SSI+ - Server Side Includes

NOTE : Server Side Includes are HTTP server specific. At present SSI+ has been implemented in the WebQuest and WebQuest95 Web servers from Questar Microsystems, Netscape and Spry Web servers and (I believe) the Microsoft Internet Information Server. You should check with your server maintainer to be sure that implementing SSI+ tokens in your HTML documents will have the desired effect. The SSI specification document is written by Mark West of Questar Microsystems. For complete and up to date versions of the specification, visit the Questar Web site at http://www.questar.com/

Server side includes (SSI) applied to an HTML document, provide for interactive real-time features such as echoing current time, conditional execution based on logical comparisons, querying or updating a database, sending an e-mail etc., with no programming or CGI scripts.

As the SSI specification is continually updated, what is presented here is just a taster of SSI+ usage. For the complete and most recent documentation visit the Questar web site (see the link above)

The possible SSI+ tags that are currently useable are :

'echo' Allows the insertion of certain variables into a HTML document

'include' Allows the insertion of files into HTML documents

'fsize' Allows for the insertion of a file's size in a HTML document

'flastmod' Allows for the insertion of a file's last modification date into a HTML document

'exec' Executes an external executable file

'config' Allows setting of various options for other SSI+ tags

'odbc' Allows the querying and updating of external ODBC databases

'email' Sends an e-mail whenever a page is accessed or a form is submitted

'if' Allows conditional execution of other SSI tags and conditional printing of documents based on logical comparisons

'goto' Makes a jump to a pre-defined SSI 'label'

'label' Provides a target for 'goto' jumps

'break' Terminates the HTML document at any point

Some brief examples

'echo'

The ‘echo’ tag is used for inserting certain variable data into a HTML document. For example :

```
You are using a <!--#echo var="HTTP_USER_AGENT" --> browser
```

would translate in the document as being (for the sake of this example, the browser is Internet Explorer 3.0 beta 2 for Windows 95) :

```
You are using a Mozilla/2.0 (compatible; MSIE 3.0A; Windows 95) browser.
```

The various different `var` values that this tag accepts are described fully in the SSI specification.

'include'

The 'include' tag provides the answer to one of the most common questions within HTML discussions. Namely, ‘How can I insert a file into an HTML page?’

By using the following in an HTML document :

```
<!--#include virtual="/includes/page.htm"-->
```

would insert the contents of the file `/includes/page.htm` into the current HTML document at the insertion point.

'email'

The 'email' tag can be used to send an e-mail message whenever (for example) a page is accessed, or some form data sent to a remote site.

Using the following in a document :

```
<!--#email tohost="htmlib.com" message="Thanks for the HTMLib"
  toaddress="response@htmlib.com" subject="The HTMLib" -->
```

would cause the mail message (specified in the message attribute) to the address specified in the toaddress attribute (if they actually existed.

As can be seen, the possibilities of SSI+ tags are extensive and it provides a mechanism for performing many common actions that would normally require CGI scripting.   As said above, SSI+ use is server specific.   If you wish to employ SSI+ tags in your HTML, consult your server maintainer to see if it is possible and also consult the Questar Web site (URL above) for the most recent specification.

HTMLib

Welcome to the HTML Reference Library

This reference, using the Internet RFC as an information base is an on-line reference library of currently supported HTML elements - their syntax, and use.
It assumes that the user has knowledge of the World Wide Web and the various HTML user agents (browsers) available. Information on the broader topic of 'The World Wide Web' can be obtained by reading the World Wide Web FAQ.

NOTE : This reference is not intended to be used as a *replacement* for any available standards documents and users are encouraged to obtain the standards documents mentioned within for more information.

Stephen Le Hunte

RFC1866 - The RFC document specifies an Internet standards track protocol for the Internet community, and is a request for discussion and suggestions for improvements.   The most recent *official* draft is HTML 3.2 (Wilbur) and is available from the W3C web site (http://www.w3.org/)
Other information provided in this document has been taken from various sources, including the Netscape, NCSA Mosaic, and Microsoft World Wide Web sites, and the various HTML authoring Usenet newsgroups.

The World Wide Web FAQ is posted (every four days) to the following UseNet newsgroups:

- news.answers
- comp.infosystems.www.users
- comp.infosystems.www.providers
- comp.infosystems.www.misc
- comp.infosystems.gopher
- comp.infosystems.wais

The most recent version is also always held at :
http://www.boutell.com/faq/
The FAQ is maintained by Thomas Boutell

HTMLib Document Conventions

Within the HTMLib there are certain typefaces that have certain meanings.

A hypertext link that is coloured blue, is a jump to an external Web site. Clicking any of these links will launch your systems default browser and load the document specified by the link. In most cases the document that will be loaded is the link, so that you know where the link will go. Where this is not the case, it is hoped that the link is obvious, i.e. The Microsoft Web site link would take you to the Microsoft web site at http://www.microsoft.com/. (The above example is an active link)

Text within a topic that is coloured dark red and bold, represents an attribute name. Normally only the first time the attribute name is mentioned will it be highlighted in red, bold text. Anywhere else, atribute names will be in a different font face.

For example, in the `<BODY>` topic, the first time the `TEXT` attribute is mentioned, it is in bold, red text : **`TEXT`**, on subsequent mentions of the attribute, it will just be in a different font : `TEXT`.

HTML Language

The vast range of HTML mark-up currently supported by available Web browsers can be divided into the following sections. Some elements featured here, may not be supported by all browsers. Where an element is known to be supported by specific browsers, the element description will be labelled as such. A more detailed comparison of the three browsers supported by this reference can be found in the Comparison Table.

Wherever this symbol : 📷 appears, a screen shot showing typical rendering of the element in question is available. To see the screenshot, click the camera.

Exactly...just like you did then. :)

Document Structure Elements

These elements are required within a HTML document for it to conform to HTML standards. Apart from the prologue document identifier, they represent the only HTML elements which are explicitly required for a document to conform to the standard.

The essential document structure elements are :

```
<HTML> ... </HTML>
<HEAD> ... </HEAD>
<BODY> ... </BODY>
```

In order to identify a document as HTML, each HTML document should start with a document prologue such as :

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">.
```

However, it is worth noting that if the document does not contain this type declaration, a browser should infer it. The above document identifier identifies the document as conforming to the HTML 2.0 DTD. There are separate prologue identifiers that should be used to define which particular DTD the document conforms to.

Prologue Identifiers

The following are various prologue identifiers that should be used in HTML documents. With the identifier is the name of the HTML DTD (document type definition) that the prologue identifier labels the HTML document as adhering to. I.e. a HTML document whose prologue identifier is `<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Level 1//EN//">` should adhere to the `HTML-1.DTD` (see the RFC for HTML 2.0 (**RFC1866**) for the DTD). Such a document should not contain Form elements for example.

The document prologue identifier should be included before the `<HTML>` element of a HTML document, in fact, it should be the first line of any HTML document.

Ways to refer to Level 3.2: most general to most specific

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN//3.2">
<!DOCTYPE HTML PUBLIC "-//W30//DTD W3 HTML 3.2//EN//">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.2//EN">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.2//EN//">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Level 3.2//EN">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 3.2 Level 3.2//EN">
```

Ways to refer to strict Level 3: most general to most specific

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Strict//EN//3.2">
<!DOCTYPE HTML PUBLIC "-//W30//DTD W3 HTML Strict 3.2//EN//">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Strict Level 3.2//EN">
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Strict 3.2 Level 3.2//EN">
```

Ways to refer to Level 2: most general to most specific
These all require conformance to the `HTML.DTD`.

```
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML Level 2//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0 Level 2//EN">
```

Ways to refer to Level 1: most general to most specific
These all require conformance to the `HTML-1.DTD`.

```
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML Level 1//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0 Level 1//EN">
```

Ways to refer to Strict Level 2: most general to most specific
These all require conformance to the `HTML-S.DTD`.

```
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML Strict//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0 Strict//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML Strict Level 2//EN">
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0 Strict Level 2//EN">
```

Ways to refer to Strict Level 1: most general to most specific
These all require conformance to the `HTML1-S.DTD`.

```
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML Strict Level 1//EN">
```

```
<!DOCTYPE HTML PUBLIC  "-//IETF//DTD HTML 2.0 Strict Level 1//EN">
```

<HTML> ... </HTML>

This element identifies the document as containing HTML elements. It should immediately follow the prologue document identifier (if used) and serves to surround all of the remaining text, including all other elements. That is, the document should be constructed thus :

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML>
Here is all the rest of the document, including any elements.
</HTML>
```

The HTML element is not visible upon HTML user agent rendering and can contain only the `<HEAD>` and `<BODY>` elements.

NOTE : The `<HTML>` element can be used to set Style Sheet informationto be used for the whole document.

<HEAD> ... </HEAD>

The head of an HTML document is an unordered collection of information about the document. It requires the Title element between `<HEAD>` and `</HEAD>` elements :

```
<HEAD>
<TITLE> Introduction to HTML </TITLE>
</HEAD>
```

The `<HEAD>` and `</HEAD>` elements do not directly affect the look of the document when rendered.

The following elements are related to the head element. They can be used to provide important information to the browser.

`<BASE>` - Allows base address of HTML document to be specified
`<ISINDEX>`- Allows keyword searching of the document
`<LINK>`- Indicate relationships between documents
`<NEXTID>`- Creates unique document identifiers
`<TITLE>`- Specifies the title of the document
`<META>` - Specifies document information useable by server/clients.
`<SYTLE>` - For the inclusion of Style Sheet information.

NOTE : The Title element is the ***only*** element described here that is ***required*** as part of the Head of a HTML document.

<BODY> ... </BODY>

Attributes See Also

The body of a HTML document contains all the text and images that make up the page, together with all the HTML elements that provide the control/formatting of the page (unless the document uses a style sheet to control presentation). The format is :

```
<BODY>
The document included here
</BODY>
```

It is possible to control the document colour scheme and background by specifying certain attributes in the `<BODY ...>` declaration. However, it should be noted that most browsers provide the user with a means to over-ride colour schemes and prevent images from loading.

The **BACKGROUND** attribute can be used to point to an image file that will be tiled across the browser window, to provide a background for the document. Specifying :

```
<BODY BACKGROUND="URL or path/filename.gif">
Document here
</BODY>
```

would cause whatever text, images, etc. appeared in that document to be placed on a background consisting of the (filename.gif) graphics file being tiled to cover the viewing area.

The **BGCOLOR** attribute, allows setting of the background colour for the document.

```
<BODY BGCOLOR="#rrggbb">
Document here
</BODY>
```

Where "#rrggbb" is a hexadecimal red-green-blue triplet used to specify the background colour

If a background image or colour is used it will probably be necessary to alter the foreground colours in order to establish a sensible contrast for the document.

TEXT
This attribute is used to control the colour of all the normal text in the document. This basically consists of all text that is not specially coloured to indicate a link. The format of `TEXT` is the same as that of `BGCOLOR`.

```
<BODY TEXT="#rrggbb">
Document here
</BODY>
```

LINK, **VLINK**, and **ALINK** attributes
These attributes allow control over the link text colouring. `VLINK` stands for visited link, and `ALINK` stands for active link. The default colouring of these is: `LINK=blue (#0000FF)`, `VLINK=purple (#400040)`, and `ALINK=red (#FF0000)`. Again, the format for these attributes is the same as that for `BGCOLOR` and `TEXT`.

```
<BODY LINK="#rrggbb" VLINK="#rrggbb" ALINK="#rrggbb">
Document here
</BODY>
```

Microsofts **Internet Explorer** supports the ability to *watermark* HTML documents, by fixing a background image (if specified) so that it doesnt scroll as a normal background image does. To give a page with a background image a watermark background, add **BGPROPERTIES=FIXED** to the BODY element as follows:

```
<BODY BACKGROUND="filename.gif" BGPROPERTIES="FIXED">
```

NOTE : This attribute is **Internet Explorer** specific.

LEFTMARGIN attribute
This **Internet Explorer** attribute allows the left hand margin of the document to be set.
For example:

```
<BODY LEFTMARGIN="60">This document is indented 60 pixels from the left hand side
  of the page</BODY>
```

TOPMARGIN attribute
This **Internet Explorer** specific attribute allows the top margin of the document to be set.
For example:

```
<BODY TOPMARGIN="60">This document is indented 60 pixels from the top of the
  page</BODY>
```

NOTE : Both of the above attributes can be set to 0, making the page start at the very top and very left hand side of the page.

For all of the background, text and link colour combinations, try the HTMLib Colour Wizard provided with this reference. This allows interactive editing of colour schemes by a variety of means without the need to constantly update a page in a browser.

Microsoft's **Internet Explorer** supports 16 and **Netscape** supports 140 discrete named colours. It should be noted that use of these colour names is browser specific, so display by other browsers will be unreliable, unless the corresponding #rrggbb triplet value is used instead. (The HTMLib Colour Wizard has support for these colour names and provides their #rrggbb triplet to ensure browser independance)
For a HTML page showing the colours, click here. **NOTE :** This page requires the Internet Explorer, or Netscape to display properly. If you don't have either of these installed, check the Netscape colours page instead, or use the HTMLib Colour Wizard. These colour names are also supported within the `<FONT COLOR="...">` element.

NOTE : All of the attributes detailed on this page can be set using Style Sheet definitions.

The following attributes are supported in the <BODY> element by the most recent browsers.

- ALINK
- BACKGROUND
- BGCOLOR
- BGPROPERTIES
- LEFTMARGIN
- LINK
- TEXT
- TOPMARGIN
- VLINK

Netscape Colours
HTMLib Colour Wizard

Netscape Colours

{button Click here to leave,SPC(16777215);CW(`newind')}

This page showing the 140 possible discrete named colours supported by the **Netscape Navigator** and **Internet Explorer** (when employing Style Sheets). Colours sourrounded by a pair of * characters are also supported by **Internet Explorer** for all elements that support colouring. It should be noted that, these colour names are not supported by other browsers and their display would be unreliable, unless the #rrggbb value is used instead.

For an indication of the colour and the #rrggbb value, click on the specific colour you are interested in, the pop-up window will be that colour. Please click the button above to leave this topic. This serves to reset the pop-up colour back to white. The text in the pop-up window can be copied by pressing **Ctrl+C**.

NOTE : The colours shown cannot be guaranteed. Due to resolution/colour depth differences of different Windows video drivers, colours may appear different on either the author's system or the end user's system. This should be considered when using any colour in documents.

Also try the HTMLib Colour Wizard

aliceblue
antiquewhite
aqua
aquamarine
azure
beige
bisque
black
blanchedalmond
blue
blueviolet
brown
burlywood
cadetblue
chartreuse
chocolate
coral
cornflowerblue
cornsilk
crimson
cyan
darkblue
darkcyan
darkgoldenrod
darkgray
darkgreen
darkkhaki
darkmagenta
darkolivegreen
darkorange
darkorchid
darkred
darksalmon
darkseagreen
darkslateblue
darkslategray
darkturquoise
darkviolet
deeppink
deepskyblue
dimgray
dodgerblue
firebrick
floralwhite
forestgreen
fuchsia
gainsboro
ghostwhite
gold
goldenrod
gray
green
greenyellow
honeydew
hotpink
indianred
indigo
ivory
khaki
lavender
lavenderblush
lawngreen
lemonchiffon
lightblue
lightcoral
lightcyan
lightgoldenrodyellow
lightgreen
lightgrey
lightpink
lightsalmon
lightseagreen
lightskyblue
lightslategray
lightsteelblue
lightyellow
lime
limegreen
linen
magenta
maroon
mediumaquamarine
mediumblue
mediumorchid
mediumpurple
mediumseagreen
mediumslateblue
mediumspringgreen
mediumturquoise
mediumvioletred
midnightblue
mintcream
mistyrose
moccasin
navajowhite
navy
oldlace
olive
olivedrab
orange
orangered
orchid
palegoldenrod
palegreen
paleturquoise
palevioletred
papayawhip
peachpuff
peru
pink
plum
powderblue
purple
red
rosybrown
royalblue
saddlebrown

salmon
sandybrown
seagreen
seashell
sienna
silver
skyblue
slateblue
slategray
snow
springgreen
steelblue
tan
teal
thistle
tomato
turquoise
violet
wheat
white
whitesmoke
yellow
yellowgreen

Phew…Thanks :)
You can leave this topic now

#F0F8FF-aliceblue

#FAEBD7-antiquewhite

#00FFFF-aqua

#7FFFD4-aquamarine

#F0FFFF-azure

#F5F5DC-beige

#FFE4C4-bisque

Oh come on, surely you know what black looks like.
Were you expecting a black pop-up here? :)

#FFEBCD-blanchedalmond

#0000FF-blue

#8A2BE2-blueviolet

#A52A2A-brown

#DEB887-burlywood

#5F9EA0-cadetblue

#7FFF00-chartreuse

#D2691E-chocolate

#FF7F50-coral

#6495ED-cornflowerblue

#FFF8DC-cornsilk

#DC143C-crimson

#00FFFF-cyan

#00008B-darkblue

#008B8B-darkcyan

#B8860B-darkgoldenrod

#A9A9A9-darkgray

#006400-darkgreen

#BDB76B-darkkhaki

#8B008B-darkmagenta

#556B2F-darkolivegreen

#FF8C00-darkorange

#9932CC-darkorchid

#8B0000-darkred

#E9967A-darksalmon

#8FBC8F-darkseagreen

#483D8B-darkslateblue

#2F4F4F-darkslategray

#00CED1-darkturquoise

#9400D3-darkviolet

#FF1493-deeppink

#00BFBF-deepskyblue

#696969-dimgray

#1E90FF-dodgerblue

#B22222-firebrick

#FFFAF0-floralwhite

#228B22-forestgreen

#FF00FF-fuchsia

#DCDCDC-gainsboro

#F8F8FF-ghostwhite

#FFD700-gold

#DAA520-goldenrod

#808080-gray

#008000-green

#ADFF2F-greenyellow

#F0FFF0-honeydew

#FF69B4-hotpink

#CD5C5C-indianred

#4B0082-indigo

#FFFFF0-ivory

#F0E68C-khaki

#E6E6FA-lavender

#FFF0F5-lavenderblush

#7CFC00-lawngreen

#FFFACD-lemonchiffon

#ADD8E6-lightblue

#F08080-lightcoral

#E0FFFF-lightcyan

#FAFAD2-lightgoldenrodyellow

#90EE90-lightgreen

#D3D3D3-lightgrey

#FFB6C1-lightpink

#FFA07A-lightsalmon

#20B2AA-lightseagreen

#87CEFA-lightskyblue

#778899-lightslategrey

#B0C4DE-lightsteelblue

#FFFFE0-lightyellow

#00FF00-lime

#32CD32-limegreen

#FAF0E6-linen

#FF00FF-magenta

#800000-maroon

#66CDAA-mediumaquamarine

#0000CD-mediumblue

#BA55D3-mediumorchid

#9370DB-mediumpurple

#3CB371-mediumseagreen

#7B68EE-mediumslateblue

#00FA9A-mediumspringgreen

#48D1CC-mediumturquoise

#C71585-mediumvioletred

#191970-midnightblue

#F5FFFA-mintcream

#FFE4E1-mistyrose

#FFE4B5-moccasin

#FFDEAD-navajowhite

#000080-navy

#FDF5E6-oldlace

#808000-olive

#6B8E23-olivedrab

#FFA500-orange

#FF4500-orangered

#DA70D6-orchid

#EEE8AA-palegoldenrod

#98FB98-palegreen

#AFEEEE-paleturquoise

#DB7093-palevioletred

#FFEFD5-papayawhip

#FFDAB9-peachpuff

#CD853F-peru

#FFC0CB-pink

#DDA0DD-plum

#B0E0E6-powderblue

#800080-purple

#FF0000-red

#BC8F8F-rosybrown

#4169E1-royalblue

#8B4513-saddlebrown

#FA8072-salmon

#F4A460-sandybrown

#2E8B57-seagreen

#FFF5EE-seashell

#A0522D-sienna

#C0C0C0-silver

#87CEEB-skyblue

#6A5ACD-slateblue

#708090-slategray

#FFFAFA-snow

#00FF7F-springgreen

#4682B4-steelblue

#D2B48C-tan

#008080-teal

#D8BFD8-thistle

#FF6347-tomato

#40E0D0-turquoise

#EE82EE-violet

#F5DEB3-wheat

#FFFFFF-white.   Surprised?

#F5F5F5-whitesmoke

#FFFF00-yellow

#9ACD32-yellowgreen

<BASE...>

See Also

The Base element allows the URL of the document to be recorded in situations in which the document may be read out of context. URLs within the document in a ″partial″ form will be resolved relative to this base address (most browsers will use the URL of the current document to resolve partial URL's if no `<BASE>` element is present). The `<BASE>` Element should only appear within the the `<HEAD>` element.

The Base element uses the **`HREF`** attribute, which identifies the base URL.

e.g. `<BASE HREF="http://www.myhost.com/">`

specifies www.myhost.com to be the base from which all relative URLs should be determined.

Frames capable browsers add one other attribute to the `BASE` element. The **`TARGET`** attribute allows the setting of a default frame pane, or targetted window for links to be loaded into.

```
<BASE TARGET="default_target">
```

NOTE : The use of the `TARGET` attribute is only supported by frames capable browsers (i.e. **Internet Explorer** or **Netscape**.

<ISINDEX...>

See Also

The ISINDEX element tells the HTML user agent that the document is an index document. As well as reading it, the reader may use a keyword search.

The document can be queried with a keyword search by adding a question mark to the end of the document address, followed by a list of keywords separated by plus signs.

> ***NOTE :*** The ISINDEX element is usually generated automatically by a server. If added manually to a HTML document, the HTML user agent assumes that the server can handle a search on the document. To use the ISINDEX element, the server must have a search engine that supports this element.

To the <ISINDEX> element Netscape authors have added the PROMPT attribute. <ISINDEX> indicates that a document is a searchable index.

PROMPT has been added so that text, chosen by the author, can be placed before the text input field of the index. This allows any author chosen message to replace the default text of :

This is a searchable index. Enter search keywords

> ***NOTE :*** The PROMPT attribute is only supported by **Netscape**.

Another **Netscape** specific attribute is **ACTION**. When used in the <ISINDEX> element, it specifies the cgi script or program to which the text string in the input box should be passed.

For example:

```
<ISINDEX ACTION="websearch">
```

would pass the text entered into the input box on the page to the cgi script "websearch".

> ***NOTE :*** "websearch" in the above example is a hypothetical cgi script. If used, the ACTION attribute must point to a properly configured script on the host machine. ACTION is **Netscape** specific.

<LINK...>

See Also

The Link element indicates a relationship between the document and some other object. A document may have any number of Link elements.

The Link element is empty (does not have a `</LINK>` closing element) and takes the same attributes as the Anchor element.

Typical uses are to indicate authorship, related indexes and glossaries, older or more recent versions, etc. Links can indicate a static tree structure in which the document was authored by pointing to a ″parent″ and ″next″ and ″previous″ document, for example.

The Link element can also be used to point to an external Style Sheet to be used within the document. Using standard Anchor attributes, the style sheet is referenced thus:

```
<LINK REL="stylesheet" TYPE="text/css" HREF="http://www.foo.com/sheet"
  TITLE="coolsheet">
```

The **REL** attribute is used to specify the `REL`ationship of the referenced file to the main document (i.e. it is a style sheet). The **TYPE** attribute is used to specify the MIME type of the style sheet (***NOTE :*** this is mandatory for **Internet Explorer** 3.0 beta 1, but may be relaxed for future versions). The **HREF** attribute provides the reference for the style sheet and the **TITLE** attribute provides the title of the style sheet to be used. For more information about Style Sheets, see the Style Sheet topic.

<NEXTID...>

See Also

The NEXTID element is a parameter read by and generated by text editing software to create unique identifiers. This element takes a single attribute which is the next document-wide alpha-numeric identifier to be allocated of the form z123 :

```
<NEXTID N=Z127>
```

When modifying a document, existing anchor identifiers should not be reused, as these identifiers may be referenced by other documents. Human writers of HTML usually use mnemonic alphabetic identifiers. HTML user agents may ignore the NEXTID element. Support for the NEXTID element does not impact HTML user agents in any way.

<TITLE> ... </TITLE>

See Also

Every HTML document must have a Title element. The title should be some arbitrary string that identifies the contents of the document and may be used in history lists, or as a label for the windows displaying the document. Unlike headings, titles are not typically rendered in the text of a document itself.

The Title element must occur within the head of the document and may not contain anchors, paragraph elements, or highlighting. Only one title is allowed in a document.

> ***NOTE :*** The length of a title is not limited, however, long titles may be truncated. To minimise the possibility, titles should be kept as succinct as possible. Also keep in mind that a short title, such as 'Introduction' may be meaningless out of context. An example of a meaningful title might be 'Introduction to HTML elements''

This is the ***only*** element that is ***required*** within the Head element. The other elements described are optional and can be used when appropriate

```
<HEAD>
<TITLE> Introduction to HTML</TITLE>
</HEAD>
```

<META...>

Attributes See Also

The META element is used within the HEAD element to embed any useful information not defined by other HTML elements. Such information can be extracted by servers/clients for use in identifying, indexing and cataloguing documents.

In addition, HTTP servers can read the content of the document head to generate response headers corresponding to any elements defining a value for the attribute HTTP-EQUIV. This provides document authors a mechanism (not necessarily the preferred one) for identifying information that should be included in the response headers for an HTTP request and forms the basis of the Client Pull mechanism.

Attributes of the META element :

HTTP-EQUIV

This attribute specifies the HTTP response header that the information pertains to. If the semantics of the HTTP response header named by this attribute is known, then the contents can be processed based on a well-defined syntactic mapping whether or not the DTD includes anything about it. HTTP header names are not case sensitive. If not present, the NAME attribute should be used to identify this meta-information and it should not be used within an HTTP response header.

NAME

Meta-information name. If the name attribute is not present, then name can be assumed equal to the value HTTP-EQUIV.

CONTENT

The meta-information content to be associated with the given name and/or HTTP response header.

Typically, the META element would be used to introduce Client Pull into HTML documents, or to specify keywords that may be used by search engines to determine a documents relevance to the current search (although some are stopping using this menthod).

Microsofts **Internet Explorer** now supports the use of HTML ratings, to restrict access to pages that whoever controls the browser deems unacceptable. This uses the W3C's PICS format of content labelling and currently supports the use of labels provided by the RSAC under their RSACi labelling vocabulary. ***NOTE :*** The PICS format devised by the W3C, defines the format for content labelling, not the actual definitions of the labels - this is up to individual organisations who (like the RSAC, in RSACi) provide *ratings services*.

Author labelling of content is a step towards responsible self-censorship in the wake of recent government intervention and provides responsible content authors with a way of labelling their web sites, so that users who do not wish to view their content, can not stumble across it by accident, as they can control the browser to block sites who provide labels for content that the user may deem offensive.

PICS rating labels are included in the document with <META> elements within the <HEAD> elements, allowing the browser to determine whether the content is 'safe' for the user, based on their ratings settings, before any of the main page content has been retrieved. It should be noted that the user should not write their own ratings within the <META> element, but should submit pages to ratings organisations for indepenent rating. With the RSAC, this involves filling out a series of short forms, detailing whether your site contains any degree of offensive language, violence, sex or nudity. They then provide the specific <META> element for the page, site directory, or whole site. An example of a RSAC PICS rating (for the HTMLib web site main page) is :

```
<META http-equiv="PICS-Label" content='(PICS-1.0
  "http://www.rsac.org/ratingsv01.html" l gen false comment "RSACi North America
  Server" by "cmlehunt@swan.ac.uk" for
  "http://subnet.virtual-pc.com/~le387818/index.html" on "1996.04.04T08:15-0500"
  exp "1997.01.01T08:15-0500" r (n 0 s 0 v 0 l 0))'>
```

The following attributes are allowed within the `<META ...>` element.

- `CONTENT`
- `HTTP-EQUIV`
- `NAME`

`<HEAD>` - The Head Element.
Dynamic Documents

<HEAD> - The Head Element.

PICS : **P**latform for **I**nternet **C**ontent **S**election is a ratings labelling format specification defined by the World Wide Web Consortium (W3C). It provides a specification for the format of content rating labels (as given by independent ratings organisations (See RSAC for more information)), not the definition of the labels themselves. For more information, see http://www.w3.org/PICS/

RSAC : **R**ecreational **S**oftware **A**dvisory **C**ouncil. The RSAC provides the most common computer game rating vocabulary, which it adapted for Web site use and called it RSACi. They provide the necessary ratings element to be placed within HTML documents after the user has submitted details of their site based on the four criteria : language, violence, sex and nudity. Each criteria has several definitions. E.g. an author can rate the site as containing language raging from ;'Inoffensive Slang' to 'Explicit or crude language' (with three settings in between). The user can set their browser (currently only **Internet Explorer**) to only accept one of the settings for each criteria, preventing access to sites that rate their content above those specified by the user. For more information, visit http://www.rsac.org/

<STYLE> ... </STYLE>

See Also

The <STYLE> element should reside within the <HEAD> element and is used to delimit Style Sheet information.

The **TYPE** and **TITLE** attributes can be used within the <STYLE> element. TYPE is used to specify the internet media (MIME) type of the style sheet definition (which is "text/css") and TITLE can be used to provide a title for the style sheet definitions. This may be used by browsers when a choice of style sheets are available.

e.g.

```
<HTML>
<HEAD>
  <TITLE>Introduction to Style Sheets</TITLE>
  <STYLE TYPE="text/css" TITLE="Bright Colours">
    BODY { color : white}
    P { color : blue;
        font-size : 12pt;
        font-family : Arial}
    H1 { color : red;
         font-size : 18pt}
  </STYLE
</HEAD>
<BODY>
...
```

Defines the styles that are specified within the <STYLE> ... </STYLE> elements to be a style sheet, having the title "Bright Colours". For more information about Style Sheets, see the Style Sheets topic.

NOTE : Use of the <STYLE> element and Style Sheet use in general is **Internet Explorer** specific.

`<HEAD>` - The Head Element
Style Sheets

Block Formatting Elements

Block formatting elements are used for the formatting of whole blocks of text within a HTML document, rather than single characters. They should all (if present) be within the body of the document. The essential block formatting elements are :

<ADDRESS> ... </ADDRESS> - Format an address section
<BASEFONT SIZE=...> - Specifying the 'default' font size for the document.
<BLOCKQUOTE> ... </BLOCKQUOTE> - To quote text from another source
<BR> - Force a line break
<CENTER> ... </CENTER>- Centering text on the page.
<COMMENT> ... </COMMENT> - To enclose text as a comment.
<DFN> ... </DFN> - Defining Instance
<DIV> … </DIV> - Allows text alignment.
<FONT ...> ... </FONT>- Setting/changing the font size, colour and type
<H1> ... </H1> - Format six levels of heading
<HR> - Renders a sizeable hard line on the page
<LISTING> ... </LISTING> - Text formatting.
<MARQUEE> - Highlighted scrolling text
<MULTICOL>- Format text in a multi-column format.
<NOBR>- Specifying that words aren't to be broken
<P> ... </P> - Specify what text constitutes a paragraph and its alignment
<PLAINTEXT> - For text formatting
<PRE> ... </PRE> - Use text already formatted
<SPACER>- Introduce hard white space blocks.
<SPAN> ... </SPAN> - Used to apply presentation mark up to text blocks
<WBR>- Specifying that a word is to be broken if necessary
<XMP> ... </XMP> - Text formatting.

<ADDRESS> ... </ADDRESS>

See Also

The Address element specifies such information as address, signature and authorship, often at the top or bottom of a document.

Typically, an Address is rendered in an italic typeface and may be indented. It carries an implied paragraph break before and after the text enclosed.

Example of use:

```
<ADDRESS>
Mr. Cosmic Kumquat<BR>
SSL Trusters Inc.<BR>
1234 Squeamish Ossifrage Road<BR>
Anywhere<BR>
NY 12345<BR>
U.S.A.
</ADDRESS>
```

📷

`<ADDRESS>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

Mr. Cosmic Kumquat
SSL Trusters Inc.
1234 Squeamish Ossifrage Road
Anywhere
NY 12345
U.S.A.

<BASEFONT ...>

Attributes See Also

NOTE : **Internet Explorer** now supports style sheets and so does not require the `<BASEFONT>` element for setting the base font style.

This changes the size of the `BASEFONT`, that all relative `<FONT SIZE ...>` changes are based on. It defaults to 3, and has a valid range of 1-7.

```
<BASEFONT SIZE=5>
```

FACE

This attribute allows changing of the face of the HTML document `BASEFONT`, exactly as for `<FONT FACE= ...>`

COLOR

This allows the `BASEFONT` colour for the HTML document to be set. Colours can either be set by using one of the colour names, or as a hex rrggbb triplet value. For details of colour names, see the colour names section `<BODY>` element.

NOTE : The `<BASEFONT...>` element, using `SIZE` and `FACE` attributes is supported only by **Netscape** and the **Internet Explorer**, with the `...COLOR` attribute being **Internet Explorer** specific.

Block Formatting Elements
Changing the `<FONT SIZE>`
Style Sheets

The following attributes are supported within the `<BASEFONT>` element. ***NOTE :*** The `FACE` attribute is specific to Microsoft's Internet Explorer. Both the Internet Explorer and Netscape support the `COLOR` attribute.

- `COLOR`
- `FACE`
- `SIZE`

<BLOCKQUOTE> ... </BLOCKQUOTE>

See Also

The BLOCKQUOTE element is used to contain text quoted from another source.

A typical rendering would be a slight extra left and right indent, and/or italic font. The BLOCKQUOTE element causes a paragraph break, and typically provides space above and below the quote.

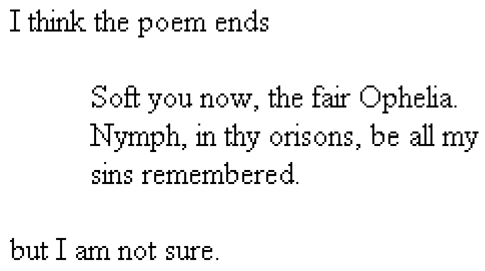
Example of use:

```
I think the poem ends
<BLOCKQUOTE>
<P>Soft you now, the fair Ophelia. Nymph,  in thy orisons, be all my sins
  remembered.
</BLOCKQUOTE>
but I am not sure.
```

📷

<BLOCKQUOTE> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

NOTE : This screenshot shows how Netscape 1.1 would display text using the `<BLOCKQUOTE>` element. Renderings using different HTML user agents may differ.



I think the poem ends

> Soft you now, the fair Ophelia.
> Nymph, in thy orisons, be all my
> sins remembered.

but I am not sure.

<BR>

See Also

The Line Break element specifies that a new line must be started at the given point. A new line indents the same as that of line-wrapped text.

Example of use:

```
<P>
Pease porridge hot<BR>
Pease porridge cold<BR>
Pease porridge in the pot<BR>
Nine days old.
```

With the introduction of floating images it became necessary to expand the `<BR>` element.   Normal `<BR>` still just inserts a line break.   A `CLEAR` attribute has been added to `<BR>`, so :

`CLEAR``=left` will break the line, and move vertically down until there is a clear left margin (no *floating* images).
`CLEAR=right` does the same for the right margin.
`CLEAR=all` moves down until both margins are clear of images.

NOTE : The screenshots on the `<IMG>` Element page use `<BR CLEAR=left>` and `<BR CLEAR=right>` respectively.

<CENTER> … </CENTER>

See Also

All lines of text between the beginning and end <CENTER> elements are centred between the current left and right margins.

```
<CENTER>All this text would be centred in the page</CENTER>
```

NOTE : Most browsers will internally work-round this element to produce the desired format, but it is an element introduced by Netscape authors.

<CENTER> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

Block Formatting Elements
<P> ... </P> Paragraph Element

<COMMENT> ... </COMMENT>

See Also

The `<COMMENT>` element can be used to 'comment' out text. As such, it is similar to the `<!-- ... -->` element.

Any text placed between the `<COMMENT>` and `</COMMENT>` elements will not render on the screen, allowing comments to be placed in HTML documents.

e.g. `<COMMENT>This text won't render.  I can say what I like here, it wont appear</COMMENT>`

Would not render on the screen.

