

# Web Authoring: Tables

Academic Computing Services  
A Division of Information Services

[www.ku.edu/acs](http://www.ku.edu/acs)

---

**Abstract:** This document accompanies the workshop on XHTML tables. It covers in a more in-depth manner concepts that have been introduced or alluded to in introductory, or more general workshops.

---

## Contents

|  |    |
|--|----|
| Introduction .....   | 3  |
| Objectives .....   | 3  |
| Prerequisites .....  | 3  |
| Related Training Available from ACS.....   | 3  |
| Tables .....   | 3  |
| The <code>TABLE</code> element .....   | 5  |
| Attributes for <code>TABLE</code> .....  | 5  |
| The <code>TR</code> element.....   | 5  |
| The <code>TD</code> element.....   | 5  |
| Table header cells: the <code>TH</code> element.....   | 6  |
| Attributes for <code>TH</code> and <code>TD</code> .....                                       | 6  |
| Table captions: The <code>CAPTION</code> element.....  | 9  |
| Grouping .....   | 9  |
| Row groups: The <code>THEAD</code> , <code>TFOOT</code> , and <code>TBODY</code> elements..... | 9  |
| Column groups: The <code>COLGROUP</code> and <code>COL</code> elements .....                   | 10 |

|  |    |
|--|----|
| The COLGROUP element .....   | 10 |
| The COL element.....   | 10 |
| Table formatting (for graphical browsers) .....                              | 11 |
| Borders and rules.....   | 11 |
| Cell margins: The cellspacing and cellpadding attributes.....                | 12 |
| Alignment .....  | 14 |
| What goes into table cells? .....  | 15 |
| Example: Adding an image to a table .....                                    | 15 |
| Example: Adding a hyperlink to a table.....                                  | 15 |
| Example: Embedding a table within a table .....                              | 16 |
| Getting Additional Help .....  | 17 |
| Appendix A—Differences between XHTML 1.0 and previous versions of HTML ..... | 18 |

## Introduction

This workshop covers a special area of web page development - working with tables. Tables are useful XHTML elements for a variety of tasks, including page layout and positioning, as well as information organization.

## Objectives

The goal of this workshop is to introduce participants to additional concepts and elements specifically related to the creation and use of XHTML tables. These topics are not covered in the *Web Authoring: Introduction* or *Web Authoring: Intermediate* workshops.

## Prerequisites

It is assumed that participants in this workshop have taken the *Web Authoring: Introduction*, and *Web Authoring: Intermediate* workshops. Specifically, participants should be comfortable creating web pages using a simple text editor.

## Related Training Available from ACS

All workshops offered by Academic Computing Services (ACS), a division of Information Services, are free to KU students, staff, faculty, and [approved affiliates](#). The general public is also welcome to most workshops, but some ACS workshops require a [registration fee](#) for them.

To learn more about or register for workshops, receive automatic announcements of upcoming workshops, and track workshops you've registered for and have attended, visit the ACS Web site at [www.ku.edu/acs/train](http://www.ku.edu/acs/train). You can also check our online schedule at [www.ku.edu/acs/schedule](http://www.ku.edu/acs/schedule) for a list of class offerings and their availability. For further workshop related questions, please email [training@ku.edu](mailto:training@ku.edu).

## Tables

Tables provide the web page designer with sophisticated layout tools for the alignment and display of graphics and text on the web page.

Tables consist of rows broken into cells, where cells stacked atop one another form columns. A table can have just one cell in one row, or it can have many cells and many rows. Tables can also have a wide variety of attributes, including merged cells, borders, data placement, headers and more.

