executable content -- scripts and applets -- in your documents. We won't, however, even begin to pretend to teach you how to write and debug your own applets. This is a book about HTML and XTHML, after all. Rather, get an expert opinion: turn to any of the many excellent texts from O'Reilly, including JavaScript: The Definitive Guide, by David Flanagan, Java in a Nutshell, by David Flanagan, and Exploring Java, Pat Niemeyer and Josh Peck.

## Applets and Objects

Applets, like client-side image maps, represent a shift in the basic model of web communications. Until recently, servers performed most of the computational work on the Web, client browsers being not much more than glorified terminals. With applets, web technology is shifting toward the client, distributing some or all of the computational load from the server to the client and its browser.

Applets also represent a way of extending a browser's features without forcing users to purchase or otherwise acquire a new browser, as is the case when developers implement new tag and attribute extensions to HTML. Nor do users have to acquire and install a special application, as is required for helper or plug-in applications. This means that once users have a browser that supports applets (Netscape and Internet Explorer do), you can deliver applets immediately, including display and multimedia innovations.

### **The Object Model**

Java-based applets -- web page-referenced programs retrieved from a network server and executed on the user's client computer -- actually are a subset of what the HTML 4 and XHTML standards call inclusions. As with images, the browser first loads the HTML document, then examines it for inclusions -- additional, separate, and discrete content that is to be handled by the client browser. A GIF image is one type of inclusion. A .wav sound file is another; an MPEG movie is another; a Java-based clock program is another.

The HTML 4 and XHTML standards generally calls the inclusion contents objects. In fact, in your document you may identify and load nearly any object file over the network through a universal <object> tag, which we discuss in detail later in this chapter. [Section 12.2.1, "The <object> Tag"](http://docstore.mik.ua/orelly/web2/xhtml/ch12_02.htm" \l "html4-CHP-12-SECT-2.1)

Once downloaded, the standards dictate that the browser somehow render the object, by internal or external mechanisms. The popular graphical browsers, for instance, have integrated software for displaying GIF and JPEG images. Otherwise, plug-ins and other helper applications may provide the necessary rendering mechanism.

### **The applet model**

With applet-based objects, the browser sets aside a portion of the document display space. You may control the size and position of this display area; the applet controls what is presented inside.

The applet object is software, an executable program. Accordingly, besides providing a display space, the browser, in tandem with the client computer environment and resources, provides the applet runtime environment -- typically Java.

During execution, the applet has access to a restricted environment within the user's computer. For instance, applets have access to the mouse and keyboard and may receive input from the user. They can initiate network connections and retrieve data from other servers on the Internet. In sum, applets are full-fledged programs, complete with a variety of input and output mechanisms, along with a full suite of network services.

Several applets may be placed in a single document; they all execute in parallel and may communicate with each other. While the browser may limit their access to its computer system, applets have complete control of their virtual environment within the browser.

### **The applet advantage**

There are several advantages of applets, not the least of which is providing more compelling user interfaces within a web page. For instance, an applet might create a unique set of menus, choices, text fields, and similar user-input tools different from those available through the browser. When the user clicks a button within the applet's interaction/display region, the applet might respond by displaying results within the region, signaling another applet, or even by loading a completely new page into the browser.

We don't mean to imply that the only use of applets is to enhance the user interface. An applet is a full-fledged program that can perform any number of computational and user-interactive tasks on the client computer. An applet might implement a real-time video display, perform circuit simulation, engage the user in a game, provide a chat interface, and so on.

### **Using applets correctly**

An applet is nothing more than another tool you may use to produce compelling and useful web pages. Keep in mind that an applet uses computational resources on the client to run and therefore places a load on the user's computer. It can degrade system performance.

Similarly, if an applet uses a lot of network bandwidth to accomplish its task (a real-time video feed, for example), it may make other network communication unbearably slow. While such applications are fun, they do little more than annoy your target audience.

To use an applet correctly, balance the load between the browser and the server. For each page, decide which tasks are best left to the server (forms processing, index searches, and the like) and which tasks are better suited for local processing (user interface enhancements, real-time data presentation, small animations, input validation, and so on). Divide the processing accordingly. Remember that many users have slower network connections and computers than you do and design your applets to satisfy the majority of your audience.

Used the right way, applets seamlessly enhance your pages and provide a satisfying experience for your audience. Used improperly, applets are just another annoying bandwidth waster, alienating your users and hurting your pages.

### **Writing applets**

Creating applets is a programming task, not usually a job for the HTML or XHTML author, and certainly well beyond the scope of this book. For details, we recommend you consult any of the many applet programming texts that have recently appeared on bookshelves everywhere, including those from O'Reilly & Associates.

Today, one language dominates the applet programming world: Java. Developed by Sun Microsystems of Mountain View, California, Java supports an object-oriented programming style wherein classes of applets can be used and reused to build complex applications.