NOTE : This element is only supported by the **Internet Explorer** and **Mosaic**.

<DFN> ... </DFN>

See Also

Use of the `<DFN>` element is currently only supported by the **Internet Explorer**.

The `<DFN>` element can be used to mark the Defining Instance of a term. For example, the first time some text is mentioned in a paragraph.
Typically, it will render italicised.

e.g. `The <DFN>Internet Explorer</DFN> is Microsoft's Web browser.`

Would render as:

The *Internet Explorer* is Microsoft's Web browser.

`<DFN>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<DIV> ... </DIV>

See Also

The <DIV> element, as described in the HTML 3.2 specification, should be used with Style Sheet attributes (as used in **Internet Explorer**), to name a section of text as being of a certain style. Netscape has implemented the DIV element to work as the <P ALIGN= ...> element. Essentially, text surrounded by the <DIV> ... </DIV> elements will be formatted according to the description attached to the **ALIGN** attribute within the <DIV> elements.

For example :

<DIV ALIGN="left">Left justify text by putting it within the DIV tags.</DIV>

<DIV ALIGN="center">Centre some text by putting it within the DIV tags.</DIV>

<DIV ALIGN="right">Right justify some text by putting it within the DIV tags.</DIV>

Internet Explorer supports the use of the <DIV> element when used with the ActiveX HTML Layout Control and Style Sheets. This allows sections of a HTML document to be authored with a fixed layout, according to the W3C 2-D layout specification. Browsers normally determine the best possible layout for HTML, but sections authored using the HTML Layout Control are fixed. Users should seek out the ActiveX Control Pad for more information about authoring using the fixed layout control.

<DIV> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

Block Formatting Elements
`<P ALIGN>` - Paragraph Alignment

<FONT ...>

Attributes See Also

NOTE : **Internet Explorer** now supports style sheets and so does not require the `<FONT>` element for individual character/block text formatting.

Netscape 1.0 (and above) and Microsoft's **Internet Explorer** support different sized fonts within HTML documents. This should be distinguished from Headings.

The element is `<FONT` **`SIZE`**`=value>`. Valid values range from 1-7 and the default font size is 3. The value given to size can optionally have a '+' or '-' character in front of it to specify that it is relative to the document basefont (if the size specified takes the font size above 7 or below 1, then values of 7 and 1 would be used). As said, the default font size is 3, and can be changed with the `<BASEFONT ...>` element.

`<FONT SIZE=4>` changes the font size to 4 `</FONT>`
`<FONT SIZE=+2>` changes the font size to `<BASEFONT SIZE ...>` + 2 `</FONT>`, or to 5 if no `<BASEFONT SIZE ...>` has been set.

NOTE : The `<FONT SIZE=...>` element is currently not supported by Mosaic.

Internet Explorer and **Netscape** support the ability to change the font colour as well as face type. They use the `COLOR` and `FACE` attributes to the `<FONT>` element

`COLOR` `= #rrggbb or colourname`

The colour attribute sets the colour which text will appear in on the screen. Rrggbb is a hexadecimal colour denoting a colour value. Alternately, the colour can be set to one of the named colours supported by the browsers. For details, see the colour names section `<BODY>` element.

Example:

```
<FONT COLOR=#ff0000>This text is red.</FONT>
```

or

```
<FONT COLOR=Red>This text is also red.</FONT>
```

`FACE``=name [,name] [,name]`

The `FACE` attribute sets the typeface that will be used to display the text on the screen. The type face displayed must already be installed on the users computer. Substitute type faces can be specified in case the chosen type face is not installed on the customers computer. If no match is found, the text will be displayed in the default type according to the browser preference settings.

Example:

```
<FONT FACE="Arial, Lucida Sans"> This text will be displayed in either Arial,
  Lucida Sans, or Times Roman, depending on which fonts you have installed on your
  system.
</FONT>
```

NOTE : When using this element, care should be taken to try to use font types that will be installed on the users computer if you want the text to appear as desired. Changing the font face is supported by **Internet Explorer** and **Netscape** only.

Block Formatting Elements
Setting the <BASEFONT> size
Style Sheets

The following attributes are supported within the `<FONT>` element. ***NOTE :*** The `FACE` attribute is specific to Microsoft's Internet Explorer. Both the Internet Explorer and Netscape support the `COLOR` attribute.

- `COLOR`
- `FACE`
- `SIZE`

<H1> ... </H1> Headings

Attributes See Also

HTML defines six levels of heading. A Heading element implies all the font changes, paragraph breaks before and after, and white space necessary to render the heading.

The highest level of headings is `<H1>`, followed by `<H2>` ... `<H6>`.

Example of use:

```
<H1>This is a heading</H1>
Here is some text
<H2>Second level heading</H2>
Here is some more text.
```

The rendering of headings is determined by the HTML user agent.

Although heading levels can be skipped (for example, from H1 to H3), this practice is discouraged as skipping heading levels may produce unpredictable results when generating other representations from HTML.

Included in the HTML level 3.2 specification is the ability to align Headings.
The **ALIGN**=`left|center|right` attribute has been added to the `<H1>` through to `<H6>` elements. e.g :

```
<H1 ALIGN=center>Hello, this is a heading</H1>
```

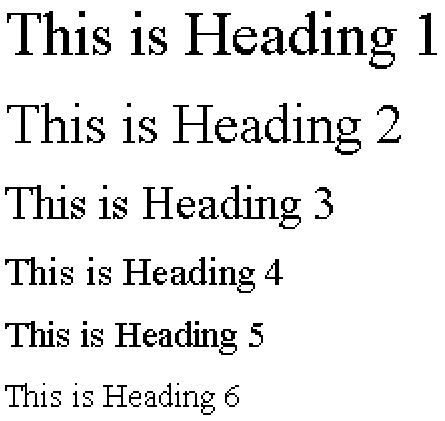
would align a heading of style 1 in the centre of the page.

`<H`*x*`>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

The following attributes are supported by recent browsers under the <H*X*> element.

ALIGN
CLASS, ID and STYLE

NOTE : These Headings are a screenshot of Mosaic heading rendering, using a default installation. The exact format can be altered within Mosaic, this screenshot is provided to show the six different headings in relation to each other.



This is Heading 1

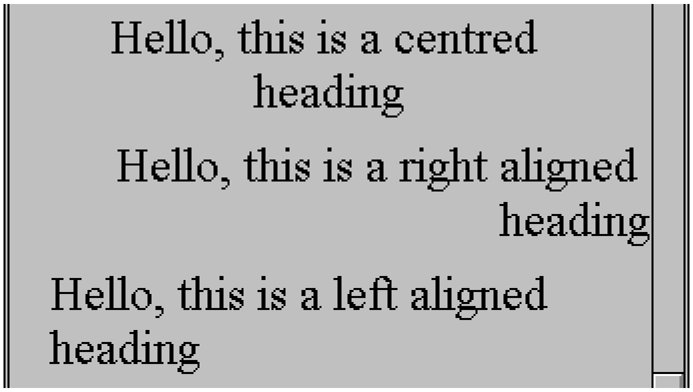
This is Heading 2

This is Heading 3

This is Heading 4

This is Heading 5

This is Heading 6



Hello, this is a centred heading

Hello, this is a right aligned heading

Hello, this is a left aligned heading

<HR>

Attributes See Also

A Horizontal Rule element is a divider between sections of text such as a full width horizontal rule or equivalent graphic.

Example of use:

```
<HR>
<ADDRESS>February 8, 1995, CERN</ADDRESS>
</BODY>
```

The <HR> element specifies that a horizontal rule of some sort (The default being a shaded engraved line) be drawn across the page. To this element recent browsers have added support for 4 new attributes which allow the document author to describe how the horizontal rule should look

<HR **SIZE**=number>
The SIZE attributes lets the author give an indication of how thick they wish the horizontal rule to be. A pixel value should be given.

<HR **WIDTH**=number|percent>
The default horizontal rule is always as wide as the page. With the WIDTH attribute, the author can specify an exact width in pixels, or a relative width measured in percent of document width.

<HR **ALIGN**=left|right|center>
Now that horizontal rules do not have to be the width of the page it possible to specify the horizontal alignment of the rule.

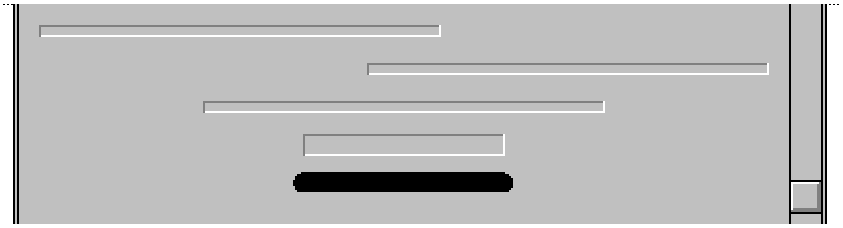
<HR **NOSHADE**>
For those times when a solid bar is required, the NOSHADE attribute specifies that the horizontal rule should not be shaded at all.

📷

<HR **COLOR=**name|#rrggbb>
Internet Explorer allows the specifying of the hard rule colour. Accepted values are any of the **Explorer** supported colour names, or any acceptable rrggbb hex triplet.

The following <HR> render as shown

```
<HR SIZE=5 WIDTH=200 ALIGN=Left>
<HR SIZE=5 WIDTH=200 ALIGN=Right>
<HR SIZE=5 WIDTH=200 ALIGN=Center>
<HR SIZE=10 WIDTH=100 ALIGN=Center>
<HR SIZE=10 WIDTH=100 ALIGN=Center NOSHADE>
```



NOTE : The page side bars are shown to emphasise the line positioning on the page.

Recent browsers support these additional attributes to the `<HR>` element

- `ALIGN`
- `COLOR`
- `NOSHADE`
- `SIZE`
- `WIDTH`

<LISTING> ... </LISTING>

See Also

The <LISTING> element can be used to present blocks of text in fixed-width font, and so is suitable for text that has been formatted on screen. As such, it is similar to the <PRE> and <XMP> elements, but has a different syntax.

Typically, it will render as fixed width font with white space separating it from other text. It should be rendered such that 132 characters fit on the line. ***NOTE :*** Only **Netscape** actually complies with this

e.g. `Some might say<LISTING>that sunshine</LISTING> follows thunder`

Would render as:

Some might say

```
that sunshine
```

follows thunder.

NOTE : The **Internet Explorer** and **Netscape** will translate any special characters included within <LISTING> elements. I.e. if characters such as &lt;, &gt; etc. are used, they will be translated to < and >. **Mosaic** treats the text contained within the elements literally.

<LISTING> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<MARQUEE> … </MARQUEE>

Attributes See Also

> ***NOTE :*** This element is currently only supported by the Microsoft **Internet Explorer**. For authors writing specifically for Netscape, a marquee effect can be obtained through the use of a Java applet, JavaScript script, or by using a multi-block GIF 'banner' (these can be constructed using the GIF construction kit).

The new `<MARQUEE>` element allows the author to create a scrolling text marquee (as the name suggests, a scrolling text region much like the Windows Marquee screen saver).

Marquees can be left- or right-aligned, like images and have a variety of attributes to control them.

ALIGN

This attribute can be set to either `TOP`, `MIDDLE` or `BOTTOM` and specifies that the text around the marquee should align with the top, middle, or bottom of the marquee.

BEHAVIOR

This can be set to `SCROLL`, `SLIDE` or `ALTERNATE`. It specifies how the text should behave. `SCROLL` (the default) means start completely off one side, scroll all the way across and completely off, then start again. `SLIDE` means start completely off on side, scroll in and stop as soon as the text touches the other margin. `ALTERNATE` means bounce back and forth within the marquee.

BGCOLOR

This specifies a background colour for the marquee, either as a rrggbb hex triplet, or as one of the Internet Explorer prenamed colours. (See `<FONT COLOR>` for more information)

DIRECTION

This specifies in which direction the text should scroll. The default is `LEFT`, which means scrolling to the left from the right. This attribute can also be set to `RIGHT`, which would cause the marquee to scroll from the left to the right.

HEIGHT

This specifies the height of the marquee, either in pixels `(HEIGHT=`*n*`)` or as a percentage of the screen height `(HEIGHT=n%)`.

WIDTH

This specifies the width of the marquee, either in pixels `(WIDTH=n)` or as a percentage of the screen height `(WIDTH=n%)`.

HSPACE

This specifies left and right margins for the outside of the marquee, in pixels.

LOOP

`LOOP` specifies how many times a marquee will loop when activated. If n=-1, or `LOOP=INFINITE` is specifies, the marquee will loop indefinitely.

SCROLLAMOUNT

Specifies the number of pixels between each successive draw of the marquee text. That is, the amount for the text to move between each draw.

SCROLLDELAY

`SCROLLDELAY` specifies the number of milliseconds between each successive draw of the marquee text. That is, it controls the speed at which text draw takes place.

VSPACE

VSPACE specifies the top and bottom margins for the outside of the marquee, in pixels.

Examples

```
<MARQUEE>This text will scroll from left to right slowly</MARQUEE>

<MARQUEE BEHAVIOR=SLIDE>This marquee will scroll in and "stick."</MARQUEE>

<MARQUEE HEIGHT=50% WIDTH=80%>This marquee, is half the height of the screen and
  80% of the screen width.</MARQUEE>

<MARQUEE SCROLLDELAY=5 SCROLLAMOUNT=50>This is a very fast marquee.</MARQUEE>
```

<MARQUEE> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

The following attributes are supported within the `<MARQUEE>` element.

- `ALIGN`
- `BEHAVIOR`
- `BGCOLOR`
- `DIRECTION`
- `HEIGHT`
- `HSPACE`
- `LOOP`
- `SCROLLAMOUNT`
- `SCROLLDELAY`
- `VSPACE`
- `WIDTH`

This element is **Internet Explorer** specific.

<MULTICOL> ... </MULTICOL>

Attributes See Also

This **Netscape** specific attribute allows text to be formatted in multiple columns. The `<MULTICOL>` element can be nested, producing multiple column layouts within multiple column layouts. It takes the following attributes :

COLS

This specifies the number of columns to be used to render the text. It accepts a numerical values. It is the only ***required*** attribute for the `<MULTICOL>` element.

GUTTER

If specified this setting (which accepts a pixel value) controls the amount of white space separating adjacent columns in the layout.

WIDTH

This specifies the width of the entire layout and can accept a pixel value, or a percentage value (which will be calculated as a percentage value of the browser window).

NOTE : As this element is **Netscape** specific, it may be wise to consider using a borderless `<TABLE>` based layout for the same effect. While this will not allow every available browser to view the content, it ensures a wider possible audience.

Example

The following HTML fragment will display as three columns, using 25 pixels white space between each column.

```
<MULTICOL COLS="3" GUTTER="25">
<P>Hello and Welcome to the HTML Reference Library.  To those of you familiar with
  the previous incarnation of this project (The HTML Reference Library - HTMLib in
  Windows .HLP format) the content and working of these pages will probably be
  obvious.
<P>To those of you new to the world of the HTML Reference Library, a little
  introduction :
</MULTICOL>
```

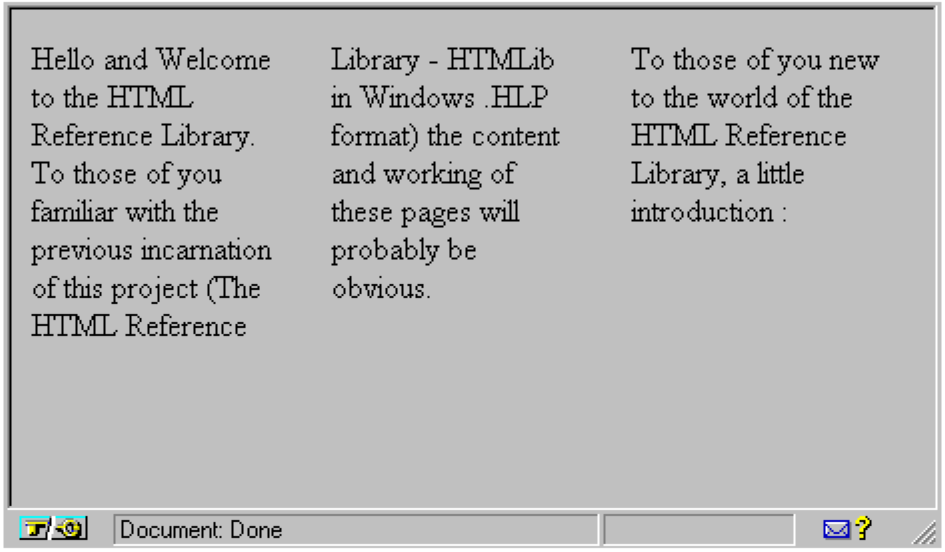
The <MULTICOL> element accepts the following attributes

COLS

GUTTER

WIDTH

The screenshot shows a **Netscape** specific `<MULTICOL>` layout.



Hello and Welcome to the HTML Reference Library. To those of you familiar with the previous incarnation of this project (The HTML Reference Library - HTMLib in Windows .HLP format) the content and working of these pages will probably be obvious.

To those of you new to the world of the HTML Reference Library, a little introduction :

<NOBR>

See Also

The <NOBR> element stands for **NO BR**eak.   This means all the text between the start and end of the <NOBR> elements cannot have line breaks inserted.   While <NOBR> is useful for character sequences that don't want to be broken, care should be taken as long text strings inside of <NOBR> elements can appear strange, especially if during viewing, the user adjust the page size by altering the window size.

NOTE : The <NOBR> Element is supported by **Netscape** and the **Internet Explorer**.

<P> ... </P>

See Also

The Paragraph element indicates a paragraph. The exact indentation, leading, etc. of a paragraph is not defined and may be a function of other elements, Style Sheets, etc.

Typically, paragraphs are surrounded by a vertical space of one line or half a line. With some browsers, the first line in a paragraph is indented.

Example of use:

```
<H1>This Heading Precedes the Paragraph</H1>
<P>This is the text of the first paragraph.
<P>This is the text of the second paragraph. Although you do not need to start
  paragraphs on new lines, maintaining this convention facilitates document
  maintenance.
<P>This is the text of a third paragraph.
```

Included in the HTML level 3.2 specification is the ability to align paragraphs
Basically, **ALIGN**=`left|center|right` attributes have been added to the `<P>` element.
e.g :

```
<P ALIGN=LEFT> ... </P>
```

All text within the paragraph will be aligned to the left side of the page layout. This setting is equal to the default `<P>` element.

```
<P ALIGN=CENTER> ... </P>
```

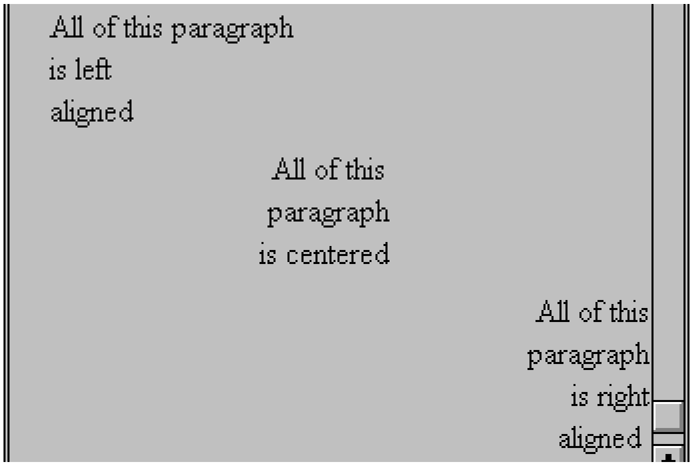
All text within the paragraph will be aligned to the centre of the page.

```
<P ALIGN=RIGHT> ... </P>
```

All text will be aligned to the right side of the page.

📷

`<P>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.



All of this paragraph
is left
aligned

All of this
paragraph
is centered

All of this
paragraph
is right
aligned

<PLAINTEXT>

See Also

The <PLAINTEXT> element can be used to represent formatted text. As such, it is similar to the <XMP> and <LISTING> element. However, the <PLAINTEXT> element should be an open element, with no closing element. Only **Netscape** supports this element according to any HTML specification. **Internet Explorer** and **Mosaic** will both allow the use of a </PLAINTEXT> closing element. **Netscape** will treat the closing element as straight text.

Typically, it will render as fixed width font with white space separating it from other text.

e.g. `We live<PLAINTEXT>as we breathe alone`

Would render as:

We live

```
as we breathe alone.
```

As said above, anything following the opening <PLAINTEXT> element should be treated as text. Only **Netscape** behaves like this. **Internet Explorer** and **Mosaic** will allow the use of a closing </PLAINTEXT> element.

<PLAINTEXT> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

Block Formatting Elements

<PRE> ... </PRE>

See Also

The Preformatted Text element presents blocks of text in fixed-width font, and so is suitable for text that has been formatted on screen.

The `<PRE>` element may be used with the optional `WIDTH` attribute, which is a Level 1 feature. The `WIDTH` attribute specifies the maximum number of characters for a line and allows the HTML user agent to select a suitable font and indentation. If the `WIDTH` attribute is not present, a width of 80 characters is assumed. Where the `WIDTH` attribute is supported, widths of 40, 80 and 132 characters should be presented optimally, with other widths being rounded up.

Within preformatted text:

- Line breaks within the text are rendered as a move to the beginning of the next line.
- The `<P>` element should not be used. If found, it should be rendered as a move to the beginning of the next line.
- Anchor elements and character highlighting elements may be used.
- Elements that define paragraph formatting (headings, address, etc.) must not be used.
- The horizontal tab character (encoded in US-ASCII and ISO-8859-1 as decimal 9) must be interpreted as the smallest positive nonzero number of spaces which will leave the number of characters so far on the line as a multiple of 8. Its use is not recommended however.

NOTE: References to the "beginning of a new line" do not imply that the renderer is forbidden from using a constant left indent for rendering preformatted text. The left indent may be constrained by the width required.

Example of use:

```
<PRE WIDTH="80">
This is an example line.
</PRE>
```

NOTE: Within a Preformatted Text element, the constraint that the rendering must be on a fixed horizontal character pitch may limit or prevent the ability of the HTML user agent to render highlighting elements specially.

`<PRE>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

<SPACER>

Attributes See Also

This **Netscape** specific attribute can be used to include hard coded white space in a document. It accepts the following attributes :

TYPE
The TYPE of <SPACER> can be either horizontal, vertical or block. A horizontal <SPACER> inserts horizontal white space in between words. A vertical <SPACER> inserts vertical space between lines and the block <SPACER> behaves like an image (except doesn't display an image.) It essentially, reserves a block of space within the document.

SIZE
When a <SPACER TYPE="horizontal"> or <SPACER TYPE="vertical"> is used, the SIZE attributes specifies its size in pixels. Any setting of this attribute is ignored if a <SPACER TYPE="block"> is used.

WIDTH
When a <SPACER TYPE="block"> is used, the WIDTH attributes specifies the WIDTH of the block to be reserved as white space.

HEIGHT
When a <SPACER TYPE="block"> is used, the HEIGHT attributes specifies the HEIGHT of the block to be reserved as white space.

ALIGN
When a <SPACER="block"> is used, the ALIGN attributes specifies the alignment of the white space block and can accept any alignments as used by the <IMG> element for embedding images.

Example
The following HTML fragment inserts a horizontal spacer of 50 pixels between two words.

```
<P>Hello and Welcome<SPACER TYPE="horizontal" SIZE="25">to the HTML Reference
  Library.  To those of you familiar with the previous incarnation of this project
  (The HTML Reference Library - HTMLib in Windows .HLP format) the content and
  working of these pages will probably be obvious.
```

The `<SPACER>` element accepts the following attributes

- `TYPE`
- `SIZE`
- `WIDTH`
- `HEIGHT`
- `ALIGN`

The screenshot shows a **Netscape** specific `<SPACER>` spacing.

<SPAN> ... </SPAN>

See Also

The SPAN element is used to apply a style to text which doesn't play any structural role, or where use of standard HTML elements is not desirable. For example, it may be useful for text to be highlighted by rendering it with a different background colour. For text such as this, using a standard HTML element such as <EM> with an applied style, would possibly be inappropriate, because browsers that don't support style sheets would render the text as italicised. The <SPAN> element is recommended in such situations as other browsers simply ignore it.

The SPAN element can be used within text blocks to apply a style as defined in a style sheet, according to a **CLASS** or **ID** attribute, or the **STYLE** can be specified within the <SPAN> attribute. As with other elements used within the <BODY> of a HTML document, SPAN can also have a certain style applied to it in the style sheet definition.

e.g. If `.redtext : { color : #FF0000}` has been defined in a style sheet, then the following :

```
some text<SPAN CLASS="redtext">some red text</SPAN>some more text
```

would render the `some red text` section in red.

```
<SPAN STYLE="color : #FF0000">some red text</SPAN>
```

would do exactly the same

See the Style Sheets topic for more information about style sheets.

Block Formatting Elements
Style Sheets

<WBR>

See Also

The `<WBR>` element stands for **W**ord **BR**eak. This is for the very rare case when a `<NOBR>` section requires an exact break. Also, it can be used any time the browser can be helped by telling it where a word is allowed to be broken. The `<WBR>` element does not force a line break (`<BR>` does that) it simply lets the browser know where a line break is allowed to be inserted if needed.

NOTE : The `<WBR>` Element is supported by **Netscape** and the **Internet Explorer**.

<XMP> ... </XMP>

See Also

The <XMP> element can be used to presents blocks of text in fixed-width font, and so is suitable for text that has been formatted on screen. As such, it is similar to the <PRE> and <LISTING> elements, but has a different syntax.

Typically, it will render as fixed width font with white space separating it from other text. It should be rendered such that 80 characters fit on the line.

e.g. `The <XMP>Netscape Navigator</XMP> doesn't support coloured tables.`

Would render as:

The

```
Netscape Navigator
```

doesn't support coloured tables.

NOTE : The **Internet Explorer** will translate any special characters included within <XMP> elements. I.e. if characters such as &lt;, &gt; etc. are used, they will be translated to < and >. Netscape and Mosaic treat the text contained within the elements literally.

<XMP> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<A...> ... </A> Anchor

Attributes

The Anchor element is the essence of HTML. It is marked text that is the start and/or destination of a hypertext link. Anchor elements are defined by the `<A>` element. The `<A>` element accepts several attributes, but either the `NAME` or `HREF` attribute is required.

Attributes of the `<A>` element :

HREF

If the `HREF` attribute is present, the text between the opening and closing anchor elements becomes hypertext. If this hypertext is selected by readers, they are moved to the document specified by the value of the attribute.

Example :

```
See <A HREF="http://subnet.virtual-pc.com/~le387818/">The HTMLib site</A>for
  updated info on the HTMLib.
```

In this example, selecting "The HTMLib" takes the reader to a document located at http://subnet.virtual-pc.com/~le387818/ (which is the HTMLib web site)

With the `HREF` attribute, the form `HREF="#identifier"` can refer to another anchor in the same document.

Example :

```
The <A HREF="document.html#glossary">glossary</A> defines terms used in the
  document.
```

In this example, selecting "glossary" takes the reader to another anchor (i.e. `<A NAME="glossary">Glossary</A>`) in the document document.html. The `NAME` attribute is described below. If the anchor is in another document and the `HREF` attribute provides a partial URL, the new document will be resolved from either the present document, or from any specified base address.

Several other forms of the `HREF` attribute are permitted by browsers. They are as follows :

`<A HREF="http://...">`	Makes a link to another document located on a World Wide Web server.
`<A HREF="ftp://...">`	Makes a link to an ftp site. Within an HTML document, normally a connection to an anonymous ftp site would be made. NCSA **Mosaic**, allows connection to private ftp sites. The Anchor would take the form ftp://jdoe@your.com. Mosaic would then prompt the user for a password.
`<A HREF="gopher://...>`	Makes a link to a gopher server.
`<A HREF="mailto:...">`	Activating such a link would bring up the browsers mailing dialog (or registered mail program) allowing

the user to send mail messages to the author of the document, or whoevers address follows the mailto attribute. **NCSA Mosaic** supports use of the **TITLE** attribute for a `mailto` link. It allows the author to specify the subject of the mail message that will be sent. **Netscape** allows specification of the subject line by using the following syntax:

```
<A HREF="mailto:cmlehunt@swan.ac.uk?subject=The HTMLib is fantastic">link text</A>
```

`<A HREF="news:...">`	Makes a link to a newsgroup. Care should be taken in using such links because the author can not know what newsgroups are carried by the local news server of the user.
`<A HREF="newsrc:...">`	Makes a link to a specific *newsrc* file.
`<A HREF="nntp://...">`	Can be used to specify a different news server to that which the user may normally use.
`<A HREF="telnet://...">`	Activating such a link would initiate a telnet session (using an external application) to the machine specified after the telnet:// label.
`<A HREF="wais://...">`	Makes a link that connects to a specified WAIS index server.

NAME

If present, the `NAME` attribute allows the anchor to be the target of a link. The value of the `NAME` attribute is an identifier for the anchor. Identifiers are arbitrary strings but must be unique within the HTML document.

Example of use:

```
<A NAME=“coffee”>Coffee</A> is an exmple of...
An example of this is <A HREF=“#coffee”>coffee</A>.
```

Another document can then make a reference explicitly to this anchor by putting the identifier after the address, separated by a has sign :

```
<A NAME=“drinks.html#coffee”>
```

TITLE

The Title attribute is informational only (unless used with a `mailto:` attribute). If present, the Title attribute should provide the title of the document whose address is given by the `HREF` attribute.

The Title attribute is useful for at least two reasons. The HTML user agent may display the title of the document prior to retrieving it, for example, as a margin note or on a small box while the mouse is over the anchor, or while the document is being loaded. Another reason is that documents that are not marked up text, such as graphics, plain text, FTP and Gopher menus, do not have titles. The `TITLE` attribute can be used to provide a title to such documents. When using the TITLE attribute, the title should be valid and unique for the destination document.

REL

The REL attribute gives the relationship(s) described by the hypertext link from the anchor to the target. The value is a comma-separated list of relationship values. Values and their semantics will be registered by the HTML registration authority. The default relationship if none other is given is void. The REL attribute is only used when the HREF attribute is present.

REV

The REV attribute is the same as the REL attribute, but the semantics of the link type are in the reverse direction. A link from A to B with REL="X" expresses the same relationship as a link from B to A with REV="X". An anchor may have both REL and REV attributes.

URN

If present, the URN attribute specifies a uniform resource name (URN) for a target document. The format of URNs is under discussion (1994) by various working groups of the Internet Engineering Task Force.

METHODS

The METHODS attributes of anchors and links provide information about the functions that the user may perform on an object. These are more accurately given by the HTTP protocol when it is used, but it may, for similar reasons as for the TITLE attribute, be useful to include the information in advance in the link. For example, the HTML user agent may chose a different rendering as a function of the methods allowed; for example, something that is searchable may get a different link appearance.

The value of the METHODS attribute is a comma separated list of HTTP methods supported by the object for public use.

TARGET

Browser windows can now have names associated with them. Links in any window can refer to another window by name. When the link is activated, the document referenced will appear in that named window. If the window is not already open, the browser will open and name a new window for you. Such an action is only supported by **Netscape** and **Internet Explorer**.

The syntax for the targeted windows is:

```
<A HREF="url.html" TARGET="window_name">Link text</A>
```

NOTE : If the targetted document is part of a frameset, there are various reserved names for target window available for smooth window transition.

See Also, <BASE TARGET= …>

<A> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic. The <A> element style can also be controlled by using the Anchor pseudo-elements within a style sheet definition.

Within a style sheet definition, provided by the <STYLE> element, the pseudo-classes A:link, A:visited and A:active can be used to set the text style of links, visited links and active links respectively (just as the LINK, ALINK and VLINK attributes of the <BODY> element do for those browsers that do not support style sheets)

The following attributes are all allowed within the `<A ...>` element

- `HREF`
- `METHODS`
- `NAME`
- `REL`
- `REV`
- `TARGET` - For opening a ‘target’ window
- `TITLE`
- `URN`
- Mailto : `TITLE`
- Anchors and Style Sheets

List Elements

HTML supports several types of lists, all of which may be nested. If used they should be present in the body of a HTML document.

`<DIR> ... </DIR>` - Directory list
`<DL> ... </DL>` - Definition list.
`<MENU> ... </MENU>` - Menu list
`<OL> ... </OL>` - Ordered list
`<UL> ... </UL>` - Unordered list

<DL> ... </DL>

Attributes See Also

A definition list is a list of terms and corresponding definitions. Definition lists are typically formatted with the term flush-left and the definition, formatted paragraph style, indented after the term.

Example of use:

```
<DL>
<DT>Term<DD>This is the definition of the first term.
<DT>Term<DD>This is the definition of the second term.
</DL>
```

📷

If the <DT> term does not fit in the <DT> column (one third of the display area), it may be extended across the page with the <DD> section moved to the next line, or it may be wrapped onto successive lines of the left hand column.

The opening list element must be <DL> and must be immediately followed by the first term (<DT>).

The definition list type can take the **COMPACT** attribute, which suggests that a compact rendering be used, because the list items are small and/or the entire list is large.

Unless you provide the COMPACT attribute, the HTML user agent may leave white space between successive <DT>, <DD> pairs. The COMPACT attribute may also reduce the width of the left-hand (<DT>) column.

```
<DL COMPACT>
<DT>Term<DD>This is the first definition in compact format.
<DT>Term<DD>This is the second definition in compact format.
</DL>
```

📷

<DL>, <DT> and <DD> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to them. For more details of these attributes, see the Style Sheets topic.

Term

This is the definition of the first term.

Term

This is the definition of the second term.

Term This is the first definition in compact format.

Term This is the second definition in compact format.

The following are all allowed within the <DL> element.

- <DT> : Definition list Term
- <DD> : Definition list definition.
- COMPACT
- CLASS, ID andSTYLE

<DIR> : Directory list
<MENU> : Menu list
<OL> : Ordered list
<UL> : Unordered list

<DIR> ... </DIR>

See Also

A Directory List element is used to present a list of items containing up to 20 characters each. Items in a directory list may be arranged in columns, typically 24 characters wide.

A directory list must begin with the `<DIR>` element which is immediately followed by a `<LI>` (list item) element:

```
<DIR>
<LI>A-H<LI>I-M
<LI>M-R<LI>S-Z
</DIR>
```

`<DIR>` and `<LI>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to them. For more details of these attributes, see the Style Sheets topic.

- A-H
- I-M
- M-R
- S-Z

<DL> : Definition list
<MENU> : Menu list
<OL> : Ordered list
<UL> : Unordered list

<MENU> ... </MENU>

See Also

A menu list is a list of items with typically one line per item. The menu list style is more compact than the style of an unordered list.

A menu list must begin with a `<MENU>` element which is immediately followed by a `<LI>` (list item) element:

```
<MENU>
<LI>First item in the list.
<LI>Second item in the list.
<LI>Third item in the list.
</MENU>
```

`<MENU>` and `<LI>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to them.   For more details of these attributes, see the Style Sheets topic.

- First item in the list.
- Second item in the list.
- Third item in the list.

<DIR> : Directory list
<DL> : Definition list
<OL> : Ordered list
<UL> : Unordered list

<OL> ... </OL>

Attributes See Also

The Ordered List element is used to present a numbered list of items, sorted by sequence or order of importance.

An ordered list must begin with the `<OL>` element which is immediately followed by a `<LI>` (list item) element:

```
<OL>
<LI>Click the Web button to open the Open the URL window.
<LI>Enter the URL number in the text field of the Open URL window. The Web
  document you specified is displayed.
<LI>Click highlighted text to move from one link to another.
</OL>
```

The Ordered List element can take the `COMPACT` attribute, which suggests that a compact rendering be used.

The average ordered list counts 1, 2, 3, ... etc. The **`TYPE`** allows different count mark types

`(TYPE=A)` - capital letters. e.g. A, B, C ...
`(TYPE=a)` - small letters. e.g. a, b, c ...
`(TYPE=I)` - large roman numerals. e.g. I, II, III ...
`(TYPE=i)` - small roman numerals. e.g. i, ii, iii ...
`(TYPE=1)` - or the default numbers. e.g. 1, 2, 3 ...