Here's an example of a simple table in XHTML:

Size Data for Canada, Mexico, and the United States

|                      | Area (sq. km) |         |           | Land Boundaries<br>(km) | Coastline<br>(km) |
|----------------------|---------------|---------|-----------|-------------------------|-------------------|
|                      | Land          | Water   | Total     |                         |                   |
| <b>Canada</b>        | 9,220,970     | 755,170 | 9,976,140 | 8,893                   | 243,791           |
| <b>Mexico</b>        | 1,923,040     | 49,510  | 1,972,550 | 4,538                   | 9,330             |
| <b>United States</b> | 9,158,960     | 470,131 | 9,629,091 | 12,248                  | 19,924            |

*A simple XHTML table*

This example illustrates the tabular arrangement of information that is the defining characteristic of tables, as well as other available features such as visible borders, captions, empty cells, cells that span multiple rows and/or columns, and header cells.

Here is a more basic table:

|       |       |       |
|-------|-------|-------|
| one   | two   | three |
| four  | five  | six   |
| seven | eight | nine  |

Don't let the absence of visible borders fool you; this *is* a table. What makes it so is the fact that the words are arranged in a grid.

The XHTML source that produces this table illustrates the basic table elements:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Basic Table</title>
</head>
<body>
  <table>
    <tr>
      <td>one</td> <td>two</td> <td>three</td>
    </tr>
    <tr>
      <td>four</td> <td>five</td> <td>six</td>
    </tr>
    <tr>
      <td>seven</td> <td>eight</td> <td>nine</td>
    </tr>
  </table>
</body>
</html>
```

The three new elements introduced here are TABLE, TR, and TD. The table is defined by the TABLE element. TR elements define table rows, within which TD elements define

table cells. Thus the layout of the table is specified and cells are defined, in order, row by row and top to bottom.

Note that all table content is contained in TD elements; none appears outside of the TD or TR elements in the TABLE.

## The TABLE element

The TABLE element contains all other elements that specify table content and formatting.

### *Attributes for TABLE*

**summary** = *text*

The `summary` attribute provides a summary of the table's purpose and structure for non-graphical browsers. The `summary` attribute debuted in HTML 4.0.

**width** = *integer number of pixels or percentage of available horizontal space*

The `width` attribute specifies the desired width for the entire table for graphical browsers. In the absence of any width specification, the table width is typically set to fit the contents.

Some of TABLE's other attributes are described in the section entitled **Table formatting (for graphical browsers)**, beginning on page 11.

## The TR element

"TR" stands for "table row." The TR element acts as a container for a row of table cells.

## The TD element

"TD" stands for "table data." The TD element defines a cell that contains data. Cells may be empty.

In the example above, the HTML source was arranged to reflect the placement of words in the resulting table, in order to reinforce the roles of the table tags and make it easy to visualize their effects. Of course, it is the tags that define the table's layout, not their arrangement in the source file. The following represents a different (and probably more typical) organization that is equivalent:

```
<table>
  <tr>
    <td>one</td>
    <td>two</td>
    <td>three</td>
  </tr>
  <tr>
    <td>four</td>
    <td>five</td>
    <td>six</td>
  </tr>
  <tr>
    <td>seven</td>
    <td>eight</td>
    <td>nine</td>
  </tr>
</table>
```

Although the cells are on different lines, they still appear in rows in the table according to their placement in TR elements. Each TR element marks a new row, the TD elements cells to be placed in the row in order. Note also that the DTD and the <HTML>, </HTML>, <HEAD>, </HEAD>, <BODY>, and </BODY> tags have been left out for clarity.

## Table header cells: the TH element

Besides TD, cells can also be defined by the TH element. The TH element defines a cell that contains header information. “TH” stands for “table header.” For example, the sample table on page 4 contains five columns of data, each headed by a column description. The column descriptions are in TH elements, which the exhibiting browser presented with a bold font and centered alignment *without style sheets or other formatting cues*. A speech synthesizer might render such cells with a distinct voice inflection.

### Attributes for TH and TD

The `id`, `headers`, `scope`, `abbr`, and `axis` attributes allow the page author to formalize associations among cells.

**`id` = a unique cell name**

The `id` attribute specifies a unique name for the cell, to be referenced in other locations, such as in the `headers` attribute. Note that names in your HTML document must all be unique.

**`headers` = space-separated list of cell names**

The `headers` attribute specifies the list of cells that provide header information for the current data cell. Listed cells must be named by setting their `id` attributes. For example, in the sample table on page 4, header information for the cell

containing “9,330” is in the cells containing “Mexico” and “Coastline (km)”. A `header` attribute (along with `id` attributes in the header cells) could make that relationship explicit. This attribute may be used in conjunction with style sheets, and/or to help non-graphical browsers render header information about data cells, e.g., by speaking header information before each cell. The `headers` attribute debuted in HTML 4.0.

**`scope = row or col or rowgroup or colgroup`**

The `scope` attribute does the reverse of `headers`: it specifies the set of data cells for which the current cell provides header information. “`row`” indicates the current cell provides header information for the rest of the row that contains it. Likewise, “`col`” indicates the rest of the column, “`rowgroup`” the rest of the row group (see **Row groups: The `THEAD`, `TFOOT`, and `TBODY` elements**, page 9), and “`colgroup`” the rest of the column group (see **Column groups: The `COLGROUP` and `COL` elements**, page 10). `Scope` is best used in place of `headers` when header relationships among cells are simple or when headers are not placed in irregular positions with respect to the data they describe. The `scope` attribute debuted in HTML 4.0.

**`abbr = text`**

The `abbr` attribute should be used to provide an abbreviated form of the cell’s content for repeated use, as when speech synthesizers speak headers relating to a particular cell before speaking that cell’s content. The `abbr` attribute debuted in HTML 4.0.

**`axis = comma-separated list of category names`**

The `axis` attribute is used to categorize cells to provide contextual or grouping information. For more information, see <http://www.w3.org/TR/html401/struct/tables.html#multi-dimension>. The `axis` attribute debuted in HTML 4.0.

The `rowspan` and `colspan` attributes enable cells to span multiple rows or columns.

**`rowspan = integer`**

The `rowspan` attribute specifies the number of rows spanned by the current cell. The default value is 1. A value greater than one makes the cell taller by the number of rows indicated, and displaces cells below it. The value 0 means that the cell spans all rows from the current row to the last row of the table section (see **Row groups: The `THEAD`, `TFOOT`, and `TBODY` elements**, page 9) in which the cell is defined.

**`colspan = integer`**

The `colspan` attribute specifies the number of columns spanned by the current cell. The default value is 1. A value greater than one makes the cell wider by the number of columns indicated, and displaces cells next to it. The value 0 means that the cell spans all columns from the current column to the last column of the column group (see **Column groups: The `COLGROUP` and `COL` elements**, page 10) in which the cell is defined.

For example, changing the previous source listing to:

```

<table border="1">
  <tr>
    <td>one</td>
    <td rowspan="2">two</td>
    <td>three</td>
  </tr>
  <tr>
    <td>four</td>
    <td>six</td>
  </tr>
  <tr>
    <td>seven</td>
    <td>eight</td>
    <td>nine</td>
  </tr>
</table>

```

would result in a table that might look like this:

|       |       |       |
|-------|-------|-------|
| one   | two   | three |
| four  |       | six   |
| seven | eight | nine  |

(Here the table border attribute has been used to add visible borders to highlight cell boundary behavior. The border attribute is discussed on page 12.)

The `rowspan` attribute with the value of 2 causes the cell to which it is applied to occupy two rows in height. Notice that the row below contains only two cells—the cell containing “five” having been deleted from the original example—since the larger cell from the row above occupies one of the spaces in the row. If this accommodation had not been made, the table would look like this:

|       |       |       |      |
|-------|-------|-------|------|
| one   | two   | three |      |
| four  |       | six   | five |
| seven | eight | nine  |      |

The table is now four columns wide, with the first and third rows padded with empty cells. It is up to you to provide the correct number of cells per row to produce the desired number of columns; including cells that span multiple rows or columns complicates this task.

The next example modifies the original table of number names to demonstrate the `colspan` attribute. The following source:

```

<table border="1">
  <tr>
    <td>one</td>
    <td>two</td>
    <td>three</td>
  </tr>
  <tr>
    <td colspan="2">four</td>
    <td>six</td>
  </tr>
  <tr>
    <td>seven</td>
    <td>eight</td>
    <td>nine</td>
  </tr>
</table>

```

produces a table like this:

|       |       |       |
|-------|-------|-------|
| one   | two   | three |
| four  |       | six   |
| seven | eight | nine  |

Here again, the “five” cell was omitted to make room for the expanded cell. Defining overlapping cells is an error.

Some of TH’s and TD’s other attributes are described in the section entitled **Table formatting (for graphical browsers)** beginning on page 11.

## Table captions: The CAPTION element

As its name implies, the CAPTION element is used to name or describe a table. It is only permitted immediately after the TABLE start tag. A TABLE element may contain no more than one CAPTION element.

## Grouping

Table rows and columns can be grouped to support advanced display features and aggregate application of style information.

### **Row groups: The THEAD, TFOOT, and TBODY elements**

Table rows may be grouped into a table head, table foot, and one or more table body sections, using the THEAD, TFOOT, and TBODY elements, respectively. This division enables browsers to support scrolling of table bodies independently of the table head and foot. When long tables are printed, the table head and foot information may be repeated on each page that contains table data.

The table head and foot should contain information about the table's columns. The table body (or bodies) should contain rows of table data.

When present, each `THEAD`, `TFOOT`, and `TBODY` contains a row group. Each row group must contain at least one row, defined by the `TR` element.

`TFOOT` must appear before `TBODY` within a `TABLE` definition. The `TBODY` element is always required except when the table contains only one table body and no table head or foot sections. The `THEAD`, `TFOOT`, and `TBODY` sections must contain the same number of columns.

The `THEAD`, `TFOOT`, and `TBODY` elements debuted in HTML 4.0.

## ***Column groups: The `COLGROUP` and `COL` elements***

Column groups enable you to create structural divisions within tables. A table may either contain a single implicit column group (no `COLGROUP` element delimits the columns) or any number of explicit column groups (each delimited by an instance of the `COLGROUP` element).

The `COL` element enables you to share attributes among several columns without implying any structural grouping. The "span" of the `COL` element is the number of columns that will share the element's attributes.

### **The `COLGROUP` element**

The `COLGROUP` element defines a table column group. The `COLGROUP` element debuted in HTML 4.0.

#### ***Attributes for `COLGROUP`***

***span = integer***

The `span` attribute, which must be an integer greater than 0, specifies the number of columns in a column group. The default value is 1. The `span` attribute debuted in HTML 4.0.

***width = integer number of pixels or percentage of available horizontal space or a relative length***

The `width` attribute specifies a default width for each column in the current column group. A relative length has the form "i\*", where "i" is an integer. Each relative length receives a portion of the available space that is proportional to the integer preceding the "\*". The value "\*" is equivalent to "1\*". The special form "0\*" means that the width of each column in the group should be the minimum width necessary to hold the column's contents. This attribute is overridden for any column in the column group whose width is specified via a `COL` element. This `width` attribute debuted in HTML 4.0.

### **The `COL` element**

The `COL` element enables you to group together attribute specifications for table columns, without grouping them together structurally. `COL` elements are empty and serve only as a

support for attributes. They may appear inside or outside a COLGROUP element. The COL element debuted in HTML 4.0.

### *Attributes for COL*

**span = integer**

The span attribute, which must be an integer greater than 0, specifies the number of columns “spanned” by the COL element; the COL element shares its attributes with all the columns it spans. The default value is 1. The span attribute debuted in HTML 4.0.

**width = integer number of pixels or percentage of available horizontal space or a relative length**

The width attribute specifies a default width for each column spanned by the current COL element. It has the same meaning as the width attribute for the COLGROUP element and overrides it. This width attribute debuted in HTML 4.0.

Some of COLGROUP’s and COL’s other attributes are described in the section entitled **Table formatting (for graphical browsers)**, below.

The following example demonstrates the use of COLGROUP and COL. It represents a table of 10 columns, each 20 pixels wide, in which the last column has special formatting.