By invention, applets built from the same language should run with any browser that supports them. In reality, certain Microsoft implementation decisions had caused some valid Java applets to fail when running on earlier versions of Internet Explorer. Microsoft has fixed these problems with Internet Explorer Version 5 and Java will remain a universal programming language for the Web. In any case, the conscientious Java programmer should keep abreast of the latest technology and create applets that are certifiably 100% pure Java. Microsoft, in particular, is trying to get programmers to use proprietary extensions to Java that will work on only Microsoft platforms and is refusing to support key parts of the standard. We recommend avoiding any vendor extensions to Java that deviate from the standard Java 1.1 version.

You can shield yourself from platform dependencies by using the Java Plugin from Sun; see http://java.sun.com/products/plugin. There are versions for both Internet Explorer and Netscape Navigator. Currently, the plugin is the only way to get support for the latest version of Java (Java 2 SDK 1.3).[[66]](http://docstore.mik.ua/orelly/web2/xhtml/ch12_01.htm#FOOTNOTE-66)

[66] The plugin has achieved some acceptance for running Java 2 applets in intranet (i.e., corporate network) environments, but we have yet to see an applet on the public Internet that required Java 2.

We should take this opportunity to also mention ActiveX, an alternative applet programming technology available from Microsoft. ActiveX is proprietary, closely coupled with various versions of Microsoft Windows, and works only when used with Internet Explorer. ActiveX applets will run on versions of Internet Explorer targeted to various versions of Windows, but a single ActiveX applet will not run on these different versions without recompilation. This is in contrast with Java applets, where a single Java applet can be written and compiled once and immediately run on a broad range of browsers and operating systems.

ActiveX also presents an unacceptably high security risk to any user whose browser supports ActiveX technology. It is ridiculously easy to penetrate and damage a computer running a browser that allows ActiveX applets to be executed. For this reason, we cannot recommend ActiveX as a viable applet implementation technology and we go so far as to recommend that users disable ActiveX capability within their browsers -- specifically, Internet Explorer.

**Style Sheets**

Cascading Style Sheets (CSS) describe how documents are presented on screens, in print, or perhaps how they are pronounced. W3C has actively promoted the use of style sheets on the Web since the Consortium was founded in 1994.

Cascading Style Sheets (CSS) provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

Example

First let's consider an example of HTML document which makes use of <font> tag and associated attributes to specify text color and font size:

<!DOCTYPE html>

<html>

<head>

<title>HTML CSS</title>

</head>

<body>

<p><font color="green" size="5">Hello, World!</font></p>

</body>

</html>

We can re-write above example with the help of Style Sheet as follows:

<!DOCTYPE html>

<html>

<head>

<title>HTML CSS</title>

</head>

<body>

<p style="color:green;font-size:24px;">Hello, World!</p>

</body>

</html>

This will produce following result:

Hello, World!

You can use CSS in three ways in your HTML document:

* **External Style Sheet** - Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
* **Internal Style Sheet** - Define style sheet rules in header section of the HTML document using <style> tag.
* **Inline Style Sheet** - Define style sheet rules directly along-with the HTML elements using **style** attribute.

Let's see all the three cases one by one with the help of suitable examples.

External Style Sheet

If you need to use your style sheet to various pages, then its always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as **.css** and it will be included in HTML files using <link> tag.

Example

Consider we define a style sheet file **style.css** which has following rules:

.red{

color: red;

}

.thick{

font-size:20px;

}

.green{

color:green;

}

Here we defined three CSS rules which will be applicable to three different classes defined for the HTML tags. I suggest you should not bother about how these rules are being defined because you will learn them while studying CSS. Now let's make use of the above external CSS file in our following HTML document:

<!DOCTYPE html>

<html>

<head>

<title>HTML External CSS</title>

<link rel="stylesheet" type="text/css" href="/html/style.css">

</head>

<body>

<p class="red">This is red</p>

<p class="thick">This is thick</p>

<p class="green">This is green</p>

<p class="thick green">This is thick and green</p>

</body>

</html>

This will produce following result:

This is red

This is thick

This is green

This is thick and green

Internal Style Sheet

If you want to apply Style Sheet rules to a single document only then you can include those rules in header section of the HTML document using <style> tag.

Rules defined in internal style sheet overrides the rules defined in an external CSS file.

Example

Let's re-write above example once again, but here we will write style sheet rules in the same HTML document using <style> tag:

<!DOCTYPE html>

<html>

<head>

<title>HTML Internal CSS</title>

<style type="text/css">

.red{

color: red;

}

.thick{

font-size:20px;

}

.green{

color:green;

}

</style>

</head>

<body>

<p class="red">This is red</p>

<p class="thick">This is thick</p>

<p class="green">This is green</p>

<p class="thick green">This is thick and green</p>

</body>

</html>

This will produce following result:

This is red

This is thick

This is green

This is thick and green

Inline Style Sheet

You can apply style sheet rules directly to any HTML element using **style** attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only.

Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element.

Example

Let's re-write above example once again, but here we will write style sheet rules along with the HTML elements using **style** attribute of those elements.

<!DOCTYPE html>

<html>

<head>

<title>HTML Inline CSS</title>

</head>

<body>

<p style="color:red;">This is red</p>

<p style="font-size:20px;">This is thick</p>

<p style="color:green;">This is green</p>

<p style="color:green;font-size:20px;">This is thick and green</p>

</body>

</html>

This will produce following result:

This is red

This is thick

This is green

This is thick and green