For lists that need to start at values other than 1 the **`START`** attribute can be used. `START` is always specified in the default numbers, and will be converted based on `TYPE` before display. Thus `START=5` would display either an 'E', 'e', 'V', 'v', or '5' based on the `TYPE` attribute.

To give even more flexibility to lists, the **`TYPE`** attribute can be used to change the list type for the item, and all subsequent items. Also, the **`VALUE`** attribute can be used to change the count for the list item and subsequent items.

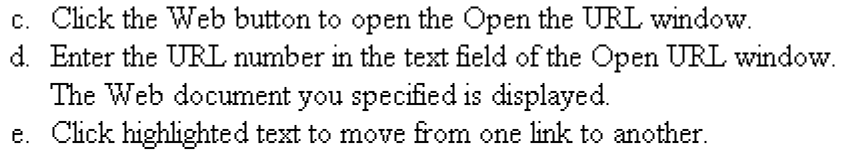
> ***NOTE :*** The `TYPE` attribute used in the `<OL>` Element and the `<LI>` Element and the `START` attribute in the `<OL>` Element are supported only by **Netscape** and the **Internet Explorer**.

`<OL>` and `<LI>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to them. For more details of these attributes, see the Style Sheets topic.

1. Click the Web button to open the Open the URL window.
2. Enter the URL number in the text field of the Open URL window. The Web document you specified is displayed.
3. Click highlighted text to move from one link to another.

<DIR> : Directory list
<DL> : Definition list
<MENU> : Menu list
<UL> : Unordered list

The following Ordered List, was rendered using `TYPE=a START=3`



c. Click the Web button to open the Open the URL window.
d. Enter the URL number in the text field of the Open URL window.
The Web document you specified is displayed.
e. Click highlighted text to move from one link to another.

The following attributes are allowed in the <OL> element.

- START
- TYPE
- VALUE
- CLASS, ID andSTYLE

<UL> ... </UL>

See Also

The Unordered List element is used to present a list of items which is typically separated by white space and/or marked by bullets.

An unordered list must begin with the `<UL>` element which is immediately followed by a `<LI>` (list item) element:

```
<UL>
<LI>First list item
<LI>Second list item
<LI>Third list item
</UL>
```

The Unordered List element can take the `COMPACT` attribute, which suggests that a compact rendering be used.

The basic bulleted list has a default progression of bullet types that change according to the level of the list item, from a solid disc, to a circle to a square. **Netscape** allows the **`TYPE`** attribute to be used within the `<UL>` element so that irrespective of the the indent level, the bullet type can be specified thus :

```
TYPE=disc
TYPE=circle
TYPE=square
```

To give even more flexibility to lists, **Netscape** allows the **`TYPE`** attribute to be used within the `<LI>` element as well. It takes the same values as `<UL>` and it changes the list type for that item, and all subsequent items.

NOTE : The `TYPE` attribute when used in the `<UL>` and `<LI>` Elements is **Netscape** specific.

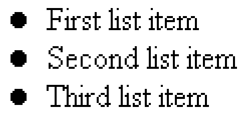
`<UL>` and `<LI>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to them. For more details of these attributes, see the Style Sheets topic.

- First list item
- Second list item
- Third list item

<DIR> : Directory list
<DL> : Definition list
<MENU> : Menu list
<OL> : Ordered list

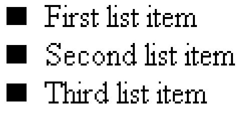
NOTE : The type's disc and circle appear the same when rendered.

`TYPE=disc/circle:`



- First list item
- Second list item
- Third list item

`TYPE=square:`



■ First list item
■ Second list item
■ Third list item

Information Type and Character Formatting Elements

Information type elements:

NOTE : Different information type elements may be rendered in the same way. The following are what are sometimes called **Logical** formatting elements. They suggest to the browser that the enclosed text should be rendered in a way according to the browser, rather than fixing the display type. Elements that do this, are character formatting elements (see below, also known as **Physical** elements) that produce strict rendering of the text.

`<CITE> ... </CITE>` - Citation
`<CODE> ... </CODE>` - An example of Code
`<EM> ... </EM>` - Emphasis
`<KBD> ... </KBD>` - User typed text
`<SAMP> ... </SAMP>` - A sequence of literal characters
`<STRONG> ... </STRONG>` - Strong typographic emphasis
`<VAR> ... </VAR>` - Indicates a variable name
`<!-- ... -->` - Defining comments.

Character formatting elements
`<B> ... </B>` - Boldface type
`<BIG> ... </BIG>` - Big text
`<BLINK> ... </BLINK>` - Blinking text
`<I> ... </I>` - Italics
`<SMALL> ... </SMALL>` - Small text
`<STRIKE> ... </STRIKE>` - Text that has been struckthrough
`<SUB> ... </SUB>` - Subscript
`<SUP> ... </SUP>` - Superscript
`<TT> ... </TT>` - TypeType (or Teletype)
`<U> ... </U>` - Underlined text

<CITE> ... </CITE>

See Also

The Citation element specifies a citation; typically rendered as italics.

```
This sentence, containing a <CITE>citation reference</CITE>
```

would look like:

This sentence, containing a *citation reference*

`<CITE>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<CODE> ... </CODE>

See Also

The Code element indicates an example of code; typically rendered as monospaced . Do not confuse with the Preformatted Text element.

```
This sentence contains an <CODE>example of code</CODE>.
```

It would look like :

This sentence contains an `example of code`

`<CODE>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

<EM> ... </EM>

See Also

The Emphasis element indicates typographic emphasis, typically rendered as italics.

```
The <EM>Emphasis</EM> element typically renders as Italics.
```

would render :

The *Emphasis* element typically render as Italics.

`<EM>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<KBD> ... </KBD>

See Also

The Keyboard element indicates text typed by a user; typically rendered as monospaced . It might commonly be used in an instruction manual.

```
The text inside the <KBD>KBD element, would typically</KBD> render as monospaced
  font.
```

would render as :

The text inside the `KBD element would typically` render as monospaced font.

`<KBD>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

<SAMP> ... </SAMP>

See Also

The Sample element indicates a sequence of literal characters; typically rendered as monospaced.

```
A sequence of <SAMP>literal characters</SAMP> commonly renders as monospaced font.
```

would render as :

A sequence of `literal characters` commonly renders as monospaced font.

`<SAMP>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<STRONG> ... </STRONG>

See Also

The Strong element indicates strong typographic emphasis, typically rendered in bold.

```
The instructions <STRONG>must be read</STRONG> before continuing.
```

would be rendered as :

The instructions **must be read** before continuing.

`<STRONG>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<VAR> ... </VAR>

See Also

The Variable element indicates a variable name; typically rendered as italic.

```
When coding, <VAR>LeftIndent()</VAR> must be a variable
```

would render as :

When coding *LeftIndent()* must be a variable.

`<VAR>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<!-- Comments -->

See Also

To include comments in an HTML document that will be ignored by the HTML user agent, surround them with <!-- and -->. After the comment delimiter, all text up to the next occurrence of --> is ignored. Hence comments cannot be nested. White space is allowed between the closing -- and >, but not between the opening <! and --.

For example:

```
<HEAD>
<TITLE>The HTML Reference Library</TITLE>
<!-- Prepared by Stephen Le Hunte, 1995, 1996 -->
</HEAD>
```

Information Type and Character formatting elements.

```
<COMMENT>
```

<B> ... </B>

See Also

The Bold element specifies that the text should be rendered in boldface, where available. Otherwise, alternative mapping is allowed.

```
The instructions <B>must be read</B> before continuing.
```

would be rendered as :

The instructions **must be read** before continuing.

`<B>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<I> ... </I>

See Also

The Italic element specifies that the text should be rendered in italic font, where available. Otherwise, alternative mapping is allowed.

```
Anything between the <I>I elements</I> should be italics.
```

would render as :

Anything between the *I elements* should be italics.

`<I>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<TT> ... </TT>

See Also

The Teletype element specifies that the text should be rendered in fixed-width typewriter font.

```
Text between the <TT> typetype elements</TT> should be rendered in fixed width
  typewriter font.
```

would render as :

Text between the `typetype elements` should be rendered in fixed width typewriter font.

`<TT>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<U> ... </U>

See Also

The `<U> ... </U>` Elements state that the enclosed text should be rendered, if practical, underlined.   This is an HTML 3.0 element and may not be widely supported.

```
The <U>main point</U> of the exercise...
```

would be rendered as :

The main point of the exercise...

`<U>` can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

<STRIKE> ... </STRIKE>

See Also

The `<STRIKE> ... </STRIKE>` element states that the enclosed text should be displayed with a horizontal line striking through the text. Alternative mappings are allowed if this is not practical.

NOTE : The actual element detailed in HTML specifications, is `<S> ... </S>`, which is also supported by the three browsers.

```
This text would be <STRIKE>struck through</STRIKE>
```

would be rendered as :

This text would be ~~struck through~~

`<STRIKE>` can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<SUB> ... </SUB>

See Also

The <SUB> element specifies that the enclosed text should be displayed as a subscript, and if practical, using a smaller font (compared with normal text).

```
This is the main text, with <SUB>this bit</SUB> being subscript.
```

📷

NOTE : The selected text will be made a superscript to the main text, formatting the selected text slightly smaller than the normal text. Browsers can be forced to make subscripts even smaller by compounding the <SUB> ... </SUB> element with the <SMALL> ... </SMALL> element, or be forced to render the subscript the same size as the normal text, by compounding the <SUB> ... </SUB> element with the <BIG> ... </BIG> element.
The exact appearance of the subscript text will change depending on any <FONT SIZE=...> and <BASEFONT SIZE=...> settings, if specified.

<SUB> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

This is the main text, with $_{\text{this bit}}$ being subscript

<SUP> ... </SUP>

See Also

The <SUP> element specifies that the enclosed text should be displayed as a superscript, and if practical, using a smaller font (compared with normal text).

```
This is the main text, with <SUP>this bit</SUP> being superscript.
```

NOTE : The selected text will be made a superscript to the main text, formatting the selected text slightly smaller than the normal text.   Browsers can be forced to make superscripts even smaller by compounding the <SUP> ... </SUP> element with the <SMALL> ... </SMALL> element, or be forced to render the superscript the same size as the normal text, by compounding the <SUP> ... </SUP> element with the <BIG> ... </BIG> element.
The exact appearance of the superscript text will change depending on any <FONT SIZE=...> and <BASEFONT SIZE=...> settings, if specified.

<SUP> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it.   For more details of these attributes, see the Style Sheets topic.

This is the main text, with $^{\text{this bit}}$ being superscript

<BIG> ... </BIG>

See Also

The <BIG> element specifies that the enclosed text should be displayed, if practical, using a big font (compared with the current font).

```
This is normal text, with <BIG>this bit</BIG> being big text.
```

would be rendered as:

This is normal text, with **this bit** being big text.

NOTE : Use of this element is currently supported by **Netscape** and the **Internet Explorer** only.
They also allow the <BIG> ... </BIG> element to be used surrounding the <SUB> ... </SUB> and <SUP> ... </SUP> elements to force rendering of the sub/superscript text as normal size text as opposed to the default slightly smaller text normally used.
The exact appearance of the big text will change depending on any <FONT SIZE=...> and <BASEFONT SIZE=...> settings, if specified.

<BIG> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<SMALL> ... </SMALL>

See Also

The <SMALL> element specifies that the enclosed text should be displayed, if practical, using a small font (compared with the current font).

```
This is normal text, with <SMALL>this bit</SMALL> being small text.
```

would be rendered as:

This is normal text, with this bit being small text.

NOTE : Use of this element is currently supported by **Netscape** and the **Internet Explorer** only.
They also allow the <SMALL> ... </SMALL> element to be used surrounding the <SUB> ... </SUB> and <SUP> ... </SUP> elements to force rendering of the sub/superscript text as text even smaller than the default slightly smaller (compared to the normal) text normally used.
The exact appearance of the small text will change depending on any <FONT SIZE=...> and <BASEFONT SIZE=...> settings, if specified.

<SMALL> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic.

<BLINK>

See Also

Surrounding any text with this element will cause the selected text to *blink* on the viewing page. This can serve to add extra emphasis to selected text.

```
<BLINK>This text would blink on the page</BLINK>
```

NOTE : The `<BLINK> ... </BLINK>` element is currently only supported by **Netscape**.

Information Type and Character formatting elements.

In line Images

Recently, new attributes have been added to the <IMG> to allow Client side Image Maps, embedded In-line video clips and also embedded In-line VRML worlds.

A note about formats.

Netscape and **Mosaic** (and most other browsers) will only support use of .GIF and .JPG images within HTML documents. This can be extended with **Netscape**, by embedding image formats within pages, providing the format is one that the user will have software to handle installed on their system (or have a plug-in that supports the format). Also, **Netscape** natively supports (i.e. the browser can display) progressive JPEG images.

The **Internet Explorer** though, will allow the use .GIF, .JPG, progressive JPEG images, .PNG (portable network graphics) images and also .BMP files, giving the author a wider variety of image formats from which to choose.

Netscape and **Internet Explorer** now fully support the GIF89a format, which means that multi-image GIF files can be used to create animation sequences. Users are encouraged to seek out the GIF Construction Kit for more details and tools for the preparation of multi-image animated GIF files.

<IMG...> In-line images

Attributes See Also

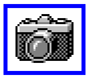
The Image element is used to incorporate in-line graphics into an HTML document. This element cannot be used for embedding other HTML text.

The Image element, which is empty (no closing element), has these attributes:

ALIGN

```
<IMG ALIGN= left|right|top|texttop|middle|
absmiddle|baseline|bottom|absbottom>
```

`ALIGN=left` image will float the image to the left margin (into the next available space there), and subsequent text will wrap around the right hand side of that image.
`ALIGN=right` will align the image aligns with the right margin, and the text wraps around the left.



`ALIGN=top` aligns itself with the top of the tallest item in the line.
`ALIGN=texttop` aligns itself with the top of the tallest text in the line (this is usually but not always the same as `ALIGN=top`).
`ALIGN=middle` aligns the baseline of the current line with the middle of the image.
`ALIGN=absmiddle` aligns the middle of the current line with the middle of the image.
`ALIGN=baseline` aligns the bottom of the image with the baseline of the current line.
`ALIGN=bottom` aligns the bottom of the image with the baseline of the current line.
`ALIGN=absbottom` aligns the bottom of the image with the bottom of the current line.

ALT

Optional text as an alternative to the graphic for rendering in non-graphical environments. Alternate text should be provided for whenever the graphic is not rendered (i.e. if the user has image loading turned off). Alternate text is mandatory for Level 0 documents. **Internet Explorer** also uses any `ALT` text set as a ToolTip to be displayed when the users mouse pauses over the image. Example of use:

e.g. : `<IMG SRC="triangle.gif" ALT="Warning:"> Be sure to read these instructions.`

ISMAP

The `ISMAP` attribute identifies an image as an image map. Image maps are graphics in which certain regions are mapped to URLs. By clicking on different regions, different resources can be accessed from the same graphic. Example of use:

e.g. :
```
<A HREF="http://machine/htbin/imagemap/sample">
<IMG SRC="sample.gif" ISMAP></A>
```

NOTE : To be able to employ image maps in HTML documents, the HTTP server which will be controlling document access must have the correct cgi-bin software installed to control image map behaviour. i.e. the document must have access to an image map handling script which is pointed to your .map file defining the graphics' 'hot-spots'

A simpler form of image map, known as client-side image maps are also possible. Currently, this type of map is a proposed extension to HTML. For details, see Client Side Image Maps.

SRC

The value of the SRC attribute is the URL of the image to be embedded. Its syntax is the same as that of the HREF attribute of the <A> element. SRC is mandatory.

```
<IMG SRC ="triangle.gif">Be sure to read these instructions.
```

```
<IMG WIDTH=value HEIGHT=value>
```

If the WIDTH and HEIGHT attributes are used, the viewer of their document will not have to wait for the image to be loaded and its size calculated. The browser can determine the layout of the text around the image and display the text first. **Netscape** is the only browser that will scale the whole image if either the WIDTH or HEIGHT attributes are specified, maintaining the aspect ratio. If both are specified then the image is displayed accordingly.

```
<IMG BORDER=value>
```

This lets the document author control the thickness of the border around an image displayed. This can be set to 0 so that if the image is surrounded by <A> elements, the normal link border will not be shown.

```
<IMG VSPACE=value HSPACE=value>
```

The VSPACE attribute controls the vertical space above and below the image, while HSPACE controls the horizontal space to the left and right of the image. They allow setting of a 'margin' around the image which is kept as white space - useful to prevent text wrapping right up to floating images.

LOWSRC

Using the LOWSRC attribute, it is possible to use two images in the same space. The syntax is :

e.g. : `<IMG SRC="highres.gif" LOWSRC="lowres.jpg">`

Browsers that do not recognise the LOWSRC attribute cleanly ignore it and simply load the image called "highres.gif".

Netscape will load the image called "lowres.jpg" on its first layout pass through the document. Then, when the document and all of its images are fully loaded, will do a second pass through and load the image called "highres.gif" in place. This means that a very low-resolution version of an image loaded initially and if the user stays on the page after the initial layout phase, a higher-resolution version of the same image can "fade in" and replace it.

Both GIF (both normal and interlaced) and JPEG images can be freely interchanged using this method. Both images will be scaled according to any WIDTH/HEIGHT attribute values present in the <IMG ...> element.

If the images are of different sizes and a fixed height and width are not specified in the IMG element, the second image (the image specified by the SRC attribute) will be scaled to the dimensions of the first (LOWSRC) image.

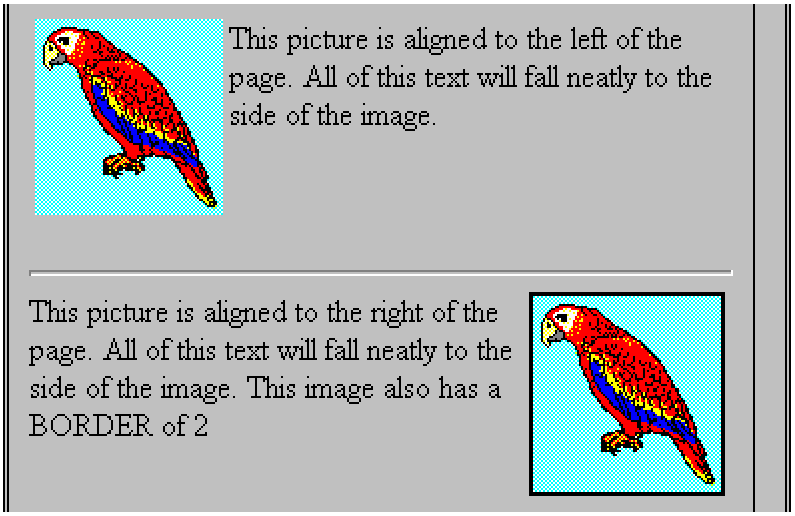
<IMG> can also take the **CLASS**, **ID** and **STYLE** attributes to allow style sheet definitions to be applied to it. For more details of these attributes, see the Style Sheets topic. The type of style sheet information that would generally be applied to the <IMG> element would be border styles, colours etc.

The following attributes are allowed within the <IMG ...> element

- ALIGN
- ALT
- BORDER
- HEIGHT
- HSPACE
- ISMAP
- *LOWSRC
- SRC
- VSPACE
- WIDTH

LOWSRC is a **Netscape** specific.

- CLASS, ID andSTYLE



This picture is aligned to the left of the page. All of this text will fall neatly to the side of the image.

This picture is aligned to the right of the page. All of this text will fall neatly to the side of the image. This image also has a BORDER of 2

In Line Images
Client side Image maps
In-line video
In Line VRML worlds

Client Side Image Maps

Attributes See Also

Adding a **USEMAP** attribute to an <IMG> element indicates that it is a client-side image map. The USEMAP attribute can be used with the ISMAP attribute to indicate that the image can be processed as either a client-side or server-side image map. The USEMAP attribute specifies which map to use with the image, in a format similar to the HREF attribute on anchors. If the argument to USEMAP starts with a '#', it is assumed to be in the same document as the IMG tag.

A few examples :

This method would only work if your browser supports client-side image maps:

```
<IMG SRC="../images/pic1.gif" USEMAP="maps.html#map1">
```

This image map will work regardless, because it specifies a server side map file as well as a client side image map file.

```
<A HREF="/cgi-bin/image map/pic2"><IMG SRC="../images/tech/pic2.gif"
  USEMAP="maps.html#map2" ISMAP></A>
```

Clicking here will take the user to a page with an error message if they don't have client-side image map support:

```
<A HREF="no_csim.html"><IMG SRC="../images/tech/pic3.gif"
  USEMAP="maps.html#map3"></A>
```

The different regions of the image are described using a **MAP** element. The map describes each region in the image and indicates where it links to. The basic format for the MAP element is as follows:

```
<MAP NAME="name">
<AREA [SHAPE="shape"] COORDS="x,y,..." [HREF="reference"] [NOHREF]>
</MAP>
```

The MAP definition can reside in the same file as the image that will use it, or in a completely separate file. This way, all map definitions can be kept separate from the main documents, allowing easier maintenance.

The **NAME** specifies the name of the map so that it can be referenced by an <IMG> element. The **SHAPE** gives the shape of this area. Currently the shapes "RECT", "CIRCLE" and "POLY" are supported (**Mosaic** only supports the RECT shape), with RECT being the default shape, if an explicit SHAPE attribute is not specified. The **COORDS** attribute gives the co-ordinates of the shape, using image pixels as the units. For a rectangle (SHAPE="RECT"), the COORDS are expressed as "left-x,top-y,right-x,bottom-y". For a circle, (SHAPE="CIRCLE"), the COORDS are expressed as "centre-x, centre-y, radius" and for a polygon (SHAPE="POLY") (an irregular shape), the COORDS are expressed in pairs of coordinates (i.e. "x1,y1,x2,y2,x3,y3... ") which defines the pixel coordinates of the various points of the polygonal image hotspot.

The **NOHREF** attribute indicates that clicks in this region should perform no action. An **HREF** tag specifies where a click in that area should lead. Note that a relative anchor specification will be expanded using the URL of the map description as a base, rather than using the URL of the document from which the map description is referenced (important if the map definition is in a file separate to the main document). If a BASE tag is present in the document containing the map description, that URL will be used as the base.

An arbitrary number of **AREA** tags may be specified. If two areas intersect, the one which appears first in the map definition takes precedence in the overlapping region. For example, a button bar in a document might use a 160 pixel by 60 pixel image and appear like this:

```
<MAP NAME="buttonbar">
<AREA SHAPE="RECT" COORDS="10,10,49,49" HREF="about_us.html">
<AREA SHAPE="RECT" COORDS="60,10,99,49" HREF="products.html">
<AREA SHAPE="RECT" COORDS="110,10,149,49" HREF="index.html">
<AREA SHAPE="RECT" COORDS="0,0,159,59" NOHREF>
</MAP>
<IMG SRC="../images/tech/bar.gif" USEMAP="#buttonbar">
```

Any region of the image that is not defined by an `AREA` tag is assumed to be `NOHREF`.

NOTE : The `TARGET` attribute can be used within the `<AREA>` element, allowing the use of Client side image maps within framed documents.

The following attributes are supported when using Client Side Image maps. They are all to be included within the bounds of a normal IMG Element

- USEMAP
- AREA
- COORDS
- MAP
- SHAPE

In Line Images
`<IMG ...>` Element
`ISMAP` attribute
In Line VRML worlds
In-line video

In line Video

Attributes See Also

Microsofts **Internet Explorer** allows the author to embed .AVI (Audio Video Interleave) video clips in HTML documents. This is done by adding several new attributes, notably DYNSRC (Dynamic Source) to the <IMG> element. Using the IMG element for this purpose makes it possible to add video clips to pages, but also have other browsers display still images in their place.

```
<IMG DYNSRC="video.avi" SRC="TheEarth.gif" WIDTH=50 HEIGHT=50 LOOP=INFINITE
  ALIGN=RIGHT>
```

DYNSRC

Specifies the address of a video clip to be displayed in the window. It stands for **Dyn**amic **S**ou**rc**e.

e.g. `<IMG SRC="foo.gif" DYNSRC="bar.avi">`
Internet Explorer will play the movie file "bar.avi"; other browsers will display the picture "foo.gif"

START

This attribute specifies when the video file should start playing. This attribute can be set to FILEOPEN or MOUSEOVER. FILEOPEN means that the video will start playing as soon as it has finished opening. This is the default setting. MOUSEOVER means that the video will start playing when the user moves the mouse cursor over the animation. Both can be specified together if necessary.

CONTROLS

This attribute specifies whether a set of controls is displayed under the clip window. This allows the user to pause/restart or skip sections of the video if desired. It is a flag and accepts no values.

LOOP

Specifies how many times a video clip will loop when activated. If n=-1, or if LOOP=INFINITE is specified, the video will loop indefinitely.

LOOPDELAY

Specifies, in milliseconds, how long a video clip will wait between play loops.

NOTE : As seen in the first example on this page, because the DYNSRC is an attribute of the IMG element, other attributes of the IMG element, such as HEIGHT and WIDTH will also apply to the in-line video, specifying the size of the video window. Video content can also be included using the ActiveMovie ActiveX control, using the <OBJECT> embedding mechanism.

The following attributes can be perceived as sub-attributes of the DYNSRC attribute, the main attribute of the IMG element that allows embedding of video clips.

- CONTROLS
- LOOP
- LOOPDELAY
- START

In Line Images
`<IMG>` Element
`<EMBED>` Used for OLE object embedding
In Line VRML worlds
Client Side Image Maps

The **V**irtual **R**eality **M**odelling **L**anguage (VRML) is a language for describing multi-participant interactive simulations - virtual worlds networked via the global **I**nternet and hyperlinked with the World Wide Web. All aspects of the virtual world display, interaction and internetworking can be specified using VRML.

In line VRML Worlds

See Also

Microsoft's **Internet Explorer** (from version 2) has added the ability to include in-line embedded VRML worlds by installing the **Virtual Explorer** plug-in module, available from the Microsoft Windows 95 Web site (http://www.microsoft.com/windows). (***NOTE :*** The Virtual Explorer module does not work with **Internet Explorer** 3.0, that requires the ActiveVRML add-in to be installed, available from the **Internet Explorer** web site at http://www.microsoft.com/ie/) It does this by adding the **`VRML`** attribute to the `<IMG>` element.

As the attribute is used in the `<IMG>` element, it supports many of the other attributes of the `<IMG>` element, such as **`HEIGHT`** and **`WIDTH`**.

As well as `.WRL` VRML virtual world files, using the Virtual Explorer, `.XAF` ActiveVRML files can be embedded into HTML documents. These allow for animated VRML objects that can be freely manipulated in 3-D space and may respond to user events, or change with time. As such, Microsoft's ActiveVRML specification represents a possible major enhancement to what may become VRML 2.0.

For example;

```
<IMG SRC="picture.gif" VRML="world.wrl" HEIGHT=250 WIDTH=300>
```

The above example, would embed the VRML world, "world.wrl" into the HTML document, with the navigation controls below the embedding pane. The pane is displayed according to the dimensions specified. For browsers, other than the **Virtual Explorer** (Internet Explorer with the VRML add-on), the picture "picture.gif" would be displayed.

> ***NOTE :*** Embedding of VRML worlds is also supported by Netscape, using the Live3D plug-in module and the `<EMBED>` element. ActiveVRML (and normal VRML worlds) files can be displayed in **Internet Explorer** 3.0 providing the user has the ActiveMovie ActiveX control installed.

ActiveVRML is at present a draft specification, available from Microsoft at http://www.microsoft.com/intdev/avr, or from the Internet Explorer web site, http://www.microsoft.com/ie/ The specification details additions to the VRML language that allows the creation of interactive animations. Using ActiveVRML, the author can create animations by describing how animation parameters vary continuously with time, user input and other parameters, instead of creating conventional frames to make an animation. That is, the author describes events influencing objects, which results in other events happening, leading to animation.

Forms

The following elements are used to create forms :

<FORM> ... </FORM> - A form within a document
<INPUT ...> ... </INPUT> - One input field
<OPTION> - One option within a Select element.
<SELECT> ... <SELECT> - A selection from a finite set of options
<TEXTAREA ...> ... </TEXTAREA> - A multi-line input field

Each variable field is defined by an INPUT, TEXTAREA, or OPTION element and must have a NAME attribute to identify its value in the data returned when the form is submitted.

Example of use :

```
<H1 ALIGN="center">Comment Form</H1>
<FORM METHOD="POST" ACTION="http://www.htmlib.com/formscript.cgi">
<CENTER>
Your name: <INPUT NAME="name" size="20">
Your e-mail address: <INPUT NAME="email" size="20">
<P>I think the HTML Reference is :
  <SELECT NAME="Choice">
    <OPTION>Outstanding
    <OPTION>Very good
    <OPTION>Good
    <OPTION>Average
    <OPTION>Below Average
    <OPTION>Awful
    <OPTION SELECTED>My response would be "indecent" under the CDA Act.
  </SELECT>
<P>If you have any further comments, please enter them here:<BR>
  <TEXTAREA NAME="Comments" ROWS="10" COLS="40" WRAP="Virtual">
  </TEXTAREA>
<P><INPUT TYPE=SUBMIT> <INPUT TYPE=RESET>
</CENTER>
</FORM>
```

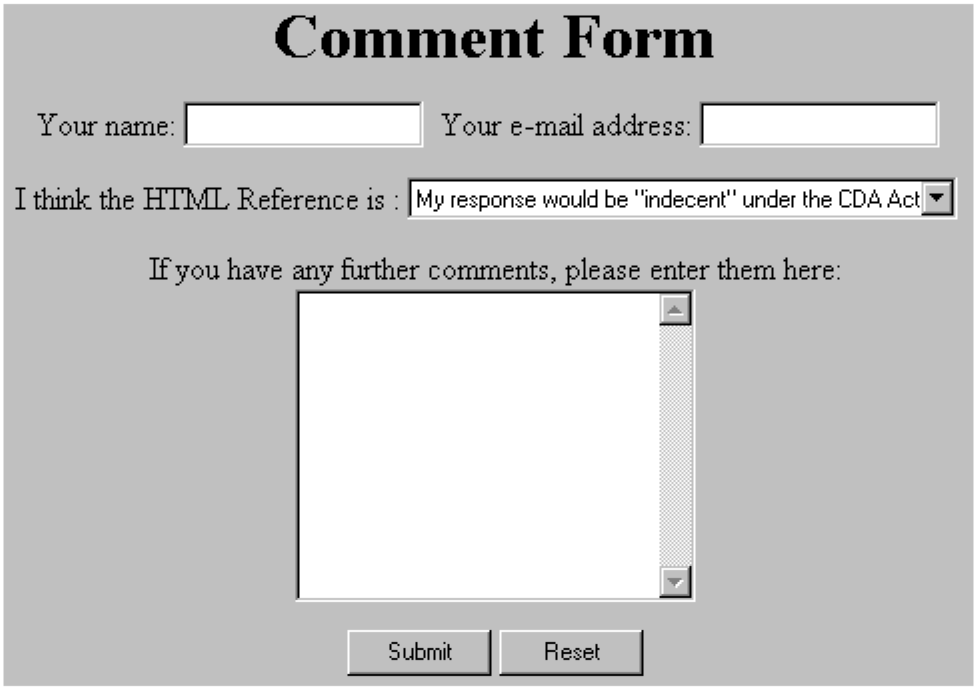
HTTP File Upload: It is possible to write forms that ask for files as input, rather than input boxes and other simple elements such as checkboxes and radio buttons.

An example of such a form would be:

```
<FORM ENCTYPE="multipart/form-data" ACTION="_URL_" METHOD=POST>
Send this file: <INPUT NAME="userfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

NOTE : This method of file upload is **Netscape specific** and is essentially adoption of another IETF Internet Draft by the Netscape authors. The Internet Draft in question, "Form based file upload in HTML", suggested adding a FILE option to the TYPE attribute of the INPUT element, allowing an ACCEPT attribute for the INPUT element (which is a list of media types or type patterns allowed for the input) and allowing the ENCTYPE of a from to be “multipart/form-data”.

Such additions to the <FORM> element could prove invaluable for example, for companies providing technical support, or service providers, requesting data files.



Comment Form

Your name:

Your e-mail address:

I think the HTML Reference is : My response would be "indecent" under the CDA Act

If you have any further comments, please enter them here:

<FORM> ... </FORM>

Attributes See Also

The Form element is used to delimit a data input form. There can be several forms in a single document, but the Form element can't be nested.

The **ACTION** attribute is a URL specifying the location to which the contents of the form are submitted to elicit a response, typically this will be a form handling CGI script. If the ACTION attribute is missing, the URL of the document itself is assumed. The way data is submitted varies with the access protocol of the URL, and with the values of the **METHOD** and **ENCTYPE** attributes.

In general:

- the METHOD attribute selects variations in the protocol.

- the ENCTYPE attribute specifies the format of the submitted data in case the protocol does not impose a format itself.

The Level 2 specification defines and requires support for the HTTP access protocol only.

When the ACTION attribute is set to an HTTP URL, the METHOD attribute must be set to an HTTP method as defined by the HTTP method specification in the IETF draft HTTP standard. The default METHOD is GET, although for many applications, the POST method may be preferred. With the post method, the ENCTYPE attribute is a MIME type specifying the format of the posted data; by default, is application/x-www-form-urlencoded.

Under any protocol, the submitted contents of the form logically consist of name/value pairs. The names are usually equal to the NAME attributes of the various interactive elements in the form.

> ***NOTE:*** The names are not guaranteed to be unique keys, nor are the names of form elements required to be distinct. The values encode the user's input to the corresponding interactive elements. Elements capable of displaying a textual or numerical value will return a name/value pair even when they receive no explicit user input.

Layout information can be applied to the <FORM> element through the use of Style Sheets. See that topic for more information.

The following are all attributes of the `<FORM>` element.

```
      ACTION
      ENCTYPE
      METHOD
```

All are detailed on this page.

Other <FORM> elements :
<INPUT> Element
<OPTION> Element
<SELECT> Element
<TEXTAREA> Element

Forms - <INPUT>

Attributes See Also

The Input element represents a field whose contents may be edited by the user.

Attributes of the Input element:

ALIGN

Vertical alignment of the image. For use only with `TYPE=IMAGE` in HTML level 2. The possible values are exactly the same as for the `ALIGN` attribute of the image element.

CHECKED

Indicates that a checkbox or radio button is selected. Unselected checkboxes and radio buttons do not return name/value pairs when the form is submitted.

MAXLENGTH

Indicates the maximum number of characters that can be entered into a text field. This can be greater than specified by the `SIZE` attribute, in which case the field will scroll appropriately. The default number of characters is unlimited.

NAME

Symbolic name used when transferring the form's contents. The `NAME` attribute is required for most input types and is normally used to provide a unique identifier for a field, or for a logically related group of fields.

SIZE

Specifies the size or precision of the field according to its type. For example, to specify a field with a visible width of 24 characters:

e.g. : `INPUT TYPE=text SIZE="24"`

SRC

A URL or URN specifying an image. For use only with `TYPE=IMAGE` in HTML Level 2.

TYPE

Defines the type of data the field accepts. Defaults to free text. Several types of fields can be defined with the type attribute:

CHECKBOX : Used for simple Boolean attributes, or for attributes that can take multiple values at the same time. The latter is represented by a number of checkbox fields each of which has the same name. Each selected checkbox generates a separate name/value pair in the submitted data, even if this results in duplicate names. The default value for checkboxes is "on".

HIDDEN : No field is presented to the user, but the content of the field is sent with the submitted form. This value may be used to transmit state information about client/server interaction.

IMAGE : An image field upon which you can click with a pointing device, causing the form to be immediately submitted. The co-ordinates of the selected point are measured in pixel units from the upper-left corner of the image, and are returned (along with the other contents of the form) in two name/value pairs. The x-co-ordinate is submitted under the name of the field with .x appended, and the y- co-ordinate is submitted under the name of the field with .y appended. Any `VALUE` attribute is ignored. The image itself is specified by the `SRC` attribute, exactly as for the Image element.

PASSWORD is the same as the `TEXT` attribute, except that text is not displayed as it is entered.

RADIO is used for attributes that accept a single value from a set of alternatives. Each radio button field in the group should be given the same name. Only the selected radio button in the group generates a name/value pair in the submitted data. Radio buttons require an explicit VALUE attribute.

RESET is a button that when pressed resets the form's fields to their specified initial values. The label to be displayed on the button may be specified just as for the SUBMIT button.

SUBMIT is a button that when pressed submits the form.   You can use the VALUE attribute to provide a non- editable label to be displayed on the button.   The default label is application-specific. If a SUBMIT button is pressed in order to submit the form, and that button has a NAME attribute specified, then that button contributes a name/value pair to the submitted data. Otherwise, a SUBMIT button makes no contribution to the   submitted data.

TEXT is used for a single line text entry fields. Use in conjunction with the SIZE and MAXLENGTH attributes. Use the Textarea element for text fields which can accept multiple lines.

TEXTAREA is used for multiple-line text-entry fields. Use in conjunction with the SIZE and MAXLENGTH attributes.

FILE   **Netscape** now supports a FILE option to the TYPE attribute of the INPUT element, allowing an ACCEPT attribute for the INPUT element (which is a list of media types or type patterns allowed for the input) and allowing the ENCTYPE of a from to be “multipart/form-data”.
This allows the inclusion of files with form information, which could prove invaluable for example, for companies providing technical support, or service providers, requesting data files.

VALUE

The initial displayed value of the field, if it displays a textual or numerical value; or the value to be returned when the field is selected, if it displays a Boolean value. This attribute is required for radio buttons.

The following are all allowed within the `<INPUT>` element.