```
<TABLE>
  <COLGROUP width="20">
    <COL span="9">
      <COL style="...">
    </COLGROUP>
  <TR>
    <TD>...
  ...
</TABLE>
```

## **Table formatting (for graphical browsers)**

Cascading style sheets offer control over a number of table formatting characteristics. This section describes formatting options in HTML.

### ***Borders and rules***

The following TABLE attributes affect a table’s external frame and internal rules.

**frame = void or above or below or hside or lhs or rhs or vside or box or border**

The frame attribute specifies which sides of the frame surrounding a table will be visible. Possible values:

- void: No sides. This is the default value.
- above: The top side only.

- `below`: The bottom side only.
- `hsides`: The top and bottom sides only.
- `vsides`: The left and right sides only.
- `lhs`: The left-hand side only.
- `rhs`: The right-hand side only.
- `box`: All four sides.
- `border`: All four sides.

The `frame` attribute debuted in HTML 4.0.

**`rules` = *none or groups or rows or cols or all***

The `rules` attribute specifies which rules will appear between cells within a table. Possible values:

- `none`: No rules. This is the default value.
- `groups`: Rules will appear between row groups and column groups only.
- `rows`: Rules will appear between rows only.
- `cols`: Rules will appear between columns only.
- `all`: Rules will appear between all rows and columns.

The `rules` attribute debuted in HTML 4.0.

**`border` = *integer number of pixels***

The `border` attribute specifies the width, in pixels, of the frame around a table. Setting `border="0"` implies `frame="void"` and, unless otherwise specified, `rules="none"`. Other values of `border` imply `frame="border"` and, unless otherwise specified, `rules="all"` (see example on page 8).

## ***Cell margins: The `cellspacing` and `cellpadding` attributes***

The `TABLE` attributes `cellspacing` and `cellpadding` affect cell margins.

**`cellspacing` = *integer number of pixels or percentage of the available horizontal space***

The `cellspacing` attribute specifies how much space the browser is to leave between cells, and between the cells and the table frame.

**`cellpadding` = *integer number of pixels or percentage of the available horizontal space***

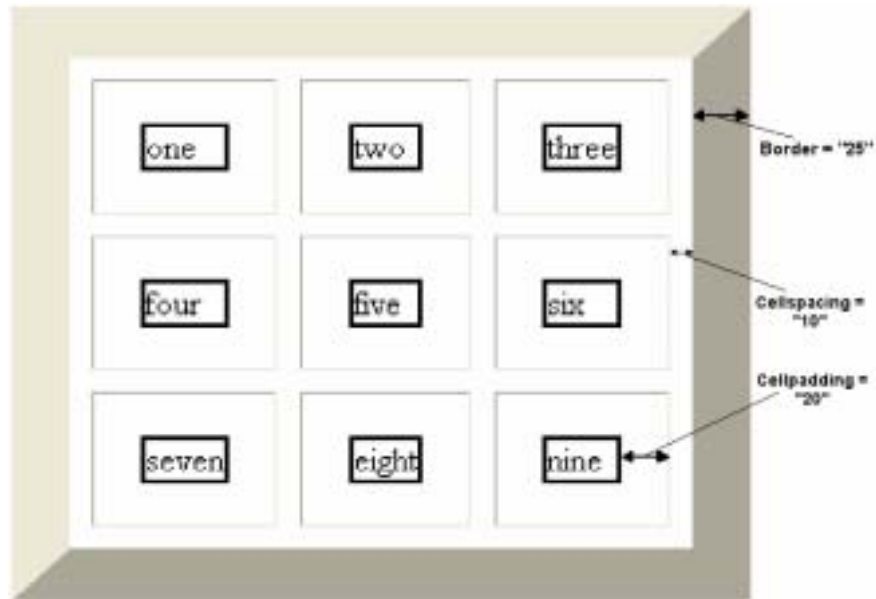
The `cellpadding` attribute specifies the amount of space between the border of the cell and its contents. If the value is a pixel length, all four margins should be this distance from the contents. If the value is a percentage length, the top and bottom margins should be equally separated from the content based on a percentage of the available vertical space, and the left and right margins should

be equally separated from the content based on a percentage of the available horizontal space. For example, a value of 20% specifies that the top margin of the cell and the bottom margin of the cell will each be separated from the cell's contents by 10% of the available vertical space (the total being 20%). Similarly, the left margin of the cell and the right margin of the cell will each be separated from the cell's contents by 10% of the available horizontal space (the total being 20%).

The following example shows borders and cell margins. Cell contents have been enclosed in black boxes to highlight their boundaries, and the specified spacing indicated in the figure. (The `STYLE` element, which is responsible for outlining the cell data, is discussed in the Cascading Style Sheets 1 class.)

