Writing Web Pages using HTML

Mark O. Stitson

September 6, 1999

Department of Computer Science
Egham, Surrey TW20 0EX, England

The main text of this document is set in Computer Modern 11pt.

This document was prepared with LaTeX using a PC running Linux.

It was printed on a 600 dpi Hewlett Packard laser printer.

Contents

Introduction

I originally wrote this book as part of a UNIX Users' Course. Since then it has developed a life of itself and now stands as a short introduction to writing HTML.

While writing this booklet I was working on a range of different systems and my experience is drawn from all of them. They included Sun SparcStations, HP Workstations, DEC MIPS Workstations, DEC Alpha Servers and PCs running UNIX, DEC VAXes and Alphas running VMS, PCs running DOS and Windows and Apple Macintosh Computers.

All systems have their advantages and disadvantages. This booklet is not specifically tied to any operating system although all examples and exercises use a UNIX system. 90 % of all Web servers run a UNIX operating system, which means most authors of web pages come into contact with a UNIX system at some point.

Writing this course would have been impossible without the help of the systems staff, Jon, Will and Mark, of the Department of Computer Science at Royal Holloway and especially Nick who encouraged me to start on this project.

September 1999

The Author

Chapter 1

The Internet

This chapter will discuss the Internet in general terms and is applicable not only to Unix, although the Internet is predominantly made up of Unix computers.

1.1 A little History and Background

The Internet is the world's largest computer network. Its basic service is to provide communication channels between computers. In this way it does not much differ from a Novell, Microsoft or Banyan Vines network, to name the largest selling PC networks. It does however differ in size and variety. As there is no central administration of the Internet nor does anybody own it, it is hard to estimate its size, but estimates five years ago speak of 3 to 4 million computers and 30 to 40 million users.

There are also many services available on the Internet, which are slowly making their way into areas not covered by standard PC networks. Although the standard services of printing and file serving are available in the Internet, these are in contrast to PC networks not usually as efficient and often organizations will use normal PC networking in parallel with Internet services on the same piece of wire.

1.1.1 A Brief History

The Internet is about 25 years old. It was originally designed as a military network (ARPA net), which could withstand partial failures, in case of bomb attacks and the like. A few years later the American National Science Foundation decided to create five linked up supercomputer centres to offer services to educational universities and research establishments. As it would have been too expensive to give everybody a direct link up, everybody was just linked to their nearest neighbour using the ARPA net technology: the Internet Protocol.

1.1.2 What makes up the Internet

A few years ago this question would have been easy to answer: All computers using the IP protocol which are connected to each other directly or indirectly. Today this question is a lot harder to answer. There are bulletin board systems which allow their user to send Internet mail and read Internet news, there are services like Compuserve or the Microsoft Network which offer commercial services next to some extracts from the Internet. These services do not offer full Internet access, but do provide the most popular services. So it is a hard question to answer and probably cannot be answered.

1.1.3 Who runs the Internet

No single organization runs the Internet, no single organization owns the Internet. The Internet is run by agreement of the participants. They decide on standards and related issues. This fact makes the Internet a network with ultimate freedom[1]. Whereas Compuserve or Microsoft can be held responsible for their world-wide bulletin board systems and in turn can hold their users directly responsible for their actions, this is not so in the Internet. This freedom of course creates dangers as well and means users of the Internet have to behave themselves in the Internet-public.

1.1.4 What does the Internet actually offer

The Internet seems to have grown into a super-information highway in the hands of advertising consultants, who themselves do not really know what it is.

The main services offered to the general public in the Internet are:

- ◇ telnet - Use of remote computers
- ◇ ftp - Retrieval of files from remote sites
- ◇ e-mail - Sending messages to people
- ◇ news - The world's largest bulletin board system

[1] Within the bounds of the applicable legislation, of course. Which legislation applies is as yet an unresolved issue though.

- ◇ archie - Finding software
- ◇ gopher - Browsing through the Internet's resources (text-based)
- ◇ WAIS - Browsing through the Internet's databases (text-based)
- ◇ World Wide Web - Browsing through the Internet's resources using hypertext

As you can see the last three services all concern browsing and we will not go into gopher or WAIS any further because they nearly all can be accessed through the World Wide Web.

There are two further widely used services available within the Internet framework, but they are not usually used or indeed designed for the use outside of a site. These services are file serving and printing. They are hardly ever available to the general public for obvious reasons.

1.2 DNS

To access any of the Internet services we have to know the name or address of the computer we are trying to get a service from. Every computer connected to the Internet has a unique IP-address. This is a 32bit number. Domain Name Services were invented to relieve the pain of remembering 32bit numerical addresses. DNS gives every computer a name. Computers can have multiple addresses and multiple names, but this does not matter for our purposes. DNS names are organized hierarchically. Each country has a short abbreviation in the DNS scheme, except America which is divided along slightly different lines. The UK has uk as its abbreviation, Switzerland has ch, France fr, Germany de and Finland fi to name just a few. Within the countries each country organizes its own naming scheme, so I will concentrate on the UK. The next part of the DNS name lists the type of organization. ac stands for academic, co for commercial and gov for governmental. The next part of the name is an abbreviation of the organizations name. After that it is up to the individual organization to subdivide its domain. Royal Holloway has decided to give Departments individual names and the computers within those departments. Say we want to ask our computer to look up the numerical address of a computer at Royal Holloway. Say its individual name was platon and it was located in the computer science department. We know Royal Holloway is an academic institution within the uk so the last part of the computer's full name is ac.uk. Note how the parts of the name are separated by a dot. Royal Holloway's abbreviation is rhbnc[2] and

[2]Royal Holloway and Bedford New College

the department of computer science is abbreviated as cs or dcs. The full name of the computer therefore would be platon.cs.rhbnc.ac.uk. Most programs we will be using will automatically find the numerical address of another computer, but sometimes it is useful to check if we have got the right name. This can be done by checking if there is a numerical address for that name. We use the nslookup program[3]:

```
$nslookup platon
Server:  platon.cs.rhbnc.ac.uk
Address:  134.219.96.1

Name:    platon.cs.rhbnc.ac.uk
Addresses:  134.219.96.1, 134.219.44.52, 134.219.97.1, 134.219.80.1
$
```

The nslookup program can also find the name of a computer given its numerical address.