```
ALIGN
CHECKED
MAXLENGTH
NAME
SIZE
SRC
TYPE :
   CHECKBOX; HIDDEN; IMAGE;
   PASSWORD; RADIO; RESET;
   SUBMIT; TEXT; TEXTAREA; FILE
VALUE
```

Other <FORM> elements :

<FORM> Element

<OPTION> Element

<SELECT> Element

<TEXTAREA> Element

Forms - <OPTION>

Attributes See Also

The Option element can only occur within a Select element. It represents one choice, and can take these attributes:

DISABLED
Proposed.

SELECTED
Indicates that this option is initially selected.

VALUE
When present indicates the value to be returned if this option is chosen. The returned value defaults to the contents of the Option element.

The contents of the Option element is presented to the user to represent the option. It is used as a returned value if the `VALUE` attribute is not present.

The following are all allowed within the `<OPTION>` element

```
DISABLED
SELECTED
VALUE
```

All are detailed on this page .

Other `<FORM>` elements :

`<FORM>` Element
`<INPUT>` Element
`<SELECT>` Element
`<TEXTAREA>` Element

Forms - <SELECT ...> ... </SELECT>

Attributes See Also

The Select element allows the user to chose one of a set of alternatives described by textual labels. Every alternative is represented by the Option element.

Attributes are:

ERROR
Proposed.

MULTIPLE
The `MULTIPLE` attribute is needed when users are allowed to make several selections, e.g. `<SELECT MULTIPLE>`.

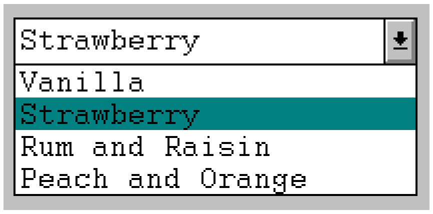
NAME
Specifies the name that will submitted as a name/value pair.

SIZE
Specifies the number of visible items. If this is greater than one, then the resulting form control will be a list.

The `SELECT` element is typically rendered as a pull down or pop-up list. For example:

```
<SELECT NAME="flavor">
<OPTION>Vanilla
<OPTION>Strawberry
<OPTION>Rum and Raisin
<OPTION>Peach and Orange
</SELECT>
```

If no option is initially marked as selected, then the first item listed is selected.



Strawberry
Vanilla
Strawberry
Rum and Raisin
Peach and Orange

The following are al allowed within the `<SELECT>` element

```
ERROR
MULTIPLE
NAME
SIZE
```

All are detailed on this page.

Other <FORM> elements :

<FORM> Element

<INPUT> Element

<OPTION> Element

<TEXTAREA> Element

Forms - <TEXTAREA> ... </TEXTAREA>

Attributes See Also

The TEXTAREA element lets users enter more than one line of text. For example:

e.g. :
```
<TEXTAREA NAME="address" ROWS=64 COLS=6>
HaL Computer Systems
1315 Dell Avenue
Campbell, California 95008
</TEXTAREA>
```

The text up to the end element (</TEXTAREA>) is used to initialise the field's value. This end element is always required even if the field is initially blank. When submitting a form, lines in a TEXTAREA should be terminated using CR/LF.

In a typical rendering, the **ROWS** and **COLS** attributes determine the visible dimension of the field in characters. The field is rendered in a fixed-width font. HTML user agents should allow text to extend beyond these limits by scrolling as needed.

> ***NOTE:*** In the initial design for forms, multi-line text fields were supported by the Input element with TYPE=TEXT. Unfortunately, this causes problems for fields with long text values. SGML's default (Reference Quantity Set) limits the length of attribute literals to only 240 characters. The HTML 2.0 SGML declaration increases the limit to 1024 characters.

Recent versions of the Netscape Navigator (from version 2.0) have introduced the **WRAP** attribute in the TEXTAREA element: Now it is possible to specify how to handle word-wrapping in text input areas in forms.

<TEXTAREA WRAP=OFF> -- the default setting - Wrapping doesn't happen. Lines are sent exactly as typed.

<TEXTAREA WRAP=VIRTUAL> -- The display word-wraps, but long lines are sent as one line without new-lines.

<TEXTAREA WRAP=PHYSICAL> -- The display word-wraps, and the text is transmitted at all wrap points.

NOTE : Word wrapping in a TEXTAREA text box is **Netscape specific.**

The following are all allowed within the <TEXTAREA> element

- COLS
- NAME
- ROWS
- WRAP

All are detailed on this page.

Other <FORM> elements :

<FORM> Element
<INPUT> Element
<OPTION> Element
<SELECT> Element

Character Data

The characters between HTML elements represent text. A HTML document (including elements and text) is encoded using the coded character set specified by the "charset" parameter of the "text/html" media type. For levels defined in this specification, the "charset" parameter is restricted to "US-ASCII" or "ISO-8859-1". ISO-8859-1 encodes a set of characters known as Latin Alphabet No. 1, or simply Latin-1. Latin-1 includes characters from most Western European languages, as well as a number of control characters. Latin-1 also includes a non-breaking space, a soft hyphen indicator, 93 graphical characters, 8 unassigned characters, and 25 control characters.

Because non-breaking space and soft hyphen indicator are not recognised and interpreted by all HTML user agents, their use is discouraged.

There are 58 character positions occupied by control characters. See Control Characters for details on the interpretation of control characters.

Because certain special characters are subject to interpretation and special processing, information providers and HTML user agent implementers should follow the guidelines in the Special Characters section.

In addition, HTML provides character entity references and numerical character references to facilitate the entry and interpretation of characters by name and by numerical position.

Because certain characters will be interpreted as mark-up, they must be represented by entity references as described in character and/or numerical references.

Special Characters

See Also

Certain characters have special meaning in HTML documents. There are two printing characters which may be interpreted by an HTML application to have an effect of the format of the text:

Space

- Interpreted as a word space (place where a line can be broken) in all contexts except the Preformatted Text element.
- Interpreted as a nonbreaking space within the Preformatted Text element.

Hyphen

- Interpreted as a hyphen glyph in all contexts
- Interpreted as a potential word space by hyphenation engine

The following entity names are used in HTML, always prefixed by ampersand (&) and followed by a semicolon. They represent particular graphic characters which have special meanings in places in the mark-up, or may not be part of the character set available to the writer.

The following table lists each of the supported characters specified in the Numeric and Special Graphic entity set, along with its name, syntax for use, and description. This list is derived from ISO Standard 8879:1986//ENTITIES Numeric and Special Graphic//EN however HTML does not provide support for the entire entity set. Only the entities listed below are supported.

Glyph	**Name**	**Syntax**	**Description**
<	lt	&lt	Less than sign
>	gt	&gt	Greater than sign
&	amp	&amp	Ampersand
"	quot	&quot	Double quote sign

Control Characters

See Also

Control characters are non-printable characters that are typically used for communication and device control, as format effectors, and as information separators.

In SGML applications, the use of control characters is limited in order to maximise the chance of successful interchange over heterogeneous networks and operating systems. In HTML, only three control characters are used: Horizontal Tab (HT, encoded as 9 decimal in US-ASCII and ISO-8859-1), Carriage Return, and Line Feed.

Horizontal Tab is interpreted as a word space in all contexts except preformatted text. Within preformatted text, the tab should be interpreted to shift the horizontal column position to the next position which is a multiple of 8 on the same line; that is, col := ((col+8) div8) * 8 (where div is integer division).

Carriage Return and Line Feed are conventionally used to represent end of line. For Internet Media Types defined as "text/*", the sequence CR/LF is used to represent an end of line. In practice, text/html documents are frequently represented and transmitted using an end of line convention that depends on the conventions of the source of the document; frequently, that representation consists of CR only, LF only, or CR/LF combination. In HTML, end of line in any of its variations is interpreted as a word space in all contexts except preformatted text. Within preformatted text, HTML interpreting agents should expect to treat any of the three common representations of end-of-line as starting a new line.

Numeric Character References

See Also

In addition to any mechanism by which characters may be represented by the encoding of the HTML document, it is possible to explicitly reference the printing characters of the ISO-8859-1 character encoding using a numeric character reference.

Two reasons for using a numeric character reference:

- the keyboard does not provide a key for the character, such as on U.S. keyboards which do not provide European characters

- the character may be interpreted as SGML coding, such as the ampersand (&), double quotes ("), the lesser (<) and greater (>) characters

Numeric character references are represented in an HTML document as SGML entities whose name is number sign (#) followed by a numeral from 32-126 and 161-255. The HTML DTD includes a numeric character for each of the printing characters of the ISO-8859-1 encoding, so that one may reference them by number if it is inconvenient to enter them directly:

the ampersand (&#38;), double quotes (&#34;), lesser (&#60;) and greater (&#62;) characters

The following entity names are used in HTML, always prefixed by ampersand (&) and followed by a semicolon.   This list, sorted numerically, is derived from ISO-8859-1 8-bit single-byte coded graphic character set:

Reference	**Description**	**Description**
&#00- &#08	Unused	Unused
&#09	Horizontal tab	Horizontal tab
&#10	Line feed	Line feed
&#11 - &#31	Unused	Unused
&#32	Space	Space
&#33	!	Exclamation mark
&#34	"	Quotation mark
&#35	Number sign	Number sign
&#36	$	Dollar sign
&#37	%	Percent sign
&#38	&	Ampersand
&#39	'	Apostrophe
&#40	(	Left parenthesis
&#41	)	Right parenthesis
&#42	*	Asterisk
&#43	+	Plus sign
&#44	,	Comma
&#45	-	Hyphen
&#46	.	Period (fullstop)
&#47	/	Solidus (slash)
&#48- &#57	Digits 0-9	Digits 0-9
&#58	:	Colon
&#59	;	Semi-colon
&#60	<	Less than
&#61	=	Equals sign
&#62	>	Greater than

&#63	?	Question mark
&#64	@	Commercial at
&#91	[	Left square bracket
&#92	\	Reverse solidus (backslash)
&#93	]	Right square bracket
&#94	^	Caret
&#95	_	Horizontal bar
&#96	´	Acute accent
&#97- &#122	Letters a-z	Letters a-z
&#123	{	Left curly brace
&#124	|	Vertical bar
&#125	}	Right curly brace
&#126	~	Tilde
&#127 - &#160	Unused	Unused
&#161	¡	Inverted exclamation
&#162	¢	Cent sign
&#163	£	Pound sterling
&#164	General currency sign	General currency sign
&#165	¥	Yen sign
&#166	¦	Broken vertical bar
&#167	§	Section sign
&#168	¨	Umlaut (dieresis)
&#169	©	Copyright
&#170	ª	Feminine ordinal
&#171	«	Left angle quote, guillemot left
&#172	¬	Not sign
&#173		Soft hyphen
&#174	®	Registered trademark
&#175	¯	Macron accent
&#176	°	Degree sign
&#177	±	Plus or minus
&#178	²	Superscript two
&#179	³	Superscript three
&#180	´	Acute accent
&#181	µ	Micro sign
&#182	¶	Paragraph sign
&#183	·	Middle dot
&#184	¸	Cedilla
&#185	¹	Superscript one
&#186	º	Masculine ordinal
&#187	»	Right angle quote, guillemot right
&#188	¼	Fraction one-fourth
&#189	½	Fraction one-half
&#190	¾	Fraction three-fourths
&#191	¿	Inverted question mark
&#192	Á	Capital A, acute accent
&#193	À	Capital A, grave accent

&#194	Â	Capital A, circumflex accent
&#195	Ã	Capital A, tilde
&#196	Å	Capital A, dieresis or umlaut mark
&#197	Ä	Capital A, ring
&#198	Æ	Capital AE dipthong (ligature)
&#199	Ç	Capital C, cedilla
&#200	É	Capital E, acute accent
&#201	È	Capital E, grave accent
&#202	Ê	Capital E, circumflex accent
&#203	Ë	Capital E, dieresis or umlaut mark
&#204	Í	Capital I, acute accent
&#205	Ì	Capital I, grave accent
&#206	Î	Capital I, circumflex accent
&#207	Ï	Capital I, dieresis or umlaut mark
&#208	Đ	Capital Eth, Icelandic
&#209	Ñ	Capital N, tilde
&#210	Ó	Capital O, acute accent
&#211	Ò	Capital O, grave accent
&#212	Ô	Capital O, circumflex accent
&#213	Õ	Capital O, tilde
&#214	Ö	Capital O, dieresis or umlaut mark
&#215	*	Multiply sign
&#216	Ø	Capital O, slash
&#217	Ú	Capital U, acute accent
&#218	Ù	Capital U, grave accent
&#219	Û	Capital U, circumflex accent
&#220	Ü	Capital U, dieresis or umlaut mark
&#221	Ý	Capital Y, acute accent
&#222	Þ	Capital THORN, Icelandic
&#223	ß	Small sharp s, German (sz ligature)
&#224	á	Small a, acute accent
&#225	à	Small a, grave accent
&#226	â	Small a, circumflex accent
&#227	ã	Small a, tilde
&#228	å	Small a, dieresis or umlaut mark
&#229	ä	Small a, ring
&#230	æ	Small ae dipthong (ligature)
&#231	ç	Small c, cedilla

&#232	é	Small e, acute accent
&#233	è	Small e, grave accent
&#234	ê	Small e, circumflex accent
&#235	ë	Small e, dieresis or umlaut mark
&#236	í	Small i, acute accent
&#237	ì	Small i, grave accent
&#238	î	Small i, circumflex accent
&#239	ï	Small i, dieresis or umlaut mark
&#240	ð	Small eth, Icelandic
&#241	ñ	Small n, tilde
&#242	ó	Small o, acute accent
&#243	ò	Small o, grave accent
&#244	ô	Small o, circumflex accent
&#245	õ	Small o, tilde
&#246	ö	Small o, dieresis or umlaut mark
&#247	÷	Division sign
&#248	ø	Small o, slash
&#249	ú	Small u, acute accent
&#250	ù	Small u, grave accent
&#251	û	Small u, circumflex accent
&#252	ü	Small u, dieresis or umlaut mark
&#253	ý	Small y, acute accent
&#254	þ	Small thorn, Icelandic
&#255	ÿ	Small y, dieresis or umlaut mark

Character Data
Character Entity References

Character Entity References

See Also

Many of the Latin alphabet No. 1 set of printing characters may be represented within the text of an HTML document by a character entity.

Two reasons for using a character entity:

- the keyboard does not provide a key for the character, such as on U.S. keyboards which do not provide European characters

- the character may be interpreted as SGML coding, such as the ampersand (&), double quotes ("), the lesser (<) and greater (>) characters

A character entity is represented in an HTML document as an SGML entity whose name is defined in the HTML DTD. The HTML DTD includes a character entity for each of the SGML mark-up characters and for each of the printing characters in the upper half of Latin-1, so that one may reference them by name if it is inconvenient to enter them directly:

the ampersand (&amp;), double quotes (&quot;), lesser (&lt;) and greater (&gt;) characters

e.g. : `Kurt G&ouml;del was a famous logician and mathematician.`

NOTE: To ensure that a string of characters is not interpreted as mark-up, represent all occurrences of <, >, and & by character or entity references.

NOTE: There are SGML features, CDATA and RCDATA, to allow most <, >, and & characters to be entered without the use of entity or character references. Because these features tend to be used and implemented inconsistently, and because they require 8-bit characters to represent non-ASCII characters, they are not used in this version of the HTML DTD. An earlier HTML specification included an Example element whose syntax is not expressible in SGML. No mark-up was recognised inside of the Example element. While HTML user agents are encouraged to support this idiom, its use is deprecated.

The following entity names are used in HTML, always prefixed by ampersand (&) and followed by a semicolon.   The following table lists each of the characters specified in the Added Latin 1 entity set, along with its name, syntax for use, and description. This list is derived from ISO Standard 8879:1986//ENTITIES Added Latin 1//EN. HTML supports the entire entity set.

Name	Syntax	Look	Description
Aacute	&Aacute	Á	Capital A, acute accent
Agrave	&Agrave	À	Capital A, grave accent
Acirc	&Acirc	Â	Capital A, circumflex accent
Atilde	&Atilde	Ã	Capital A, tilde
Aring	&Aring	Å	Capital A, ring
Auml	&Auml	Ä	Capital A, dieresis or umlaut mark
AElig	&AElig	Æ	Capital AE dipthong

			(ligature)
Ccedil	&Ccedil	Ç	Capital C, cedilla
Eacute	&Eacute	É	Capital E, acute accent
Egrave	&Egrave	È	Capital E, grave accent
Ecirc	&Ecirc	Ê	Capital E, circumflex accent
Euml	&Euml	Ë	Capital E, dieresis or umlaut mark
Iacute	&Iacute	Í	Capital I, acute accent
Igrave	&Igrave	Ì	Capital I, grave accent
Icirc	&Icirc	Î	Capital I, circumflex accent
Iuml	&Iuml	Ï	Capital I, dieresis or umlaut mark
ETH	&ETH	Ð	Capital Eth, Icelandic
Ntilde	&Ntilde	Ñ	Capital N, tilde
Oacute	&Oacute	Ó	Capital O, acute accent
Ograve	&Ograve	Ò	Capital O, grave accent
Ocirc	&Ocirc	Ô	Capital O, circumflex accent
Otilde	&Otilde	Õ	Capital O, tilde
Ouml	&Ouml	Ö	Capital O, dieresis or umlaut mark
Oslash	&Oslash	Ø	Capital O, slash
Uacute	&Uacute	Ú	Capital U, acute accent
Ugrave	&Ugrave	Ù	Capital U, grave accent
Ucirc	&Ucirc	Û	Capital U, circumflex accent
Uuml	&Uuml	Ü	Capital U, dieresis or umlaut mark;
Yacute	&Yacute	Ý	Capital Y, acute accent
THORN	&THORN	Þ	Capital THORN, Icelandic
Szlig	&szlig	ß	Small sharp s, German (sz ligature)

aacute	&aacute	á	Small a, acute accent
agrave	&agrave	à	Small a, grave accent
acirc	&acirc	â	Small a, circumflex accent
atilde	&atilde	ã	Small a, tilde
aring	&aring	å	Small a, ring
auml	&auml	ä	Small a, dieresis or umlaut mark
aelig	&aelig	æ	Small ae dipthong (ligature
ccedil	&ccedil	ç	Small c, cedilla
eacute	&eacute	é	Small e, acute accent
egrave	&egrave	è	Small e, grave accent
ecirc	&ecirc	ê	Small e, circumflex accent
euml	&euml	ë	Small e, dieresis or umlaut mark
iacute	&iacute	í	Small i, acute accent
igrave	&igrave	ì	Small i, grave accent
icirc	&icirc	î	Small i, circumflex accent
iuml	&iuml	ï	Small i, dieresis or umlaut mark
eth	&eth	ð	Small eth, Icelandic
ntilde	&ntilde	ñ	Small n, tilde
oacute	&oacute	ó	Small o, acute accent
ograve	&ograve	ò	Small o, grave accent
ocirc	&ocirc	ô	Small o, circumflex accent
otilde	&otilde	õ	Small o, tilde
ouml	&ouml	ö	Small o, dieresis or umlaut mark
oslash	&oslash	ø	Small o, slash
uacute	&uacute	ú	Small u, acute accent
ugrave	&ugrave	ù	Small u, grave accent
ucirc	&ucirc	û	Small u, circumflex accent

uuml	&uuml	ü	Small u, dieresis or umlaut mark
yacute	&yacute	ý	Small y, acute accent
thorn	&thorn	þ	Small thorn, Icelandic
yuml	&yuml	ÿ	Small y, dieresis or umlaut mark
reg	&reg	®	Registered TradeMark
copy	&copy	©	Copyright
trade	&trade	™	TradeMark
nbsp	&nbsp	Non breaking space	Non breaking space

NOTE : The last four character entities are only supported in recent versions of **Mosaic** and **Netscape** (except for the &trade; code). They may not appear as planned in early versions of these, or different browsers.

Character Data
Numerical Character References

Character Data

Tables

At present, the table HTML elements are :

`<TABLE> ... </TABLE>` - The Table delimiter.
`<TR ...> ... </TR>` - Used to specify number of rows in a table.
`<TD ...> ... </TD>` - Specifies table data cells.
`<TH ...> ... </TH>` - Table Header cell.
`<CAPTION ...> ... </CAPTION>` - Specifies the table Caption.

Internet Explorer has introduced support for various HTML 3.0 table elements. Those introduced are:
`<THEAD> ... </THEAD>` - specifies the Table head.
`<TBODY> ... </TBODY>` - specifies the Table body.
`<TFOOT> ... </TFOOT>` - specifies the Table footer
`<COLGROUP> ... </COLGROUP>` - used to group column alignments
`<COL> ... </COL>` - used to specify individual column alignments.

Also, some new attributes have been introduced. These are:
`<TABLE BACKGROUND="...">` - specifies a background image for the table.
`<TH BACKGROUND="...">` - specifies a background image for the table header.
`<TD BACKGROUND="...">` - specifies a background image for table data cell.
`<TABLE FRAME="...">` - specifies the appearance of the Table frame
`<TABLE RULES="...">` - specifies the appearance of the Table dividing lines.

Tables and Style Sheets.

All table elements (in fitting with being allowed in the `<BODY>` of a HTML document) can have style sheet information applied to them. This means that they can take the **`CLASS`**, **`ID`** and **`STYLE`** attributes within the elements as well as having their own definitions within the `<STYLE> ... </STYLE>` settings in the `<HEAD>` element. There are special considerations for tables within the style sheet specification though. Some detail about this can be found in the Tables and Style Sheets topic.

If these details are too confusing, there is also a graphical guide to Tables giving some example tables.

Table widths and heights.

There is some confusion about the support for the **`WIDTH`** and **`HEIGHT`** attributes within the `<TABLE>`, `<TH>` and `<TD>` elements. I have tried to define the various nuances of these attributes below.

In the `<TABLE>` element.

Both **Netscape** and **Internet Explorer** support the use of **`WIDTH`** and **`HEIGHT`** `"value%"` and `"pixel_value"` settings for this element.

In the `<TH>` element.

Netscape will support use of the **`WIDTH`** `"value%"` and `"pixel_value"` for this element (it only supports the **`HEIGHT`** attribute for the main `<TABLE>` element). **Internet Explorer** supports only use of the `"pixel_value"` setting for the **`WIDTH`** attribute, but both percentage and absolute pixel values for the **`HEIGHT`** attribute within this element. If only one header has a **`WIDTH`**, or **`HEIGHT`** attribute set, then the setting is used for all the columns/rows of the table (for **Internet Explorer**) that the header is part of. If more than one **`WIDTH`**/**`HEIGHT`** setting is used for header elements within a table column, or row, then the largest of all the settings will be used. Any setting of the attributes within this element (percentage, or absolute pixel values), will be proportioned according to any percentage, or absolute widths set in the

<TABLE> element

In the <TD> element.

Netscape will support use of the **WIDTH** "value%" and "pixel_value" for this element (it only supports the **HEIGHT** attribute for the main <TABLE> element). **Internet Explorer** supports only use of the "pixel_value" setting for the **WIDTH** attribute, but both percentage and absolute pixel values for the **HEIGHT** attribute within this element. If only one cell has a **WIDTH**, or **HEIGHT** attribute set, then the setting is used for all the columns/rows of the table that the cell is part of. If more than one **WIDTH**/**HEIGHT** setting is used for header elements within a table column, or row, then the largest of all the settings will be used. Any setting of the attributes within this element (percentage, or absolute pixel values), will be proportioned according to any percentage, or absolute widths set in the <TABLE> element.

<TABLE> ... </TABLE>

Attributes See Also

This is the main wrapper for all the other table elements, and other table elements will be ignored if they aren't wrapped inside of a `<TABLE> ... </TABLE>` element.

The `<TABLE>` element has the following attributes :

BORDER

This attribute appears in the `<TABLE>` element. If present, borders are drawn around all table cells. If absent, there are no borders, but by default space is left for borders, so the same table with and without the `BORDER` attribute will have the same width. By default, tables are rendered without borders. The value should be a pixel value.

CELLSPACING=<value>

This attribute allows control over the space used between cells in a table. The value should be a pixel value.

CELLPADDING=<value>

This attribute allows control over the space inserted between the cell data and cell wall in a table. The value should be a pixel value.

```
<TABLE BORDER=0 CELLSPACING=0 CELLPADDING=0>
```

gives the most compact table possible.

WIDTH=<value_or_percent>

This attribute is used to describe the desired width of this table, either as an absolute width in pixels, or a percentage of document width. Ordinarily complex heuristics are applied to tables and their cells to attempt to present a pleasing looking table. Setting the `WIDTH` attribute overrides those heuristics and instead effort is put into fitting the table into the desired width as specified.

See the note about table widths and heights for some extra information of these attributes.

HEIGHT=<value_or_percent>

This attribute describes the height of the table, either as a particular pixel value, or as a percentage of the display window. As for the `WIDTH` attribute above, setting this attribute will override the browsers internal display mechanism and it will render the table according to the values given.

ALIGN

Microsofts **Internet Explorer** (version 2.0 and above) and **Netscape** support the `ALIGN` attribute to the `<TABLE>` element. Like that used for floating images, it allows a table to be aligned to the **left** or **right** of the page, allowing text to flow around the table. Also, as with floating images, it is necessary to have knowledge of the `<BR CLEAR=...>` element, to be able to organise the text so as to minimise any unwanted clashing.

VALIGN

Microsofts **Internet Explorer** (version 2.0 and above) supports the `VALIGN` attribute to the `<TABLE>` element. It specifies that the text can be **top**- or **bottom**-aligned. The default is centre-aligned.

BGCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) supports use of the `BGCOLOR` attribute in the `TABLE` element. It allows the background colour of the table to be specified, using either the specified colour names, or a rrggbb hex triplet.

BORDERCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) includes support for this attribute which sets the border colour of the table. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

BORDERCOLORLIGHT

Microsofts **Internet Explorer** (version 2.0) allows use of the BORDERCOLORLIGHT attribute to set independently, the lighter colour to be displayed on a 3-dimensional <TABLE> border. It is the opposite of BORDERCOLORDARK. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

BORDERCOLORDARK

Microsofts **Internet Explorer** (version 2.0) allows use of the BORDERCOLORDARK attribute to set independently, the darker colour to be displayed on a 3-dimensional <TABLE> border. It is the opposite of BORDERCOLORLIGHT. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

NOTE : The BGCOLOR, BORDERCOLOR, BORDERCOLORLIGHT and BORDERCOLORDARK attributes can also be used in <TH>, <TR> and <TD> elements, with the colour defined in the last element over-riding those defined before. E.g. if a <TD> element contains a BORDERCOLOR attribute setting, the setting specified will be used instead of any colour settings that may have been specified in the <TR> element, which in turn over-rides any colour settings in the <TABLE> element.

BACKGROUND

Internet Explorer supports the placing of images inside the <TABLE> element. (Also in the <TD> and <TH> elements) If used in the <TABLE> element, the image in question will be tiled behind all of the table cells. The syntax is the same as that for <BODY BACKGROUND ...>

FRAME

Only **Internet Explore** supports the use of this attribute. It requires the BORDER attribute to be set and affects the display of the table borders. It can accept any of the following values:

void — this removes all the external borders

above — this displays external borders at the top of the table only

below — this displays external borders at the bottom of the table only

hsides — this displays external borders at the horizontal sides of the table. I.e. at the top and bottom of the table.

lhs — this displays external borders at the left hand edges of the table only

rhs — this displays external borders at the right hand edges of the table only.

vsides — this displays external borders at both left and right hand edges of the table

box — this displays a box around the table (i.e. top, bottom, left and right hand sides)

RULES

Internet Explorer supports this new attribute. It requires the BORDER value to be set and may only be used in tables where the <THEAD>, <TBODY> and <TFOOT> sections have been set. It affects the display of the internal table borders ("rules"). It can accept the following values:

none this removes all the internal rules
basic this displays horizontal borders between the <THEAD>, <TBODY> and <TFOOT> sections
rows this displays horizontal borders between all rows
cols this displays horizontal borders between all columns
all this displays all the internal rules.

The following are all allowed within the `<TABLE>` element

- `ALIGN`
- `BACKGROUND`
- `BGCOLOR`
- `BORDER`
- `BORDERCOLOR`
- `BORDERCOLORDARK`
- `BORDERCOLORLIGHT`
- `CELLSPACING`
- `CELLPADDING`
- `FRAME`
- `HEIGHT`
- `RULES`
- `VALIGN`
- `WIDTH`

Tables

<TR> Specifies number of rows in a table.

<TD> Data cell element.

<TH> Header element.

<CAPTION> Caption element.

<THEAD>

<TBODY>

<TFOOT>

<COLGROUP>

<COL>

Graphical examples of Tables

Table - <TR ...> ... </TR>

Attributes See Also

This specifies a table row. The number of rows in a table is exactly specified by how many <TR> elements are contained within it, regardless of cells that may attempt to use the ROWSPAN attribute to span into non-specified rows.

The <TR> element can have the following attributes.

ALIGN

This controls whether text inside the table cell(s) is aligned to the left side of the row, the right side of the cell, or centred within the cell. Values are **left**, **center**, and **right**.

VALIGN

This attribute controls whether text inside the table cell(s) is aligned to the top of the row, the bottom of the cell, or vertically centred within the cell. It can also specify that all the cells in the row should be vertically aligned to the same baseline. Values are **top**, **middle**, **bottom** and **baseline**.

BGCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) has also included use of the BGCOLOR attribute in the TR element. It allows the background colour of the row to be specified, using either the specified colour names, or a rrggbb hex triplet.

BORDERCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) includes support for this attribute which sets the border colour of the row. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

BORDERCOLORLIGHT

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the BORDERCOLORLIGHT attribute to set independently, the lighter colour to be displayed on a 3-dimensional <TABLE> border. It is the opposite of BORDERCOLORDARK. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

BORDERCOLORDARK

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the BORDERCOLORDARK attribute to set independently, the darker colour to be displayed on a 3-dimensional <TABLE> border. It is the opposite of BORDERCOLORLIGHT. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the BORDER attribute to be present in the main <TABLE> element for border colouring to work.

> ***NOTE :*** The BGCOLOR, BORDERCOLOR, BORDERCOLORLIGHT and BORDERCOLORDARK attributes can also be used in <TABLE>, <TH>, and <TD> elements, with the colour defined in the last element over-riding those defined before. E.g. if a <TD> element contains a BORDERCOLOR attribute setting, the setting specified will be used instead of any colour settings that may have been specified in the <TR> element, which in turn over-rides any colour settings in the <TABLE> element.

Tables
<TABLE> - The Table element.
<TD> Table data cell element.
<TH> Table Header element.
<CAPTION> Caption element.
<THEAD>
<TBODY>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

The following are all allowed within the <TR> element

- ALIGN
- BGCOLOR
- BORDERCOLOR
- BORDERCOLORLIGHT
- BORDERCOLORDARK
- VALIGN

Table - <TD ...> ... </TD>

Attributes See Also

This stands for table data, and specifies a standard table data cell. Table data cells must only appear within table rows. Each row need not have the same number of cells specified as short rows will be padded with blank cells on the right. A cell can contain any of the HTML elements normally present in the body of an HTML document. The default alignment of table data is `ALIGN=left` and `VALIGN=middle`. These alignments are overridden by any alignments specified in the containing `<TR>` element, and those alignments in turn are overridden by any `ALIGN` or `VALIGN` attributes explicitly specified on this cell. By default lines inside of table cells can be broken up to fit within the overall cell width. Specifying the `NOWRAP` attribute for a `<TD>` prevents line breaking for that cell.

`<TD ...> ... </TD>` can accept the following attributes.

ALIGN

This attribute controls whether text inside the table cell(s) is aligned to the left side of the cell, the right side of the cell, or centred within the cell. Values are **left**, **center**, and **right**.

VALIGN

The `VALIGN` attribute controls whether text inside the table cell(s) is aligned to the top of the cell, the bottom of the cell, or vertically centred within the cell. It can also specify that all the cells in the row should be vertically aligned to the same baseline. Values are **top**, **middle**, **bottom** and **baseline**.

WIDTH

This attribute is used to describe the desired width of the cell, either as an absolute width in pixels, or a percentage of table width. Ordinarily complex heuristics are applied to tables and their cells to attempt to present a pleasing looking table. Setting the `WIDTH` attribute overrides those heuristics and instead effort is put into fitting the table into the desired width as specified.

See the note about table widths and heights for some extra information of these attributes.

HEIGHT=`<value_or_percent>`

This attribute describes the height of the cell, either as a particular pixel value, or as a percentage of the table width. As for the `WIDTH` attribute above, setting this attribute will override the browsers internal display mechanism and it will render the table according to the values given.

NOWRAP

If this attribute appears in any table cell (`<TH>` or `<TD>`) it means the lines within this cell cannot be broken to fit the width of the cell. Be cautious in use of this attribute as it can result in excessively wide cells.

COLSPAN

This attribute can appear in any table cell (`<TH>` or `<TD>`) and it specifies how many columns of the table this cell should span. The default `COLSPAN` for any cell is 1.

ROWSPAN

This attribute can appear in any table cell (`<TH>` or `<TD>`) and it specifies how many rows of the table this cell should span. The default `ROWSPAN` for any cell is 1. A span that extends into rows that were never specified with a `<TR>` will be truncated.

BGCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) has also included use of the `BGCOLOR` attribute in the `TD` element. It allows the background colour of the data cell to be specified, using either the specified colour names, or a rrggbb hex triplet.

BORDERCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) includes support for this attribute which sets the border colour of the data cell. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

BORDERCOLORLIGHT

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the `BORDERCOLORLIGHT` attribute to set independently, the lighter colour to be displayed on a 3-dimensional `<TABLE>` border. It is the opposite of `BORDERCOLORDARK`. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

BORDERCOLORDARK

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the `BORDERCOLORDARK` attribute to set independently, the darker colour to be displayed on a 3-dimensional `<TABLE>` border. It is the opposite of `BORDERCOLORLIGHT`. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

NOTE : The `BGCOLOR, BORDERCOLOR, BORDERCOLORDARK` and `BORDERCOLORLIGHT` attributes can also be used in `<TABLE>`, `<TH>`, and `<TR>` elements, with the colour defined in the last element over-riding those defined before. E.g. if a `<TD>` element contains a `BORDERCOLOR` attribute setting, the setting specified will be used instead of any colour settings that may have been specified in the `<TR>` element, which in turn over-rides any colour settings in the `<TABLE>` element.

BACKGROUND

Internet Explorer supports the placing of images inside the `<TD>` element. (Also in the `<TABLE>` and `<TH>` elements) If used in the `<TD>` element, the image in question will be tiled behind the table data cell specified. The syntax is the same as that for `<BODY BACKGROUND ...>`

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TH> Table Header element.
<CAPTION> Caption element.
<THEAD>
<TBODY>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

The following are all allowed within the `<TD>` element

- ALIGN
- BACKGROUND
- BGCOLOR
- BORDERCOLOR
- BORDERCOLORLIGHT
- BORDERCOLORDARK
- COLSPAN
- HEIGHT
- NOWRAP
- ROWSPAN
- VALIGN
- WIDTH

Table - <TH ...> ... </TH>

Attributes See Also

This stands for table header. Header cells are identical to data cells in all respects, with the exception that header cells are in a **bold** FONT, and have a default `ALIGN=center`.

`<TH ...> ... </TH>` can contain the following attributes

ALIGN

The `ALIGN` attribute controls whether text inside the table cell(s) is aligned to the left side of the cell, the right side of the cell, or centred within the cell. Values are **left**, **center**, and **right**.

VALIGN

This attribute controls whether text inside the table cell(s) is aligned to the top of the cell, the bottom of the cell, or vertically centred within the cell. It can also specify that all the cells in the row should be vertically aligned to the same baseline. Values are **top**, **middle**, **bottom** and **baseline**.

WIDTH

This attribute is used to describe the desired width of the header cell, either as an absolute width in pixels, or a percentage of table width. Ordinarily complex heuristics are applied to tables and their cells to attempt to present a pleasing looking table. Setting the `WIDTH` attribute overrides those heuristics and instead effort is put into fitting the table into the desired width as specified.

See the note about table widths and heights for some extra information of these attributes.

HEIGHT=<value_or_percent>

This attribute describes the height of the header cell, either as a particular pixel value, or as a percentage of the table width. As for the `WIDTH` attribute above, setting this attribute will override the browsers internal display mechanism and it will render the table according to the values given.

NOWRAP

This attribute specifies that the lines within this cell cannot be broken to fit the width of the cell. Be cautious in use of this attribute as it can result in excessively wide cells.

COLSPAN

The `COLSPAN` attribute specifies how many columns of the table this cell should span. The default `COLSPAN` for any cell is 1.

ROWSPAN

This attribute specifies how many rows of the table this cell should span. The default `ROWSPAN` for any cell is 1. A span that extends into rows that were never specified with a `<TR>` will be truncated.

BGCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) has also included use of the `BGCOLOR` attribute in the `TH` element. It allows the background colour of the heading cell to be specified, using either the specified colour names, or a rrggbb hex triplet.

BORDERCOLOR

Microsofts **Internet Explorer** (version 2.0 and above) includes support for this attribute which sets the border colour of the heading cell. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

BORDERCOLORLIGHT

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the `BORDERCOLORLIGHT` attribute to set independently, the lighter colour to be displayed on a 3-dimensional `<TABLE>` border. It is the opposite of `BORDERCOLORDARK`. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

BORDERCOLORDARK

Microsofts **Internet Explorer** (version 2.0 and above) allows use of the `BORDERCOLORDARK` attribute to set independently, the darker colour to be displayed on a 3-dimensional `<TABLE>` border. It is the opposite of `BORDERCOLORLIGHT`. Any of the pre-defined colour names can be used, as well as any colour defined by a rrggbb hex triplet. It is necessary for the `BORDER` attribute to be present in the main `<TABLE>` element for border colouring to work.

NOTE : The `BGCOLOR, BORDERCOLOR, BORDERCOLORDARK` and `BORDERCOLORLIGHT` attributes can also be used in `<TABLE>`, `<TD>`, and `<TR>` elements, with the colour defined in the last element over-riding those defined before. E.g. if a `<TD>` element contains a `BORDERCOLOR` attribute setting, the setting specified will be used instead of any colour settings that may have been specified in the `<TR>` element, which in turn over-rides any colour settings in the `<TABLE>` element.

BACKGROUND

Internet Explorer supports the placing of images inside the `<TH>` element. (Also in the `<TD>` and `<TABLE>` elements) If used in the `<TH>` element, the image in question will be tiled behind the table header. The syntax is the same as that for `<BODY BACKGROUND ...>`

Tables

<TABLE> - The Table element.

<TR> Specifies number of rows in a table.

<TD> Table data cell element.

<CAPTION> Caption element.

<THEAD>

<TBODY>

<TFOOT>

<COLGROUP>

<COL>

Graphical examples of Tables

The following are all allowed within the `<TH>` element

- `ALIGN`
- `BACKGROUND`
- `BGCOLOR`
- `BORDERCOLOR`
- `BORDERCOLORLIGHT`
- `BORDERCOLORDARK`
- `COLSPAN`
- `HEIGHT`
- `NOWRAP`
- `ROWSPAN`
- `VALIGN`
- `WIDTH`

Table - <CAPTION ...> ... </CAPTION>

See Also

This represents the caption for a table. `<CAPTION>` elements should appear inside the `<TABLE>` but not inside table rows or cells. Like table cells, any document body HTML can appear in a caption.

The `<CAPTION>` element can accept the following attributes.

ALIGN

The `ALIGN` attribute controls whether the caption appears above or below the table, and can have the values **top** or **bottom**, defaulting to top. Microsofts **Internet Explorer** allows use of **left**, **right** and **center** for the alignment of the `<CAPTION>` element. This forces the caption text to be aligned flush-left, flush-right, or centred within the boundaries of the table.

VALIGN

Microsofts **Internet Explorer** allows use of the `VALIGN` attribute inside the `<CAPTION>` element. It specifies whether the caption text should be displayed at the **top** or **bottom** of the table.

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TD> Table data cell element.
<TH> Table Header element.
<THEAD>
<TBODY>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

Table - <THEAD> ... </THEAD>

See Also

This element, which is **Internet Explorer** specific, is used to specify the head section of the table. It is somewhat analogous to the <HEAD> element. It does not render on the screen, but is required is RULES wish to be set in the <TABLE>.

For an example, see the HTML 3 tables examples.

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TD> Table data cell element.
<TH> Table Header element.
<CAPTION>
<TBODY>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

Table - <TBODY> ... </TBODY>

See Also

This element, which is **Internet Explorer** specific, is used to specify the body section of the table. It is somewhat analogous to the `<BODY>` element. It does not render on the screen, but is required is `RULES` wish to be set in the `<TABLE>`.

For an example, see the HTML 3 tables examples.

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TD> Table data cell element.
<TH> Table Header element.
<CAPTION>
<THEAD>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

Table - <TFOOT> ... </TFOOT>

See Also

This element, which is **Internet Explorer** specific, is used to specify the footer section of the table. It does not render on the screen, but is required is RULES wish to be set in the <TABLE>.

For an example, see the HTML 3 tables examples.

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TD> Table data cell element.
<TH> Table Header element.
<CAPTION>
<THEAD>
<TBODY>
<COLGROUP>
<COL>
Graphical examples of Tables

Table - <COLGROUP> ... </COLGROUP>

Attributes See Also

This element, which is **Internet Explorer** specific, can be used to group columns together to set their alignment properties. It accepts the following attributes:

ALIGN="center|justify|left|right"
This sets the text alignment within the column group. The default value is "center"

VALIGN="baseline|bottom|middle|top"
This sets the vertical text alignment within the column group.

SPAN=
This can be used to set the number of columns upon which the ALIGN and VALIGN attributes are to act.

For examples of this element's use, see the HTML 3 tables examples.

The `<COLGROUP>` element accepts the following attributes:

```
ALIGN
VALIGN
SPAN
```

Tables
<TABLE> - The Table element.
<TR> Specifies number of rows in a table.
<TD> Table data cell element.
<TH> Table Header element.
<CAPTION>
<THEAD>
<TBODY>
<TFOOT>
<COL>
Graphical examples of Tables

Table - <COL> ... </COL>

Attributes See Also

This element, which is **Internet Explorer** specific, can be used to specify the text alignment for table columns. It accepts the following attributes:

ALIGN="center|justify|left|right"
This sets the text alignment within the column group. The default value is "center"

SPAN=
This can be used to set the number of columns upon which the ALIGN attribute is to act.

For examples of this element's use, see the HTML 3 tables examples.

The `<COL>` element accepts the following attributes:

```
    SPAN
    ALIGN
```

Tables

`<TABLE>` - The Table element.

`<TR>` Specifies number of rows in a table.

`<TD>` Table data cell element.

`<TH>` Table Header element.

`<CAPTION>`

`<TBODY>`

`<THEAD>`

`<TFOOT>`

`<COLGROUP>`

Graphical examples of Tables

Tables and Style Sheets

See Also

NOTE : presented here is an overview of the table section of the Style Sheet specification. More complete information can be obtained from the Style Sheet specification, available from http://www.w3.org/pub/WWW/Style

For simple styling of tables, normal Style Sheet methods can be employed (i.e. defining the elements style within the `<STYLE> ... </STYLE>` elements, or by using **CLASS**, **ID** and **STYLE** attributes. For more information, see the Style Sheets topic). For example :

```
<STYLE>
  TABLE { font-size : 12pt;
          color : #FF0000}
  TH { font-size : 14pt;
       font-weight : bold}
</STYLE>
```

are acceptable.

The problem with tables comes when considering the borders present and therefore any border styling methods. According to the HTML 3 table model, there are only three sections of a table that can be considered as having well defined borders (the `<TABLE>` itself, row groups (as defined by`<THEAD>`, `<TBODY>` and `<TFOOT>`) and table cells - `<TD>` and `<TH>`). Table rows can be considered to be nested within the table and table cells can be considered to be nested within the table rows. It is counter-intuitive to think of tables like this (as a three level nested object) and so the Style Sheet table model treats tables as single level objects (i.e. the border/edge of a cell at the edge of a table and the table are considered to share a border). Essentially, considering elements in the nesting order of a table, elements higher in the nesting order will take precedence. That is, the border style of a `<TABLE>` will be used, but the border style of rows, or cells will only be used when there is no border specified for the table.

The style sheet table model adds a couple of values to the **border-style** property and adds a new **elevation** property. For details of style sheet properties and values, see the Properties and Values topic.

Additions to the **'border-style'** property.

The table model adds the **blank** and **bevel** values to the border property. Blank differs from the 'none' property allowed for borders by preventing the border style being over-ridden by elements lower in the table nesting order. Bevel specifies that the table has a 3-D border style (necessary to allow the **elevation** property below)

The **'elevation'** property.

This property specifies the elevation (in the z-direction - towards the viewer from the screen) of the table border-styles. It will only work if the **border-style** of **bevel** is used. The elevations of `<TR>`, `<THEAD>`, `<TBODY>` and `<TFOOT>` are used to determine the elevation of table cells (but aren't displayed) while the elevation of the `<TABLE>` element is used to determine the elevation of the cells contained within the table. It requires a unitless value which is used as the size of the 3-D border.

Tables
<TABLE>
<TR> Specifies number of rows in a table.
<TD> Data cell element.
<TH> Header element.
<CAPTION> Caption element.
<THEAD>
<TBODY>
<TFOOT>
<COLGROUP>
<COL>
Graphical examples of Tables

Graphical Examples of Tables

See Also

A Simple table
Using `ROWSPAN`
Using `COLSPAN`
Using Headers
Using all of the above
Using `ALIGN/VALIGN`
Nested tables

Internet Explorer specific attributes for tables.
Floating tables
Colouring tables
Border Colouring
Caption Alignment

HTML 3 specific tables (supported by **Internet Explorer** only)
Using `COL` and `COLGROUP`
Specifying `FRAMES` and `RULES`
An example of it all
3D tables

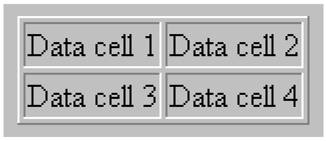
A Simple Table

See Also

The following will render a simple 2x2 table.

```
<TABLE BORDER>
 <TR>
  <TD>Data cell 1</TD><TD>Data cell 2</TD>
 </TR>
 <TR>
  <TD>Data cell 3</TD><TD>Data cell 4</TD>
 </TR>
</TABLE>
```

📷



Data cell 1	Data cell 2
Data cell 3	Data cell 4

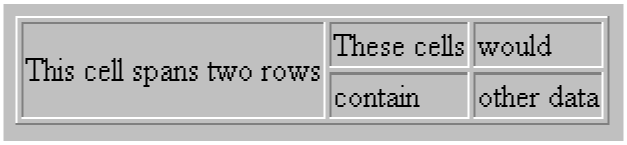
Using ROWSPAN

See Also

The following will render a table with the first data cell spanning two rows.

```
<TABLE BORDER>
 <TR>
  <TD ROWSPAN=2>This cell spans two rows</TD>
  <TD>These cells</TD><TD>would</TD>
 </TR>
 <TR>
  <TD>contain</TD><TD>other data</TD>
 </TR>
</TABLE>
```

📷



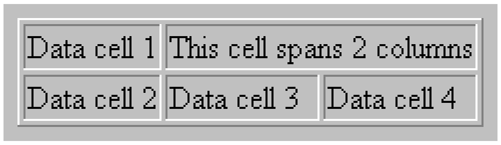
This cell spans two rows	These cells	would
	contain	other data

Using COLSPAN

See Also

The following will render a table where the right uppermost data cell spans two columns

```
<TABLE BORDER>
 <TR>
  <TD>Data cell 1</TD>
  <TD COLSPAN=2>This cell spans 2 columns</TD>
 </TR>
 <TR>
  <TD>Data cell 2</TD><TD>Data cell 3</TD><TD>Data cell 4</TD>
 </TR>
</TABLE>
```



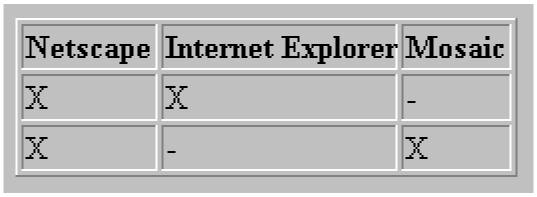
Data cell 1	This cell spans 2 columns	
Data cell 2	Data cell 3	Data cell 4

Using Headers

See Also

The following will render a table that uses column headers

```
<TABLE BORDER>
 <TR>
  <TH>Netscape</TH><TH>Internet Explorer</TH><TH>Mosaic</TH>
 </TR>
 <TR>
  <TD>X</TD><TD>X</TD><TD>-</TD>
 </TR>
 <TR>
  <TD>X</TD><TD>-</TD><TD>X</TD>
 </TR>
</TABLE>
```



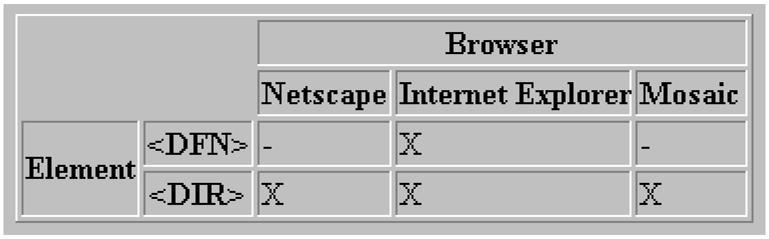
Netscape	Internet Explorer	Mosaic
X	X	-
X	-	X

Using all of the above

See Also

The following renders a table that uses ROWSPAN, COLSPAN and table header settings.

```
<TABLE BORDER>
 <TR>
  <TD><TH ROWSPAN=2></TH>
  <TH COLSPAN=3>Browser</TH></TD>
 </TR>
 <TR>
  <TD><TH>Netscape</TH><TH>Internet Explorer</TH>
  <TH>Mosaic</TH></TD>
 </TR>
 <TR>
  <TH ROWSPAN=2>Element</TH>
  <TH><DFN></TH><TD>-</TD><TD>X</TD>
  <TD>-</TD>
 </TR>
 <TR>
  <TH><DIR></TH><TD>X</TD><TD>X</TD>
  <TD>X</TD>
 </TR>
</TABLE>
```



		Browser		
		Netscape	Internet Explorer	Mosaic
Element	<DFN>	-	X	-
	<DIR>	X	X	X

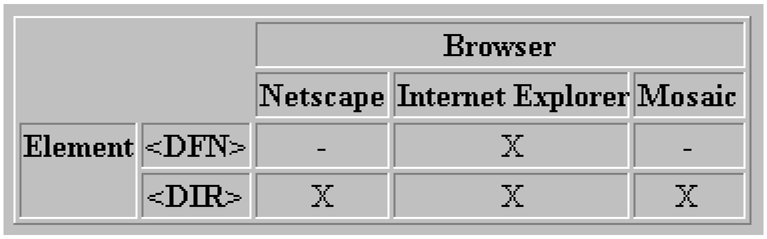
Using ALIGN/VALIGN

See Also

The following will render a table using ALIGN and VALIGN attributes to provide better formatting of the example table used previously.

```
<TABLE BORDER>
 <TR>
  <TD><TH ROWSPAN=2></TH>
  <TH COLSPAN=3>Browser</TH></TD>
 </TR>
 <TR>
  <TD><TH>Netscape</TH><TH>Internet Explorer</TH>
  <TH>Mosaic</TH></TD>
 </TR>
 <TR>
  <TH ROWSPAN=2 VALIGN=top>Element</TH>
  <TH><DFN></TH>
  <TD ALIGN=center>-</TD>
  <TD ALIGN=center>X</TD>
  <TD ALIGN=center>-</TD>
 </TR>
 <TR>
  <TH><DIR></TH>
  <TD ALIGN=center>X</TD>
  <TD ALIGN=center>X</TD>
  <TD ALIGN=center>X</TD>
 </TR>
</TABLE>
```

📷



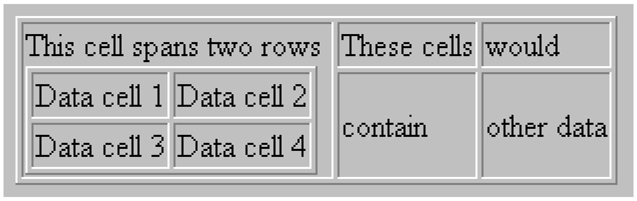
		Browser		
		Netscape	Internet Explorer	Mosaic
Element	<DFN>	-	X	-
	<DIR>	X	X	X

Nested Tables

See Also

To show that tables can be nested, the following will render a table using ROWSPAN attrbutes to nest the Simple Table insideanother table.

```
<TABLE BORDER>
 <TR>
  <TD ROWSPAN=2>This cell spans two rows
   <TABLE BORDER>
    <TR>
     <TD>Data cell 1</TD><TD>Data cell 2</TD>
    </TR>
    <TR>
     <TD>Data cell 3</TD><TD>Data cell 4</TD>
    </TR>
   </TABLE>
  </TD>
  <TD>These cells</TD><TD>would</TD>
 </TR>
 <TR>
  <TD>contain</TD><TD>other data</TD>
 </TR>
</TABLE>
```



<table>
<tr><td rowspan="2">This cell spans two rows<table><tr><td>Data cell 1</td><td>Data cell 2</td></tr><tr><td>Data cell 3</td><td>Data cell 4</td></tr></table></td><td>These cells</td><td>would</td></tr>
<tr><td>contain</td><td>other data</td></tr>
</table>

Floating tables

See Also

The following will render two small tables, both of which are floating, with text wrapping around the tables. ***NOTE :*** Only the Internet Explorer and Netscape support the alignment of tables within a HTML document. Other browsers may not display the tables as they are shown here.

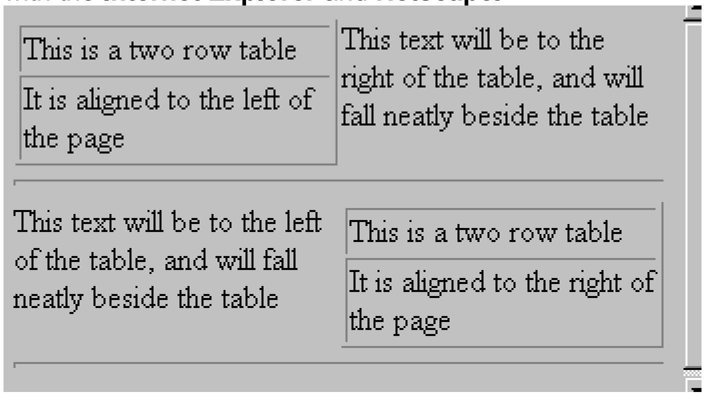
```
<TABLE ALIGN=left BORDER WIDTH=50%>
 <TR>
  <TD>This is a two row table</TD>
 </TR>
 <TR>
  <TD>It is aligned to the left of the page</TD>
 </TR>
</TABLE>

This text will be to the right of the table, and will fall neatly beside the
table
<BR CLEAR=all>
<HR>

<TABLE ALIGN=right BORDER WIDTH=50%>
 <TR>
  <TD>This is a two row table</TD>
 </TR>
 <TR>
  <TD>It is aligned to the right of the page</TD>
 </TR>
</TABLE>

This text will be to the left of the table, and will fall neatly beside the
table
<BR CLEAR=all>
<HR>
```

NOTE : Table alignment is currently only possible with the **Internet Explorer** and **Netscape.**



This is a two row table
It is aligned to the left of the page

This text will be to the right of the table, and will fall neatly beside the table

This text will be to the left of the table, and will fall neatly beside the table

This is a two row table
It is aligned to the right of the page

Colouring Tables

See Also

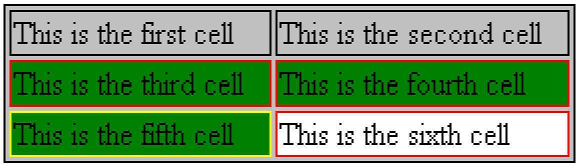
The following table shows the use of the BGCOLOR and BORDERCOLOR attributes in the <TABLE>, <TR>, <TD> and <TH> elements. It shows the order of rendering as well. I.e. the BORDERCOLOR and BGCOLOR settings used in the <TD> elements over-ride those set in the <TR> elements, which in turn over-ride those set in the <TABLE> element.

```
<TABLE BORDER BGCOLOR=Silver BORDERCOLOR=Black WIDTH=50%>
 <TR>
  <TD>This is the first cell</TD>
  <TD>This is the second cell</TD>
 </TR>
 <TR BORDERCOLOR=Red BGCOLOR=Green>
  <TD>This is the third cell</TD>
  <TD>This is the fourth cell</TD>
 </TR>
 <TR BORDERCOLOR=Red BGCOLOR=Green>
  <TD BORDERCOLOR=Yellow>This is the fifth cell</TD>
  <TD BGCOLOR=White>This is the sixth cell</TD>
 </TR>
</TABLE>
```

NOTE : For the fifth cell, where a BGCOLOR is not specified, it adopts the background colour most recently specified, i.e. Green. The same is true for the sixth cell that adopts the most recent border colour of Red.

NOTE : For table cell and row border colouring to work, the BORDER attribute of the <TABLE> element must be present.

The following table was rendered in Microsofts Internet Explorer. Other browsers may not support colouring of tables.



This is the first cell	This is the second cell
This is the third cell	This is the fourth cell
This is the fifth cell	This is the sixth cell

The first and second cells have no colour settings, so adopt those set in the main `<TABLE>` element. I.e. `BORDERCOLOR=Black` and `BGCOLOR=Silver`.

The third and fourth cells have `BORDERCOLOR=Red` and `BGCOLOR=Green` specified for their whole row, over riding the settings in the main `<TABLE>` element.

The fifth and sixth cells have `BORDERCOLOR=Yellow` and `BGCOLOR=White` respectively, over-riding the settings in their parent `<TR>` and `<TABLE>` elements.

Border Colouring

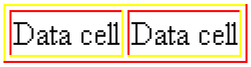
See Also

The following table shows the use of the BORDERCOLORLIGHT and BORDERCOLORDARK attributes, specifying the respective light and dark colours to be used when rendering the 3-D border of a table.

NOTE : Use of these attributes is **Internet Explorer** specific.

```
<TABLE BORDER BORDERCOLORDARK=Red BORDERCOLORLIGHT=Yellow>
 <TR>
  <TD>Data cell</TD>
  <TD>Data cell</TD>
 </TR>
</TABLE>
```

NOTE : Border colouring is currently only possible with the **Internet Explorer.**



Data cell	Data cell

Caption Alignment

See Also

The following tables use various **Internet Explorer** specific <CAPTION> attributes.

This table renders with the caption at the top left of the table.

```
<TABLE BORDER>
<CAPTION VALIGN=top ALIGN=left>Table caption</CAPTION>
  <TR>
    <TD>Cell no. 1</TD>
    <TD>Cell no. 2</TD>
  </TR>
  <TR>
    <TD>Cell no. 3</TD>
    <TD>Cell no. 4</TD>
  </TR>
</TABLE>
```

📷

This renders a table with the caption at the bottom right of the table.

```
<TABLE BORDER>
<CAPTION VALIGN=bottom ALIGN=right>Table caption</CAPTION>
  <TR>
    <TD>Cell no. 1</TD>
    <TD>Cell no. 2</TD>
  </TR>
  <TR>
    <TD>Cell no. 3</TD>
    <TD>Cell no. 4</TD>
  </TR>
</TABLE>
```

📷

NOTE : This table are currently only possible with the **Internet Explorer.**

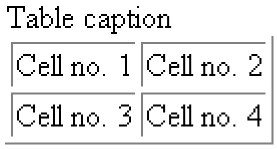
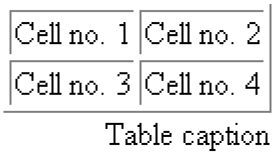


Table caption

Cell no. 1	Cell no. 2
Cell no. 3	Cell no. 4

NOTE : This table is currently only possible with the **Internet Explorer.**



Cell no. 1	Cell no. 2
Cell no. 3	Cell no. 4

Table caption

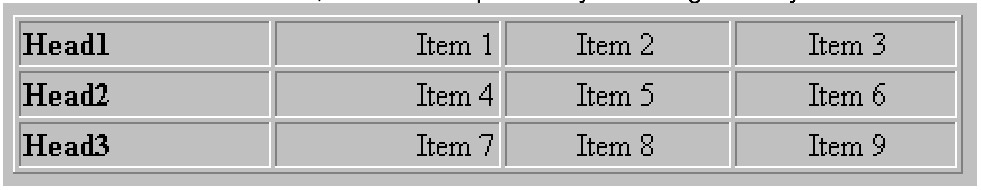
Using COL and COLGROUP

See Also

In this example, (which will only render correctly in the **Internet Explorer**), the two <COL> elements specify the text alignment for the first two columns, and the <COLGROUP> specifies alignments for both the last two columns.

```
<TABLE BORDER width=75%>
  <COL ALIGN=left>
  <COL ALIGN=right>
  <COLGROUP SPAN=2 ALIGN=center VALIGN=top>
  <TR>
   <TD><B>Head1</B></TD><TD>Item 1</TD> <TD>Item 2</TD> <TD>Item 3</TD>
  </TR>
  <TR>
   <TD><B>Head2</B></TD><TD>Item 4</TD> <TD>Item 5</TD> <TD>Item 6</TD>
  </TR>
  <TR>
   <TD><B>Head3</B></TD><TD>Item 7</TD> <TD>Item 8</TD> <TD>Item 9</TD>
  </TR>
</TABLE>
```

The following table gives examples of the <COL> and <COLGROUP> HTML 3 table elements and attributes. At the moment, such mark up will only be recognised by the **Internet Explorer.**



Head1	Item 1	Item 2	Item 3
Head2	Item 4	Item 5	Item 6
Head3	Item 7	Item 8	Item 9

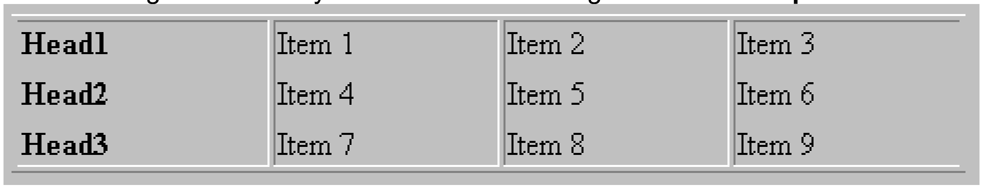
Specifying FRAMES and RULES

See Also

In this example, (which will only render correctly in the **Internet Explorer**), the <COLGROUP> element specifies alignments for all the columns, with the FRAME and RULES attributes determining the table border..

```
<TABLE BORDER width=75% FRAME=hsides RULES=cols>
 <THEAD>
  <COLGROUP SPAN=4 ALIGN=left VALIGN=top>
 </THEAD>
  <TBODY>
  <TR>
   <TD><B>Head1</B></TD><TD>Item 1</TD> <TD>Item 2</TD> <TD>Item 3</TD>
  </TR>
  <TR>
   <TD><B>Head2</B></TD><TD>Item 4</TD> <TD>Item 5</TD> <TD>Item 6</TD>
  </TR>
  <TR>
   <TD><B>Head3</B></TD><TD>Item 7</TD> <TD>Item 8</TD> <TD>Item 9</TD>
  </TR>
  </TBODY>
 <TFOOT></TFOOT>
</TABLE>
```

The following table will only render as shown using the **Internet Explorer**



Head1	Item 1	Item 2	Item 3
Head2	Item 4	Item 5	Item 6
Head3	Item 7	Item 8	Item 9

An example of it all

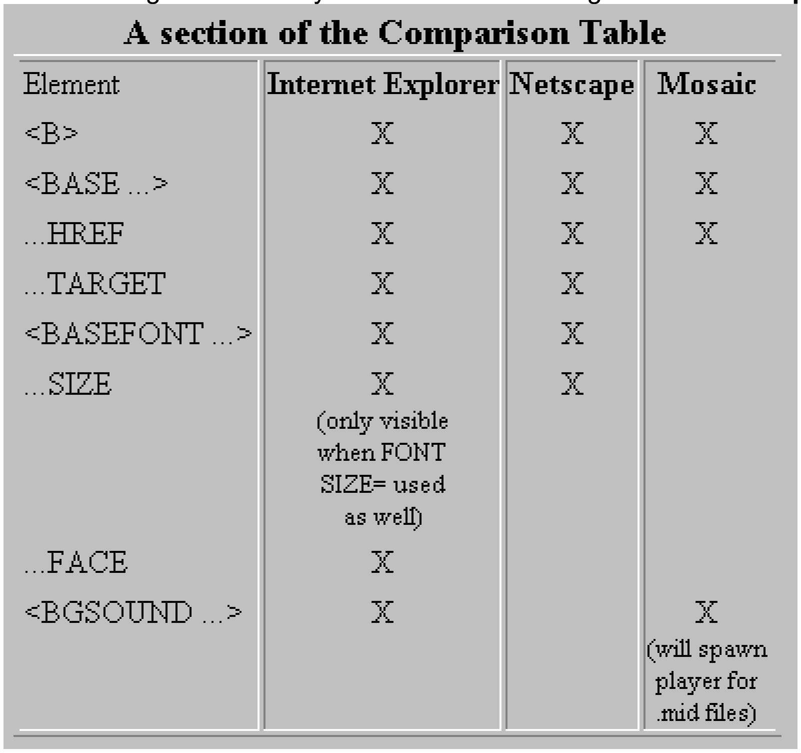
See Also

The following table gives examples of all the above mentioned HTML 3 table elements and attributes. At the moment, such mark up will only be recognised by the **Internet Explorer.**

In the example, the `<COL>` element is used to specify the alignment of the first column, with the `<COLGROUP>` element specifying the alignment for the remaining three columns in one group.

```
<TABLE BORDER FRAME=hsides RULES=cols>
  <COL ALIGN=left>
  <COLGROUP SPAN=3 ALIGN=center VALIGN=middle>
   <THEAD>
    <CAPTION ALIGN=center><FONT SIZE=+1><B>A section of the Comparison
    Table</B></FONT>
    </CAPTION>
   <TR>
    <TD>Element</TD><TD><B>Internet
     Explorer</B></TD><TD><B>Netscape</B></TD><TD><B>Mosaic</B></TD>
   </TR>
   </THEAD>
   <TBODY>
   <TR>
    <TD>&lt;B&gt;</TD><TD>X</TD><TD>X</TD><TD>X</TD>
   </TR>
   <TR>
    <TD>&lt;BASE ...&gt;</TD><TD>X</TD><TD>X</TD><TD>X</TD>
   </TR>
   <TR>
    <TD>  ...HREF</TD><TD>X</TD><TD>X</TD><TD>X</TD>
   </TR>
   <TR>
    <TD>  ...TARGET</TD><TD>X</TD><TD>X</TD><TD></TD>
   </TR>
   <TR>
    <TD>&lt;BASEFONT ...&gt;</TD><TD>X</TD><TD>X</TD><TD></TD>
   </TR>
   <TR>
    <TD VALIGN=top>  ...SIZE</TD><TD>X<BR><FONT SIZE=-1>(only visible<BR>when
     FONT<BR>SIZE= used<BR>as well)</FONT></TD><TD VALIGN=top>X</TD><TD></TD>
   </TR>
   <TR>
    <TD>  ...FACE</TD><TD>X</TD><TD></TD><TD></TD>
   </TR>
   <TR>
    <TD VALIGN=top>&lt;BGSOUND ...&gt;</TD><TD
     VALIGN=top>X</TD><TD></TD><TD>X<BR><FONT SIZE=-1>(will spawn<BR>player
     for<BR>.mid files)</FONT></TD>
   </TR>
   </TBODY>
   <TFOOT></TFOOT>
</TABLE>
```

The following table will only render as shown using the **Internet Explorer**



A section of the Comparison Table

Element	Internet Explorer	Netscape	Mosaic
<B>	X	X	X
<BASE ...>	X	X	X
...HREF	X	X	X
...TARGET	X	X	
<BASEFONT ...>	X	X	
...SIZE	X (only visible when FONT SIZE= used as well)	X	
...FACE	X		
<BGSOUND ...>	X		X (will spawn player for .mid files)

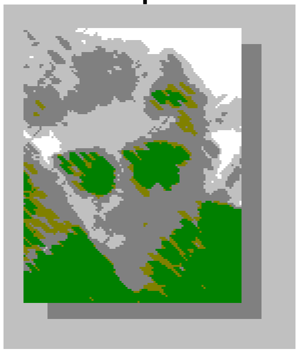
'3D' Tables

See Also

This is an example of how to use HEIGHT and WIDTH attributes to present a table that appears to have a shadow. It requires **Internet Explorer** to display properly. (To make it easier to see how it's done, try setting the BORDER value.)

