

# Getting Started with HTML

Jukka Korpela

*This document is a generic introduction to HTML, the language used for World Wide Web documents ("WWW pages"). Some familiarity with WWW as a user (reader of documents) is assumed, but with regard to HTML no previous knowledge is assumed.*

## Contents

- Introduction
- Get ready...
- The requisites you need
- A closer look at our example and HTML notation
- Typing text - living with character code problems
- Organizing the contents: sections, headings, paragraphs, lists
- Emphasis and other classification
- Adding links
- Adding images
- Miscellaneous
- Check it!
- What next
- Summary

## Introduction

This document is generic in the sense of being independent of the particular version of HTML in use. It is not exhaustive with respect to any HTML specification. It aims at presenting the most basic and recommendable HTML constructs. The author has written a separate document about all features of HTML 3.2.

This document aims at being as easy as possible, but not easier. (But if you feel that you don't learn from this, don't give up; try some other introduction to HTML which might be better suited for you.) I have tried to avoid assuming any particular prior knowledge about HTML and Web authoring. But if you are wondering whether and why to create Web pages, please take a look at my essay *So you want to create a home page?*.

## Get ready...

When you wish to put text, images, or other pieces or sets of data (collectively called *documents* here) onto the World Wide Web, the normal way is to write an HTML file. There are other ways which can be taken in special cases, but HTML (HyperText Markup Language) is the "native" language of Web documents.

An HTML file is a text file, and it can be read by human beings, not only by computers. It contains special *markup*, however, so you will have to learn how to read such notation. The following is a simple HTML file:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<TITLE>Demonstration</TITLE>
<H1>Demo</H1>
<P>
```

For demonstration purposes only.

Please do not regard this as an example of a

# Getting Started with HTML

Jukka Korpela

