Tables

HTML table elements, first introduced in Netscape 1.1, were developed to give authors a way to present rows and columns of tabular data. In fact, that has always been and remains their intended use. But it didn't take long for designers, fed up with the one-column, full-width web pages, to co-opt tables as a tool for controlling page layout. For the last 10 years, complex table-based layouts have been the norm. Nobody cared much that it was a misuse of the table elements—there weren't any other options. Today, we do have an option. Cascading Style Sheets offer the ability to create multicolumn pages and sophisticated layouts that were previously achievable only with tables. With improved browser support, pure style sheet layouts are finally a viable solution.

So tables-for-layout are out, but that doesn't mean that the whole set of table elements has been tossed in the dustbin. In fact, tables are still the appropriate markup choice for real tabular data, such as schedules, statistics, and so on.

This chapter takes on the topic of HTML tables, starting with their basic structure and markup and moving on to methods that make data tables accessible when rendered non-visually. Tips for using layout tables responsibly are included as well. Along the way, the following table-related elements will be addressed.

`table`	Establishes a table
`tr`	Table row
`td`	Table cell
`th`	Table header cell
`caption`	Provides a table caption
`thead`	Identifies a table header
`tbody`	Identifies the body of the table
`tfoot`	Identifies a table footer
`col`	Declares a column
`colgroup`	Declares a group of columns

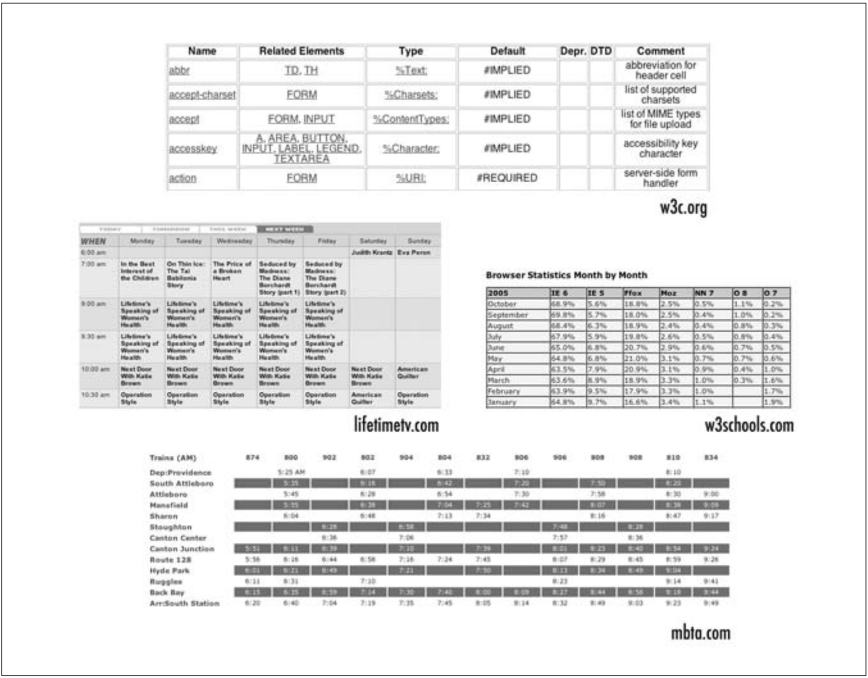
Table Uses

HTML tables fall into two broad categories: data tables and layout tables. This section takes a look at both types.

Data Tables

Data tables, the arrangement of information in rows and columns, are the intended use of HTML table elements. In visual browsers, the arrangement of data in rows and columns gives users an instant understanding of the relationships between data cells and their respective header labels. These relationships may be lost for users without the benefit of visual presentation unless care is taken to author the data table with accessibility in mind. These techniques are discussed in the upcoming "Accessible Tables" section.

Tables may be used to present calendars, schedules, statistics, or other types of information as shown in Figure 13-1. Note that "data" doesn't necessarily mean numbers. A table cell may contain any sort of information, including numbers, text elements, even images or multimedia objects.