```
<TABLE CELLPADDING=0 CELLSPACING=0>
  <TR>
   <TD COLSPAN=2 ROWSPAN=2>
   <IMG SRC="GLAST.GIF">
   </TD>
   <TD HEIGHT=8 WIDTH=10></TD>
  </TR>
  <TR>
   <TD BGCOLOR=GRAY></TD>
  </TR>
  <TR>
   <TD HEIGHT=8 WIDTH=10></TD>
   <TD BGCOLOR=GRAY WIDTH=75></TD>
   <TD BGCOLOR=GRAY></TD>
  </TR>
</TABLE>
```

The following table will only render as shown using the **Internet Explorer**



Tables - An Introduction

Dynamic Documents

Recent browsers (i.e. Netscape and the Internet Explorer) support various mechanisms for inroducing dynamic content.

Server push
The server sends down a chunk of data; the browser display the data but leaves the connection open; whenever the server wants it sends more data and the browser displays it, leaving the connection open; at some later time the server sends down yet more data and the browser displays it; etc.

Client pull
The server sends down a chunk of data, including a directive (in the HTTP response or the document header) that says "reload this data in 5 seconds", or "go load this other URL in 10 seconds". After the specified amount of time has elapsed, the client loads the given URL.

JavaScript
Netscape Navigator 2.0 and Netscape Navigator Gold 2.0 provide flexible, lightweight programmability via the Netscape scripting language, a programmable API that allows cross-platform scripting of events, objects, and actions. It allows the page designer to access events such as startups, exits, and user mouse clicks. Based on the Java language, the Netscape scripting language extends the programmatic capabilities of Netscape Navigator to a wide range of authors and is easy enough for anyone who can compose HTML. Use the Netscape scripting language to glue HTML, inline plug-ins, and Java Applets to each other.

Java
From Sun Microsystems comes Java, the platform independent programming language for creating executable content within web pages. Based on C++, this is not a language to be taken lightly, but when mastered, could prove as limitless as the authors imagination. Indeed, some have even gone so far as to predict that the computer software industry is seriously under threat, because in a few years, all applications will be in the form of applets, downloaded as and when they are required.

ActiveX
The recent technology from Microsoft based around ActiveX controls (previously known as OLE controls). A new control requirement specification has meant that OLE controls previously burdened with code inappropriate for use on the Internet can now be much more streamlined, making it possible to embed them as `<OBJECT>`s into web pages. This (like Java) allows almost limitless activity and interactivity within web pages. (**Internet Explorer** for Windows 95/NT specific)

Visual Basic Script
A lightweight, yet fully compatible version of Visual Basic, designed for use on the Internet, Visual Basic Script allows full automation, customisation and scripting within web pages. Coming into its own when used to control ActiveX controls, Visual Basic is the easiest of the available methods for creating dynamic content to learn. (**Internet Explorer** specific)

Server Side Includes
Server side includes (SSI) applied to an HTML document, provide for interactive real-time features such as echoing current time, conditional execution based on logical comparisons, querying or updating a database, sending an e-mail etc., with no programming or CGI scripts.

Server Push

See Also

Server Push allows for dynamic document updating via a server to client connection that is kept open. This method (as opposed to Client Pull) is totally controlled by the server, but the perpetual open connection occupies valuable server resources. It's main advantage over Client Pull though, is that using Server Push, it is possible to replace single elements within a HTML document.

The exact Server Push mechanism is technically complex and is outside the scope of this reference. Those that are interested in utilising Server Push in CGI scripts, or web server based executable applications should visit the Netscape site (http://www.netscape.com/) for more information. It should be noted that only **Netscape** supports the use of Server Push.

Dynamic Documents

Client Pull

See Also

The main use of client pull is to cause a document to be automatically reloaded on a regular basis. For example, consider the following document :

```
<META HTTP-EQUIV="Refresh" CONTENT="x">
<TITLE>Document ONE</TITLE>

<H1>This is a document</H1>

Here's some text. <P>
```

If loaded into a browser supporting Client Pull, it would re-load itself every *x* seconds.

In most cases, Client Pull is used to automatically load another document (automatic re-direction). For this, the HTTP response header needs to be :

```
Refresh: x; URL=http://foo.bar/blatz.html
```

and the corresponding `META` element syntax is:

```
<META HTTP-EQUIV="Refresh" CONTENT="x; URL=http://foo.bar/blatz.html">
```

This loads the document referenced by the URL part of the `CONTENT` attribute after *x* seconds.

Dynamic Documents
<META> Element

JavaScript

See Also

NOTE : JavaScript is currently supported by the **Netscape Navigator** (version 2 and above) and **Internet Explorer** (version 3.0 and above). It is still under development and is liable to change. For more information on JavaScript, point your browser to http://home.netscape.com/. The information provided here only details how to include JavaScript scripts within HTML documents, not how to author actual scripts. Such information is well beyond the scope of this document.

A script is embedded in HTML within a `<SCRIPT>` element.

```
<SCRIPT>...</SCRIPT>
```

The text of a script is inserted between `<SCRIPT>` and its end element.

Attributes within the `<SCRIPT>` element can be specified as follows:

```
<SCRIPT LANGUAGE="JavaScript">
</SCRIPT>
```

The **LANGUAGE** attribute is mandatory unless the **SRC** attribute is present and specifies the scripting language. ***NOTE :*** Use of the `LANGUAGE` attribute is important now that **Internet Explorer** supports JavaScript. In order for the **Internet Explorer** to know it is to process the script as JavaScript, it needs to know that it is JavaScript.

The **SRC** attribute is optional and, if given, specifies a URL that loads the text of a script.

```
<SCRIPT LANGUAGE="language" SRC=url>
```

Both attributes may be present.
NOTE : For **Netscape** to be able to properly use external JavaScript files, the server on which the files are to reside must have the MIME type `application/x-javascript` mapped into its list of MIME types (with a suitable extension mapped to it - **Netscape** recommend "`.js`") for **Netscape** to properly respond to the script when it is loaded.

Scripts should be placed inside comment fields to ensure that the script is not displayed when the page's HTML is viewed with a browser unaware of the `<SCRIPT>` element. The entire script is encased by HTML comment elements:

```
 <!-- Begin to hide script contents from old browsers.
 <!-- End the hiding here.-->
```

For more information about scripts, the language and some examples, see http://home.netscape.com

Netscape now supports use of the `<NOSCRIPT> ... </NOSCRIPT>` container in HTML, which allows authors to write alternative content to cater for those users who browse with JavaScript capabilities disabled in the browser (or for those using non JavaScript aware browsers). If your intended HTML document comlpetely relies on JavaScript for its effect, then the alternative content should be contained within the `<NOSCRIPT> ... </NOSCRIPT>` container.
For example :

```
<HTML>
<HEAD><TITLE>Welcome to JavaScript World</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
```

```
  All of the JavaScript that controls the page goes here
// -->
</SCRIPT>
</HEAD>
<BODY>
<NOSCRIPT>
  Sorry, you need a JavaScript capable browser to get the best from this page
</NOSCRIPT>
</BODY>
</HTML>
```

Users browsing with a JavaScript capable browser (**Netscape** and **Internet Explorer**) that have JavaScript enabled will get the full effect from the Script contained in the document. For those browsing with JavaScript disabled, or using a non JavaScript capable browser, they will see the message "`Sorry, you need a JavaScript capable browser to get the best from this page`" (The alternative, standard HTML content should go there)

The `<NOSCRIPT> ... </NOSCRIPT>` container can be placed either inside or outside the `<BODY>` element.

Dynamic Documents
VisualBasic Script

<EMBED> - Embedded Objects

See Also

NOTE : This element is currently the focus of the HTML working groups Compound Document draft specification which would extend its capabilities beyond those described here.

The <EMBED> element allows you to put objects directly into an HTML page.

The syntax is:

```
<EMBED SRC="object_to_embed">
```

The <EMBED> element will allow you to embed documents of any type. Your user only needs to have an application which can view the data installed correctly on their machine, or have a plug-in (for **Netscape** and **Internet Explorer**) that can manipulate the embedded file format.

The <EMBED> element can be used more or less the same as the <IMG> element and so accepts typical image embedding attributes such as **WIDTH**, **HEIGHT**, **BORDER**, **HSPACE**, **VSPACE**.

Netscape Plug-ins

Netscapes plug-ins makes use of the <EMBED> element. Essentially, plug-ins are dynamic code modules which are associated with a MIME data type that the Netscape client has no native support for. When Netscape encounters an unknown data type from a server, it will search for a plug-in that is associated with that MIME type and load it, enabling viewing/transforming of the data object.

Netscape now comes (if the standard version is downloaded) with some plug-ins pre-installed allowing the embedding of video, audio, VRML worlds and quicktime movies. The syntax for including any of these formats is as above.

For more information about plug-ins, see the Netscape web site. http://home.netscape.com/. Lots of plug-in modules are available, the additional attributes for which are beyond the scope of this document. The documentation with the plug-in should be consulted for useable HTML element attributes.

NOTE : Microsofts **Internet Explorer** (from version 3.0 beta 2) fully supports Netscape plug-ins. When **Internet Explorer** loads a page designed to use a (**Netscape**) plug-in module, it will use either the correct plug-in, or a pre-installed ActiveX control to display the content as authored. Microsoft recently released the ActiveVRML and ActiveMovie add-ons for **Internet Explorer** which can handle VRML worlds, AVI, QuickTime, MPEG video/audio, WAV, Sun AU and AIFF sound files respectively, either using the respective ActiveX <OBJECT> code, or <EMBED> code as used by popular **Netscape** plug-ins.

Dynamic Documents
`<OBJECT>`

Visual Basic Script

See Also

Visual Basic Script recently introduced by Microsoft represents a further step towards active web pages. Visual Basic Script provides scripting, automation and customisation capabilities for **Internet Explorer**. It is a subset of the Visual Basic programming language that is fully compatible with Visual Basic and Visual Basic for Applications.

To use Visual Basic Script within a HTML document, the code needs to be wrapped in `<SCRIPT> ... </SCRIPT>` elements, just like Netscape's JavaScript. As with JavaScript, the **LANGUAGE** attribute is required, in this case needing the value `"VBScript"`. Visual Basic Script comes into its own when used in conjunction with ActiveX OLE controls, which allow for full automation with any OLE compliant application, but can be used for almost any purpose on a page, allowing for truly interactive web sites to be created easily.

E.g. The following code section, assumes that a button, named 'btnHello' has been created somewhere on the HTML document.

```
<SCRIPT LANGUAGE="VBScript">
<!-- These comment delimiters ensure the code is hidden from those
     browsers that do not support Visual Basic Script
Sub btnHello_OnClick
   MsgBox "Hello, It's a fine day"
End Sub
-->
</SCRIPT>
```

The button, called 'btnHello' responds to being clicked by displaying a message box with the text "Hello, It's a fine day".

As with JavaScript, a complete description of Visual Basic Script is well outside the scope of this reference and you are encouraged to visit http://www.microsoft.com/vbscript/ for more information and complete (and more recent) language reference documentation.

Dynamic Documents
Java Script
ActiveX technology
<OBJECT>

Microsoft ActiveX technology

See Also

Microsoft's ActiveX technology, recently announced and supported by **Internet Explorer 3.0** (for Windows 95/NT) only, represents a huge advance in the capabilities of the **Internet Explorer**. ActiveX has relaxed the OLE control requirements to practically nothing. While previous OLE controls (such as the .OCX files shipped with Visual Basic) contained a lot of baggage inappropriate to use on the internet, new ActiveX controls (conforming to the re-designed control requirements specification) can be a lot more streamlined, facilitating the easy production of high quality dynamic content for HTML documents.

Internet Explorer 3.0 allows for the use of ActiveX controls, active scripts (such as Visual Basic Script) and active documents. (**NOTE :** The embedding mechanism, using the `<OBJECT>` element has been designed in coalition with the W3C, a technical report of which can be found at http://www.w3.org/pub/www/TR/WD-object.html) ActiveX can be used encapsulate practically any application or applet, for use within HTML documents.

The specific method of construction of ActiveX controls is outside the scope of this reference. Users interested in the use of ActiveX components are encouraged to visit the Microsoft Internet Explorer web site at http://www.microsoft.com/ie/

There, Microsoft has made available a number of ActiveX controls, which can be readily downloaded and used within HTML documents. Also, the ActiveX development kit can be downloaded, this kit contains many ActiveX/ActiveX Script/Visual Basic Script examples as well as all the specifications necessary for ActiveX control authoring.

The Ncompass plug-in for Netscape allows embedded ActiveX controls to work within **Netscape**.

Dynamic Documents
VisualBasic Script
JavaScript
Examples
ActiveX / Visual Basic Script examples
`<OBJECT>`

<OBJECT> ... </OBJECT>

See Also

NOTE : The <OBJECT> element is currently only supported by **Internet Explorer**. The object insertion mechanism is the subject of a W3C Working draft available at http://www.w3.org/pub/WWW/TR/WD-object.html. For this and other W3C working drafts, you should visit the W3C site at http://www.w3.org/pub/World Wide Web/TR/

The <OBJECT> element provides a way for the ActiveX controls and other media to be embedded directly into HTML documents. It subsumes the role of the <IMG> element, providing an insertion mechanism for media other than static images. As far as the Internet Explorer is concerned, the <OBJECT> element (at present) is used for the inclusion of ActiveX controls.

An example of the syntax is as follows:

```
<OBJECT CLASSID="clsid:99B42120-6EC7-11CF-A6C7-00AA00A47DD2"
  ID=lbl1
  WIDTH=150
  HEIGHT=90>
  <PARAM NAME="angle" VALUE="30" >
  <PARAM NAME="alignment" VALUE="2" >
  <PARAM NAME="BackStyle" VALUE="0" >
  <PARAM NAME="caption" VALUE="Hello there">
  <PARAM NAME="FontName" VALUE="Arial">
  <PARAM NAME="FontSize" VALUE="20">
  <PARAM NAME="FontBold" VALUE="1">
  <PARAM NAME="frcolor" VALUE="8421376">
</OBJECT>
```

(The above example inserts a 'label' ActiveX control into the page.)

The object being inserted into the HTML document in this case, is referred to by its **CLASSID**. This is a unique identifier for the control, according to the Component Object Model "class id" URL scheme. The **ID** attribute identifies the specific label with a unique name, allowing interaction with and dynamic updating of the object's properties via active OLE scripting (i.e. Visual Basic Script).

Where the ActiveX control that is to be inserted may not be present on the users system, the automatic download mechanism of ActiveX controls (using the **CODEBASE** attribute) can be employed. This attribute can be used to provide a location from which the control will be downloaded.

For example :

```
<OBJECT CLASSID="clsid:99B42120-6EC7-11CF-A6C7-00AA00A47DD2"
  CODEBASE=”http://www.mysite.com/controls/mycontrol.ocx”
  ID=lbl1
  WIDTH=90
...
```

would look within the system registry to see if the control with the given **CLASSID** is present on the system and if not, the control will be retrieved from the URL given by the **CODEBASE** attribute. Full version checking can also be employed using this mechanism.

For example :

```
<OBJECT CLASSID="clsid:99B42120-6EC7-11CF-A6C7-00AA00A47DD2"
  CODEBASE="http://www.mysite.com/controls/mycontrol.ocx#ver=3,10,0,1000"
  ID=lbl1
  WIDTH=90
...
```

would only download the control to the users system if the version present on the system is older than that given by the #ver setting. I.e. if it is older than version 3.10.0.1000.

Some objects will require certain code to implement the object. This should be referenced by using the **CODE** attribute. Also, the **DATA** attribute can be used to point to a persistent data stream to initialise the object's state. The use of the above attributes is control dependant, so exhaustive examples cannot be given.

In keeping with the role of the <OBJECT> element as a media insertion element (c.f. <IMG>) various standard formatting attributes, such as **HEIGHT**, **WIDTH**, **ALIGN BORDER**, **HSPACE**, **VSPACE** etc. can also be used to define the positioning of the object on the page.

The **PARAM** element allows a list of named property values (used to initialise a control, plug-in module or Java applet) to be represented as a sequence of PARAM elements. Note that PARAM is an empty element and should appear without an end tag (i.e. no </PARAM>. The **NAME** attribute defines the property to be defined and the **VALUE** attribute defines the property value. For instance, in the above example, the line:

```
<PARAM NAME="caption" VALUE="Hello there">
```

sets the value of the property 'Caption' to be "Hello there" (In this case, this property represents the text that will be displayed for the label.) Object properties are entirely control dependant, so the reference documentation for any control should be read to find out what properties can be set using the PARAM element.

Microsoft recently released the **ActiveX Control Pad**, a text editor with added <OBJECT> insertion capabilities. Using this, authoring HTML documents that contain ActiveX components is a breeze, as control insertion/property setting is done through a Visual Basic like forms interface (the ActiveX Control Pad has been jokingly referred to as Visual Notepad). The Control Pad also allows authoring of fixed-layout sections, viewable using the **ActiveX HTML Layout Control** (available from Microsoft). This allows for presentation of fixed content, where the content will display as authored (something which can't be guaranteed with normal HTML. Users interested in authoring 'Activated' HTML documents, then download the ActiveX Control Pad from the Microsoft Web site at http://www.microsoft.com/workshop/author/cpad/.

Dynamic Documents
ActiveX technology
ActiveX / Visual Basic Script examples
VisualBasic Script
JavaScript
<EMBED>

ActiveX / Visual Basic Script Examples

As the ActiveX controls available from Microsoft keep changing (and their number increases), the topics previously topics contained in this section have been removed, so as not to provide confusing/outdated information. Instead, what I have done is included (embedded within this HLP file) a fully working Web site that extensively uses ActiveX/Visual Basic Script (and Style Sheets). Indeed, the archive contained within this HLP file is a complete copy of the HTMLib web site (as of the time of release of this version of the HTMLib). Those wishing to pursue the use of ActiveX controls and Visual Basic Script are welcome to go through the HTML documents here, in order to see how things have been done.

To extract the archive file from the HLP file, **click here**.

Once the archive has been extracted (it will have been placed in the directory that this HLP file is in), use a suitable .ZIP file extraction tool (i.e. PKUNZIP, or WinZip) to extract the files contained into a separate directory and then load the file index.html into **Internet Explorer**.

NOTE : This web ‘site’ ***requires*** that you use **Internet Explorer** (at least 3.0 beta 1) to view it. Also, some controls will need to be installed. The necessary controls are detailed in the files. Some editing of the file hlp.htm is also required for the Activated HTMLib portion of the site to work properly on a local machine. Skim through the file until you see the section highlighted by rows of ***** characters (it's the **sub cmdForward** routine) for instructions.

Quick Reference

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

Exit

<!--...
<!--#...
<!DOCTYPE ...>

<A ...>
<ADDRESS>
<APPLET ...>
<AREA ...>

<B>
<BASE ...>
<BASEFONT ...>
<BGSOUND ...>
<BIG>
<BLINK>
<BLOCKQUOTE>
<BODY>
<BR>

<CAPTION>
<CENTER>
<CITE>
<CODE>
<COL>
<COLGROUP>
<COMMENT>

<DD>
<DFN>
<DIR>
<DIV>
<DL>
<DT>

<EM>
<EMBED ...>

<FONT ...>
<FORM>
<FRAME ...>
<FRAMESET ...>

<H ALIGN= ...>
<H1>
<H2>
<H3>
<H4>
<H5>
<H6>
<HEAD>
<HR>
<HTML>

<I>
<IFRAME>
<IMG ...>
<INPUT>
<ISINDEX>

<KBD>

<LI>
<LINK ...>
<LISTING>

<MAP ...>
<MARQUEE ...>
<MENU>
<META ...>
<MULTICOL>

<NEXTID ...>
<NOBR>
<NOFRAMES>
<NOSCRIPT>

<OBJECT>
<OL>
<OPTION>

<P>
<PLAINTEXT>
<PRE>

<S>
<SAMP>
<SCRIPT ...>
<SELECT>
<SMALL>
<SOUND ...>
<STRIKE>
<STRONG>
<SPACER>
<SPAN>
<STYLE>
<SUB>
<SUP>

<TABLE ...>
<TBODY>
<TD>
<TEXTAREA>
<TFOOT>
<TH>
<THEAD>
<TITLE>
<TR>
<TT>

<U>
<UL>

<VAR>

<WBR>

<XMP>

<BODY ...>

...ALINK

...BACKGROUND

...BGCOLOR

...BGPROPERTIES

...LEFTMARGIN

...LINK

...TEXT

...TOPMARGIN

...VLINK

<A ...>

- ...HREF
- Mailto : ...TITLE
- ...METHODS
- ...NAME
- ...REL
- ...REV
- ...TARGET
- ...TITLE
- ...URN

<APPLET ...>

- ...ALIGN
- ...ALT
- ...CODEBASE
- ...CODE
- ...NAME
- ...PARAM NAME/VALUE
- ...VSPACE/HSPACE
- ...WIDTH/HEIGHT

<FRAME...>

- ...FRAMEBORDER
- ...FRAMESPACING
- ...MARGINWIDTH
- ...MARGINHEIGHT
- ...NAME
- ...NORESIZE
- ...SCROLLING
- ...SRC
- ...BORDERCOLOR

WIDTH, HEIGHT HSPACE, VSPACE and ALIGN are also supported when employing floating frames.

<FRAMESET...>

- ...COLS
- ...ROWS
- ...FRAMEBORDER
- ...BORDER
- ...BORDERCOLOR

<HR...>

...ALIGN

...COLOR

...NOSHADE

...SIZE

...WIDTH

<DL...>

- <DT>
- <DD>
- ...COMPACT

<OL...>
   ...START
   ...TYPE
   ...VALUE

<IMG ...>

- ...ALIGN
- ...ALT
- ...BORDER
- ...DYNSRC
- ...HEIGHT
- ...HSPACE
- ...ISMAP
- ...LOWSRC
- ...SRC
- ...USEMAP
- ...VRML
- ...VSPACE
- ...WIDTH

- <FONT...>
    - ...COLOR
    - ...FACE
    - ...SIZE

<BASEFONT...>

...COLOR

...FACE

...SIZE

<AREA>

...COORDS

...SHAPE

<MAP...>
   ...NAME

<META ...>

- ...HTTP-EQUIV
- ...CONTENT
- ...NAME

```
<INPUT...>
   ...ALIGN
   ...CHECKED
   ...MAXLENGTH
   ...NAME
   ...SIZE
   ...SRC
   ...TYPE :
      CHECKBOX; HIDDEN; IMAGE;
      PASSWORD; RADIO; RESET;
      SUBMIT; TEXT; TEXTAREA; FILE
   ...VALUE
```

<TEXTAREA...>

- ...COLS
- ...NAME
- ...ROWS
- ...WRAP

- <TABLE...>
  - ...ALIGN
  - ...BACKGROUND
  - ...BGCOLOR
  - ...BORDER
  - ...BORDERCOLOR
  - ...BORDERCOLORLIGHT
  - ...BORDERCOLORDARK
  - ...CELLSPACING
  - ...CELLPADDING
  - ...FRAME
  - ...HEIGHT
  - ...RULES
  - ...VALIGN
  - ...WIDTH

<TR...>
- ...ALIGN
- ...BGCOLOR
- ...BORDERCOLOR
- ...BORDERCOLORLIGHT
- ...BORDERCOLORDARK
- ...VALIGN

<TD...>

- ...ALIGN
- ...BACKGROUND
- ...BGCOLOR
- ...BORDERCOLOR
- ...BORDERCOLORLIGHT
- ...BORDERCOLORDARK
- ...COLSPAN
- ...HEIGHT
- ...NOWRAP
- ...ROWSPAN
- ...VALIGN
- ...WIDTH

<TH...>

- ...ALIGN
- ...BACKGROUND
- ...BGCOLOR
- ...BORDERCOLOR
- ...BORDERCOLORLIGHT
- ...BORDERCOLORDARK
- ...ROWSPAN
- ...COLSPAN
- ...HEIGHT
- ...NOWRAP
- ...VALIGN
- ...WIDTH

<SCRIPT...>

- ...LANGUAGE
- ...SRC

<MARQUEE...>

- ...ALIGN
- ...BEHAVIOR
- ...BGCOLOR
- ...DIRECTION
- ...HEIGHT
- ...HSPACE
- ...WIDTH
- ...LOOP
- ...SCROLLAMOUNT
- ...SCROLLDELAY
- ...VSPACE

<COL>
...ALIGN
...SPAN

<COLGROUP>
   ...ALIGN
   ...SPAN
   ...VALIGN

<MULTICOL>

- ...COLS
- ...GUTTER
- ...WIDTH

<SPACER>

- ...ALIGN
- ...HEIGHT
- ...SIZE
- ...WIDTH

Contacting the Author

After gathering together information about the specification of HTML (level 2.0 and draft versions of the HTML 3.2 and *'Cougar'* (the next level above 3.2) available at the time of writing) I realised that no concise (useable) reference existed (apart from the (Internet Engineering Task Force) IETF HTML-working group Internet Drafts and RFC documents), so I decided to construct a HTML reference guide.

This is said reference.

While I take credit (and criticism) for the actual construction/layout of this reference, I cannot take any responsibility for its original content. The original HTML document type was designed by **Tim Berners-Lee** at CERN in 1990. In 1992, **Dan Connolly** wrote the HTML Document Type Definition (DTD) and a brief HTML specification. Since 1993, a wide variety of people have contributed to the evolution of the HTML specification and in 1994, Dan Connolly and Karen Olson Muldrow rewrote the HTML Specification. A complete list of those contributors can be found in the HTML specification, which is now written by **Dave Raggett** and can be obtained from :

http://www-uk.hpl.hp.co.uk/people/dsr/

It should be noted that, while this index of HTML elements was complete at the time of writing (covering all HTML elements supported by commonly available browsers), the World Wide Web *initiative* is evolving at an exponential rate, hence this reference can only be considered a *work in progress*, parallel to the actual specifications themselves. For more information, the archives of the IETF HTML working group discussions and the current HTML 3.0 specification should be consulted at the W3 Consortium web site:

http://www.w3.org/

Therefore, while I am willing to accept comments and criticisms of the document, I cannot take responsibility for the status of the specification contained within. I will however, endeavour to keep up with the specifications and update this document when necessary.

Netscape is a registered trademark of Netscape Communications Corp. Updated information about any of the Netscape products mentioned in this reference (Netscape, JavaScript, Plug-in information etc.) can always be found at the Netscape Web site ; http://www.netscape.com.

Windows, Windows95, ActiveX, ActiveMovie, ActiveVRML, Visual Basic, Visual Basic Script, Internet Assistant and Internet Explorer are either registered trademarks or trademarks of Microsoft Corp. in the United States and/or other countries.

Portions reprinted with permission granted by Microsoft Corporation.

Updated information about any of the Microsoft products described in this reference (Internet Explorer, ActiveX, ActiveMovie, Windows95, Virtual Explorer, Internet Assistant, Visual Basic Script etc.) can always be found at the Microsoft Corporation Web site ; http://www.microsoft.com.

The Server Side Includes section of this document is taken from information kindly provided by Mark West of Questar Microsystems Inc. Updated SSI+ information can be obtained from http://www.questar.com/ssi/

Java and HotJava are trademarks of Sun Microsystems in the US and/or other countries.

NOTE : The whereabouts of the most recent version of this reference can always be found by going to the HTMLib web site at http://subnet.virtual-pc.com/~le387818/ or by mailing me at **cmlehunt@swan.ac.uk**. Normally, the most recent version can be found att ftp.swan.ac.uk in the directory pub/in.coming/htmlib. A mailing list is also kept of those people who have expressed an interest in the reference and will be used to notify them of future releases. To subscribe to the mailing list, send mail to me at **cmlehunt@swan.ac.uk**. and let me know you want to be included. ***NOTE :*** This is not an autoreplier (its me) nor is the mailing list a general HTML mailing list. Basically, it is a list I maintain to do bulk mailings of release notes and any messages about updated information. Users wishing to join HTML discussion lists should consider the HTML Writers Guild (at http://www.hwg.org/) or Microsoft (http://www.microsoft.com/) for their respective HTML discussion lists.

Acknowledgements

Following is a list of people (in no particular order) without whom, the HTML Reference Library would either not have existed, or wouldn't be what it is today. To these people, I wish to extend my eternal gratitude. I'll also apologise to anybody who feels they should be mentioned here, but aren't - I'm not excluding anybody, I'll just have forgotten.

Thanks to

Hayley
Nick
Nona (huge thanks), Jeff and Gale at Microsoft
Hans big thanks for loads of recommendations.
Steve at Sausage Software, Frank at FLFSoft, Adam at SoftQuad, Art at IST and Todd at American Cybernetics.
Mark at Sams.
Mark West at Questar Microsystems
Cub Lea. For valuable WinHelp related advice.
Steve Pruitt at RhoTech(Microsoft) for his tireless help on the WinHelp mailing list and particularly for advice about the Activated HTMLib portion of the HTMLib Web Site.
Christian for his help/advice about the HTMLib Web Site.

Everybody that stores the HTMLib for me
Everybody who has mailed me with comments

Java - Adding an Applet

Attributes See Also

> ***NOTE :*** The information provided here only describes the necessary HTML elements that allow pre-compiled HotJava applets to be added to your HTML documents. It does not describe how to actually write HotJava applets. Such information is well beyond the scope of this document. More information can be found about writing HotJava applets and how to obtain a copy of the HotJava browser from :

http://java.sun.com/

Also, the information provided here is relevant for the beta-1 development version of HotJava and the Java development kit. Existing applets (developed under the 1.0 alpha 3 version) can be *upgraded* to this new version. More details can be found at the Sun MicroSystems Web site (URL above).

To add an applet to an HTML page, you need to use the `<APPLET>` HTML element.

```
<APPLET CODE="MyApplet.class" WIDTH=100 HEIGHT=140>
</APPLET>
```

This tells the viewer or browser to load the applet whose compiled code is in MyApplet.class (in the same directory as the current HTML document), and to set the initial size of the applet to 100 pixels wide and 140 pixels high.

Below is a more complex example of an `APPLET` element:

```
<APPLET CODEBASE="http://java.sun.com/JDK-prebeta1/applets/NervousText" CODE="NervousText.class"
  width=400 height=75 align=center >
<PARAM NAME="text" VALUE="This is the Applet Viewer.">
<BLOCKQUOTE>
<HR>
If you were using a Java(tm)-enabled browser, you would see dancing text instead of this paragraph.
<HR>
</BLOCKQUOTE>
</APPLET>
```

This tells the viewer or browser to load the applet whose compiled code is at the URL http://java.sun.com/JDK-prebeta1/applets/NervousText/NervousText.class, to set the initial size of the applet to 400x75 pixels, and to align the applet in the centre of the line. The viewer/browser must also set the applet's "text" attribute (which customises the text this applet displays) to be "This is the Applet Viewer." If the page is viewed by a browser that can't execute applets written in the Java Programming Language, then the browser will ignore the `APPLET` and `PARAM` elements, displaying the HTML between the `<BLOCKQUOTE>` and `</BLOCKQUOTE>` elements.
Java-enabled browsers *ignore* that HTML.

Here's the complete syntax for the `APPLET` element:

```
<APPLET
  [CODEBASE = codebaseURL]
  CODE = appletFile
  [ALT = alternateText]
  [NAME = appletInstanceName]
  WIDTH = pixels HEIGHT = pixels
  [ALIGN = alignment]
  [VSPACE = pixels] [HSPACE = pixels]
>
```

```
    [<PARAM NAME = appletAttribute1 VALUE = value>]
    [<PARAM NAME = appletAttribute2 VALUE = value>]
   . . .
    [alternateHTML]
</APPLET>
```

CODEBASE = *codebaseURL*

This optional attribute specifies the base URL of the applet -- the directory that contains the applet's code. If this attribute is not specified, then the document's URL is used.

CODE = *appletFile*

This required attribute gives the name of the file that contains the applet's compiled Applet subclass. This file is relative to the base URL of the applet. It cannot be absolute.

ALT = *alternateText*

This optional attribute specifies any text that should be displayed if the browser understands the `APPLET` element but can't run applets written in the Java(tm) Programming Language.

NAME = *appletInstanceName*

This optional attribute specifies a name for the applet instance, which makes it possible for applets on the same page to find (and communicate with) each other.

WIDTH = *pixels* **HEIGHT** = *pixels*

These required attributes give the initial width and height (in pixels) of the applet display area, not counting any windows or dialogs that the applet brings up.

ALIGN = *alignment*

This required attribute specifies the alignment of the applet. The possible values of this attribute are the same as those for the `IMG` element: left, right, top, texttop, middle, absmiddle, baseline, bottom, absbottom.

VSPACE = *pixels* **HSPACE** = *pixels*

These option attributes specify the number of pixels above and below the applet (`VSPACE`) and on each side of the applet (`HSPACE`). They're treated the same way as the IMG element's `VSPACE` and `HSPACE` attributes.

<**PARAM NAME** = *appletAttribute1* **VALUE** = *value*>

This element is the only way to specify an applet-specific attribute. Applets access their attributes with the getParameter() method.

The following attributes are used in the `<APPLET>` element for the inclusion of HotJava applets in HTML documents.

- `ALIGN`
- `ALT`
- `CODEBASE`
- `CODE`
- `NAME`
- `PARAM NAME/VALUE`
- `VSPACE/HSPACE`
- `WIDTH/HEIGHT`

Dynamic Documents
JavaScript

Frames - Advanced page formatting

Attributes See Also

NOTE : The use of Frames is currently **only** supported by recent versions of the Netscape Navigator (from version 2.0) and Internet Explorer (3.0 and above). Frames can be authored according to specific style sheet mechanisms detailed in the W3C tehcnical report *"Frame based layout via Style Sheets"*. Users wishing to author styled frame layouts are encouraged to obtain this report from the W3C technical report archive at http://www.w3.org.pub/WWW/TR/.

Frames extend the layout flexibility of web pages by allowing the visible client area to be divided into more than one sub-region. Each sub-region, or frame, has several properties:

1) It can load a URL, independently of the other frames.
2) It can be given a NAME, allowing it to be targeted by other URL's
3) It resizes itself dynamically in response to changes in the size of the visible client area, and it can choose to allow or disallow itself to be manually resized by the user.

Netscape supports standard fixed frame documents, while the **Internet Explorer** also supports the use of floating frames.

Frames are generated by three things: FRAMESET tags, FRAME tags, and FRAME Documents.

Frame Document

A Frame document has a basic structure very much like your normal HTML document, except the BODY container is replaced by a FRAMESET container which describes the sub-HTML documents, or Frames, that will make up the page.

