Introduction to HTML

Before you start

To understand this article, it is recommended to be comfortable with the use of a Browser.

If you are also familiar with the manipulation (creating and editing) text files, you can test the examples shown in this article.

When you look at a web page in a web browser, you see, at the simplest level, words. It is rare to see a web page containing only *plain text.* Instead the words you see and read have *style* characteristics. Modern web page designers have access to hundreds of different fonts, font sizes, colors, and even alphabets (e.g. Spanish, Japanese, Russian). Modern browsers can display most of them accurately. Web pages may also contain images, video clips, and background music. They may include drop down menus, search boxes, clickable links to products or other pages within the same website, or links to external web pages. Some web sites even provide options to customize the page display to accomodate personal preference or inabilities such as limited vision, deafness, or color blindness. Often a page contains content boxes that move (scroll) while the rest of the page remains static.

Several technologies (such as CSS, JavaScript, Flash, AJAX, JSON) can be used to define the elements of a web page. However, at the base level, a web page is created using HTML (HyperText Markup Language). Without HTML, there is no web page. HTML is what a browser "reads" to present the page on the client-side computer.

HTML's international standards and specifications are maintained by the World Wide Web Consortium (W3C) and the Web Hypertext Application Technology Working Group (WHATWG). The WHATWG considers HTML a "living standard" which is constantly evolving, whereas the W3C works on both "snapshots" of HTML, the most recent of which is HTML5 and on the evolution of HTML (HTML 5.1).

The HTML specification defines a single language that can be written using the relaxed HTML syntax, but also the more strict XML (Extensible Markup Language) one, and also addresses the needs of web applications. HTML does **not** describe the style and formatting of content, but only the content itself and its **meaning**.

The creator of a web page can use CSS to define the appearance and layout of texts and other materials.

Current best practices in web development encourage the use of CSS rather than explicit presentational HTML.

This article provides an introduction to HTML. If you've ever wondered what goes on behind the scenes in your web browser, this article is the place to start learning.

A brief history of HTML

In the late 1980s, Tim Berners-Lee was working as a physicist at CERN (the European Organization for Nuclear Research). He devised a way for scientists to share documents over the internet. Prior to his invention, communication via the internet was limited to plain text, using technologies such as email, FTP (File Transfer Protocol), and Usenet-based discussion boards. The invention of HTML made use of a model of content stored on a central server that could be transferred and displayed on a local workstation via a browser. It simplified access to content and enabled the display of "rich" content (such as sophisticated text formatting and the display of images).

What is HTML?

HTML is a **markup language**. The word *markup* was used by editors who *marked up* manuscripts (usually with a blue pencil) when giving instructions for revisions. The term remains in use, though with slightly different meaning. A markup language as it relates to browsers is a language with specific syntax that gives intructions to a web browser about how to display a page. HTML separates "content" (words, images, audio, video, and so on) from "presentation" (the definition of the type of content and the instructions for how that type of content should be displayed). HTML uses a pre-defined set of elements to identify content types. Elements contain one or more "tags" that contain or express content. Tags are surrounded by angle brackets, and the "closing" tag (the one that indicates the end of the content) is prefixed by a forward slash.

> *Most browsers allow the user to view the HTML of any webpage. Using Firefox, for example, click on menu item:Tools:Web Developer:Page Source to view the HTML of a page. Beginners will find the code nearly unreadable for a complex page, but if you spend some time looking at the code for a simple page and comparing it to the page the code renders, you will soon develop a clear understanding of some of the basic syntax requirements. Try it next time you pull up a "sign in" page that contains few elements such as the sign in at www.ecrater.com.*

The paragraph element consists of the start tag " " and the closing tag " ". The following example shows a paragraph contained within the HTML paragraph element but remember it will not preserve more than one white spaces:

```
1 <p>You are beginning to learn HTML.</p>
```

When this content is displayed in a web browser, it looks like this:

You are beginning to learn HTML.

The browser uses the tags as an indicator of how to display the content in the tags.

Elements that contain content can usually also contain other elements. For example, the emphasis element ("") can be embedded within a paragraph element, to add emphasis to a word or phrase:

```
<p>You are <em>beginning</em> to learn HTML.</p>
```

When displayed, this looks like:

You are *beginning* to learn HTML.

Some elements do not contain other elements. For example, the image tag ("<img>") specifies the file name of the content (an image) as an attribute:

```
<img src="smileyface.jpg" alt="Smiley face" >
```

Often a forward slash is placed before the final angle bracket to indicate the end tag of an empty element in XHTML (which is an XML schema that implements HTML elements).

The rest of this article goes into greater detail regarding the concepts introduced in this section. However, if you want to see HTML in action, check out Mozilla Thimble, which is an interactive online editor that helps you learn how to write HTML markup. Also, see HTML Elements for a list of available elements and a description of their use.

Elements — the basic building blocks

HTML consists of a set of **elements**. Elements define the **semantic** meaning of their content. Elements include everything between two matching element tags, including the tags themselves. For example, the "<p>" element indicates a paragraph; the "<img>" element indicates an image. See the HTML Elements page for a complete list.

Some elements have very precise meaning, as in "this is an image", "this is a heading", or "this is an ordered list." Others are less specific, such as "this is a section on the page" or "this is part of the text." Yet others are used for technical reasons, such as "this is identifying information for the page that should not be displayed." Regardless, in one way or another all HTML elements have a semantic value.