```
<EM>useful</EM> Web page!  
</P>  
<P>  
This document was written by  
<A HREF="personal.html">Jukka Korpela</A> .  
</P>
```

We'll discuss the contents of the file later. But it wasn't that bad, was it? At least it is something that pretty normal people have learned to read and write. The first line probably looks confusing, however, and it is obligatory and really has a meaning. Fortunately you can, for most purposes, just copy it into your documents and otherwise ignore it. That particular line specifies that the version of HTML used in the document is HTML 3.2, and you need to change that if you use some other version.

If you think that the HTML file looks awful, you may feel tempted to listen to people who offer you "Web editors", which would hide the dirty HTML behind the scenes where it belongs. Look out! *It won't be any easier to learn to use a "Web editor" than to learn HTML* (and use whatever text editor you are used to or can learn in a few minutes). Moreover, Web editors (such as FrontPage) often produce HTML files which work only in some particular environment, since they don't obey HTML standards. Some Web editors *might* sometimes be useful e.g. for specifying tables (an advanced topic which will not be discussed here), since they might save the user from some routine work. But in order to be able to use a Web editor in a useful way - saving your time and work instead of wasting them - you have to know HTML so well that you *could* write it yourself!

When a user (you or someone else) accesses something which is commonly called a "Web page", the actual process behind the scenes is developed the following manner:

1. The user asks his *Web browser* (such as Netscape, Internet Explorer, Lynx, or Amaya) to show a document. This involves specifying the *Web address*, also known as *URL* (Uniform Resource Locator), either by just typing it or by following a *link* in another Web document.
2. The browser looks at the address. One part of the address specifies a *Web server* on which the document resides. The browser sends a request to the server, specifying a file name (which is another part of the Web address). - A Web server is a computer which hosts Web documents and makes them available upon request.
3. The server locates the requested HTML file on its disk and sends it (well, a copy of it, to be exact) to the browser.
4. The browser interprets the contents of the HTML file and formats it for display on the user's screen.

Perhaps you didn't quite follow? Well, the essential thing - from your point of view as a future author of Web documents - is that **a Web document is formatted for display every time it is fetched and presented to a user**. Even if the same user asked for the same document again after ten seconds, he might see it differently. He might, for example, have narrowed the window of his Web browser, which in general means that at least the division into lines is different. (The document itself might have been changed by its author, too!)

Let's presume that a user asks for a document with the simple contents of our trivial example. Different browsers will display it differently. For example, the text might be black on white, or something else - surprisingly many browsers use grey background by default.

Due to the presence of the funny looking EM (for emphasis) tags around the word "useful", that word might be displayed in *italics* or underlined or in **red** or even blinking. You cannot, as an author, know

# Getting Started with HTML

Jukka Korpela

what it really looks on the user's screen; such features can even be configurable by users themselves. The only thing which you can reasonably assume as an author about the EM tags is that they cause text to be presented in *some* manner which makes them look more prominent than normal text in your document. Generally speaking, HTML elements specify *logical relationships* rather than any particular physical representation.

The division of text into *lines* in the HTML file has *no* effect on what the document looks like on the user's screen. (There is an exception to this, called PRE element, but we'll come to that later.) This is natural, since the width of the users screen (and the window used by a Web browser) varies, and each user has his own preferences; I might wish to command my browser display 130 mm wide lines even if the screen is much wider, and it's really my own business.

Thus, a Web browser essentially throws all newline indicators away, taking the text as long piece of text, and inserts newlines as suitable, i.e. splits the text into lines as appropriate under the particular circumstances. Now you probably (and hopefully) say that this may screw things up badly. Indeed. This is one reason why HTML has special tags for indicating things like *paragraph division*. The tags <P> and </P> say that the text between them forms one paragraph, and a browser is expected to format it as such and *somehow* indicate the division into paragraphs to the reader. Typically, there is some extra vertical space between paragraphs. (You cannot know how much, and you shouldn't even ask, as an author.) But alternatively a Web browser could, for example, be in fact a speech synthesizer, which probably uses pauses for the same purpose.

The lesson is that you must think somewhat more *abstractly* than you are accustomed to, especially if you have used text processing or layout programs. Think in terms *what* you wish to express (e.g. emphasis or paragraph division), not *how* it should be expressed.

## The requisites you need

Anyone who can read and write can learn HTML. However, you probably want to learn HTML in order to create Web documents. You may hear different opinions about this, but the truth is that you only need

- some knowledge about the HTML language
- a text editor
- access to a Web server

You can use any text editor, provided only that it can save files in "text only" format (sometimes called "ASCII format" or "plain text format"). For instance, on Windows systems you can use NotePad; on Unix systems, you can use vi or Emacs. You can also use text processing systems like WordPerfect or MS Word, but this causes extra problems (e.g. you have to select the format to be "text only" instead of the default internal format) and doesn't bring real benefits. But the essential thing is that you can use your favorite text editor. (If you don't know any editor, you should find out what is the simplest editor in your system and learn to use it.)

Access to a Web server may sound like a complicated technical thing. However, here it only means that you can store files onto the disk of a computer acting as a Web server. You really have to find local information about the details, since they vary a lot. You need adequate permission to store files and you need related information about how to do that. Typically that information should tell you some directory names (e.g. on Web servers running Unix, a user should create a directory named public\_html into his home directory and use it for his personal Web pages) and recommended names for HTML document files (normally they should end with .html or .htm, depending on the system).

# Getting Started with HTML

Jukka Korpela

The Web server might be a computer to which you can log on and do all your Web authoring. Alternatively, you might have to create the documents elsewhere (e.g. on your PC) and transfer them onto a Web server (using e.g. the FTP program in a local area network or a modem connection).

## A closer look at our example and HTML notation

Let's have a closer look at our tiny example, reproduced here for your convenience:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<TITLE>Demonstration</TITLE>
<H1>Demo</H1>
<P>
For demonstration purposes only.
Please do not regard this as an example of a
<EM>useful</EM> Web page!
</P>
<P>
This document was written by
<A HREF="personal.html">Jukka Korpela</A>.
</P>
```

We already mentioned that the first line in a document is something you should put into the beginning of each HTML file without thinking about it too much. But it has a purpose: It specifies that the file contains an HTML document, as opposite to other possible document formats, and it specifies the HTML version used (in our examples, version 3.2). Usually Web browsers pay no attention to that line, but it is essential to so-called validators, which you should use to check your HTML code.

The next line begins with `<TITLE>` and ends with `</TITLE>`. As you may have guessed, these are the *start and end tags* for an HTML construct, the TITLE element. You can read the slash / in such contexts as *end of...*

The `<` and `>` characters are most appropriately called *angle brackets* in this context, but you should notice that in other contexts they are called and used as *less than* and *greater than* character. (The traditional character set used in computers, ASCII, is so small that language and software designers have to use one character for several purpose, which may of course cause confusion.)