Name	Related Elements	Type	Default	Depr.	DTD	Comment
abbr	TD, TH	%Text;	#IMPLIED			abbreviation for header cell
accept-charset	FORM	%Charsets;	#IMPLIED			list of supported charsets
accept	FORM, INPUT	%ContentTypes;	#IMPLIED			list of MIME types for file upload
accesskey	A, AREA, BUTTON, INPUT, LABEL, LEGEND, TEXTAREA	%Character;	#IMPLIED			accessibility key character
action	FORM	%URI;	#REQUIRED			server-side form handler

WHEN	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
6:00 am						Judith Krantz	Eva Peron
7:00 am	In the Best Interest of the Children	On Thin Ice: The Tai Babilonia Story	The Price of a Broken Heart	Seduced by Madness: The Diane Borchardt Story (part 1)	Seduced by Madness: The Diane Borchardt Story (part 2)		
9:00 am	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health		
9:30 am	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health	Lifetime's Speaking of Women's Health		
10:00 am	Next Door With Katie Brown	Next Door With Katie Brown	Next Door With Katie Brown	Next Door With Katie Brown	Next Door With Katie Brown	Next Door With Katie Brown	American Quilter
10:30 am	Operation Style	Operation Style	Operation Style	Operation Style	Operation Style	American Quilter	Operation Style

Browser Statistics Month by Month

2005	IE 6	IE 5	Ffox	Moz	NN 7	O 8	O 7
October	68.9%	5.6%	18.8%	2.5%	0.5%	1.1%	0.2%
September	69.8%	5.7%	18.0%	2.5%	0.4%	1.0%	0.2%
August	68.4%	6.3%	18.9%	2.4%	0.4%	0.8%	0.3%
July	67.9%	5.9%	19.8%	2.6%	0.5%	0.8%	0.4%
June	65.0%	6.8%	20.7%	2.9%	0.6%	0.7%	0.5%
May	64.8%	6.8%	21.0%	3.1%	0.7%	0.7%	0.6%
April	63.5%	7.9%	20.9%	3.1%	0.9%	0.4%	1.0%
March	63.6%	8.9%	18.9%	3.3%	1.0%	0.3%	1.6%
February	63.9%	9.5%	17.9%	3.3%	1.0%		1.7%
January	64.8%	9.7%	16.6%	3.4%	1.1%		1.9%

Trains (AM)	874	800	902	802	904	804	832	806	906	808	908	810	834
Dep:Providence		5:25 AM		6:07		6:33		7:10				8:10	
South Attleboro		5:35		6:16		6:42		7:20		7:50		8:20	
Attleboro		5:45		6:29		6:54		7:30		7:58		8:30	9:00
Mansfield		5:55		6:38		7:04	7:25	7:42		8:07		8:38	9:09
Sharon		6:04		6:48		7:13	7:34			8:16		8:47	9:17
Stoughton			6:28		6:58				7:48		8:28		
Canton Center			6:36		7:06				7:57		8:36		
Canton Junction	5:51	6:11	6:39		7:10		7:39		8:01	8:23	8:40	8:54	9:24
Route 128	5:56	6:16	6:44	6:58	7:16	7:24	7:45		8:07	8:29	8:45	8:59	9:26
Hyde Park	6:01	6:21	6:49		7:21		7:50		8:13	8:34	8:49	9:04	
Ruggles	6:11	6:31		7:10					8:23			9:14	9:41
Back Bay	6:15	6:35	6:59	7:14	7:30	7:40	8:00	8:09	8:27	8:44	8:58	9:18	9:44
Arr:South Station	6:20	6:40	7:04	7:19	7:35	7:45	8:05	8:14	8:32	8:49	9:03	9:23	9:49

Figure 13-1. Examples of data tables

Layout Tables

Layout tables, unlike data tables, are used purely as a presentational device for controlling the layout of a page. The HTML 4.01 Recommendation specifically discourages this use of tables, but it wasn't until CSS became a viable alternative that they have been condemned by the professional web community at large as well.

You can't turn around on the Web without bumping into a site—even big-name sites—that still uses tables for layout. Some sites use tables as a minimal framework; others have complex tables nested several layers deep to hold things together. Figure 13-2 shows just a few examples of layout tables of varying levels of complexity. The borders have been enhanced to reveal the table structure. (As not to point any fingers, these "old-school" examples are all my own work; I assure you, I've changed my ways.)