```
<title>Sample Table Exhibiting Cell Margins</title>
<style type="text/css">
<!--
  P { border: thin solid #000000; }
-->
</style>

<table border="25" cellspacing="10" cellpadding="20">
  <tr>
    <td><p>one</p></td>
    <td><p>two</p></td>
    <td><p>three</p></td>
  </tr>
  <tr>
    <td><p>four</p></td>
    <td><p>five</p></td>
    <td><p>six</p></td>
  </tr>
  <tr>
    <td><p>seven</p></td>
    <td><p>eight</p></td>
    <td><p>nine</p></td>
  </tr>
</table>
```



## Alignment

The following attributes may be set for THEAD, TFOOT, TBODY, COLGROUP, COL, TR, TD, and TH. Alignment attributes may be inherited from enclosing elements.

**align = left or center or right or justify or char**

---

**Note:** Application of align to the TABLE element is deprecated. The align attribute specifies the alignment of data and the justification of text in a cell. Possible values:

---

- **left:** Left-flush data/Left-justify text. This is the default value for table data.
- **center:** Center data/Center-justify text. This is the default value for table headers.
- **right:** Right-flush data/Right-justify text.
- **justify:** Double-justify text.
- **char:** Align text around a specific character.

This align attribute debuted in HTML 4.0.

**valign = top or middle or bottom or baseline**

The valign attribute specifies the vertical position of data within a cell. Possible values:

- **top:** Cell data is flush with the top of the cell.
- **middle:** Cell data is centered vertically within the cell. This is the default value.

- `bottom`: Cell data is flush with the bottom of the cell.
- `baseline`: All cells in the same row as a cell whose `valign` attribute has this value should have their textual data positioned so that the first line occurs on a baseline common to all cells in the row.

**`char` = *character***

The `char` attribute specifies a single character to act as an axis for alignment. The default value for this attribute is the decimal point character for the current language (e.g., the period in English). Cell data containing more than one instance of the alignment character should be avoided. The `char` attribute debuted in HTML 4.0.

**`charoff` = *integer number of pixels or percentage of the available horizontal space***

The `charoff` attribute specifies the offset to the first occurrence of the alignment character on each line. If a line doesn't include the alignment character, it should be horizontally shifted to end at the alignment position. The `charoff` attribute debuted in HTML 4.0.

## What goes into table cells?

Table cells can contain anything that a normal web page can contain (except scripts). Tables can contain text, images, hyperlinks, paragraphs, lists, or even other tables. As stated before, you can also leave cells empty, by simply including the open tag for the cell, and immediately following that tag with the corresponding closing tag.

To include an image or a hyperlink, simply include the elements, text or tags following the `<td>` or `<th>` tag.

Examples:

### ***Example: Adding an image to a table***

```
<table border="3">
  <tr>
    <td> </td>
    <td><br />
      plus some text </td>
  </tr>
</table>
```

### ***Example: Adding a hyperlink to a table***