Generally speaking, an HTML tag consists of

- opening angle bracket `<`
- a slash /, if the tag is to be an end tag
- tag name, such as TITLE; there is a fixed (relatively small) set of allowed tag names, although this set is partly different in different versions of HTML
- optionally, one or more *attributes*, separated from the name and from each other with some white space; each tag has its own set (possibly empty set) of allowed attributes; the attributes are of the form *name=value*
- closing angle bracket `>`

Tag names can be written in upper case or in lower case or even in mixed case, just as you like. For instance, `<Title>` and `<TITLE>` are equivalent.

# Getting Started with HTML

Jukka Korpela

Normally start and end tags are paired. The two tags together with everything between them constitutes an *HTML element*. (There are some tags, such as <BR> for line break, which neither need nor allow an end tag, and such a tag is an HTML elements by itself. In HTML 4.0, these tags are: AREA, BASE, BASEFONT, BR, COL, FRAME, HR, IMG, INPUT, ISINDEX, LINK, META, PARAM.) It depends on the element what is allowed as the contents of an element. For example, the TITLE element allows plain text only, whereas many other elements allow other elements as well, e.g. elements can be *nested* - not arbitrarily, but according to specific rules.

The detailed rules for using TITLE are the following: Every HTML document must contain a TITLE element, and only one TITLE per document is allowed. The contents of the TITLE element is to be used, in a browser-dependent manner, as an informative title for the entire document. It *does not appear as part of the document itself*, although a browser may display it in a special area of its window. You need a separate element if you wish to have a heading in the document itself (as you normally do). This may sound confusing, but there is wisdom behind it. The title is information *about* the document and it can be used in several contexts outside the document when referring to it; typically, so-called hotlists (or lists of favorites, or whatever they are called in different browsers) contain titles of documents for easy reference.

Thus, there are good reasons for writing a descriptive title. Our tiny example doesn't even try, for obvious reasons, but the title of the document you are reading right now is HTML primer. Notice that this happens to be different from the main heading you see at the beginning of the document.

The next line specifies a *header*, using H1 tags, which stand for 1st level (most prominent) headers. As you may guess, there are also H2, H3 and H4 tags available for 2nd, 3rd and 4th level headers. (Actually, the repertoire contains H5 and H6, too, but there are several good reasons to avoid using them.) Typically 1st level headers are displayed very visibly, using a very large font if available.

Then we have two *paragraphs*, enclosed between start and end P tags.

In the childhood of HTML, the <P> tag was regarded as paragraph separator, and several people still think that way. You can in fact omit the end P tags (</P>), but it is better to adopt the habit of thinking structurally.

The first of our paragraph contains one word within EM tags. This means that the word (or more generally, any text within EM start and end tags) is *emphasized*. It might, for example, be presented in italics or underlined or even in different color, according to browser features and user preferences. (If you are going to ask how to enforce **your** preferences in such issues, think twice. Are you writing the document for yourself?)

The second paragraph contains, in addition to normal text, the strange-looking construct

```
<A HREF="personal.html">Jukka Korpela</A>
```

This may look strange, but you really have to learn how to read and write such things, because they belong to the core of HTML. The construct sets up a hypertext link (hyperlink), or *link* for short. In this case, the text "Jukka Korpela" will be presented to the user in a distinguished manner, e.g. underlined (the presentation depends on the browser), and by selecting this phrase (typically, by a mouse click) the user will be able to view the document to which the links points - here it happens to be a personal home page with relative address personal.html. We will discuss the details of setting up links later.

# Getting Started with HTML

Jukka Korpela

## Typing text - living with character code problems

Since the characters `<` and `>` are used to designate HTML tags, they should not be entered as such, if they are to appear as data characters (e.g. in a mathematical expression like  $a < b$ ). For this purpose, so-called *escape sequences* are defined: `&lt;` for `<` and `&gt;` for `>`. This in turn makes the ampersand `&` a special character, and when it is to appear in the data, it shall be escaped as `&amp;`. Thus, for example, to produce the notation R&D you should type `R&amp;D`.

A bit clumsy, but such situations are pretty rare. But there are other character problems which may appear more frequently.

You can safely use the so-called printable ASCII characters:

```
! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ 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 [ \ ] ^ _
` 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 { | } ~
```

provided that you represent `&` and `<` and `>` as explained above.

All other characters, such as national letters `ä` and `é`, cause difficulties of some sort, partly because they have different internal representations in different computers. If you need non-ASCII characters, you will have to read additional information. In that case you might start from my more technical notes on character issues in HTML.