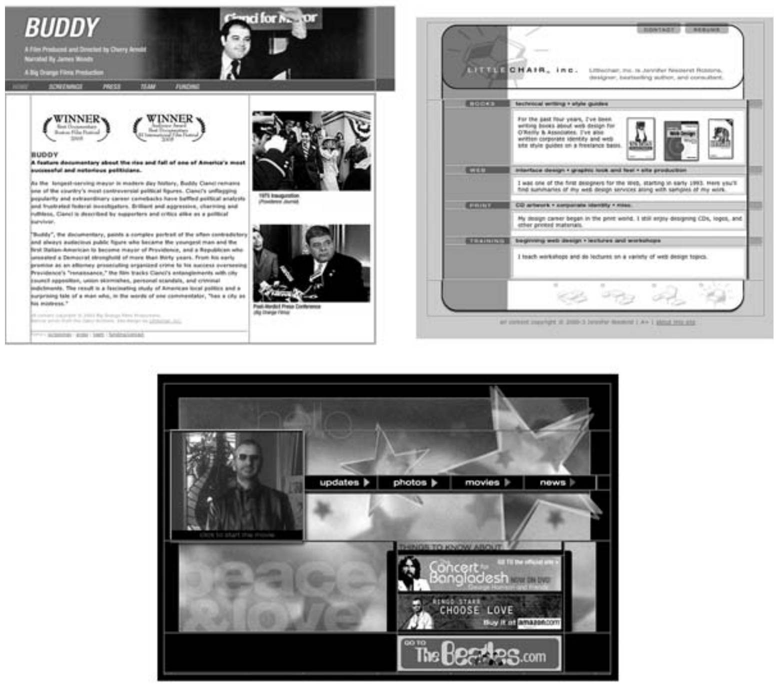


Figure 13-2. Examples of layout tables

While we are still in a period of transition from table-based design to totally CSS-based design (with flawless browser support, of course), some authors still choose to use tables to establish the basic column structure of the page. While not ideal, it can be done responsibly by using style sheets to keep the table markup minimal and with a mind toward accessibility. These strategies are discussed in the "Responsible Layout Tables" section at the end of this chapter.

Basic Table Structure

Put simply, web tables are made up of cells (which is where the content goes), arranged into rows. The HTML table model is said to be "row primary" because rows are identified explicitly in the document structure, while columns are just implied. The following examples illustrate the basic structure of an HTML table.

Rows and Cells

The minimum elements for defining a table are `table`, for establishing the table itself, `tr` for declaring a table row, and `td` for creating table cells within the row. Explanations and examples of how these elements fit together follow these element and attribute listings.

table

`<table>...</table>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization*, *Events*
`border="`*`number`*`"`
`cellpadding="`*`number of pixels or %`*`"`
`cellspacing="`*`number of pixels or %`*`"`
`frame="void|above|below|hsides|lhs|rhs|vsides|box|border"`
`rules="all|cols|groups|none|rows"`
`summary="`*`text`*`"`
`width="`*`number, percentage`*`"`

Deprecated attributes

`align="left|right|center"`
`bgcolor="`*`#rrggbb`*`"` or `"`*`color name`*`"`

Nonstandard attributes

`height="`*`number, percentage`*`"`

tr

`<tr>...</tr>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization*, *Events*
`align="left|center|right|justify|char"`
`char="`*`character`*`"`
`charoff="`*`length`*`"`
`valign="top|middle|bottom|baseline"`

Deprecated attributes

`bgcolor="`*`#rrggbb`*`"` or `"`*`color name`*`"`

td

```
<td>...</td>
```

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization*, *Events*
`abbr="`*`text`*`"`
`align="left|right|center|justify|char"`
`axis="`*`text`*`"`
`char="`*`character`*`"`
`charoff="`*`length`*`"`
`colspan="`*`number`*`"`
`headers="`*`id references`*`"`
`rowspan="`*`number`*`"`
`scope="row|col|rowgroup|colgroup"`
`valign="top|middle|bottom|baseline"`

Deprecated attributes

`bgcolor="`*`#rrggbb`*`"` or `"`*`color name`*`"`
`height="`*`pixels, percentage`*`"`
`nowrap="nowrap"`
`width="`*`pixels, percentage`*`"`

To see how the basic table elements are applied, consider a simple table with two rows and two columns (four content or "data" cells). The diagram on the left in Figure 13-3 shows the table with its cells and rows labeled in the way they are recognized in HTML. The diagram on the right shows the HTML elements that correspond with each component.