```
<table border="3">
  <tr>
    <td><a href="http://www.ku.edu/"> KU </a> </td>
    <td><a href="nearby-file.html">A relative URL</a> </td>
  </tr>
</table>
```

## Example: Embedding a table within a table

```
<table border="5" cellpadding="10">
  <tr>
    <td>This is the first cell of the outer table.</td>
    <td>This is the next cell of the outer table. </td>
  </tr>
  <tr>
    <td>
      <table border="1">
        <tr>
          <td>This is a cell in the inner table.</td>
          <td>This is another cell.</td>
          <td>This is a third cell.</td>
        </tr>
      </table>
    </td>
    <td>This is the last outer cell.</td>
  </tr>
</table>
```

This example would be rendered as (note the difference in border widths):

|   |   |                       |                       |                              |
|---|---|-----------------------|-----------------------|------------------------------|
| This is the first cell of the outer table.  | This is the next cell of the outer table. |                       |                       |                              |
| <table border="1"><tbody><tr><td>This is a cell in the inner table.</td><td>This is another cell.</td><td>This is a third cell.</td></tr></tbody></table> | This is a cell in the inner table.        | This is another cell. | This is a third cell. | This is the last outer cell. |
| This is a cell in the inner table.  | This is another cell.                     | This is a third cell. |                       |                              |

## Getting Additional Help

ACS provides consulting and Q&A help in a variety of ways:

785/864-0410

[workshop@ku.edu](mailto:workshop@ku.edu)

[www.ku.edu/acs/help](http://www.ku.edu/acs/help)

*Last Update: 09/09/2003*

## Appendix A—Differences between XHTML 1.0 and previous versions of HTML

The nature of web development has been undergoing a dramatic shift in recent years. Increasingly, the creation of web content is being modularized. This is seen in part by the advent of Cascading Style Sheets. By making use of CSS, web authors separate the content and layout of a web page (contained in the HTML file) from the visual style and presentation (contained in the CSS file).

This trend of modularization continued with the development of XML (Extensible Markup Language). XML is a markup language whose purpose is to *describe* data - its focus is on what data *is*. Its markup elements (which are not predefined, as they are in HTML) are used to describe the data that the file contains. This is in contrast to HTML, whose purpose is to *render* data (usually visually) - its focus is on how data *is presented*, i.e., how the data are laid out. (Note that this is still distinct from the *style* of the data, defined by CSS.) It is important to note that even though XML does not have predefined markup elements it does have a strict syntax. For example, all markup must be done in lowercase, all elements must have both open and close tags, and all tags must be nested properly.

XHTML 1.0 a new version of HTML, with the following properties:

- XHTML stands for Extensible Hypertext Markup Language
- XHTML is aimed to replace HTML
- XHTML is almost identical to HTML 4.01
- XHTML is a stricter and cleaner version of HTML
- XHTML is HTML defined as XML

Some of these points may be clear, but some explanation is required for others. As mentioned above, XML does not have predefined markup - the author is free to create tags that logically describe the data. So if the data is a web page and we wish to mark up this data using XML, our XML markup tags should describe that web page. But that is exactly what HTML markup did, so it makes perfect sense to use the 'old' HTML markup tags in our 'new' markup using XML - after all, in XML, we are free to use any markup tags we wish. This is the essence of XHTML - it is HTML reformulated as XML. Because we are using the 'old' HTML tags in our 'new' XHTML markup, XHTML is almost identical to HTML 4.01; the only real difference between XHTML and HTML is syntax - for example, all markup must be done in lowercase, elements must have both open and close tags, etc. The stricter syntax rules of XML make XHTML much cleaner and more consistent than HTML.

The complete text of the XHTML 1.0 specification can be viewed on the W3C (WWW Consortium) web site: <http://www.w3.org/TR/xhtml1>. A list of the differences between HTML 4.01 and XHTML 1.0 can be found at <http://www.w3.org/TR/xhtml1/#diffs>.