Typing text is easy, but you should remember the following:

- line breaks in HTML do **not** imply line breaks in the visual appearance of the document; in fact, in HTML line breaks are generally equivalent to blanks
- thus, you may use lines of any length you like, but for the ease of your work, use relatively short lines
- **do not divide a word** into two lines.

## Organizing the contents: sections, headings, paragraphs, lists

You should start with dividing your document into *major parts*, sections, according to its logical structure. Then you may proceed with dividing sections into subsections and perhaps even subsections into subsubsections. (For further division into pieces, using just paragraphs or lists is usually preferable to having named subsubsubsections.) Of course, very short documents do not need such divisions; you can just write a main heading and a few paragraphs.

Write descriptive *headings* for sections, subsections, and subsubsections. Basically there are H1, H2, and H3 tags for headings of different levels. There is, however, a problem here: HTML does not provide a separate element for specifying a heading for the *entire* document. (Remember that the TITLE element is for another purpose.) Therefore you have to *choose*:

- use H1 for both the overall document heading and top-level section headings, *or*

# Getting Started with HTML

Jukka Korpela

- use H1 only for the overall document heading, using H2 for sections, H3 for subsections, and H4 for subsubsections.

In the first case you may consider using the attribute `ALIGN=CENTER` to make the overall heading centered, e.g.

```
<H1 ALIGN=CENTER>Theory of Everything in a Nutshell</H1>
```

In the second alternative you may have problems with subsubsections, since browsers may display H4 headings in a non-distinctive way.

As regards to H5 and H6 headings, which are theoretically available, notice that popular browsers may display them as *smaller* than normal text! Even more ridiculously, if possible, some people have started using those tags to *intentionally* get smaller text as if all browsers behaved so strangely.

If you have a section with, say, H2 heading and containing H3 headings, avoid inserting text between the H2 heading and the first H3 heading. Such "homeless" text can be acceptable if it only contains *very short* notes such as general orientation, some remarks about the section, or a motto. Long homeless texts confuse the reader who does not see your good intentions; therefore, use a subsection with a heading of the appropriate level and with text like *Introductory remarks*, *Generalities* or *Summary*.

When entering text into the structure, under the lowest-level headings, use the P tags to mark up *paragraph*. Generally, there should be more than one and at most seven (or so) paragraphs under one heading. Otherwise you should consider changing the structure.

Thus, the structure of your document should be something like the following (assuming you follow the second strategy of using H1 tags mentioned above):

```
<H1 ALIGN=CENTER>Document heading</H1>
  <H2>First section</H2>
    <H3>First subsection of first section</H3>
      <P>Paragraph one. Bla bla bla...</P>
      <P>Paragraph two. Bla bla bla...</P>
    <H3>Second subsection of first section</H3>
      <P>A paragraph. Bla bla bla...</P>
      <P>Another paragraph. Bla bla bla...</P>
  <H2>Second section</H2>
  ...
```

(The indentation is here just to show the structure. You don't need it in actual HTML files.)

In addition to paragraphs, you may also use some special kind of blocks of text:

## BLOCKQUOTE

For quotations from external sources, such as a piece of text from a book. The use of these tags makes browsers present the text in a manner which distinguishes it from your normal flow of text (e.g. using italics or indentation or both). If the quotation contains several paragraphs, use P tags to show this.

## ADDRESS

For address information about the author. Might contain an E-mail address or a postal address. This too is presented in some distinctive manner suitable for such information.

# Getting Started with HTML

Jukka Korpela

## PRE

For preformatted text. This means that contrary to all normal rules of HTML, the text between `<PRE>` and `</PRE>` is presented as such with respect to the division into lines and use of blanks. This implies that a monospaced ("teletype") font is used, in contrast with proportional fonts normally used by graphic browsers. Suitable for presenting computer output and primitive "hand-formatted" tabular information.

If you have information which is most suitably presented as a *list*, you can use

- *unordered* list where items appear in the order written but with e.g. bullets instead of ordinal numbers
- *ordered* list where items have numbers 1, 2, 3, ... generated by the Web browser
- *definition* list which consists of term-definition pairs

An *unordered list* is presented in HTML as follows:

- begin with the start tag `<UL>`
- for each item of the list, type the list item start tag `<LI>` followed by the item itself; the list item end tag `</LI>` is optional and usually omitted
- end with the end tag `</UL>`

Example:

```
We sell:
<UL>
<LI> apples
<LI> oranges
<LI> bananas
</UL>
```

An *ordered list* is marked up in exactly the same way except for the start and end tags which are `<OL>` and `</OL>`.