We now know the other computer exists, but is it switched on? We can find this out by 'pinging' the other computer using the ping command. The ping command sends a reply request to the other computer and measures the time it takes the other computer to reply[4]:

```
$ping platon
PING platon (134.219.96.1): 56 data bytes
64 bytes from 134.219.96.1: icmp_seq=0 ttl=255 time=0.8 ms
64 bytes from 134.219.96.1: icmp_seq=1 ttl=255 time=0.7 ms
64 bytes from 134.219.96.1: icmp_seq=2 ttl=255 time=0.7 ms

--- platon ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.7/0.7/0.8 ms
$
```

Ping can be stopped on UNIX systems by typing CTRL-C on Windows systems it will run just four pings.

Exercises

1. Find the numerical address of the following computers:

[3] This is not available under Windows.

[4] Under Windows open a DOS box and run the ping program.

- ◇ www.cs.rhbnc.ac.uk
- ◇ www.bmw.co.uk
- ◇ www.idg.co.uk
- ◇ freud.cs.rhbnc.ac.uk

2. Find the name corresponding to the following IP-addresses:
   - ◇ 199.41.254.163
   - ◇ 134.219.96.129
3. Now ping these computers to find if they are alive using either the IP-address or name.

1.3 Telnet

Accessing a remote machine might not seem too useful at first, but does have its advantages. Obviously if you want to use a more powerful computer, you are allowed access to, but it would take five hours to get there, this is useful. Yet even when you are a normal user it can have advantages. You can read the mail you received on your computer at home while you are hundreds or thousands of miles away. All you need is a basic Internet connection. Other services that can often be accessed remotely using telnet are library services, directory services and other information services like archie.

The telnet command usually takes the remote machine's name as a command line argument and in some special cases the specific port number[5]. To use telnet we always have to know the remote machine's DNS name or IP address. Say we want to connect to the remote machine einstein. After the connect message appears the remote machine takes over and starts its login process[6]:

```
$telnet einstein
Trying 134.219.96.116...
Connected to einstein.cs.rhbnc.ac.uk.
Escape character is '^]'.

Linux 2.0.17 (einstein.cs.rhbnc.ac.uk) (ttyp2)


einstein login: markst
```

[5] A port determines to which service on the remote machine a connection will be made.
[6] Under Windows you can run telnet from a DOS box.

```
Password:
Last login: Thu Mar 21 20:51:40 on tty5
Linux 2.0.17. (POSIX).

$logout
Connection closed by foreign host.
$
```

We can now use the remote machine as we would use the local machine. If we want to use applications which require their own windows we have to give the remote machine some more information, such as which display to use etc.

Exercises

1. Try and telnet to the machines that responded to your `pings` in the last set of exercises.
2. Now telnet back to `platon` from the remote computer.
3. Log out of `platon` using the `logout` command. This will put you back to the remote computer. Then log out of the remote computer.

1.4 ftp and Archie

`ftp` is the internet's file transfer protocol. It allows users using the `ftp` program to retrieve files from computers offering ftp services. Archie is a services which allows the location of files in the worlds major public ftp archives.

1.4.1 ftp

The `ftp` program[7] is started like the telnet program with the remote computer's name on the command line. The remote computer asks us the name we want to log in as and then for our password, depending which username we use different files become available to us[8].

```
$ ftp einstein
```

[7]The `ftp` program discussed here is the standard program which come with most UNIX systems and Windows 95. There are of course `ftp` programs with graphical user interfaces which provide exactly the same functionality.

[8]Under Windows you can run ftp from a DOS box.

```
Connected to einstein.cs.rhbnc.ac.uk.
220 einstein FTP server (Version wu-2.4(1) Tue Jun 6 13:20:57 MET DST 1995)
ready.
Name (einstein:markst):
331 Password required for markst.
Password:
230 User markst logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

We can now issue ftp commands:

- ◇ ls - List the current remote directories contents
- ◇ cd - Change the current remote directory
- ◇ lcd - Change the current local directory
- ◇ bin - Switch to binary transfer mode
- ◇ ascii - Switch to text transfer mode
- ◇ hash on/off - Switch printing of file transfer progress on or off
- ◇ get - Retrieve a file (can be followed by filenames like cp)
- ◇ mget - Retrieve multiple files (can be followed by filenames like cp[9])
- ◇ put - Send a file (can be followed by filenames like cp)
- ◇ mput - Send multiple files (can be followed by filenames like cp)
- ◇ help - Lists all commands for this ftp program (can be followed by a command name to get more information)
- ◇ quit - Quit the ftp program

If we ftp to a public archive, we cannot log in with our user name, as the archive would have to know all user names and passwords in the world to allow this. Instead we have to log in as user anonymous and give our e-mail address instead of a password:

[9] cp is the UNIX file copying command.