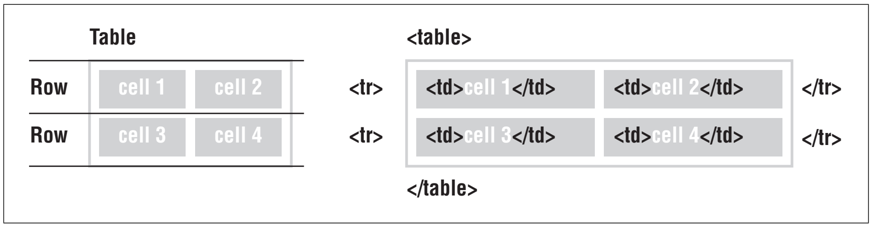


Figure 13-3. Basic table structure

Written out in an HTML source document, the markup for the table in Figure 13-3 would look more like this:

```
<table>
<tr>
     <td>cell 1</td><td>cell 2</td>
</tr>
<tr>
     <td>cell 3</td><td>cell 4</td>
</tr>
</table>
```

The entire table is indicated by the table element, which has no content of its own, but acts as a containing element for one or more of table row elements (tr). The table in the example contains two rows. Each tr element, in turn, contains two *data cells*, which are indicated by the td elements. The cells are the elements that contain real content; the table and tr elements are purely for table structure. A table cell may contain any data that can be displayed in a document, including formatted text, images, multimedia elements, and even other tables.

As mentioned earlier, the table system in HTML is row-primary. Rows are labeled explicitly, but the number of columns is just implied by the number of cells in the longest row. In other words, if all the rows have three cells (three td elements), then the table has three columns. If one row contains four td elements and all the others contain two, the browser displays the table with four columns, adding blank cells to the shorter rows. HTML 4.01 introduced an advanced standard system for describing table structure that includes explicit column elements. This system is discussed in the "Columns and Column Groups" section of this chapter.

Spanning Rows and Columns

Data cells in a table can occupy more than one space in the grid created by the rows and columns. You expand a td element horizontally or vertically using the colspan and rowspan attributes, respectively.

Column span

In Figure 13-4, <td colspan="2"> tells the browser to make "cell 1" occupy the same horizontal space as two cells—to make it "span" over two columns. The resulting spanned cell is indicated in Figure 13-4. Note that the row containing the spanned cell now only has one td element instead of two.

```
<table>
<tr>
<td colspan="2">Cell 1</td>
</tr>
<tr>
<td>Cell 3</td><td>Cell 4</td>
</tr>
</table>
```

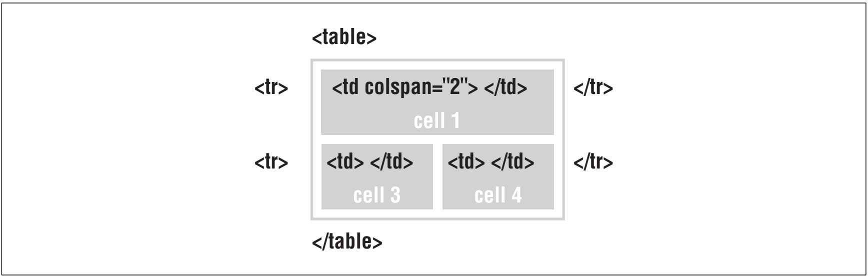


Figure 13-4. The colspan attribute expands cells horizontally to the right

Setting the colspan to a number greater than the actual number of columns (such as colspan="4" for the example) may cause some browsers to add empty columns to the table, possibly throwing your elements out of alignment.

Row span

Similar to colspan, the rowspan attribute stretches a cell to occupy the space of cells in rows below. Include the rowspan attribute in the row where you want the cell to begin and set its value equal to the number of rows you want it to span downward.