A *definition list* is somewhat more complicated. The start and end tags are `<DL>` and `</DL>`. Within them, you will write items as follows:

```
<DT>term<DD>definition data for that term
```

Example:

```
Some hypertext concepts:
<DL>
<DT> anchor
  <DD> one of two ends of a hyperlink
<DT> hyperlink
  <DD> a relationship between two anchors, called
  the <DFN>head</DFN> and the <DFN>tail</DFN>;
  the link goes from the tail to the head
<DT>  <DD>
</DL>
```

There are some practical problems with definition lists, and you might find more appropriate tools among the more advanced features of modern HTML, such as tables. Here we will only remark that a natural way of *avoiding overly large lists* is to use normal text paragraphs preceded by headings. That is, make each term a heading (of suitable level) and provide its definition as text under the heading.

# Getting Started with HTML

Jukka Korpela

## Emphasis and other classification

When writing text by hand or using a typewriter or a text processing program, there are various means of emphasizing things, e.g. underlining, bolding, different colors, etc. In HTML, you should use the EM tags or, for stronger emphasis, STRONG tags. Leave it to the various browsers how they physically present the emphasis.

For example:

```
Avoid emphasizing too much, since emphasis means <em>separating</em> some important things from the normal flow of text. If the entire flow of text is "emphasized", you are in fact not emphasizing anything!
```

There are various other ways of classifying pieces of text (typically, a word or phrase):

### CITE

citation (title of a book or article or equivalent)

### CODE

computer program code or equivalent

### SAMP

sample output from e.g. computer program

### KBD

text to be typed by a user; this is typically used when giving instructions about computer usage

### I

text to be presented in italics, e.g. scientific names of animal species

### SMALL

text to be presented in a font smaller than normal

The classification tags are allowed to contain text and even other (nested) classification tags but not paragraph markers (P tags). If you really need to emphasize several consecutive paragraphs, you have to put the emphasis tags into each paragraph separately (e.g. <P><EM>paragraph text</EM></P>).

## Adding links

Presumably you roughly know, as a Web user, what a link looks like: typically there are words which are designated in some particular, browser-dependent manner (using e.g. underlining or special colors), and if you *select* a link (typically by "clicking it" with the mouse if you use a graphical browser), your browser will display a new document - the *target* of the link - to you. (Please notice that not all use of the Web involves mouses and clicking, so you should not use stupid phrases like *Click here!* in your documents.)

Now, to *construct a link* as an HTML author, you just type the following:

- the start tag <A HREF="*URL*">  
where *URL* is the target address (for the document to which the link shall point)
- the text which you wish to appear in your document as the link text (underlined, colored, or in whatever special way each browser presents it)

# Getting Started with HTML

Jukka Korpela

- the end tag `</A>`

The URL, or Web address, typically begins with `http://`. When looking at a document in a Web browser, you can normally see its URL somewhere in the browser window; it might be labelled URL or Netsite or Address or Location, for example. Copy it (using cut and paste, if possible) into your document. Notice that URLs published in newspapers or other printed media very often contain typos, and even URLs published in the Internet aren't always correct. For this reason, and for checking the real contents as well, you should check the URL with your browser before constructing a link.

The tag name `A` comes from the word *anchor*. (Some metaphors in the Web world are really strange.)

That's all it takes to set up a link. Our simple example HTML file contains the following:

```
<A HREF="personal.html">Jukka Korpela</A>
```

Here the link points to the personal home page of the author (that's me, by the way). This is a comfortable way to allow people find information about you without bothering those who don't like to see that information right now.

Other typical uses of links include

- separating details from main discussion, by just mentioning a document detail with a short phrase and making that phrase a link
- creating a file which acts as a *table of contents* by containing just a list of links
- referring to a related document, such as a discussion of the same theme by someone else or a translation of the current document into another language

A typical example of a link into detailed information is the following:

```
The exact specifications of the syntax of URLs are in  
<a href="ftp://ftp.funet.fi/pub/doc/rfc/rfc1738.txt">RFC 1738</a>  
(absolute URLs) and  
<a href="ftp://ftp.funet.fi/pub/doc/rfc/rfc1808.txt">RFC 1808</a>  
(relative URLs).
```

The example is somewhat atypical in the sense of using [ftp URLs](#) instead of `http` URLs. However, there are many useful resources accessible that way.