```
$ftp nietzsche
Connected to nietzsche.cs.rhbnc.ac.uk.
220 nietzsche FTP server (Version wu-2.4(1) Tue Jun 6 13:20:57 MET DST 1995) ready.
Name (nietzsche:markst): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-Welcome, archive user!  This is an experimental FTP server.  If have any
230-unusual problems, please report them via e-mail to root@nietzsche
230-If you do have problems, please try using a dash (-) as the first character
230-of your password -- this will turn off the continuation messages that may
230-be confusing your ftp client.
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> quit
221 Goodbye.
$
```

1.4.2 Archie

Archie is the Internet service we use when we are looking for a file. This could be an update for our software, a bug fix or even a recipe. Archie can be accessed in three ways. Accessing archie via e-mail is very cumbersome and replies take a while. Accessing archie via telnet[10] is not very user-friendly but very fast. The best way to access archie is by using a local archie client, which gives you a nice user interface and takes care of all the messy archie commands.

On many UNIX systems the xarchie program is installed. It provides a friendly user interface[11]:

[10] In the UK archie can be accessed by telnetting to archie.doc.ic.ac.uk.
[11] Archie does not come with Windows by default.

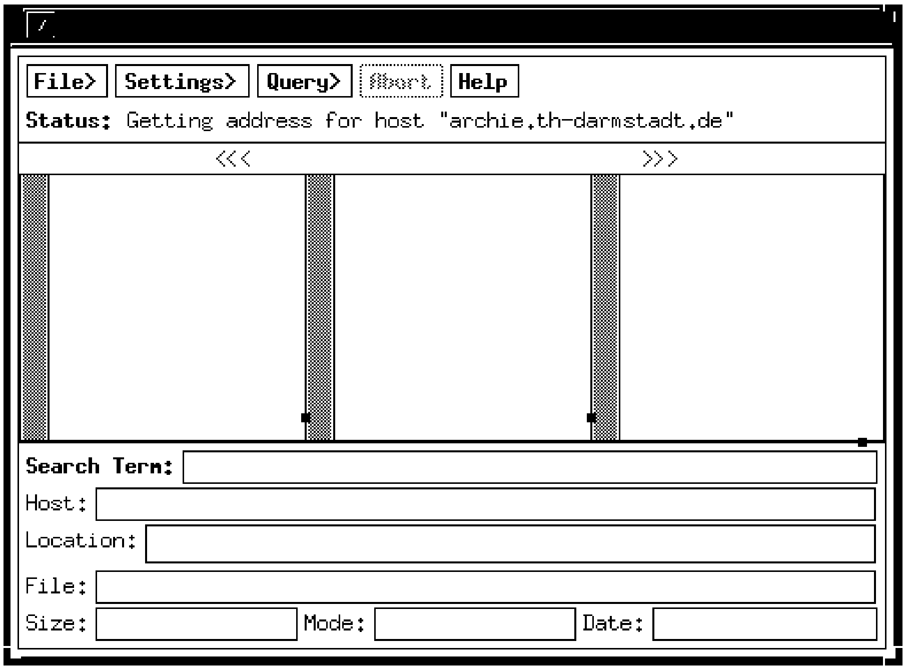


Figure 1.1 Xarchie

We can simply type the filename or a substring of it into the "Search Term" field and hit return. We will be returned a list of archives on which the files found reside, can choose them and retrieve them.

Exercises

1. ftp to einstein and find out what files are in the current directory on the remote machine.
2. Retrieve the file `/tmp/welcome`
3. Use xarchie to find the file called netscape[12].
4. Now retrieve one of the files you found.

1.5 Interactive Communication

There are two ways of directly communicating with other users over the Internet: Electronic Mail and Talk.

[12] You might have to change the archie host in the setting menu, if the default host does not work.

Talk is being replaced by a multitude of chat programs and has fallen into disuse, so we wont discuss it here.

1.5.1 Finding someone's user name

There are directory services available for e-mail systems, which find a users login name. One is the X.500 service which is available on the world wide web at address http://vms.rhbnc.ac.uk. An easier way of finding somebody's login name is to ask the machine he is using. The finger command can find a user with a given real name on a computer[13]:

```
$finger stitson@nietzsche
[nietzsche.cs.rhbnc.ac.uk]
Login: markst                         Name: Mark O. Stitson
Directory: /home/local/markst         Shell: /bin/bash
Last login Wed Mar 27 21:37 (GMT) on ttyp2 from :0.0
Mail last read Mon Apr  1 21:05 1996 (GMT)
No Plan.
```

Exercises

1. finger the following users:
   - markst@@einstein
   - markst@@platon
   - will
   - root
2. Find the user names of the other students on the course using the finger command.
3. Group up with another student on the course and talk to them.

1.6 Electronic Mail

Electronic Mail allows users to communicate with each other, by sending messages which can be read by the receiver at a later time.

[13] This service does not come with Windows.

Usually it is sufficient to know the recipient's user name and the name of the machine he receives mail at. Some recipients though are not using services directly connected the Internet and therefore have to have their mail addressed in other ways. A few examples of this are given below.

An Internet e-mail address is usually written in the form *user*@@*computer*, the computer name is case insensitive, but the user name can be case sensitive. It is hard to guess other peoples e-mail address. Some organizations though have intelligent mail hosts which allow you to specify the name instead of the user name, e.g. M.Stitson@@rhbnc.ac.uk. This can often take longer to arrive; at RHBNC up to eight hours later! It is generally safer to use the proper e-mail address, e.g. markst@@dcs.rhbnc.ac.uk or M.Stitson@@dcs.rhbnc.ac.uk. Note that rhbnc.ac.uk and dcs.rhbnc.ac.uk are not actually computers, but synonyms for the computers handling the e-mail. Your e-mail address will consist of your user name followed by @@dcs.rhbnc.ac.uk.

Mail in the Internet is directly transferred to the recipient's e-mail handling computer. This means that if that computer is switched off, the e-mail will be returned to the sending computer and after a certain number of retries will be returned to the sender with an appropriate message.

1.7 Internet News

Network news is the Internet's answer to the bulletin board systems of the 70s and 80s. Although Network News existed before the Internet as USENET, it has become widespread through the massive expansion of the Internet. Network News is divided into groups along topic lines. Each group is a discussion forum for a certain topic. It is possible just to browse through the groups or to actively participate in the discussions.

Although Internet News is a very useful system it lacks the nice graphical front end of many of the new bulletin board systems. Internet News connects many people all over the world, whereas the new bulletin board systems only see themselves and stand isolated.

1.8 World Wide Web - An Introduction

The World Wide Web is a very recent addition to the Internet (well it was when this course was written in 1995). It was born at CERN, the famous Swiss Particle Research Institute. The amount of written information had grown to a stage at which it was impracticable to have it in paper form. It was decided to computerize all information using a Markup Language. In addition it was decided to make it possible to insert electronic cross-references into the documents. Thus the Hypertext Markup Language HTML was born. It was not the first of its kind, nor is it the most powerful or easy to administer. Yet its initial simplicity gave it a head start over better concepts such as Hyper-G.

HTML defines roughly the appearance of a document, yet the actual image the user will see depends on the browser program used. The very basic browsers offer a text-only interface, which cannot display any graphics, yet modern browsers can display moving images and play sounds, not to mention running Java and Javascript programs.

1.9 Publishing on the Web

There are two sides to using the web. The first is the provider of information who runs a HTTP[14] server, which gives out information. The other side is the user with a browser, that requests information from a server. I am assuming you are familiar with browsing the web and this section will only cover how information can be provided for retrival.

This is a short and basic introduction for producing documents in HTML, the markup language used by the World Wide Web. It is based on the National Center for Supercomputing Applications' primer / pubs@@ncsa.uiuc.edu. The original document is available at:

http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html

1.9.1 Acronym Expansion

WWW World Wide Web (or Web, for short).

SGML Standard Generalized Markup Language – this is a standard for describing markup languages.

DTD Document Type Definition – this is a specific markup language, written using SGML.

HTML HyperText Markup Language – HTML is an SGML DTD. In practical terms, HTML is a collection of styles (indicated by markup tags) that define the various components of a World Wide Web document. HTML was invented by Tim Berners-Lee while at CERN. He is now director of the W3 Consortium.

[14] Hypertext Transfer Protocol

1.9.2 Creating HTML Documents

HTML documents are in plain (also known as ASCII) text format and can be created using any text editor (e.g., Emacs or vi on UNIX machines or notepad or Word on Windows machines). Some Web browsers include rudimentary HTML editors in a WYSIWYG[15] environment. There are also some WYSIWIG editors available now (e.g. HotMetal for Sun Sparcstations, HTML Edit for Macintoshes or saving from Word as HTML). These WYSIWIG editors often introduce a lot of unnecessary instructions into your documents. These instructions are completely ignored by the browsers, but slow down browsing the web significantly.

While editing HTML documents it is possible to preview them with your Web browser. Documents can be opened as local files in the file menu of your browser. After editing the source HTML file and saving the changes, hit the reload button on your browser and the changes will be reflected in the reloaded document.

1.9.2.1 The Minimal HTML Document

Here is a bare-bones example of HTML:

```
<TITLE>The simplest HTML example</TITLE>

<H1>This is a level-one heading</H1>

<P> Welcome to the world of HTML.

This is one paragraph.</P>

<P> And this is a second.</P>
```

The formatted version looks like this:

[15] What You See Is What You Get

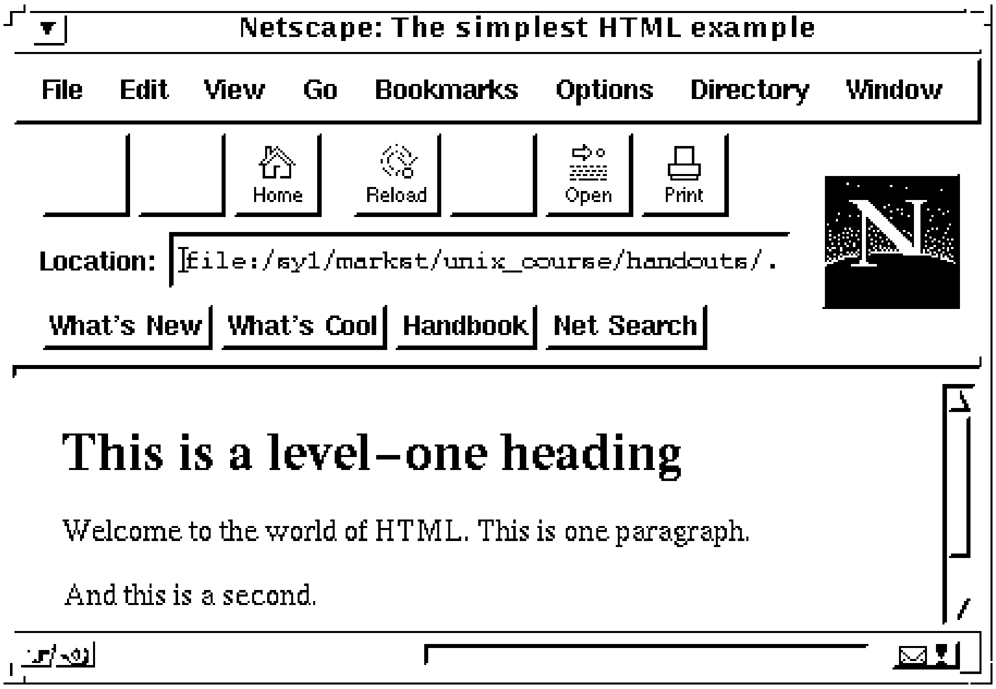


Figure 1.2 A basic HTML example

HTML uses markup tags to tell the Web browser how to display the text. The above example uses:

- the `<TITLE>` tag (and corresponding `</TITLE>` tag), which specifies the title of the document
- the `<H1>` header tag (and corresponding `</H1>`)
- the `<P>` paragraph-separator tag (and corresponding `</P>`)

HTML tags consist of a left angle bracket (<) followed by name of the tag and closed by a right angule bracket (>). Tags are usually paired, e.g. `<H1>` and `</H1>`. The ending tag looks just like the starting tag except a slash (/) precedes the text within the brackets. In the example, `<H1>` tells the Web browser to start formatting a level-one heading; `</H1>` tells the browser that the heading is complete. The primary exception to the pairing rule is the `<HR>` tag, which produces a horizontal line. There is no such thing as `</HR>`.

NOTE: HTML is not case sensitive. `<title>` is equivalent to `<TITLE>` or `<TiTlE>`.

Not all tags are supported by all World Wide Web browsers. If a browser does not support a tag, it just ignores it.

1.9.2.2 Basic Markup Tags

Title Every HTML document should have a title. A title is generally displayed separately from the document and is used primarily for document identification in other contexts (e.g., a WAIS search). Choose about half a dozen words that describe the document's purpose. In the X Window System and Microsoft Windows versions of Netscape, the **Document Title** field is at the top of the screen above the pulldown menus.

Headings HTML has six levels of headings, numbered 1 through 6, with 1 being the most prominent. Headings are displayed in larger and/or bolder fonts than normal body text. The first heading in each document should be tagged `<H1>`. The syntax of the heading tag is:

`<H` *y* `>` *Text of heading* `</H` *y* `>`

where *y* is a number between 1 and 6 specifying the level of the heading.

Title versus first heading In many documents, the first heading is identical to the title. For multipart documents, the text of the first heading should be suitable for a reader who is already browsing related information (e.g. a chapter title), while the title tag should identify the document in a wider context (e.g., include both the book title and the chapter title, although this can sometimes become overly long).

Say we have a book about the internet which includes a chapter on HTML with a section on tags. The first heading of the document should be "Tags" whereas the title should be "Internet HTML Tags" or similar.

Paragraphs Unlike documents in most word processors, carriage returns in HTML files are not significant. Word wrapping can occur at any point in your source file, and multiple spaces are collapsed into a single space. (There are couple of exceptions: spaces following a `</P>` or `<H` *y* `>` tag, for example, are ignored.) Notice that in the bare-bones example, the first paragraph is coded as