In Figure 13-5, note that the bottom row now contains only one cell. The other has been incorporated into the vertical spanned cell. Browsers ignore overextended rowspan values. There can never be more rows than explicitly stated tr elements.

```
<table>
<tr>
<td rowspan="2">Cell 1</td><td>Cell 2</td>
</tr>
<tr>
<td>Cell 4</td>
</tr>
</table>
```

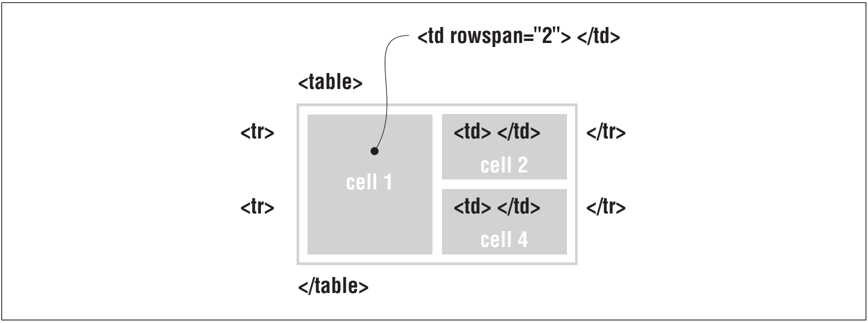


Figure 13-5. The rowspan attribute expands cells vertically

You may combine colspan and rowspan attributes to create a cell that spans both rows and columns.

Descriptive Elements

The basic table model also includes two elements that provide descriptions of the table's contents. Table header cells (th) are used to describe the cells in the row or column that they precede. The caption element gives a title to the whole table.

Table headers

Table header cells (indicated by the `th` element) are used to provide important information or context about the cells in the row or column that they precede. The `th` element accepts the same list of attributes as `td`.

th

`<th>...</th>`

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*
`abbr="`*`text`*`"`
`align="left|right|center|justify|char"`
`axis="`*`text`*`"`
`char="`*`character`*`"`
`charoff="`*`length`*`"`
`colspan="`*`number`*`"`
`headers="`*`id references`*`"`
`rowspan="`*`number`*`"`
`scope="row|col|rowgroup|colgroup"`
`valign="top|middle|bottom|baseline"`

Deprecated attributes

`bgcolor="`*`#rrggbb`*`"` or `"`*`color name`*`"`
`height="`*`pixels, percentage`*`"`
`nowrap="nowrap"`
`width="`*`pixels, percentage`*`"`

In terms of markup and table structure, headers are placed in the `tr` element, the same as a `td`, as shown in this example.

```
<table>
<tr><th>Planet</th><th>Distance from Earth</th></tr>
<tr><td>Venus</td><td>pretty darn far</td></tr>
<tr><td>Neptune</td><td>ridiculously far</td></tr>
</table>
```

User agents usually render the contents of table headers slightly differently than regular table cells (most often in bold, centered text); however, their appearance may easily be changed with style sheets.

The difference between `th` and `td` elements is not merely presentational, however. Table headers perform an important function in binding descriptive labels to table cells for non-visual browsers. They are discussed in more detail in the “Accessible Tables” section later in this chapter. Table header elements should not be used in layout tables.

Captions

The `caption` element provides a title or brief description of the table.

caption

`<caption>...</caption>`

Attributes

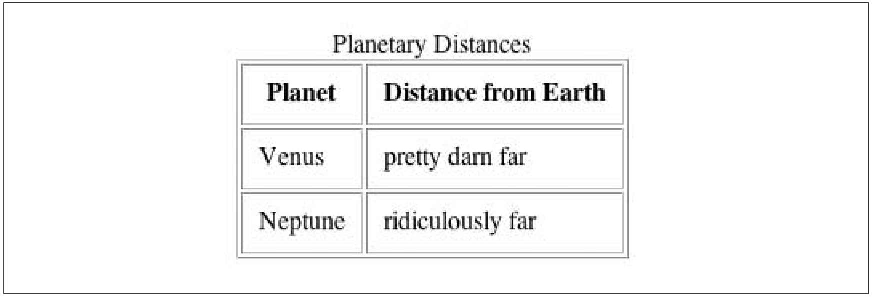
Core (id, class, style, title), *Internationalization, Events*