```
<HTML>
<HEAD>
</HEAD>
<FRAMESET>

</FRAMESET>
</HTML>
```

Frame Syntax

Frame syntax is similar in scope and complexity to that used by tables, and has been designed to be quickly processed by browsers

<FRAMESET>

This is the main container for a Frame. A standard frame document has no BODY, and no tags that would normally be placed in the BODY can appear before the FRAMESET tag, or the FRAMESET will be ignored (unless the frame is a *floating* frame, supported by **Internet Explorer** only). The FRAMESET tag has a matching end tag, and within the FRAMESET you can only have other nested FRAMESET tags, FRAME tags, or the NOFRAMES tag.

ROWS="row_height_value_list"

The ROWS attribute takes as its value a comma separated list of values. These values can be absolute pixel values, percentage values between 1 and 100, or relative scaling values. The number of rows is implicit in the number of elements in the list. Since the total height of all the rows must equal the height of the window, row heights might be normalised to achieve this. A missing ROWS attribute is interpreted as a single row arbitrarily sized to fit.

Syntax of value list.

`value`
A simple numeric value is assumed to be a fixed size in pixels. This is the most dangerous type of value to use since the size of the viewer's window can and does vary substantially. If fixed pixel values are used, it will almost certainly be necessary to mix them with one or more of the relative size values described below. Otherwise the client engine will likely override your specified pixel value to ensure that the total proportions of the frame are 100% of the width and height of the user's window.

`value%`
This is a simple percentage value between 1 and 100. If the total is greater than 100 all percentages are scaled down. If the total is less than 100, and relative-sized frames exist, extra space will be given to them. If there are no relative-sized frames, all percentages will be scaled up to match a total of 100%.

`value*`
The value on this field is optional. A single '*' character is a "relative-sized" frame and is interpreted as a request to give the frame all remaining space. If there exist multiple relative-sized frames, the remaining space is divided evenly among them. If there is a value in front of the '*', that frame gets that much more relative space. "2*,*" would give 2/3 of the space to the first frame, and 1/3 to the second.

Example for 3 rows, the first and the last being smaller than the centre row:

```
<FRAMESET ROWS="20%,60%,20%">
```

Example for 3 rows, the first and the last being fixed height, with the remaining space assigned to the middle row:

```
<FRAMESET ROWS="100,*,100">
```

COLS="column_width_list"
The `COLS` attribute takes as its value a comma separated list of values that is of the exact same syntax as the list described above for the `ROWS` attribute.

The `FRAMESET` tag can be nested inside other `FRAMESET` tags. In this case the complete subframe is placed in the space that would be used for the corresponding frame if this had been a `FRAME` tag instead of a nested `FRAMESET`.

BORDER
Netscape allows the global setting of frame border thicknesses by using this attribute within the `<FRAMESET>` element. It accepts a pixel value, which determines the thickness of any borders used within the frame set.

BORDERCOLOR
This **Netscape** specific attribute sets the colours for the border of the specified frameset. It can also be used in the `<FRAME>` element (see below) for setting the border colours of a specific frame. It accepts any #rrggbb hex triplet as a value. Any `BORDERCOLOR` setting in a `<FRAMESET>` element is over-ridden by a setting present in the `<FRAME>` element.

FRAMEBORDER
Netscape allows the use of this attribute (in a similar fashion to that supported by **Internet Explorer** in the `<FRAME>` element) to set the borders globally for an entire `<FRAMESET>`. Values can be either "yes" or "no".

<FRAME>
This tag defines a single frame in a frameset. It has 6 possible attributes: `SRC`, `NAME`, `MARGINWIDTH`, `MARGINHEIGHT`, `SCROLLING`, and `NORESIZE`. The `FRAME` tag is not a container so it has no matching end tag.

SRC="url"

The SRC attribute takes as its value the URL of the document to be displayed in this particular frame. FRAMEs without SRC attributes are displayed as a blank space the size the frame would have been.

NAME="window_name"

The NAME attribute is used to assign a name to a frame so it can be targeted by links in other documents (These are usually from other frames in the same document.) The NAME attribute is optional; by default all windows are unnamed.

Names must begin with an alphanumeric character. However, several reserved names have been defined, which start with an underscore.

These are currently:

_blank	Always load this link into a new, unnamed window.
_self	Always load this link over yourself.
_parent	Always load this link over your parent. (becomes self if you have no parent).
_top	Always load this link at the top level. (becomes self if you are at the top).

NOTE : Although these are reserved names for the NAME attribute of the <FRAME> element, they should only be referred to using an Anchor Target. That is, used to target specific windows, allowing smoother transition between framed documents and between framed and *normal* documents.

MARGINWIDTH="value"

The MARGINWIDTH attribute is used when the document author wants some control of the margins for this frame. If specified, the value for MARGINWIDTH is in pixels. Margins can not be less than one-so that frame objects will not touch frame edges-and can not be specified so that there is no space for the document contents. The MARGINWIDTH attribute is optional; by default, all frames default to letting the browser decide on an appropriate margin width.

MARGINHEIGHT="value"

The MARGINHEIGHT attribute is just like MARGINWIDTH above, except it controls the upper and lower margins instead of the left and right margins.

SCROLLING="yes|no|auto"

The SCROLLING attribute is used to describe if the frame should have a scrollbar or not. Yes results in scrollbars always being visible on that frame. No results in scrollbars never being visible. Auto instructs the browser to decide whether scrollbars are needed, and place them where necessary. The SCROLLING attribute is optional; the default value is auto.

NORESIZE

The NORESIZE attribute has no value. It is a flag that indicates that the frame is not resizable by the user. Users typically resize frames by dragging a frame edge to a new position. Note that if any frame adjacent to an edge is not resizable, that entire edge will be restricted from moving. This will effect the resizability of other frames. The NORESIZE attribute is optional; by default all frames are resizable.

FRAMEBORDER

This attribute allows control of the frame border display. With this attribute set to "0" (**Internet Explorer**), the borders for the specific frame are not drawn. **Netscape** now also supports use of this attribute (using values of "yes|no") and also supports it in the <FRAMESET> element (see above) for globally setting the borders of a whole frame set. ***NOTE :*** In **Netscape** frames share borders and for the borders to not be drawn, **all** the frames sharing a common border must have their **FRAMEBORDER** attribute set to "no"

FRAMESPACING="value"

This attribute is also **Internet Explorer** specific and allows the setting of extra space around frames, to give the appearance of floating frames. The "value" should be the distance required around the frame in pixels.

i.e. `<FRAME FRAMESPACING="55" ...>`
would present the frame with a spacing of 55 pixels.

BORDERCOLOR

This **Netscape** specific attribute sets the colours for the border of the specified frame. It can also be used in the `<FRAMESET>` element (see above) for globally setting the border colours of a whole frameset. It accepts any #rrggbb hex triplet as a value. Setting the `BORDERCOLOR` attribute in the `<FRAME>` element over-rides any setting given in the `<FRAMESET>` element, of which the `<FRAME>` is a part.

<NOFRAMES>

This tag is for content providers who want to create alternative content that is viewable by non-Frame-capable clients. A Frame-capable Internet client ignores all tags and data between start and end `NOFRAMES` tags.

Internet Explorer floating frames.

The **Internet Explorer** (version 3 and above) has introduced the concept of floating frames. These are much like standard frames, except they can be anywhere within a standard HTML document. A floating frame must be enclosed within `<IFRAME> ... </IFRAME>` elements. Any HTML between the start and end elements will be displayed by browsers that do not support floating frames, while the frame specified in the `<IFRAME>` elements will be displayed by those browsers that do.

For example :

```
<IFRAME NAME="content_frame" width="488" height="244" SRC="welcome.htm">This site
  uses floating frames</IFRAME>
```

The above HTML fragment would display the file `"content_frame"` in **Internet Explorer**, but would display the text `This site uses floating frames` to any other browser.

The other main difference between floating and normal frames, is that floating frames accept **WIDTH** and **HEIGHT** and standard `<IMG>` **HSPACE**, **VSPACE** and **ALIGN** attributes, to set the display size and alignment of the frame within the document. They can have other files loaded into them exactly as normal frames (by use of the `TARGET` attribute in the link and the `NAME` attribute of the frame). For an example of floating frame use, see the frames examples, or go to the ActiveX/Visual Basic Script examples page and read about the example web site enclosed with this version of the HTMLib.

Frames are controlled by the following elements and their attributes:

- <FRAMESET>
  - COLS
  - ROWS
  - BORDER
  - BORDERCOLOR
  - FRAMEBORDER
- <FRAME>
  - FRAMEBORDER
  - FRAMESPACING
  - MARGINWIDTH
  - MARGINHEIGHT
  - NAME
  - NORESIZE
  - SCROLLING
  - SRC
  - BORDERCOLOR
- <NOFRAMES>
- <IFRAME>

A Frames example

Frames examples

See Also

For the purposes of this example of standard frames (see below for a floating frames example), a portion of this HLP reference was formatted into HTML, using frames.

The example actually consists of six different HTML documents. These are detailed below:

1) The main frame set up document
2) The title document
3) The contents document
4) The main text document
5) The navigation buttons document
6) The HTML language document

Documents 2, 4, 5 and 6 have no MarkUp relevant to the use of Frames and have only been included here for reasons of completeness.

To see how the document displays on a frames compliant browser (i.e. Netscape 2.0, Internet Explorer 3.0 and above), click the camera:

📷

Floating frames

The **Internet Explorer** allows *floating* frames to be used within standard documents. To do this, it only requires that a frame be employed within a standard HTML document. For example, for the main interface to the 'Activated HTMLib' section of the HTMLib web site, a floating frame is specified for all of the subsequent HTML documents (both framed and normal) to be displayed in.

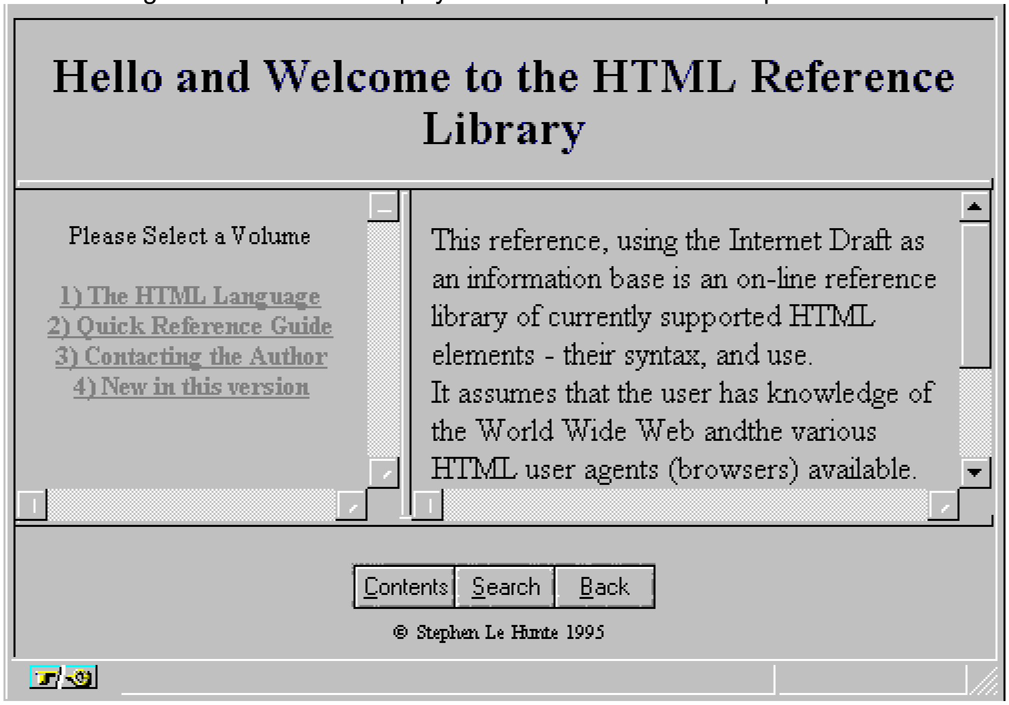
The HTML used to do this is :

```
<IFRAME NAME="content_frame" width="488" height="244" SRC="welcome.htm"></IFRAME>
```

This sets the frame as being called `"content_frame"` and being initially loaded with the file `"welcome.htm"`. The `WIDTH` and `HEIGHT` attributes serve to set the size of the frame within the page. View the screenshot to see the effect. Care must be taken when specifying the `WIDTH` and `HEIGHT` attributes in absolute pixel amounts as you can not guarantee the screen resolution of the users system.

📷

The resulting frame document displays like this on a frames compliant browser:



Hello and Welcome to the HTML Reference Library

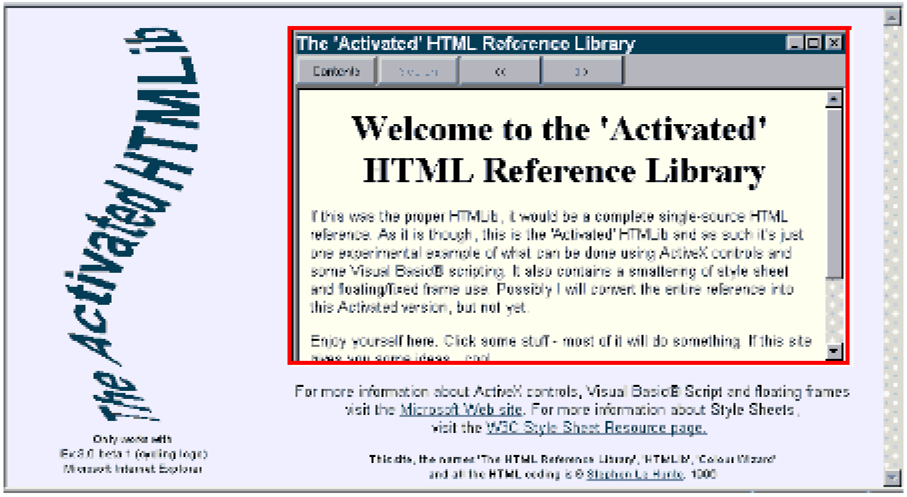
Please Select a Volume

1) The HTML Language
2) Quick Reference Guide
3) Contacting the Author
4) New in this version

This reference, using the Internet Draft as an information base is an on-line reference library of currently supported HTML elements - their syntax, and use.
It assumes that the user has knowledge of the World Wide Web andthe various HTML user agents (browsers) available.

© Stephen Le Hunte 1995

The 'Activated HTMLib' web site looks like this:



Welcome to the 'Activated' HTML Reference Library

If this was the proper HTMLib, it would be a complete single-source HTML reference. As it is though, this is the 'Activated' HTMLib and as such it's just one experimental example of what can be done using ActiveX controls and some Visual Basic® scripting. It also contains a smattering of style sheet and floating/fixed frame use. Possibly I will convert the entire reference into this Activated version, but not yet.

Enjoy yourself here. Click some stuff - most of it will do something. If this site gives you some ideas... cool

Only works with
IE3.0 beta 1 (swirling logo)
Microsoft Internet Explorer

For more information about ActiveX controls, Visual Basic® Script and floating frames visit the Microsoft Web site. For more information about Style Sheets, visit the W3C Style Sheet Resource page.

This site, the names 'The HTML Reference Library', 'HTMLib', 'Colour Wizard' and all the HTML coding is © Stephen Le Hunte, 1996

The floating frame is highlighted with a red border.   All the other pages, as loaded by the contents menu, or the browse buttons, are displayed in this frame.

The main frame set up document

See Also

The main document that sets up the example frame is as follows:

```
<HTML>
<!--HTMLIB.HTM-- >
 <HEAD>
 <TITLE>The HTML Reference Library</TITLE>
 </HEAD>
<BASEFONT SIZE=3>

<FRAMESET ROWS="85,*,65">
 <FRAME SCROLLING="no" NAME="title" NORESIZE SRC="title.htm">
 <FRAMESET COLS="40%,60%">
  <FRAME SCROLLING="yes" NAME="toc" SRC="toc.htm">
  <FRAME SCROLLING="yes" NAME="main page" SRC="main.htm">
 </FRAMESET>
 <FRAME SCROLLING="no" NAME="HLP buttons" NORESIZE SRC="buttons.htm">

<NOFRAME>

</NOFRAME>
</FRAMESET>
```

A line by line breakdown

```
<FRAMESET ROWS="85,*,65">
```

This line divides the page into three regions, the top region being 85 pixels in height, the bottom region being 65 pixels in height, the middle region occupying the rest of the Netscape window.

```
<FRAME SCROLLING="no" NAME="title" NORESIZE SRC="title.htm">
```

This line sets the top region of the window (the region that is 85 pixels high) to be a non-scrolling, non-resizable region. Its name is title (so, any other link that specifies "title" with its `TARGET` attribute would be displayed in this region.)

```
<FRAMESET COLS="40%,60%">
```

This splits the middle region of the Netscape window into two sections horizontally. The left hand section being 40% of the window width, the right hand section being the remaining 60% of the window width.

```
<FRAME SCROLLING="yes" NAME="toc" SRC="toc.htm">
<FRAME SCROLLING="yes" NAME="main page" SRC="main.htm">
```

These two lines (as the other `FRAME` line above) set the attributes for the two middle sections of the page. That is, it names the regions "toc" and "main page" respectively and links to the two pages to be displayed in the regions.

```
</FRAMESET>
```

This line closes the sub-frames that were opened in the middle section of the main framed regions.

```
<FRAME SCROLLING="no" NAME="HLP buttons" NORESIZE SRC="buttons.htm">
```

This line defines the properties of the remaining main region of the window - namely the bottom region, 65 pixels high. It defines it as a non-scrolling, non-resizable region (ideal for navigation tools)

The use of the `NAME` attribute in setting up the page regions is so that each different page can have linked pages sent to it by name. This way, where the document called (when a link is activated) is displayed can be controlled by the author.

NOTE : Here, no information has been provided between the `<NOFRAME>` … `</NOFRAME>` elements. This is where the author must put information that s/he wishes to be displayed on browsers that do not support the use of frames.

📷

The title document

See Also

NOTE : This document contains no mark-up relevant to the use of the frames, but has been included for reasons of completeness.

This document is the Title for the paged document.   It resides in the top frame, which is a non-scrolling non-resizeable frame.   Hence the title will always be displayed in the same place.   Note that for frame sub-documents titles are not required.   The title of the site will always be taken from the main frame page.

```
<HTML>
<!--TITLE.HTM-- >
<BODY>
<BASEFONT SIZE=3>
<CENTER>
<H2 ALIGN=center>Hello and Welcome to the HTML Reference Library</H2>
<BR>
</CENTER>
</BODY>
</HTML>
```

📷

The contents document

See Also

This is the Table of Contents page. It appears on the left scrolling frame region. This section has been used (in this example) for a stationary table of contents.

```
<HTML>
<!--TOC.HTM-- >
<BODY>
<BASEFONT SIZE=2>
<CENTER>
Please Select a Volume<BR><BR>
 <A HREF="lang.htm" TARGET="main page"><B>1) The HTML Language</B></A><BR>
 <A HREF="qr.htm" TARGET="main page"><B>2) Quick Reference Guide</B></A><BR>
 <A HREF="author.htm" TARGET="main page"><B>3) Contacting the Author</B></A><BR>
 <A HREF="new.htm" TARGET="main page"><B>4) New in this version</B></A><BR>
</CENTER>
</BODY>
</HTML>
```

The use of the `TARGET` attribute in the anchor means that when each link is activated the document accessed will be displayed in the frame region named "main page". Thus, any documents accessed from the table of contents will appear in the framed region to the right of the table of contents.

The main text document

See Also

NOTE : This document contains no mark-up relevant to the use of the frames, but has been included for reasons of completeness.

This document is the document that appears in the right hand framed region of the page the first time the page is accessed.

```
<HTML>
<!--MAIN.HTM-- >
<BODY>
 This reference, using the Internet Draft as an information base is an on-line
 reference library of currently supported HTML elements - their syntax, and
 use.<BR>
 It assumes that the user has knowledge of the World Wide Web and the various HTML
 user agents (browsers) available.  Information on specific browsers, or the
 broader topic of 'The World Wide Web' can be obtained by reading the World Wide
 Web FAQ.<BR>
</BODY>
</HTML>
```

The navigation buttons document

See Also

NOTE : This document contains no mark-up relevant to the use of the frames, but has been included for reasons of completeness.

This document resides at the bottom of the framed document.   This region is a non-scrollable non-resizable region.   As such, is ideal for a set of navigation buttons or other tools, as these could be.   For the purposes of this example, the buttons are just a graphic image.

```
<HTML>
<!--BUTTONS.HTM-- >
<BODY>
<CENTER>
 <IMG SRC="buttons.jpg"><BR>
 <FONT SIZE=1>&copy; Stephen Le Hunte 1995</FONT>
</CENTER>
</BODY>
</HTML>
```

The HTML language document

See Also

NOTE : This document contains no mark-up relevant to the use of the frames, but has been included for reasons of completeness.

This document is accessed from choosing the first option from the table of contents. When accessed, it would be displayed in the right hand section of the middle regions.

```
<HTML>
<!--LANG.HTM-- >
<BODY>
<CENTER><B>The HTML Language</B></CENTER>
<BR>
 The vast range of HTML MarkUp currently supported by available HTML user agents
 (Web browsers, such as Netscape, Mosaic etc.) can be divided into the following
 sections.  Some elements featured here, may not be supported by all browsers.
 Where an element is known to be supported by specific browsers, the element
 description will be labelled as such.<BR>
</BODY>
</HTML>
```

Frames - Advanced page formatting

Document Sound

NOTE : Two different elements now exist for employing in-line sound in a HTML document. The first is BGSOUND, this element is currently only supported by the Microsoft **Internet Explorer**. The other is SOUND, which is currently only supported by **NCSA Mosaic**. Mosaic does also support a limited version of Microsoft's BGSOUND element. (see below)

The new **BGSOUND** tag allows you to create pages with background sounds or "soundtracks." Sounds can either be samples (.WAV or .AU format) or MIDI (.MID format).

The HTML used to insert a background sound into a page is:

```
<BGSOUND SRC="start.wav">
```

The attributes associated with the BGSOUND element are SRC and LOOP.

SRC
This attribute specifies the address of a sound to be played.

LOOP
This attribute specifies how many times a sound will loop when activated. If n=-1 or LOOP=INFINITE is specified, the sound will loop indefinitely.

Examples
```
<BGSOUND SRC="boing.wav">
```
This would play the specified WAV file as soon as Internet Explorer has downloaded the file.

```
<BGSOUND SRC="boing.wav" LOOP=INFINITE>
```
This would play the specified WAV file as soon as it has finished being loaded and would continuously play the file until another page is loaded.

NCSA Mosaic supports use of the **SOUND** element for playing in-line sound. This element allows the playing of *.WAV files in pages. ***NOTE :*** The SOUND element is only supported by **Mosaic**

The syntax is:

```
<SOUND SRC="filename.wav">
```

This element is not currently supported by the HTML working group, although the Mosaic authors believe it will be included. Inline sound files can be placed in any part of the document and Mosaic will act on the sound file when its position is visible to the document view window. Sound files can implemented as a background sounds using the **LOOP** attribute. An HTML author can also delay the play of an inline sound for *x* number of seconds using the **DELAY** attribute.

The Attributes of the sound tag are:

LOOP=infinite and DELAY=sec.

Examples:

```
<SOUND SRC="*.wav" LOOP=infinite>
<SOUND SRC="*.wav" DELAY=10>
```

NOTE : Although Mosaic will support the use of the BGSOUND element, it will not play in-line *.MID

MIDI files. For this, it will launch an external application. It will play *.WAV files using the BGSOUND element though. **Netscape** supports in-line document sounds by using the standard audio plug-in, using the <EMBED> element.

New in this version

Previous versions

The HTMLib now has a support Web site at http://subnet.virtual-pc.com/~le387818/ at which news about new releases, updated information etc., will be posted.

The biggest inclusion in this version of the HTMLib is Style Sheet information. Microsofts **Internet Explorer** 3.0 has introduced support for the much vaunted and very powerful presentation mechanism. Mainly it employs the new `<STYLE>` and `<SPAN>` elements (and the `STYLE`, `CLASS` and `ID` attributes) and also introduces some new attributes to the `<LINK>` element.

New **Netscape** **`BORDER`**, **`FRAMEBORDER`** and **`BORDERCOLOR`** attributes to the `<FRAME>` and `<FRAMESET>` elements. New syntax for Microsofts floating frames implementation in **Internet Explorer** - `<IFRAME>`

Support for **Netscape**s new `<MULTICOL>` and `<SPACER>` elements.

An all-new Colour Wizard is included with this release of the HTMLib. This new version is significantly smaller and offers more ways of choosing the document colours. The code can be exported as style sheet code (***NOTE :*** it sets the `TEXT` colour by using the color attribute in the `BODY` style declaration, not by defining a `P` style. This means that all body text that is not otherwise styled will be of the colour defined in the colour wizard).

The HTMLib Tips 'n' Tricks Reference is also included with this version of the HTMLib. This is an attempt to provide some solutions to common HTML questions posed in the various newsgroups/mailng lists. While being separate to the actual HTMLib, the TNT reference is closely linked to the HTMLib. I am anticipating updating this reference more often than the whole HTMLib - check the HTMLib web site for updates.

Loads of tidying up (a never ending job) has been done as well.

More reference links have been added (any dark blue coloured link within a topic body is a hyperlink to a Web site) and users are encouraged to obtain any standards/information documents referenced within the HTMLib. Automatic mailto: links have also now been employed, to save the arduous task of copying and pasting any e-mail addresses into mailing software. (This only works if you have e-mail software installed and properly registered to handle e-mail shortcuts).

Version 2.2

In the wake of certain situations, the License terms for the HTMLib have been changed. Please take a moment to review the License.

Observant users will have noticed the inclusion of an executable within the HTMLib archive for this release. This is the HTMLib Colour Wizard which can be used to get an idea of what different colour schemes will look like when used in HTML documents.

There is a large amount of new information in this release of the HTMLib. All the new information pertaining to the Internet Explorer is based on the Alpha developers release of Internet Explorer and is subject to change.

New Internet Explorer `<FRAME ...>` attributes.

New Internet Explorer colouring of hard rules.

New Internet Explorer Table elements, `BACKGROUND,` `FRAME,` `RULES,` (background is supported in `<TH>` and `<TD>` as well.)

Internet Explorer now supports some HTML 3 table mark up : `<THEAD>`, `<TBODY>`, `<TFOOT>`, `<COLGROUP>`, `<COL>` elements.

Internet Explorer now supports `<BIG>` and `<SMALL>`

The Internet Explorer also now supports ActiveX technology and Visual Basic Script.

A large inclusion in this version of the HTML Reference Library, is the Comparison Table. This is a table which (like the Quick Reference) shows all the elements currently supported by the browsers that the HTMLib supports (i.e. Internet Explorer, Netscape and Mosaic), with an indication of what elements/attributes are supported by each browser. If it has a tick in the table, then it is supported. Where that support is conditional, it is indicated. The Comparison Table is always available, by choosing its option from the HTMLib menu, or on the context menu (right mouse click). The Quick Reference button has also been removed and an option added to both the HTMLib and context menu's.

New elements/attributes.

FACE and COLOR attributes for the BASEFONT element (Internet Explorer specific)

Netscape now supports floating tables, by using the ALIGN attribute for <TABLE> elements.

Netscape now supports 140 discrete colour names for use in the <BODY BGCOLOR="..."> or <FONT COLOR="..."> Markup.

Descriptions of the <XMP>, <LISTING>, <COMMENT> and <PLAINTEXT> elements, previously missing from the HTMLib have been added.

The usual amount of typo/formatting corrections have also been done.

Version 2.1

This version of the HTML Reference Library contains mostly updated information and aesthetic/functional changes. There is a lot of new information though.

The Quick Reference section is now displayed in a separate window. This allows the Quick Reference section to be kept open for as long as required. It is now accessed through the **Quick Ref.** Button on the button bar at the top of the main window, or by pressing Alt+Q. Also, in the Quick Reference section, with elements that accept many attributes that don't fit into the initial topic display, a list of the attributes are displayed in a pop-up box from the Quick Reference entry, allowing quicker targeting of desired attribute information.

Here are the element/attribute changes/additions detailed in this version.

Reserved names for the <FRAME> element.

<APPLET> element replaces the previous <APP> element.

Netscape JavaScript elements.

Netscapes plug-ins. Description and some detail of a VRML plug-in module.

Server Side Includes (from the SSI+ 1.0 specification).

LEFTMARGIN and TOPMARGIN attributes for the <BODY> element - Allow margin setting for the Internet Explorer.

BGCOLOR, BORDERCOLOR, BORDERCOLORLIGHT and BORDERCOLORDARK attributes for flexible colouring of tables using the Internet Explorer. There are also many new alignment options for various <TABLE> related elements, supported by the new Internet Explorer.

Embedding of VRML worlds (and ActiveVRML animated objects) supported by Microsofts Virtual Explorer.

Version 2.0

This version of the HTML Reference Library contained probably the most new information of any release. This section was an addition. Here are quick links to the elements or attributes that were new to version 2.0 (released just after Netscape 2.0, Internet Explorer 2.0 and Mosaic 2.0)

New Elements

In-line video support for Internet Explorer
Document sound capabilities of Internet Explorer and Mosaic
Highlighted scrolling Marquee text - Internet Explorer
HotJava applet inclusion - Netscape
Frames - Advanced page formatting.
The <DIV> element. Text alignment

New Attributes

Watermarking now available in Internet Explorer
Floating tables
Colouring of tables
Client Pull is now supported by Internet Explorer
Targetted windows and target setting of base pages
Font colouring now supported by Netscape
Netscape now supports Client Side Image Maps
Wrapping the text in a TEXTAREA box on a form is now available in Netscape

Version 2.2
Version 2.1
Version 2.0

mailto : **cmlehunt@swan.ac.uk**

HTMLib - Licence

For the purposes of this document, the name 'HTMLib' refers to the 'HTML Reference Library' package (including the HTMLib Colour Wizard). The 'HTMLib package' consists of the original archive file in which the HTMLib is contained. The 'author' is Stephen Le Hunte, who can be contacted at **cmlehunt@swan.ac.uk**

Distribution terms.

You are free to distribute the HTMLib package on any public Web/FTP/Gopher site without necessarily obtaining the authors permission. (Notification will be appreciated). (Corporate/ISP administrators wishing to install the HTMLib for multiple users, see the 'Registration' section below) If you wish to distribute the HTMLib on or with ANY OTHER MEDIA (i.e. if you want to include the HTMLib on a magazine cover CD, or on a CD accompanying a book, or possibly with other software, or any distribution situation similar to these) then the EXPRESS PERMISSION of the author is REQUIRED. The HTMLib is NOT allowed to be distributed as part of any larger package (such as those described above) if the permission of the author has not been granted.
The HTMLib Colour Wizard is NOT to be distributed outside of the total HTMLib package.

Registration.

The 'HTMLib' is totally free for personal and private use. However, if you feel that the 'HTMLib' is worth money, or you have made money as a result of using the HTMLib and wish to register your approval financially, then contact me to arrange payment. (It should be noted that there is no formal registration/support deal with the HTMLib. Essentially, if you are willing to pay then I will not turn down any payments.)

If you are responsible for a company/ISP etc. network and you wish to install the HTMLib for multiple users, then you should consider some payment as a reflection of the time that the HTMLib may save users within the company. Contact me if you are in this position.

It should be noted that each different release of the 'HTMLib' package from now on should be treated as a completely separate product. Any agreements made for previous versions of the HTMLib should be agreed upon again. Contact the author for details.

The HTMLib Colour Wizard is not installed.

It appears that the Colour Wizard is not installed on this system.   It should reside in the same directory as the HLP file.   It may be wise to re-install the download archive of the HTMLib.

Style Sheets

See Also

NOTE : Presented here is a brief overview of the Style Sheet specification. For a more complete (and possibly new) reference, visit the Style Sheet resource page at the W3C (at http://www.w3.org/pub/WWW/Style/). Also, **Microsoft** have produced an excellent "Users guide to stylesheets" document avaiable from the Microsoft *Web Development Workshop* web site at http://www.microsoft.com/workshop/

Style Sheets allow the HTML author to separate presentation definitions from content in HTML documents. HTML was designed primarily as a content based mark up language and the introduction of various text level formatting elements and attributes only served to confuse the issue. Microsoft have led the way by implementing style sheet support in **Internet Explorer** 3.0 (at the time of writing, this was the only common browser supporting style sheets).

Anyone who does a lot of work using a word processor should be familiar with the concept of style sheets. Basically, styles are defined and then applied to blocks of text (or even single characters) by referring to the definition in the available style range.

The simplest form of style sheet could be:

```
P {color : #800000}
```

This would cause any text that is classified as a paragraph (i.e. uses a `<P>` element) to be rendered using a dark red colour. For a breakdown of the possible properties for use in style sheets, see the Properties and Values topic.

Applying a style sheet to a document

Style definitions can be applied to documents and their elements in on of four ways.

Using the `<LINK>` attribute to point to an external style sheet. (See `<LINK>` for details)

Using the `<STYLE>` element within the `<HEAD>` of a document. (See `<STYLE>` for details)

Using the @import mechanism (similar to the `<LINK>` method above, this allows the importing of external style sheets.)

Using the `STYLE` attribute in an element. This is allowed for any elements allowed within the `<BODY>` of a HTML document.

The fourth option above, while allowed, is not recommended as it mixes style definitions with content, which goes against the style sheet concept. An example would be :

```
<H1 STYLE="color : #FF0000">Heading 1</H1>
```

There are various methods for denoting the style declaration and all use separate methods within the HTML to determine the style to be used :

`CLASS` as selector

The `CLASS` attribute was first proposed in the now expired HTML 3 specification. All elements that can reside within the `<BODY>` of a HTML document can be addressed using a `CLASS` attribute.

Within the style sheet definition, the style class is set thus:

```
P.redtext { color : #FF0000}
```

So, any paragraphs denoted by a `<P CLASS="redtext">text</P>` HTML sequence would be rendered using a red coloured text. If the style class is defined with no major element, i.e. :

```
.redtext { color : #FF0000}
```

then all elements that use `CLASS="redtext"` would be coloured red.

`ID` as selector

Like the `CLASS` attribute above, the `ID` attribute was first introduced in the HTML 3 specification. The difference between it and the `CLASS` attribute is that the `ID` attribute must be guaranteed a unique identifier within the document. It must also be addressed within the style sheet preceded with a # character. I.e.:

```
#redtext { color : #FF0000}
```

`<P ID="redtext">some text</P>` would be coloured red.

Context sensitive selectors

Suppose that you wanted to make all text that was defined as strong within a paragraph be rendered using a lime green colour. You could use :

```
STRONG { color : #00FF00}
```

in the style sheet definition. However, this would cause all text that was enclosed in `<STRONG> ... </STRONG>` elements to be rendered lime green. This is where context sensitive selectors are useful. If you use :

```
P STRONG { color : #00FF00}
```

in the style sheet definition, then only text enclosed in `<STRONG> ... </STRONG>` elements enclosed in `<P> ... </P>` elements would be lime green. For instance :

```
<P><STRONG>This will be lime green</STRONG></P> This <STRONG>other bit</STRONG>
  won't be.
```

As with the two methods of style definition above, this can be used for all elements normally contained within the body of a document.

Hopefully, this has provided enough starter information about style sheets. Users interested in employing style sheet presentation to their documents should obtain the Cascading Style Sheet specification from the W3C Style Sheet Resource page for more detailed information.

`<LINK>`
`<STYLE> ... </STYLE>`
`<SPAN> ... </SPAN>`
Properties and Values
Style Sheet Examples

Style Sheet Properties and Values

NOTE : The information presented here is abridged from the style sheet specification (along the lines of the properties quick reference guide at the W3C web site (http://www.w3.org/pub/WWW/Style/)) For more detailed information, consult the Style Sheet specification.

The Style Sheet properties can be split into the following groups :
Font properties
Colour and background properties
Text properties
Box properties
Classification properties
Also, some information about units
Length units
Percentage units
Colour units
URL

Style Sheet Font properties

Properties See Also

font-size
Value: xx-small | x-small | small | medium | large | x-large | xx-large | <number> | <length> | <percentage>
Initial: medium
Applies to: all elements
Inherited: yes
Percentage values: relative to parent's font size

The keyword values can be used to specify a font-size that will be determined by the browser, according to an internal table. <number> specifies an absolute font size, or a font-size relative to the elements parent (using '+' or '-' settings).'

For the 'font-size' property' <length> units (e.g. 'em' and 'ex'), refer to the font size of the parent element.

font-family
Value: [[<family-name> | <generic-family>],]* [<family-name> | <generic-family>]
Initial: UA specific
Applies to: all elements
Inherited: yes
Percentage values: N/A

The value is a prioritized list of font family names and/or generic family names. Values are separated by a comma to indicate that they are alternatives:

```
BODY { font-family: gill, helvetica, sans-serif }
```

There are two types of list values:

<font-family>
The name of a font family of choice. In the last example, "gill" and "helvetica" are font families.

<generic-family>
In the example above, the last value is a generic family name. The following generic families are defined:

'serif' (e.g. Times)
'sans-serif' (e.g. Helvetica)
'cursive' (e.g. Zapf-Chancery)
'fantasy' (e.g. Western)
'monospace' (e.g. Courier)

It is useful to offer a generic font family as a last alternative, in case the specified font is not available on the viewers system.

Font names containing white space should be quoted:

```
BODY { font-family: "new century schoolbook", serif }
<BODY STYLE="font-family: 'My own font', fantasy">
```

font-weight
Value: extra-light | light | demi-light | medium | demi-bold | bold | extra-bold | bolder | lighter
Initial: medium

Applies to: all elements
Inherited: yes
Percentage values: N/A

The keywords can either be absolute or relative:
For example :

```
P { font-weight: medium }
STRONG { font-weight: bolder }
```

This would lead to `'STRONG'` elements being *'bolder'* than their parent. If the parent is 'medium', a `'STRONG'` element will have the value of 'bold'. I.e., a relative value indicates a change of two positions relative to the list of absolute values. Child elements inherit the resultant weight, not the keyword value.

font-style
Value: normal | italic || small-caps | oblique || small-caps | small-caps
Initial: normal
Applies to: all elements
Inherited: yes
Percentage values: N/A

The keyword values can be combined. Legal combinations of the values are:

one of the four values ('normal', 'italic', 'oblique', 'small-caps')
'italic' or 'oblique', combined with 'small-caps'

As with other font properties, if the specified style cannot be achieved, the browser should select an approximation.

line-height
Value: <number> | <length> | <percentage>
Initial: UA specific (see below)
Applies to: block-level elements
Inherited: yes
Percentage values: refers to the font size of the element itself

This property sets the the distance between two adjacent lines' baselines. It only applies to block-level elements, (i.e. `<BLOCKQUOTE>`, `<ADDRESS>` etc.)

Negative values are not allowed.

font
Value: [<font-weight> || <font-style>]? <font-size> [/ <line-height>]? <font-family>
Initial: not defined
Applies to: all elements
Inherited: yes
Percentage values: allowed on <font-size> and <line-height> only

This property provides a wrapper for defining multiple font properties, using the properties described above). See the above properties for their allowed values. Setting of this property is equivalent to setting of the individual properties.

font-size
font-family
font-weight
font-style
line-height
font

Style Sheet Colour/Background properties

See Also

color
Value: <color>
Initial: UA specific
Applies to: all elements
Inherited: yes
Percentage values: N/A

This property describes the text colour of an element, i.e. the "foreground" colour. There are different ways to specify red:

```
EM { color: red }
EM { color: rgb(255,0,0) }
```

See the Colour Units sectionfor details on how to specify colours.

background
Value: transparent | <color> [/ <color>]? || <url> || <blend-direction> || <repeat> || <scroll> || <position>
Initial: transparent
Applies to: all elements
Inherited: no
Percentage values: N/A

This property describes the background of an element, i.e. the surface onto which the text content is displayed. The background can be transparent, one colour, two colours (bleneded in the direction specified by the <blend-direction> (see below) value, or an image, possibly combined with one or two colours.

For example :

```
P { background: transparent }
BODY { background: red }
H1 { background: blue / red }
BODY { background: url(chess.png) 50% repeat fixed }
```

This property does not inherit, but the parent element's background will shine through by default due to the initial transparent value. If neither an image or a colour is found, a value of 'transparent' is assumed.

If an image is specified and found through the URL, it will be overlaid on top of any colour specified. The colour (or colour combination) will be used to fill any transparency in the image, or to fill the image space while the image is loading.

<blend-direction>

If two colours have been specified, they will be blended according to the value of <blend-direction>. Legal values are [N | NW | W | SW | S | SE | E | NE]. If no value has been set, 'S' is assumed. The value specifies the direction of blending from the first colour into the second colour. The values are shorthands for north, north-west, west etc where 'N' is the top of the element's box, 'NW' is the upper left corner etc:

For example :

```
TABLE { background: blue/green NW }
```

If only one background colour is specified, <blend-direction> is ignored.

<repeat>

This determines the extent of repetition of any specified background image. Values are [repeat | repeat-x | repeat-y | no-repeat]. If no value is set, 'repeat' is assumed, i.e. the image is repeated both horizontally and vertically.