```
<P> Welcome to HTML.
This is the first paragraph. </P>
```

In the source file, there is a line break between the sentences. A Web browser ignores this line break and starts a new paragraph only when it reaches a `</P>` tag.

Important You must enclose paragraphs in <P> - </P> pairs. The browser ignores any indentations or blank lines in the source text. HTML relies almost entirely on the tags for formatting instructions, and without the <P> - </P> tags, the document becomes one large paragraph. (The exception is text tagged as "preformatted," which is explained below.) For instance, the following would produce identical output as the first bare-bones HTML example:

```
<TITLE>The simplest HTML example</TITLE><H1>This is a level

one heading</H1><P>Welcome to the world of HTML. This is one

paragraph.</P><P>And this is a second.</P>
```

However, to preserve readability in HTML files, headings should be on separate lines, and paragraphs should be separated by blank lines (in addition to the <P> - </P> tags).

The advantage of using the pair <P> - </P> of paragraph enclosing tags as opposed to earlier HTML which used an end of paragraph tag only, is that it is possible to specify formatting options for a paragraph. For example it is possible to center a paragraph by coding

```
<P ALIGN=CENTER>

This is a centered paragraph.

</P>
```

Decorative Tags There are three more basic tags which have a mainly decorative function.

The <HTML> tag is used to indicate the start of the HTML. This must be the first tag in your document, as your document may only contain HTML. A closing </HTML> tag has to be appended to the end of the document.

The <HEAD> and <BODY> tags are more useful which will be explained later. Their use is to separate the title which appears separately on most browsers from the body of the HTML.

Our above example would look like this, if we inserted the extra tags:

```
<HTML>

<HEAD>
<TITLE>The simplest HTML example</TITLE>
</HEAD>

<BODY>
```

```
<H1>This is a level one heading</H1>
<P>Welcome to the world of HTML. This is one paragraph.</P>
<P>And this is a second.</P>
</BODY>
```

1.9.2.3 Putting you documents on the web

We know how to edit and display HTML files using an editor and the Open File function on our browser, this however has not put our document on the web. To put documents on the web our system administrator has to assign us special directories. Luckily most system administrators assign every user a directory by default. In our case, which is the standard way of doing this, the directory we have to use for our personal web pages is ~/public_html. This directory automatically translates to the URL http://www.dcs.rhbnc.ac.uk/~*username*.

To create a new web page[16] we first have to go to our public HTML directory:

```
$ cd /CS/www/home/
```

username

Every web server also defines a default file to load from a directory. Usually this is `index.html`. So our top-level page should be `index.html`. If this file does not exist the server will show the directory contents of that directory:

```
$ emacs index.html &
```

Now we can edit the file, save it, and test it giving our web browser the appropriate URL.

Exercise

1. Try setting up a basic web page for yourself now. It should contain a little description of yourself and a line giving the author of the page and the last modification date.

1.9.2.4 Linking to Other Documents

The chief power of HTML comes from its ability to link regions of text (and also images) to another document. The browser highlights these regions (usually

[16] If you now about UNIX this description is a simplification and not completely accurate, but will produce the required results.

with color and/or underlines) to indicate that they are hypertext links (often shortened to *hyperlinks* or simply *links*).

HTML's single hypertext-related tag is `<A>`, which stands for *anchor*. To include an anchor in your document:

1. Start the anchor with `<A` . (There's a space after the A.)
2. Specify the document that's being pointed to by entering the parameter
   `HREF="`*filename*`"`
   followed by a closing right angle bracket: >
3. Enter the text that will serve as the hypertext link in the current document.
4. Enter the ending anchor tag:
   `</A>`.

Here is an sample hypertext reference: `<A HREF="MaineStats.html">Maine</A>`

This entry makes the word "Maine" the hyperlink to the document `MaineStats.html`, which is in the same directory as the first document. You can link to documents in other directories by specifying the relative path from the current document to the linked document. For example, a link of a file `NJStats.html` located in the subdirectory `AtlanticStates` would be:

`<A HREF="AtlanticStates/NJStats.html">New Jersey</A>`

These are called *relative links*. You can also use the absolute pathname of the file if you wish. Pathnames use the standard UNIX syntax.

Relative Links Versus Absolute Pathnames In general, you should use relative links, because

1. You have less to type.
2. It is easier to move a group of documents to another location, because the relative path names will still be valid.

However, use absolute pathnames when linking to documents that are not directly related. For example, consider a group of documents that comprise a user manual. Links within this group should be relative links. Links to other documents (perhaps a reference to related software) should use full path names. This way, if you move the user manual to a different directory, none of the links will have to be updated.

Uniform Resource Locator The World Wide Web uses Uniform Resource Locators (URLs) to specify the location of files on other servers. A URL includes the type of resource being accessed (e.g. http, gopher, WAIS etc.), the address of the server, and the location of the file. The syntax is:

scheme`://`*host.domain*`[:`*port*`]/` *path*`/`*filename*

where *scheme* is one of

file a file on your local system, or a file on an anonymous FTP server

http a file on a World Wide Web server

gopher a file on a Gopher server

WAIS a file on a WAIS server

news an Usenet newsgroup

telnet a connection to a Telnet-based service

The *port* number can generally be omitted. (That means unless someone tells you otherwise, leave it out.)

For example, to include a link to this primer in your document, you would use

```
<A HREF="http://nietzsche.cs.rhbnc.ac.uk/cpd/courses/html_course/book/">
UNIX Course Text</A>
```

This would make the text `UNIX Course Text` a hyperlink to this primer.

Links to Specific Sections in Other Documents Anchors can also be used to move to a particular section in a document. Suppose you wish to set a link from document A and a specific section in document B. (Call this file `documentB.html`.) First you need to set up a *named anchor* in document B. For example, to set up an anchor named “Jabberwocky” to document B, enter

```
Here's <A NAME="Jabberwocky">some text</A>
```

Now when you create the link in document A, include not only the filename, but also the named anchor, separated by a hash mark (#).