Deprecated attributes

align="top|bottom|left|right"

The caption element must immediately follow the opening table tag and precede all other table elements, as shown in this example and Figure 13-6.

```
<table>
<caption>Planetary Distances</caption>
<tr><th>Planet</th><th>Distance from Earth</th></tr>
<tr><td>Venus</td><td>pretty darn far</td></tr>
<tr><td>Neptune</td><td>ridiculously far</td></tr>
</table>
```



Planetary Distances

Planet	Distance from Earth
Venus	pretty darn far
Neptune	ridiculously far

Figure 13-6. A table with a caption

By default, the caption appears at the top of the table. Its width is determined by the width of the table. You can use the caption-side style property to move the caption below the table. There is also a deprecated align attribute that does the same thing. The left and right values are not well supported, so authors generally have the option of putting the caption above or below the table.

Captions are a useful tool for table accessibility and will be addressed again briefly in the "Accessible Tables" section later in this chapter.

Row Groups

HTML and XHTML define three "row group" elements that enable authors to organize rows into a table header (thead), footer (tfoot), and a table body (tbody). Because these elements share syntax and attributes, they have been aggregated into one element listing, presented here.

thead, tbody, tfoot

```
<thead>...</thead>, <tbody>...</tbody>, <tfoot>...</tfoot>
```

Attributes

Core (`id`, `class`, `style`, `title`), *Internationalization, Events*
`align="left|center|right|justify|char"`
`char="`*`character`*`"`
`charoff="`*`length`*`"`
`valign="top|middle|bottom|baseline"`

Internet Explorer 3.0 first introduced this system for grouping rows so they can be treated as units by user agents or style sheets. The W3C included the row group elements in the HTML 4.0 Recommendation as a way to allow more meaningful labeling, improve accessibility, and provide more flexibility for applying style sheet properties. Row groups are advantageous for data tables but should be avoided for layout tables.

The rows in a table may be grouped into a table head (`thead`), a table footer (`tfoot`), and one or more table bodies (`tbody`). The head and footer should contain information about the document and may someday be used to display fixed elements while the body scrolls independently. Another possibility is that the table head and foot would print on every page of a long table that has been divided over several pages.

The W3C requires that the `tfoot` element (if there is one) appear before `tbody` in the markup so the table can render the foot before downloading all the (potentially numerous) rows of data. An example of a simple table marked up with row groups is shown here.

```
<table>

<thead>
<tr><th>Employee</th><th>Salary</th><th>Start date</th></tr>
</thead>

<tfoot>
<tr><td colspan="3">Compiled by Buster D. Boss</td></tr>
</tfoot>

<tbody>
<tr><td>Wilma</td><td>5,000</td><td>April 6</td></tr>
<tr>... more data cells...</tr>
<tr>... more data cells...</tr>
</tbody>

</table>
```

Columns and Column Groups

As mentioned earlier in this chapter, the columns in a table are just implied by the number of cells in the longest row. In some instances, however, it is desirable to identify conceptual columns of data cells or groups of columns. The col (column) and colgroup (column group) elements allow authors to conceptually join a group of cells that appear in a column (or columns).

Column and column groups offer a number of conveniences. Their original intent was to speed up the display of tables in visual user agents. By specifying the width of each column, the user agent does not need to parse the contents of the entire table in order to calculate column and table. Columns and column groups are also useful for applying attributes (such as width or align) to all the cells they include. They may also be used as "hooks" for a limited number of style properties (see note). When used with the scope attribute (discussed in the upcoming accessibility section), they may also provide helpful context for screen readers and other non-visual browsing devices.