More generally, links are a powerful tool for creating documents which allow each user to select what interests him. Unfortunately, there is no well-supported structured way of telling what kind of information the link points to, i.e. what the user is going to get by selecting the link, though there are some techniques for expressing the nature of a link. In many cases you can rely on intuition, but you might need to consider formulating your sentences so that the context is a sufficient clue to the idea behind each link. As the last resort, add a parenthetic remark after the link, such as "(technical details)". If our simple example which refers to RFCs appears in a document for a wide audience, we should probably add a note which tells what RFCs are - possibly referring, with a link, to a longer explanation of the nature of RFCs.

In addition to linking to a document as a whole, you can make a link point to a particular **location** in the same or another document. When a user selects such a link, the document will be displayed positioned to that location. (What this really means depends on the browser. It is not always obvious to the user exactly which location is pointed to.) To create such a link, use the same method as above

# Getting Started with HTML

Jukka Korpela

but append the character # and a location name to the URL; for example, such a link above has been coded as follows:

```
- - there are  
<a href="4.7.html#linknature">some techniques for  
expressing the nature of a link</a>.
```

This example links to location named linknature in the document 4.7.html This only works if that document really contains such a location; in practise this means that it contains a structure of the form `<A NAME="rfc">some text</A>`

Thus, such a structure specifies that the beginning of *some text* is a location to which links can refer. In our example, the markup in the other document is

```
<H3><A name="linknature">Expressing the nature of a link</A></H3>
```

Note the correct nesting of A elements and heading elements. Since A elements may contain text-level elements only, they need to appear *inside* heading elements.

You can, of course, refer to locations in *other people's documents* only if they have named locations that way. Using your Web browser (with a function named *View source* or *View HTML* or something like that) you can peek at people's HTML constructs to see if they contain named locations and what the names are. In *your own documents* you can yourself decide where to put named locations. It might be a good idea to put them at the beginning of each major part, since this allows others to link to such parts when appropriate.

As a special case, when referring to a location *within the same document*, the URL can be omitted, using just the location specifier. Thus, if your document contains a preface with the heading

```
<H1><A NAME="preface">Preface</A></H1>
```

then you could refer to it as follows:

```
As mentioned in the <A HREF="#preface">preface</A>, our  
discussion bla bla bla...
```

## Adding images

The ability to link to images and to embed images into documents is an essential feature of HTML. An image could be a photograph, or a drawing, or graphic presentation produced by a computer, for instance.

If you want to put images onto Web, the most difficult and time-consuming job is to create a file which contains a JPEG or GIF presentation of the image. You need not know what JPEG and GIF are, except that they are specific standardized formats of presenting graphic information in digital (i.e. computer readable) form; JPEG is more suitable for photographs, GIF for other graphics. But you need to find programs which can produce a JPEG or GIF presentation for your graphics. For example, you might find a scanner which can read a paper or photograph and produce JPEG or GIF, or an interactive graphics program with which you can do some drawing on the screen and get it stored as JPEG or GIF. But be prepared for some complications; for instance, you may have to use a scanner which can only produce some strange graphics format *X*, and then you have to find a conversion program which converts from *X* to GIF.

# Getting Started with HTML

Jukka Korpela

There is a large number of other graphics formats, too, and even formats for movies and sounds. However, JPEG and GIF are the most universally supported in Web browsers, so use them for images to be put onto the Web, unless there is a very good reason do otherwise.

Now let us assume that you have made your way thru that. For definiteness, let's assume that you have a photo of yourself in JPEG format in a file named myphoto.jpg, in the same directory where your HTML files are (on a Web server). The rest, the HTML side of the issue, is quite easy:

- To put a *link* to the photo into a Web document, use the A tags described above, specifying the file name as URL, e.g.
  - There is a `<A HREF="myphoto.jpg">photo of me</A>` available.
- To *embed* the photo into a Web document, use the IMG tag as follows:
  - `<IMG SRC="myphoto.jpg" ALT="a photo of me">`

Notice that IMG is an element by itself; there is no end tag. You may wish to put an explanatory text under the image. It requires no special tags; just write the text. The ALT attribute is displayed *instead of* the image, if a browser cannot display the image for some reason (such as text-only terminal!), or as a *description of* the image, when automatic loading of images has been disabled by the user. This makes design of good ALT texts sometimes difficult, but normally you should aim at helping the latter group of users to decide whether to load the image or not. If the image is purely decorative, use `ALT=""`.