```
This is my <A HREF="documentB.html#Jabberwocky">link</A> to document
B.
```

Now clicking on the word `link` in document A sends the reader directly to the words `some text` in document B.

Links to Specific Sections Within the Current Document The technique is exactly the same except the filename is omitted.

For example, to link to the Jabberwocky anchor from within the same file (Document B), use

```
This is <A HREF="#Jabberwocky">Jabberwocky link</A> from within Document
B.
```

Exercises

1. Create a second web page in your public HTML directory and make a link from `index.html` to your second document.
2. Now try creating a link from the second document back to the end of the first document.
3. Now make a link from the bottom of the first document to the top of the first document.
4. Create a link to an external document like `http://einstein.cs.rhbnc.ac.uk/MJ`.

1.9.3 Additional Markup Tags

The preceding is sufficient to produce simple HTML documents. For more complex documents, HTML has tags for several types of lists, preformatted sections, extended quotations, character formatting, and other items.

1.9.3.1 Lists

HTML supports unnumbered, numbered, and definition lists.

Unnumbered Lists To make an unnumbered list,

1. Start with an opening list `<UL>` tag.
2. Enter the `<LI>` tag followed by the individual item. (No closing `</LI>` tag is needed.)
3. End with a closing list `</UL>` tag.

Below an example two-item list:

```
<UL>
<LI> apples
<LI> bananas
</UL>
```

The output is:

- ⋄ apples
- ⋄ bananas

The <LI> items can contain multiple paragraphs. Just separate the paragraphs with the <P> and </P> paragraph tags.

Numbered Lists A numbered list (also called an ordered list, from which the tag name derives) is identical to an unnumbered list, except it uses <OL> instead of <UL>. The items are tagged using the same <LI> tag. The following HTML code

```
<OL>
<LI> oranges
<LI> peaches
<LI> grapes
</OL>
```

produces this formatted output:

1. oranges
2. peaches
3. grapes

Definition Lists A definition list usually consists of alternating a term (abbreviated as DT) and a definition (abbreviated as DD). Web browsers generally format the definition on a new line.

The following is an example of a definition list:

```
<DL>
```

```
<DT> NCSA

<DD> NCSA, the National Center for Supercomputing Applications, is
located on the campus of the University of Illinois at Urbana-Champaign.
NCSA is one of the participants in the National MetaCenter for Computational
Science and Engineering.

<DT> Cornell Theory Center

<DD> CTC is located on the campus of Cornell University in Ithaca,
New York.  CTC is another participant in the National MetaCenter for
Computational Science and Engineering.

</DL>
```

The output looks like:

NCSA NCSA, the National Center for Supercomputing Applications, is located on the campus of the University of Illinois at Urbana-Champaign. NCSA is one of the participants in the National MetaCenter for Computational Science and Engineering.

Cornell Theory Center CTC is located on the campus of Cornell University in Ithaca, New York. CTC is another participant in the National MetaCenter for Computational Science and Engineering.

The <DT> and<DD> entries can contain multiple paragraphs (separated by <P> paragraph tags), lists, or other definition information.

The COMPACT attribute can be used routinely in case your definition terms are very short. If, for example, you are showing some computer options, the options may fit on the same line as the start of the definition.

```
<DL COMPACT>

<DT> -i

<DD> invokes NCSA Mosaic for Microsoft Windows using the initialization
file defined in the path

<DT> -k

<DD> invokes NCSA Mosaic for Microsoft Windows in kiosk mode

</DL>
```

The output looks like:

```
-i    invokes NCSA Mosaic for Microsoft Windows using the
```

```
      initialization file defined in the path.
-k    invokes NCSA Mosaic for Microsoft Windows in kiosk mode.
```

Nested Lists Lists can be arbitrarily nested, although in practice you probably should limit the nesting to three levels. You can also have a number of paragraphs, each containing a nested list, in a single list item.

An example nested list:

```
<UL>
<LI> A few New England states:
<UL>
<LI> Vermont
<LI> New Hampshire
</UL>
<LI> One Midwestern state:
<UL>
<LI> Michigan
</UL>
</UL>
```

The nested list is displayed as

- ◇ A few New England states:
  - – Vermont
  - – New Hampshire
- ◇ One Midwestern state:
  - – Michigan

Exercises

1. Create a simple list of your choice.
2. Create a list of links to all the anchors and documents you previously linked to.

3. Optional: Change the list of links to group links to different parts of one document in sub-lists.

1.9.3.2 Tables

Tables allow you to present your information in a row/column format. Consider the style of a newspaper, a magazine or even the Periodic Table of Elements.

Tables are surrounded by a `<TABLE>` ... `</TABLE>` pair of tags. They can have a caption and a number of rows containing information. Each row is enclosed by a `<TR>` ... `</TR>` pair and contains a number of cells which are enclosed by a `<TH>` ... `</TH>` (table heading) or `<TD>` ... `</TD>` (table data) pair. Any HTML element can be placed in a cell:

- Text
- Images
- Forms
- Lists
- Anchors
- Table
- Any combination of the above

A table has the general format:

HTML	Description
`<TABLE>`	start of table definition
`<CAPTION> caption contents </CAPTION>`	caption definition
`<TR>`	start of first row definition
`<TH> cell contents </TH>`	first cell in row 1
...	
`<TD> cell contents </TD>`	last cell in row 1
`</TR>`	end of first row definition
`<TR>`	start of second row definition
`<TH> cell contents </TH>`	first cell in row 2
...	
`<TD> cell contents </TD>`	last cell in row 2

continued on next page

continued from previous page

HTML	Description
`</TR>`	end of second row definition
...	
`<TR>`	start of last row definition
`<TH> cell contents </TH>`	first cell in lastrow
...	
`<TD> cell contents </TD>`	last cell in last row
`</TR>`	end of last row definition
`</TABLE>`	end of table definition

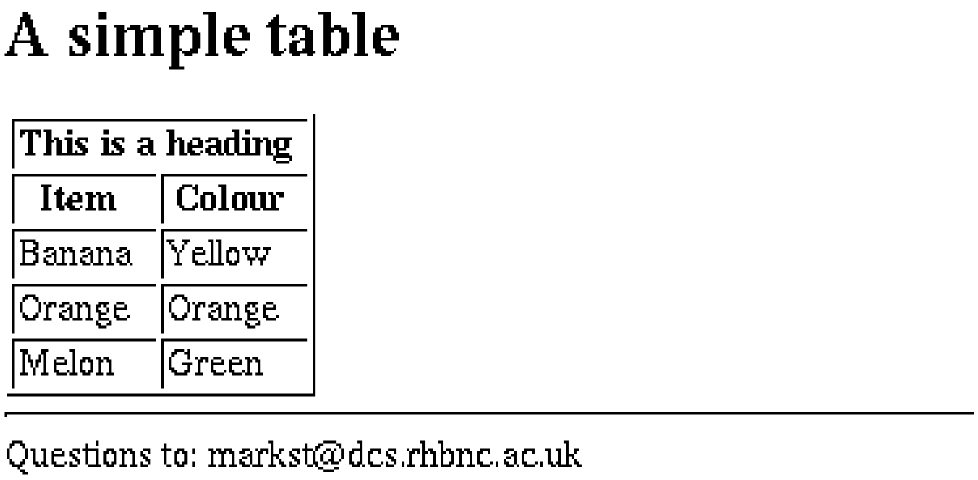
The relevant tags for tables are:

TAG	Description	Example
<TABLE> ... </TABLE>	The tag that defines a Table in HTML. If the attribute BORDER is present, then Mosaic will display the table with a border.	<TABLE BORDER> ... </TABLE>
<CAPTION> ... </CAPTION>	This tag defines the caption for the title of the table. The default position of the Title is centered at the top of the table. The attribute ALIGN=BOTTOM can be used to position the caption below the table. Note: Any kind of markup can exist in the caption.	<CAPTION ALIGN=BOTTOM> ... </CAPTION>
<TR> ... </TR>	This tag specifies a Table Row within a table. You may define default attributes for the entire row. These attributes are: ALIGN (LEFT, CENTER, RIGHT) and/or VALIGN (TOP, MIDDLE, BOTTOM).	<TR VALIGN=BOTTOM ALIGN=CENTER> ... </TR>
<TH> ... </TH>	This tag defines a Table Header cell. By default the text in this cell is bold and centered. Table Header cells may contain other attributes to determine the characteristics of the cell and/or its contents. See Alignment Attributes at the end of this table for more information.	<TH ALIGN=CENTER VALIGN=TOP COLSPAN=4 ROWSPAN=2 NOWRAP> ... </TH>
<TD> ... </TD>	This tag defines a Table Data cell. By default the text in this cell is aligned left and centered vertically. Table Data cells may contain other attributes to determine the characteristics of the cell and/or its contents. See Alignment Attributes at the end of this table for more information.	<TD ALIGN=CENTER VALIGN=TOP COLSPAN=2 ROWSPAN=3 NOWRAP> ... </TD>

Alignment Attribute	Values	Description
Note: Attributes defined within <TH> ... </TH> or <TD> ... </TD> cells will override the default alignment set in <TR> ... </TR>.		
ALIGN	LEFT, CENTER, RIGHT	Specifies the horizontal alignment of a cell.
VALIGN	TOP, MIDDLE, BOTTOM	Specifies the vertical alignment of a cell.
COLSPAN=X	X is an integer	The cell will span X columns in the table.

continued on next page

continued from previous page		
Alignment Attribute	Values	Description
ROWSPAN=Y	Y is an integer	The cell will span Y rows in a table.
NOWRAP		Word wrapping is turned off within a cell.



A simple table

This is a heading	
Item	**Colour**
Banana	Yellow
Orange	Orange
Melon	Green

Questions to: markst@dcs.rhbnc.ac.uk

Figure 1.3 A simple table

A simple example

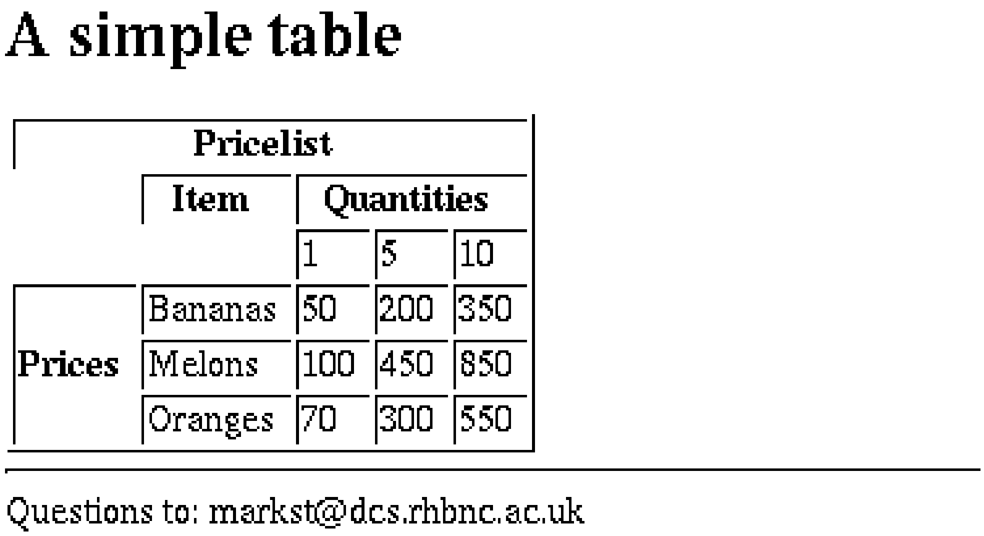
The above table 1.3 is produced by the following HTML[17]:

```
<TABLE BORDER>

<TR> <TD ALIGN=CENTER COLSPAN=2> This is a heading </TD> </TR>
<TR> <TH> Item   </TH> <TH> Colour </TH> </TR>
<TR> <TD> Banana </TD> <TD> Yellow </TD> </TR>
<TR> <TD> Orange </TD> <TD> Orange </TD> </TR>
<TR> <TD> Melon  </TD> <TD> Green  </TD> </TR>

</TABLE>
```

Exercise

1. Write a table that looks like figure 1.4.

[17] The headers and footers have been removed for clarity's sake.



A simple table

Pricelist				
	Item	Quantities		
		1	5	10
Prices	Bananas	50	200	350
	Melons	100	450	850
	Oranges	70	300	550

Questions to: markst@dcs.rhbnc.ac.uk

Figure 1.4 Another simple table

1.9.3.3 Preformatted Text

Use the<PRE> tag (which stands for "preformatted") to generate text in a fixed-width font and cause spaces, new lines, and tabs to be significant. (That is, multiple spaces are displayed as multiple spaces, and lines break in the same locations as in the source HTML file.) This is useful for program listings. For example, the following lines