The CSS 2.1 Recommendation states that only the following four style properties may be applied to the col and colgroup elements: border, background, width, and visibility. For an in-depth explanation of why this is the case, read Ian Hickson's blog entry, "The mystery of why only four properties apply to table columns" at *ln.hixie.ch/?start=1070385285&count=1*. See also Chapter 22 of this book for more information on style properties for tables.

col

`<col />`

Attributes

Core (id, class, style, title), *Internationalization*, *Events*
align="left|center|right|justify|char"
char="*character*"
charoff="*length*"
span="*number*"
valign="top|middle|bottom|baseline"
width="*pixels, percentage, n**"

colgroup

`<colgroup>...</colgroup>`

Attributes

Core (id, class, style, title), *Internationalization*, *Events*
align="left|center|right|justify|char"
char="*character*"
charoff="*length*"
span="*number*"
valign="top|middle|bottom|baseline"
width="*pixels, percentage, n**"

The col element is used to label or to apply attribute specifications to an individual column (or across several columns via the span attribute) without actually grouping the columns together structurally or conceptually. An empty element, col is used only to apply attributes or styles to the columns to which it refers.

The colgroup element defines a conceptual group of columns. The number of columns included in the group is indicated with the span attribute or by the total of col elements (with their span values) within the column group. Attributes, such as width or align, applied to the colgroup element apply to every column within that group.

The colgroup and/or col elements must appear before any row or row group elements. They are placed either immediately after the table start tag or immediately after the caption element, if there is one. In this example, column group information has been added to the previous sample table markup.

```
<table>

<colgroup id="employinfo">
   <col span="2" width="100" />
   <col span="1" width="50" class="date" />
</colgroup>

<thead>
<tr><th>Employee</th><th>Salary</th><th>Start date</th></tr>
</thead>

<tfoot>
<tr><td colspan="3">Compiled by Buster D. Boss</td></tr>
</tfoot>

<tbody>
<tr><td>Wilma</td><td>5,000</td><td>April 6</td></tr>
<tr>... more data cells...</tr>
<tr>... more data cells...</tr>
</tbody>

</table>
```

The colgroup element identifies the three columns as part of the same structural group. (There may be many column groups in a table, but for simplicity's sake, this example has just one.) Within the colgroup, the first col element identifies two columns (span="2"), each with a width of 100 pixels. The remaining col has a width of 50 pixels. If all the columns in the table were to be the same width, the width could have been specified in the colgroup element. The third column is identified with a class attribute that could later be targeted with a style property (such as background).

Table Presentation

As for all matters of presentation, style sheets are the preferred method for changing the appearance of tables and offer more fine-tuned control than HTML attributes. See Chapter 22 for more information on CSS specifically for tables.

That said, there are a number of non-deprecated attributes that may be used to control cell spacing, dimensions, borders, and alignment (although, even most of those have style sheet alternatives). This section takes a look at those presentation-related attributes and also points out the preferred CSS methods.

Table Cell Spacing

There are two types of space that can be added in and around table cells: cell padding and cell spacing. The `cellpadding` and `cellspacing` attributes are used with the `table` element and apply to the whole table; you can't specify padding or spacing for individual cells using HTML alone.

Cell spacing

Cell spacing refers to the amount of space that is held between the cells in a table. It is specified with the `cellspacing` attribute in the `table` element. Values are specified in number of pixels. Increasing the cell spacing results in wider shaded borders between cells. In the second image in Figure 13-7, the darker gray areas indicate the 10 pixels of cell spacing added between cells. The default value for `cellspacing` is 2; therefore, if no `cellspacing` is specified, browsers will automatically place two pixels of space between cells.

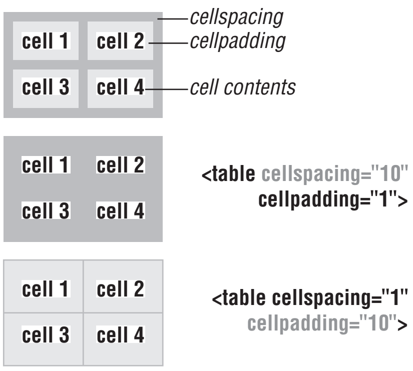


Figure 13-7. Cell spacing versus cell padding

Cell padding

Cell padding refers to the amount of space between the cell's border and the contents of the cell (as indicated by the third image in Figure 13-7). It is specified using the `cellpadding` attribute in the `table` element. Values are specified in number of pixels; the default value is 1. Relative values (percentages of available space) may also be used.