```
BODY { background: url(marble.png) repeat-x }
```

Will cause the background image to be repeated horizontally only.

<scroll>

This controls whether any specified background image should be fixed to the background, or allowed to scroll as the document is viewed. Possible values are [fixed | scroll], defaulting to 'scroll'.

<position>

<position> specifies the initial position of any specified bakcground image. Values are [<percentage> | left | center | right [<percentage> | top | middle | bottom]]. If no value is specified, '0% 0%' is assumed.

With a value pair of '0% 0%', the upper left corner of the image is placed in the upper left corner of the element. A value pair of '100% 100%' places the lower right corner of the image in the lower right corner of the element. With a value pair of '14% 84%', the point 14% across and 84% down the image is to be placed at the point 14% across and 84% down the element.

Specifiying only one value sets both the horizontal and vertical offset of the background image. If two values are given, the first position is used as the horizontal position.

If the background image is fixed with regard to the canvas (see the <scroll> value above), the image is placed relative to the canvas instead of the element. E.g.:

Style Sheet Text properties

Properties See Also

word-spacing
Value: normal | <length>
Initial: normal
Applies to: all elements
Inherited: yes
Percentage values: N/A

The length unit indicates an addition to the default space between words. Negative values can be set, but browsers may restrict the appearance of such settings. The specific spacing algorithm is also browser dependant and may be influenced by text alignment such as justification (see text-align below)
See the Length units topicfor details of specific lenght units.

letter-spacing
Value: normal | <length>
Initial: normal
Applies to: all elements
Inherited: yes
Percentage values: N/A

Similar to word spacing above, this property indicates an addition to the default space between characters. Also as above, negative values are allowed, the character spacing algorith is browser dependant and character spacing may be influenced by any text alignment properties set.

text-decoration
Value: none | [underline | overline | line-through | blink]+
Initial: none
Applies to: all elements
Inherited: no, but see clarification below
Percentage values: N/A

This property can be used to decorate text blocks. Any colours required for the decoration are inherited from the any color property value settings.

For example :

```
A:link, A:visited, A:active { text-decoration: underline }
```

Would underline all links within the document.

vertical-align
Value: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage>
Initial: baseline
Applies to: all elements
Inherited: yes
Percentage values: refer to the 'line-height' of the element itself

This property determines the vertical positioning of the element. The defined keywords set the alignment of the element realtive to the parent element.

baseline'
align the baseline of the element with the baseline of the parent
middle'

align the middle of the element with the baseline plus half the x-height of the parent
sub'
subscript the element
super'
superscript the element
text-top'
align the top of the element with the top of the parent element's font
text-bottom'
align the bottom of the element with the bottom of the parent element's font

The remaining values set the vertical alignment of the element relative to the formatted line that the element is a part of:

top'
align the top of the element with the tallest element on the line
bottom'
align the bottom of the element with the lowest element on the line

Percentage values refer to the 'line-height' of the element itself. E.g., a value of '-100%' will lower the element to where the baseline of the next line should have been.

text-transform
Value: capitalize | uppercase | lowercase | none
Initial: none
Applies to: all elements
Inherited: yes
Percentage values: N/A

capitalize'
uppercases the first character of each word
uppercase'
uppercases all letters of the element
lowercase'
lowercases all letters of the element
none'
neutralizes inherited value.

The transformation that actually occurs is browser dependant. Also, it may differ for different languages.

text-align
Value: left | right | center | justify
Initial: UA specific
Applies to: block-level elements
Inherited: yes
Percentage values: N/A

This property describes how text is aligned within the element. The actual justification algorithm used is UA and human language dependent. As above, implementation of this property is browser dependant and may differ for different languages.

text-indent
Value: <length> | <percentage>
Initial: 0
Applies to: block-level elements
Inherited: yes

Percentage values: refer to parent's width

This specifies the indent that should occur before the first formatted content lineand accepts a negative value.

word-spacing
letter-spacing
text-decoration
vertical-align
text-transform
text-align
text-indent

Style Sheet Box properties

Properties See Also

margin-left, margin-right, margin-top, margin-bottom, margin
Value: [<length> | <percentage> | auto]{1,4} (for 'margin' property)
Initial: 0
Applies to: all elements
Inherited: no
Percentage values: refer to parent's width

These properties set the margin of an element, while the 'margin' property can be used to set the border for all four sides. The other four properties only set their respective side.

For the 'margin' property, the four lengths apply to top, right, bottom and left respectively. If there is only one value, it applies to all sides, if there are two or three, the missing values are taken from the opposite side.

For example :

```
BODY { margin: 1em 2em 3em }
```

and

```
BODY {
 margin-top: 1em;
 margin-right: 2em;
 margin-bottom: 3em;
 margin-left: 2em
}
```

would produce the same result.

padding
Value: [<length> | <percentage> | auto]{1,4}
Initial: 0
Applies to: all elements
Inherited: no
Percentage values: refer to parent's width

This property describes how much space is inserted between the border and the content of the element. The order is top, right, bottom, left. A single value applies to all sides, while using two or three values set the respective sides, with the missing values being copied from the opposite side.

The surface of the padding area is set with the 'background' property.

border-top, border-right, border-bottom, border-left, border
Value: <border-width> || <border-style> || <url> || <color>
Initial: medium none
Applies to: all elements
Inherited: no
Percentage values: N/A

These properties set the border of an element. As with the above properties, the 'border' property sets the border for all four sides while the other properties only set their respective side.

The border is drawn in the image pointed to by the URL. If no URL is specified, or when the image is not available, the colour value is used.

If an image is specified and found through the URL, it will be used as a texture (i.e. repeated throughout the border) to draw the border. The <color> value, if specified, will be used to fill transparent portions of the image and while the image is loading

If no <color> value is expressly specified for the element border, the value of the 'color' property will take its place:

<border-width>
Possible values are [thin | medium | thick | <length>], defaulting to 'medium'.
The keyword widths are constant throughout a document:

<border-style>
Possible values are [none | dotted | dashed | solid | double | groove | ridge | inset | outset], defaulting to 'none'

width
Value: <length> | <percentage> | auto
Initial: auto
Applies to: all elements
Inherited: no
Percentage values: refer to parent's width

This property can be applied to text elements, but it is most useful with inline images and similar insertions. The width is to be enforced by scaling the image if necessary, preservong the aspect ratio of the image if the 'height' property is 'auto'.

height
Value: <length> | auto
Initial: auto
Applies to: block-level and replaced elements
Inherited: no
Percentage values: N/A

As above, this property can be applied to text, but it is most useful with inline images and similar insertions. The height is to be enforced by scaling the image if necessary, preserving the aspect if the 'width' property is 'auto'.

float
Value: left | right | none
Initial: none
Applies to: all elements
Inherited: no
Percentage values: N/A

This property is most often used with inline images, to allow floating images (c.f. `<IMG ALIGN ...>`

clear
Value: none | left | right | both
Initial: none
Applies to: all elements
Inherited: no
Percentage values: N/A

This property specifies if elements allow floating elements (normally images) to the left or right. With 'clear' set to 'left', an element will be moved below any floating element on the left side. (c.f. `<BR CLEAR= ...>`)

[margin]
padding
border/border-top/right/bottom/left
width
height
float
clear

Style Sheet Classification properties

See Also

These properties classify elements into categories rather than setting visual display styles

[display] [list-style][white-space]

display
Value: block | inline | list-item | none
Initial: according to HTML
Applies to: all elements
Inherited: no
Percentage values: N/A

This property indicates if an element is inline (e.g. 'EM' in HTML), block-level (e.g. 'H1' in HTML), or a block-level list item (e.g. 'LI' in HTML). For HTML documents, the initial value will be taken from the HTML specification.
A value of 'none' turns the display of the element, including children elements and the surrounding box, off.

list-style
Value: [disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none] || <url>
Initial: none
Applies to: elements with 'display' property value 'list-item'
Inherited: yes
Percentage values: N/A

This property can be set on any element, and it will inherit normally down the tree. However, the 'list-style' will only be displayed on elements with a 'display' value of 'list-item'. In HTML this is typically the case for the 'LI' element.

A URL value can be combined with any other value:
For example :

```
UL { list-style: url(http://png.com/ellipse.png) disc }
```

Will display unordered lists as using the image `ellipse.png`. If the image is unavailable, 'disc' will be used instead

white-space
Value: normal | pre
Initial: according to HTML
Applies to: block-level elements
Inherited: yes
Percentage values: N/A

Declares how white space inside the element should be handled: the 'normal' way (where white space is collapsed) or as the 'PRE' element in HTML.

Style Sheet Length units

See Also

The format of length values is an optional sign character ('+' or '-', with '+' being the default) immediately followed by a number (with or without a decimal point) immediately followed by a unit identifier (a two-letter abbreviation).

There are three types of length units: relative, pixel and absolute. Relative units specify a length relative to another length property.

The following relative units are supported: (***NOTE :*** The elements shown are used as an example, the settings can be applied to any element)

```
H1 { margin: 0.5em }      /* ems, the height of the element's font */
H1 { margin: 1ex    }     /* x-height, ~ the height of the letter 'x' */
P  { font-size: 12px }    /* pixels, relative to canvas */
```

Pixel units, as used in the last rule, are relative to the resolution of the screen display.

The following absolute units are supported: (***NOTE :*** The elements shown are used as an example, the settings can be applied to any element)

```
H1 { margin: 0.5in }        /* inches, 1in = 2.54cm */
H2 { line-height: 3cm }     /* centimeters */
H3 { word-spacing: 4mm }    /* millimeters */
H4 { font-size: 12pt }      /* points, 1pt = 1/72 in */
H4 { font-size: 1pc }       /* picas, 1pc = 12pt */
```

Style Sheet Percentage units

See Also

The format of a percentage value is a number (with or without a decimal point) immediately followed by '%'.   Percentage values are always relative to a length unit.   Each property that allows percentage units also define what length unit they refer to. Most often this is the font size of the element itself:

For example :

`P { line-height: 120% }` /* 120% of the element's 'font-size' */

Child elements inherit the resultant value, not the percentage value.

Style Sheet Colour units

See Also

A colour is a either a colour name or a numerical RGB specification. The Style Sheet specification (and style sheets as implemented by Microsoft in the **Internet Explorer**) supports all of the Netscape colour names.

The following all specify a red colour

```
EM { color: #F00 }              /* #RGB */
EM { color: #FF0000 }           /* #RRGGBB */
EM { color: rgb(255,0,0)        /* integer range 0 - 255 */
EM { color: rgb(100%, 0%, 0%)   /* float range 0.0% - 100.0% */
```

Note that a three-digit RGB notation (#RGB) is converted into six-digit form (#RRGGBB) by replicating digits, not by adding zeros. For example, #fb0 expands ro #ffbb00.

Values outside the numerical ranges will be rounded off. The three rules below are therefore equivalent:

```
EM { color: rgb(255,0,0)        /* integer range 0 - 255 */
EM { color: rgb(300,0,0)        /* rounded off to 255 */
EM { color: rgb(110%, 0%, 0%)   /* rounded off to 100% */
```

Style Sheet URL units

See Also

A Uniform Resource Locator (URL) is identified with a functional notation.   For example :

```
BODY { background: url(http://www.bg.com/pinkish.gif) }
```

Partial URLs are interpreted relative to the source of the style sheet, not relative to the document:

Style Sheets
Style Sheet properties and Values
Style Sheet Examples

Style Sheet Examples

See Also

For this example, consider the following document :

```
<HTML>
<HEAD>
  <TITLE>The HTML Reference Library</TITLE>
</HEAD>
<BODY>
<P>The HTML Reference Library is a Windows HLP file, detailing all currently
 useable HTML syntax.  It is available in the following formats :
<UL>
  <LI>Windows 3.x
  <LI>Windows 95/NT
</UL>
<P>For more information about the HTMLib, contact <A
 HREF="mailto:cmlehunt@swan.ac.uk">cmlehunt@swan.ac.uk</A>
</BODY>
</HTML>
```

As above, it appears dull and lifeless. The first simple thing that can be done with it is to use some different fonts for the sections of the page. For example :

```
<HTML>
<HEAD>
  <TITLE>The HTML Reference Library</TITLE>
<STYLE>
  P.main {font-family : Arial;
          font-size : 12pt}
  p.info { font-family : Arial;
           font-style : italic}
</STYLE>
</HEAD>
<BODY>
<P CLASS="main">The HTML Reference Library is a Windows HLP file, detailing all
 currently useable HTML syntax.  It is available in the following formats :
<UL>
  <LI>Windows 3.x
  <LI>Windows 95/NT
</UL>
<P CLASS="info">For more information about the HTMLib, contact <A
 HREF="mailto:cmlehunt@swan.ac.uk">cmlehunt@swan.ac.uk</A>
</BODY>
</HTML>
```

note the use of **CLASS** selectors to separate the different paragraphs and the way that the unordered list inherits the style attached to the previous paragraph..

This looks slightly more appealing, but would be better perhaps with some colour and highlighting of the link text and HTMLib references (note the way the highlighting, using the <SPAN> element inherits the colour of the paragraph in which it resides) :

```
<HTML>
<HEAD>
  <TITLE>The HTML Reference Library</TITLE>
<STYLE>
  BODY { background : #006000}
  P.main { font-family : Arial;
           font-size : 12pt;
           color : white}
  P.info { font-family : Arial;
           font-style : italic;
           color : #80C000}
  P SPAN { font-style : italic;
           font-size : 14pt}
  A { color : #C0C8FF;
      font-weight : bold}
</STYLE>
</HEAD>
<BODY>
<P CLASS="main">The <SPAN>HTML Reference Library</SPAN> is a Windows HLP file,
  detailing all currently useable HTML syntax.  It is available in the following
  formats :
<UL>
  <LI>Windows 3.x
  <LI>Windows 95/NT
</UL>
<P CLASS="info">For more information about the <SPAN>HTMLib</SPAN>, contact <A
  HREF="mailto:cmlehunt@swan.ac.uk">cmlehunt@swan.ac.uk</A>
</BODY>
</HTML>
```

The nest step in enhancing the presentation of the document is to force some aesthetically grand text over-lay effects. Adding the `text` and `highlight` **CLASS** descriptions into the style sheet definition defines two styles. By using the negative `top-margin` property in the `highlight` class, text specified as using this class can be over-laid on other text.

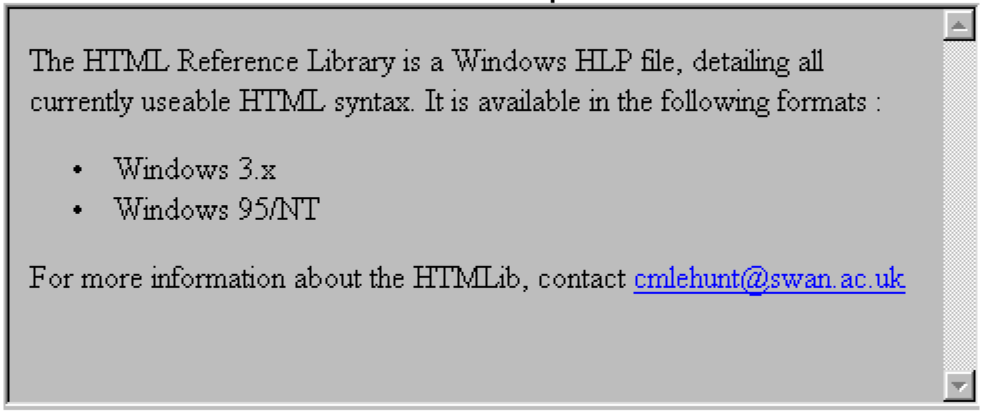
```
<HTML>
<HEAD>
  <TITLE>The HTML Reference Library</TITLE>
<STYLE>
  BODY { background : #006000}
  P.main { font-family : Arial;
           font-size : 12pt;
           color : white}
  P.info { font-family : Arial;
           font-style : italic;
           color : #80C000}
  P SPAN { font-style : italic;
           font-size : 14pt}
  A { color : #C0C8FF;
      font-weight : bold}
  .text { color: red;
          margin-left: 10px;
          font-size: 28px;
          font-family: Arial Black }
  .highlight { margin-top: -38px;
               margin-left: 8px;
               color: darkred;
               font-size: 28px;
               font-family: Arial Black }
</STYLE>
</HEAD>
<BODY>
<P CLASS="main"><DIV CLASS="text" STYLE="{color : white; font-style : italic}">The
  HTML Reference Library</DIV>
                <DIV CLASS="highlight" STYLE="{color : gray; font-style :
  italic}">The HTML Reference Library</DIV>
<P CLASS="main">is a Windows HLP file, detailing all currently useable HTML syntax.
  It is available in the following formats :

<UL>
  <LI><DIV CLASS="text">Windows 3.x</DIV>
      <DIV CLASS="highlight">Windows 3.x</DIV>
  <LI><DIV CLASS="text" STYLE="{color : blue"}>Windows 95/NT</DIV>
      <DIV CLASS="highlight" STYLE="{color : darkblue"}>Windows 95/NT</DIV>
</UL>
<P CLASS="info">For more information about the <SPAN>HTMLib</SPAN>, contact <A
  HREF="mailto:cmlehunt@swan.ac.uk">cmlehunt@swan.ac.uk</A>
</BODY>
</HTML>
```

Also note how the `text` and `highlight` class descriptions can have certain properties altered within the `<DIV>` element, by using the **STYLE** attribute. This means that the style can be described once and then have colours attached to it at the text level, rather than specifying new classes for the different coloured sections.

NOTE : The above is a (***very***) simple example of how a page can be transformed easily, using style sheet definitions in the `<STYLE>` elements in the `<HEAD>` of the document. Using style sheets makes changing the appearance of a page much easier as it no longer becomes necessary to dig through pages altering separate colouring attributes. This can be especially useful, if the style sheet is used externally to the documents and `<LINK>`ed to.

This is how the document looks in **Internet Explorer** 3.0 beta 2

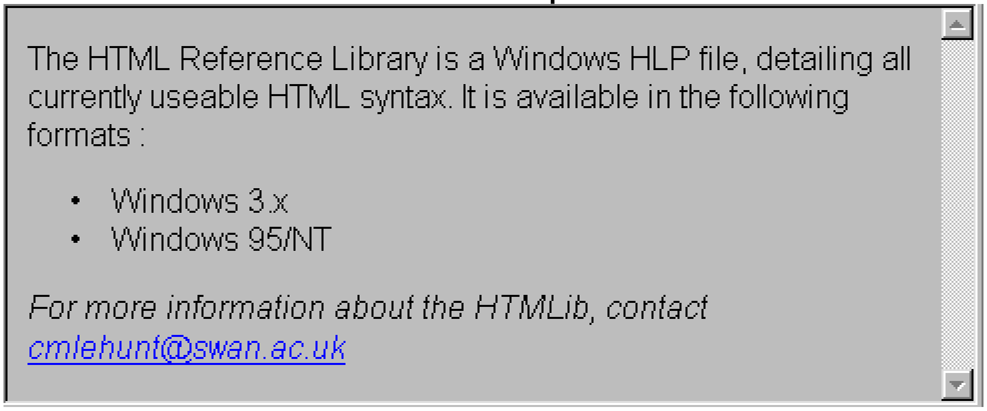


The HTML Reference Library is a Windows HLP file, detailing all currently useable HTML syntax. It is available in the following formats :

- Windows 3.x
- Windows 95/NT

For more information about the HTMLib, contact cmlehunt@swan.ac.uk

This is how the document looks in **Internet Explorer** 3.0 beta 2

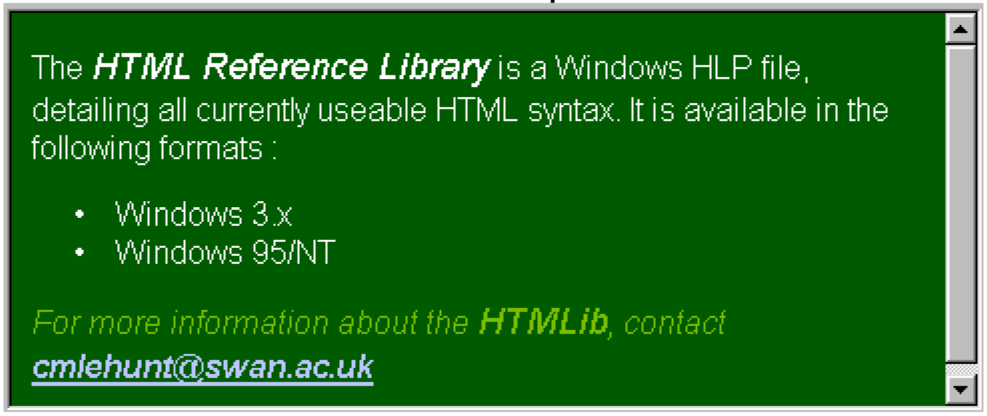


The HTML Reference Library is a Windows HLP file, detailing all currently useable HTML syntax. It is available in the following formats :

- Windows 3.x
- Windows 95/NT

For more information about the HTMLib, contact cmlehunt@swan.ac.uk

This is how the document looks in **Internet Explorer** 3.0 beta 2

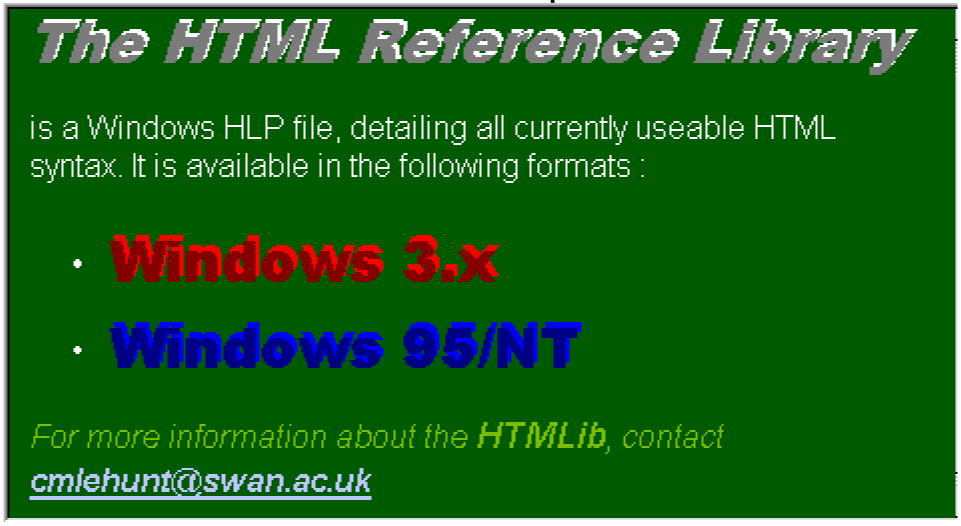


The ***HTML Reference Library*** is a Windows HLP file, detailing all currently useable HTML syntax. It is available in the following formats :

- Windows 3.x
- Windows 95/NT

*For more information about the **HTMLib**, contact*
cmlehunt@swan.ac.uk

This is how the document looks in **Internet Explorer** 3.0 beta 2



The HTML Reference Library

is a Windows HLP file, detailing all currently useable HTML syntax. It is available in the following formats :

- Windows 3.x
- Windows 95/NT

For more information about the ***HTMLib****, contact*

cmlehunt@swan.ac.uk

Comparison Table

📷
📷
📷

Element / Attribute

NOTE : All elements that can be used within the `<BODY>` of a HTML document can also take the **`CLASS`**, **`ID`** and **`STYLE`** attributes as well as those listed below. They have not been added for reasons of brevity and clarity. See the Style Sheets topic for more information. The browsers used for this comparison table (and all the HTMLib information) are :

Netscape Navigator version 3.0 beta 6 (32-bit version)
Internet Explorer version 3.0 beta 2 (4.70.1117 Windows 95/Windows NT 4 version)
NCSA Mosaic (version 2.1.1 Windows 95 version)

`<!-- ...`	✓	✓	✓
`<!DOCTYPE ...>`	✓	✓	✓
`<A ...>`	📷	📷	📷
`...HREF`	📷	📷	📷
`Mailto : ...TITLE`			📷
`...NAME`	📷	📷	📷
`...TITLE`	📷	📷	📷
`...REL`	📷	📷	📷
`...REV`	📷	📷	📷
`...URN`	📷	📷	📷
`...METHODS`	📷	📷	📷
`...TARGET`		📷	
`<ADDRESS>`	📷	📷	📷
`<APPLET ...>`	📷	📷	
`...CODEBASE`	📷	📷	
`...CODE`	📷	📷	

...ALT	📷	📷	
...NAME	📷	📷	
...WIDTH/HEIGHT	📷	📷	
...ALIGN	📷	📷	
...VSPACE/HSPACE	📷	📷	
...PARAM NAME/VALUE	📷	📷	
<AREA>	📷	📷	📷
...SHAPE	📷	📷	📷 RECT only
...COORDS	📷	📷	📷
<B>	📷	📷	📷
<BASE ...>	📷	📷	📷
...HREF	📷	📷	📷
...TARGET		📷	
<BASEFONT ...>	📷	📷	
...SIZE	📷 (only visible when FONT SIZE= used as well)	📷	
...FACE	📷		
...COLOR	📷		
<BGSOUND ...>	📷		📷 (will spawn player for .mid files)
...LOOP	📷		📷
...DELAY			📷

<BIG>	📷	📷	
<BLINK>		📷	
<BLOCKQUOTE>	📷	📷	📷
<BODY ...>	📷	📷	📷
...BACKGROUND	📷	📷	📷
...TEXT	📷	📷	📷 (Using color names is unreliable)
...LINK	📷	📷	📷
...VLINK	📷	📷	📷
...ALINK		📷	
...BGCOLOR	📷	📷	📷
...BGPROPERTIES	📷		
...LEFTMARGIN	📷		
...TOPMARGIN	📷		
<BR>	📷	📷	📷
...CLEAR	📷	📷	
<CAPTION>	📷	📷	📷
...ALIGN	📷 (top, bottom, left, right, center)	📷 (top, bottom)	📷 (top, bottom)
...VALIGN	📷 (top, bottom)		
<CENTER>	📷	📷	📷
<CITE>	📷	📷	📷

<CODE>	📷	📷	📷
<COL>	📷		
...SPAN	📷		
...ALIGN	📷		
<COLGROUP>	📷		
...SPAN	📷		
...ALIGN	📷		
...VALIGN	📷		
<COMMENT>	📷		📷
<DFN>	📷		
<DIR>	📷 (no bullet)	📷	📷
<DIV>	📷	📷	
...ALIGN	📷	📷 (left, right, center)	
<DL>	📷	📷	📷
<DT>	📷	📷	📷
<DD>	📷	📷	📷
...COMPACT		📷	
<DT>	📷	📷	📷
<EM>	📷	📷	📷
<EMBED ...>		📷	
<FONT ...>	📷	📷	
...SIZE	📷	📷	

...COLOR	📷	📷	
...FACE	📷	📷	
<FORM>	📷	📷	📷
<FRAME ...>	📷	📷	
...SRC	📷	📷	
...NAME	📷	📷	
...MARGINWIDTH	📷	📷	
...MARGINHEIGHT	📷	📷	
...SCROLLING	📷	📷	
...NORESIZE	📷	📷	
...BORDERCOLOR		📷	
...FRAMEBORDER	📷 "0" only for no borders	📷 "yes|no"	
...FRAMESPACING	📷		
...WIDTH	📷		
...HEIGHT	📷		
...HSPACE	📷		
...VSPACE	📷		
...ALIGN	📷		
<FRAMESET ...>	📷	📷	
...ROWS	📷	📷	
...COLS	📷	📷	
...BORDER		📷	
...FRAMEBORDER		📷	

...BORDERCOLOR		📷	
<H ALIGN= ...>	📷	📷	📷
<H1>	📷	📷	📷
<H2>	📷	📷	📷
<H3>	📷	📷	📷
<H4>	📷	📷	📷
<H5>	📷	📷	📷
<H6>	📷	📷	📷
<HEAD>	📷	📷	📷
<HR...>	📷	📷	📷
...SIZE	📷	📷	📷
...WIDTH	📷	📷	📷
...ALIGN	📷	📷	📷
...NOSHADE	📷	📷	📷
...COLOR	📷		
<HTML>	📷	📷	📷
<I>	📷	📷	📷
<IFRAME>	📷		
<IMG ...>	📷	📷	📷
...ALIGN	📷	📷	📷 (top, middle, bottom only)
...ALT	📷	📷	📷
...ISMAP	📷	📷	📷

...SRC	📷	📷	📷
...WIDTH	📷	📷	📷
...HEIGHT	📷	📷	📷 (image can't be distorted)
...BORDER	📷 (only when image is a link)	📷	
...VSPACE	📷	📷	
...HSPACE	📷	📷	
...LOWSRC		📷	
...USEMAP	📷	📷	📷
...VRML	📷		
<INPUT ...>	📷	📷	📷
...ALIGN	📷	📷	📷
...CHECKED	📷	📷	📷
...MAXLENGTH	📷	📷	📷
...NAME	📷	📷	📷
...SIZE	📷	📷	📷
...SRC	📷	📷	📷
...TYPE	📷	📷	📷
...VALUE	📷	📷	📷
<ISINDEX ...>	📷	📷	📷
...PROMPT	📷	📷	
<KBD>	📷	📷	📷

<LI>	📷	📷	📷
<LINK ...>	📷	📷	📷
<LISTING>	📷 will translate special characters.	📷 renders 132 characters to the line and translates special characters	📷
<MAP ...>	📷	📷	📷
...NAME	📷	📷	📷
<MARQUEE ...>	📷		
...ALIGN	📷		
...BEHAVIOR	📷		
...BGCOLOR	📷		
...DIRECTION	📷		
...HEIGHT	📷		
...WIDTH	📷		
...HSPACE	📷		
...LOOP	📷		
...SCROLLAMOUNT	📷		
...SCROLLDELAY	📷		
...VSPACE	📷		
<MENU>	📷 (no bullet)	📷	📷
<META ...>	📷	📷	📷
...HTTP-EQUIV	📷	📷	📷
...NAME	📷	📷	📷

...CONTENT	📷	📷	📷
<MULTICOL>		📷	
...COLS		📷	
...GUTTER		📷	
...WIDTH		📷	
<NEXTID ...>	📷	📷	📷
<NOBR>	📷	📷	
<NOFRAMES>	📷	📷	
<NOSCRIPT>		📷	
<OBJECT>	📷		
<PARAM>	📷		
<OL ...>	📷	📷	📷
...TYPE	📷	📷	
...START	📷	📷	
...VALUE	📷	📷	
<OPTION>	📷	📷	📷
<P>	📷	📷	📷
...ALIGN	📷 (center only)	📷 (left, right, center)	📷 (left, right, center)
<PLAINTEXT>	📷 allows closing element	📷	📷 allows closing element
<PRE>	📷	📷	📷
<S>	📷	📷	📷

<SAMP>	📷	📷	📷
<SCRIPT ...>	📷	📷	
...LANGUAGE	📷	📷	
...SRC	📷	📷	
<SELECT>	📷	📷	📷
<SMALL>	📷	📷	
<SOUND ...>			📷 (.wav only)
...SRC			📷
...DELAY			📷
<SPACER>		📷	
...ALIGN		📷	
...SIZE		📷	
...WIDTH		📷	
...TYPE		📷	
<SPAN>	📷		
<STRIKE>	📷	📷	📷
<STRONG>	📷	📷	📷
<STYLE>	📷		
<SUB>	📷	📷	📷
<SUP>	📷	📷	📷
<TABLE ...>	📷	📷	📷
...BORDER	📷	📷	📷
...CELLSPACING	📷	📷	📷

...CELLPADDING	📷	📷	📷
...WIDTH	📷	📷	📷
...HEIGHT	📷	📷	📷
...ALIGN	📷	📷	
...VALIGN	📷		
...BGCOLOR	📷	📷	
...BORDERCOLOR	📷		
...BORDERCOLORLIGHT	📷		
...BORDERCOLORDARK	📷		
...BACKGROUND	📷		
...FRAME	📷		
...RULES	📷		
<TBODY>	📷		
<TD ...>	📷	📷	📷
...ROWSPAN	📷	📷	📷
...COLSPAN	📷	📷	📷
...ALIGN	📷	📷	
...VALIGN	📷	📷	📷
...WIDTH	📷 absolute pixel values only	📷	
...HEIGHT	📷		
...NOWRAP	📷	📷	
...BGCOLOR	📷	📷	
...BORDERCOLOR	📷		

...BORDERCOLORLIGHT	📷		
...BORDERCOLORDARK	📷		
...BACKGROUND	📷		
<TEXTAREA ...>	📷	📷	📷
...NAME	📷	📷	📷
...ROWS	📷	📷	📷
...COLS	📷	📷	📷
...WRAP		📷	
<TFOOT>	📷		
<TH ...>	📷	📷	📷
...ROWSPAN	📷	📷	📷
...COLSPAN	📷	📷	📷
...ALIGN	📷	📷	📷
...VALIGN	📷	📷	📷
...WIDTH	📷 absolute pixel values only	📷	
...HEIGHT	📷		
...NOWRAP	📷	📷	
...BGCOLOR	📷	📷	
...BORDERCOLOR	📷		
...BORDERCOLORLIGHT	📷		
...BORDERCOLORDARK	📷		
...BACKGROUND	📷		
<THEAD>	📷		

<TITLE>	📷	📷	📷
<TR ...>	📷	📷	📷
...ALIGN	📷	📷	📷
...VALIGN	📷	📷	📷
...BGCOLOR	📷	📷	
...BORDERCOLOR	📷		
...BORDERCOLORLIGHT	📷		
...BORDERCOLORDARK	📷		
<TT>	📷	📷	📷
<U>	📷		📷
<UL>	📷	📷	📷
<VAR>	📷	📷	📷
<WBR>	📷	📷	
<XMP>	📷 will translate special characters.	📷	📷

Microsoft's Internet Explorer

See
http://www.microsoft.com/ie/
for more information

NCSA Mosaic

See
http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html
for more information

The `<SCRIPT>` element is supported by Netscape for JavaScript and by the Internet Explorer for Visual Basic Script.   Which do you want to see:

JavaScript
Visual Basic Script