Most elements may contain other elements, forming a hierarchic structure. A very simple but complete web page looks like this:

```
<html>
  <body>
    <p> you are in your beginning stage of HTML</p>
  </body>
</html>
```

As you can see, the <html> element surround the rest of the document, and the <body> element surround the page content. This structure is often thought of as a tree with branches (in this case, the <body> and <p> elements) growing from the trunk (<html>). This hierarchical structure is called the **DOM**: the **Document Object Model**.

Tags

HTML documents are written in plain text. They can be written in any text editor that allows content to be saved as plain text, such as Notepad, Notepad++, or Sublime Text, but most HTML authors prefer to use a specialized editor that highlights syntax and shows the DOM. Tag names may be written in either upper or lower case. However, the W3C (the global consortium that maintains the HTML standard) recommends using lower case (and XHTML requires lower case).

HTML attaches special meaning to anything that starts with the less-than sign ("<") and ends with the greater-than sign (">"). Such markup is called a **tag**. Make sure to close the tag, as some tags are closed by default, whereas others might produce unexpected errors if you forget the end tag.

Here is a simple example:

```
<p>This is text within a paragraph.</p>
```

In this example there is a *start* tag and a *closing* tag. Closing tags are the same as the start tag but also contain a *forward slash* immediately after the leading less-than sign. Most elements in HTML are written using both start and closing tags. Start and closing tags should be properly **nested**--that is, closing tags should be written in the opposite order of the start tags. Proper nesting is one rule that must be obeyed in order to write **valid** code.

This is an an example of *valid* code:

```
<em>I <strong>really</strong> mean that</em>.
```

This is an example of *invalid* code:

```
Invalid: <em>I <strong>really</em> mean that</strong>.
```

Note that in the valid example, the closing tag for the nested element is placed before the closing tag for the element in which it is nested. In the invalid code, they are nested.

Some elements do not contain any text content or any other elements. These are **empty** elements and need no closing tag. This is an example:

```
<img src="smileyface.jpg" alt="Smiley face" >
```

Empty elements in XHTML mode are usually closed using a trailing forward slash.

```
<img src="smileyface.jpg" alt="Smiley face" />
```

In HTML this slash has a meaning that is not implemented in Firefox so it should not be used. Empty elements must not be closed in HTML mode.

Attributes

The start tag may contain additional information, as in the preceding example. Such information is called an **attribute**. Attributes usually consist of 2 parts:

- An attribute **name**
- An attribute **value**

A few attributes can only have one value. They are **Boolean** attributes and may be shortened by only specifying the attribute name or leaving the attribute value empty. Thus, the following 3 examples have the same meaning:

```
<input required="required">
```

```
<input required="">

<input required>
```

Attribute values that consist of a single word or number may be written as they are, but as soon as there are two or more strings of characters in the value, it must be written within quotation marks. Both single quotes (') and double quotes (") are allowed. Many developers prefer to always use quotes to make the code less ambiguous to the eye and to avoid mistakes. The following is such a mistake:

```
<p class=foo bar> (Beware, this probably does not mean what you think it means.)
```

In this example the value was supposed to be "foo bar" but since there were no quotes the code is interpreted as if it had been written like this:

```
<p class="foo" bar="">
```

Named character references

Named character references (often casually called *entities*) are used to print characters that have a special meaning in HTML. For example, HTML interprets the less-than and greater-than symbols as tag delimiters. When you want to display a greater-than symbol in the text, you can use a named character reference. There are four common named character references one must know:

- `>` denotes the greater than sign (>)
- `<` denotes the less than sign (<)
- `&` denotes the ampersand (&)
- `&quot;` denotes double quote (")

There are many more entities, but these four are the most important because they represent characters that have a special meaning in HTML.

Doctype and comments

In addition to tags, text content, and entities, an HTML document must contain a **doctype** declaration as the first line. The **doctype** declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

In HTML 4.01, doctype refers to a DTD (Document Type Definition) as it was based on SGML. There are three different **doctype** declarations in HTML 4.01.

HTML 4.01 Strict

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font). Framesets are not allowed.

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

HTML 4.01 Transitional

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed.

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

HTML 4.01 Frameset

This DTD is equal to HTML 4.01 Transitional, but allows the use of frameset content.

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

In HTML 5, there is only one declaration and is written like this:

```
1 <!DOCTYPE html>
```

The doctype has a long and intricate history, but for now all you need to know is that this doctype tells the browser to interpret the HTML and CSS code according to W3C standards and not try to pretend that it is Internet Explorer from the 90's. (See quirks mode.)

HTML has a mechanism for embedding **comments** that are not displayed when the page is rendered in a browser. This is useful for explaining a section of markup, leaving notes for other people who might work on the page, or for leaving reminders for yourself. HTML comments are enclosed in symbols as follows:

```
<!-- This is comment text -->
```

A complete but small document

Putting this together, here is a tiny example of an HTML document. You can copy this code to a text editor, save it as *myfirstdoc.html*, and load it in a browser. Make sure you are saving it using the character encoding UTF-8. Since this document uses no styling it will look very plain, but it is only a small start.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>A RMScreenprint</title>
</head>
<body>
  <h1>Main heading in my document</h1>
  <!-- Note that it is "h" + "1", not "h" + the letters "one" -->
  <p>Look Ma, I am coding <abbr title="Hyper Text Markup Language">HTML</abbr>.</p>
</body>
</html>
```