CSS alternatives

Cell padding may be handled by applying the `padding` property to the `td` element. By using `class`, `id`, or more specific selectors, it is possible to apply different amounts of padding to different cells within a table (the `cellpadding` attribute applies the same amount of padding to all cells).

There is no CSS property that is exactly equivalent to the `cellspacing` attribute, although you can adjust the amount of space between cells by setting the `border-collapse` property for the table to `separate` and then use the `border-spacing` property to specify the amount of space between cell borders. The difference is that with the `cellspacing` attribute, the border is rendered thicker between cells, while the `border-spacing` property adds empty space between them.

Unfortunately, the `border-spacing` property is not supported by Internet Explorer 6 and earlier (support in IE 7 is not documented as of this writing), so authors are left with no practical CSS `cellspacing` substitute for the time being.

Many authors also explicitly set both the `cellspacing` and `cellpadding` to 0 (zero) to override default browser settings and clear the way for style sheet properties.

Table and Cell Dimensions

By default, a table will render just wide enough to contain all of its contents. You can explicitly specify the width of a table using the `width` attribute in the `table` element. The HTML specifications provide no way to specify the height of a table, preferring the height to be automatically determined by the table's contents. However, there is a nonstandard `height` attribute that is well-supported for providing minimum height for the overall table.

You can control the width and height of individual cells by using the (you guessed it) `width` and `height` attributes in the `td` or `th` element. Height values are considered to be minimum heights and cells may expand downward to accommodate their contents.

The `width` and `height` attributes have been deprecated for `td` and `th` elements, but they are not deprecated for use in the `table` element. Style sheet properties are still the preferred method for specifying table dimensions.

CSS alternative

Use the `width` and `height` properties to set the size of any table-related element. Heights set on table and table cells are considered minimum heights, and the actual height may expand to fit the content.

Borders

The `table` element accepts the following attributes for controlling borders and rules between cells and around the table. All of the attributes introduced here apply to the `table` element only. None of these attributes are deprecated, but authors are urged to use CSS for drawing borders around table elements instead.

`border`
: Controls the width of the frame around the table. The default value is 1.

`frame`
: Specifies the sides of the table on which the frame should render. By default, the frame is rendered as a shaded, 3D style rule. The `frame` attribute uses these keyword values: `void` (no frame), `above` (top side only), `below` (bottom side only), `hsides` (horizontal sides), `lhs` (lefthand side), `rhs` (righthand side), `vsides` (vertical sides), `box` (all four sides), and `border` (all four sides).

`rules`
: Specifies which rules render between the cells of the table. One use for this attribute might be to display rules only between certain sets of columns or rows, as defined by `colgroup` or the row group elements (`thead`, `tbody`, and `tfoot`). The accepted values for the `rules` attribute are `all`, `cols`, `groups`, `none`, and `rows`.

CSS alternative

The collection of `border` properties in CSS allows you to specify the style (such as solid, dotted, dashed, and so on), color, and width of borders around any table-related element. With style sheets, it is possible to apply different borders to different sides of tables, their rows, or cells. See Chapter 19 for details on the border properties and Chapter 22 for how borders are handled in tables specifically.

Cell Content Alignment

The `align` and `valign` attributes are used to specify the horizontal and vertical alignment (respectively) of content within cells. Alignment may be specified for the following elements: `td`, `th`, `tr`, `thead`, `tbody`, `tfoot`, `col`, and `colgroup`.

> Adding the `align` attribute to the `table` element aligns the entire table in the width of its containing element and does not affect the alignment within the cells.

Horizontal alignment

The `align` attribute accepts the usual values `left`, `right`, `center`, and `justify`. Text is left-justified by default in left-to-right reading languages.

The `align` attribute also includes the `char` value that specifies that the table contents should be aligned on a specific character, such as a decimal point for a column of currency amounts. The character used for alignment is provided by the `char` attribute. The `charoff` attribute specifies the offset distance to the first alignment character. Although it's a nifty idea, the `char` and `charoff` attributes are not supported by current browsers.

Alignment settings for individual cells (`td` or `th`) always override settings at the higher levels. Alignment set on elements within a cell (a `p` element, for example) override the cell's alignment. If the table includes a `col` or `colgroup`, the `align`