For several reasons, *it is usually better to link to images than to embed them*, especially if images are large or there are lots of them.

## Miscellaneous

You can force a **line break** using the `<BR>` tag. It is an element by itself: no end tag is needed or allowed, so there can be no contents either. It does not cause a paragraph break (e.g. empty line). The most typical use is in address information like the following:

```
<ADDRESS>
Jukka Korpela<BR>
Päivänsäteenkuja 4 as. 1<BR>
FIN-02210 Espoo<BR>
Finland
</ADDRESS>
```

You can indicate **change of topic** using the `<HR>` tag. It is an element by itself, too. It typically causes a full-width horizontal rule to be drawn. Authors often use `<HR>` after the document content proper, to separate that from final remarks about authorship, copyright, etc. You may also, for example, separate the major sections of your document from each other in such a manner.

## Check it!

People make typing errors a lot. In HTML, a simple typo may have a serious effect on the entire document. For instance, if you forget the slash from the end tag `</H1>`, you would typically get *all the rest* of your document printed as a top-level heading!

Therefore, and for other good reasons as well, you should ask a suitable program to do some basic formal checking of your HTML file. Several **validation programs**, such as the WDG Validator, and miscellaneous checkers such as the W3C Link Checker exist. for a short list.

# Getting Started with HTML

Jukka Korpela

Passing such a check does not mean that your HTML file is sensible in every respect, but the checks do find a lot of errors.

Checking what the document *looks like* on your browser is a good thing to do. But remember that it only shows what it looks like to you under the particular circumstances. Therefore, it is important to follow HTML specifications and to use validators, in order to try to make your document reasonably presented in any environment, on any browser, current and future. If your document is important, consider the additional check of looking at it on different browsers and different screens and windows.

## What next

This document has described only a basic part of HTML. There are lots of other tags, and even the tags described here may have additional options and features which do not belong to this fundamental introduction.

When you have read this document and exercised the basic features of HTML in practice, consult *Learning HTML 3.2 by Examples*. In addition to describing all elements and attributes of HTML 3.2, the current HTML standard, it contains links to suggested reading, such as good documents on Web authoring in general.

Especially if you are going to begin with a *personal home page* on the Web, you should actually read the document *So you want to create a home page?* first.

## Summary

Title and headings	
<TITLE>	title associated with the document
<H1 ALIGN=CENTER>	top-level heading, centered
<H1>	top-level heading
<H2>	second-level heading
<H3>	third-level heading
Blocks of text	
<P>	normal paragraph
<BLOCKQUOTE>	quotation from external source
<ADDRESS>	address info about author
<PRE>	preformatted text
Lists	
<UL>	unordered list
<OL>	ordered list
<LI>	list item
<DL>	definition list

# Getting Started with HTML

Jukka Korpela

<DT>	term in definition list
<DD>	definition data for term
<b>Classification of phrases (text markup)</b>	
<EM>	emphasized text
<STRONG>	strongly emphasized text
<CITE>	citation (title of a book or article or equivalent)
<CODE>	computer program code or equivalent
<SAMP>	sample output from e.g. computer program
<KBD>	text to be typed by a user
<I>	text to be presented in italics
<SMALL>	text to be presented in a font smaller than normal
<b>Hypertext links</b>	
<A HREF="URL">text</A>	link to a document
<A HREF="URL#name">text</A>	link to a named location in a document
<A HREF="#name">text</A>	link to a named location within the same document
<A NAME="name">text</A>	names a target location for links
<b>Other elements</b>	
<IMG SRC="URL" ALT="text">	image to be embedded
 	forced line break
<HR>	change of topic (horizontal rule)

---

Copyright © 1997 - 2005 [Jukka Korpela](http://www.cs.tut.fi/~jkorpela/html-primer.html).

The author hereby gives general permission to copy and distribute this document or parts thereof in any medium, provided that all copies contain, in a manner appropriate for the medium, an acknowledgement of authorship and the URL of the original document, i.e. <http://www.cs.tut.fi/~jkorpela/html-primer.html>

The permission granted above does not imply permission to distribute this document in a modified form or as a translation.