```
<PRE>

#!/bin/csh

cd $SCR

cfs get mysrc.f:mycfsdir/mysrc.f

cfs get myinfile:mycfsdir/myinfile

fc -02 -o mya.out mysrc.f

mya.out

cfs save myoutfile:mycfsdir/myoutfile

rm *

</PRE>
```

display as

```
      #!/bin/csh
```

```
cd $SCR
cfs get mysrc.f:mycfsdir/mysrc.f
cfs get myinfile:mycfsdir/myinfile
fc -02 -o mya.out mysrc.f
mya.out
cfs save myoutfile:mycfsdir/myoutfile
rm *
```

Hyperlinks can be used within <PRE> sections. You should avoid using other HTML tags within <PRE> sections, however.

Note that because <, >, and & have special meaning in HTML, you have to use their escape sequences (`<`, `>`, and `&`, respectively) to enter these characters. See the section 1.9.4.3 Special Characters for more information.

1.9.3.4 Extended Quotations

Use the <BLOCKQUOTE> tag to include quotations in a separate block on the screen. Most browsers generally indent to separate it from surrounding text.

An example:

```
<BLOCKQUOTE>

I still have a dream.  It is a dream deeply rooted in the American
dream.  <P>

I have a dream that one day this nation will rise up and live out the
true meaning of its creed.  We hold these truths to be self-evident
that all men are created equal.  <P>

</BLOCKQUOTE>
```

The result is:

> I still have a dream. It is a dream deeply rooted in the American dream.
>
> I have a dream that one day this nation will rise up and live out the true meaning of its creed. We hold these truths to be self-evident that all men are created equal.

1.9.3.5 Addresses

The <ADDRESS> tag is generally used to specify the author of a document and a means of contacting the author (e.g., an email address). This is usually the last item in a file.

For example, the last line of the online version of this guide is

`<ADDRESS>`

`A Beginner's Guide to HTML / NCSA / pubs@@ncsa.uiuc.edu`

`</ADDRESS>`

The result is

A Beginner's Guide to HTML / NCSA / pubs@@ncsa.uiuc.edu

NOTE: <ADDRESS> is *not* used for postal addresses. See "Forced Line Breaks" in section 1.9.4.3 on how to format postal addresses.

1.9.4 Character Formatting

You can code individual words or sentences with special styles. There are two types of styles: logical and physical. *Logical styles* tag text according to its meaning, while *physical styles* specify the specific appearance of a section. For example, in the preceding sentence, the words "logical styles" were tagged as a "definition." The same effect (formatting those words in italics), could have been achieved via a different tag that specifies merely "put these words in italics."

1.9.4.1 Physical Versus Logical: Use Logical Styles When Possible

If physical and logical styles produce the same result on the screen, why are there both? We devolve, for a couple of paragraphs, into the philosophy of SGML, which can be summed in a Zen-like mantra: "Trust your browser."

In the ideal SGML universe, content is divorced from presentation. Thus, SGML tags a level-one heading as a level-one heading, but does not specify that the level-one heading should be, for instance, 24-point bold Times centered on the top of a page. The advantage of this approach (it's similar in concept to style sheets in many word processors) is that if you decide to change level-one headings to be 20-point left-justified Helvetica, all you have to do is change the definition of the level-one heading in the presentation device (i.e., your World Wide Web browser).

The other advantage of logical tags is that they help enforce consistency in your documents. It's easier to tag something as <H1> than to remember that level-one headings are 24-point bold Times or whatever. The same is true for character styles. For example, consider the <STRONG> tag. Most browsers render it in bold text. However, it is possible that a reader would prefer that these sections be displayed in red instead. Logical styles offer this flexibility.

Logical Styles

`<DFN>` for a word being defined. Typically displayed in italics. (*NCSA Mosaic* is a World Wide Web browser.)

`<EM>` for emphasis. Typically displayed in italics. (*Watch out for pick pockets.*)

`<CITE>` for titles of books, films, etc. Typically displayed in italics. (*A Beginner's Guide to HTML*)

`<CODE>` for snippets of computer code. Displayed in a fixed-width font. (The `<stdio.h>` header file)

`<KBD>` for user keyboard entry. Should be displayed in a bold fixed-width font, but many browsers render it in the plain fixed-width font. (Enter **passwd** to change your password.)

`<SAMP>` for computer status messages. Displayed in a fixed-width font. (`Segmentation fault:  Core dumped.`)

`<STRONG>` for strong emphasis. Typically displayed in bold. (**Important**)

`<VAR>` for a "metasyntactic" variable, where the user is to replace the variable with a specific instance. Typically displayed in italics. (**rm** *filename* deletes the file.)

Physical Styles

`<B>` bold text

`<I>` italic text

`<TT>` typewriter text, e.g. fixed-width font.

1.9.4.2 Using Character Tags

To apply a character style,

1. Start with < *tag* >, where *tag* is the desired character formatting tag, to indicate the beginning of the tagged text.
2. Enter the tagged text.
3. End the passage with </ *tag* >.

1.9.4.3 Special Characters

Escape Sequences Four characters of the ASCII character set – the left angle bracket (<), the right angle bracket (>), the ampersand (&) and the double quote (") – have special meaning within HTML and therefore cannot be used "as is" in text. (The angle brackets are used to indicate the beginning and end of HTML tags, and the ampersand is used to indicate the beginning of an escape sequence.)

To use one of these characters in an HTML document, you must enter its *escape sequence* instead:

< the escape sequence for <

> the escape sequence for >

& the escape sequence for &

" the escape sequence for "

Additional escape sequences support accented characters. For example:

ö the escape sequence for a lowercase o with an umlaut: ö

ñ the escape sequence for a lowercase n with an tilde: ñ

È the escape sequence for an uppercase E with a grave accent: È

Check

http://www.w3.org/hypertext/WWW/MarkUp/html-spec/html-spec_13.html#SEC106

for a longer list of special characters.

NOTE: Unlike the rest of HTML, the escape sequences are case sensitive. You cannot, for instance, use < instead of <.

Forced Line Breaks The
 tag forces a line break with no extra space between lines. (By contrast, most browsers format the <P> paragraph tag with an additional blank line to more clearly indicate the beginning the new paragraph.)

One use of
 is in formatting addresses:

National Center for Supercomputing Applications

605 East Springfield Avenue

Champaign, Illinois 61820-5518

Horizontal Rules The <HR> `tag` produces a horizontal line the width of the browser window.

1.9.5 In-line Images

Most Web browsers can display inline images (that is, images next to text) that are in X Bitmap (XBM), GIF, or JPEG format. Other image formats are being incorporated into Web browsers [e.g., the Portable Network Graphic (PNG) format]. Each image takes time to process and slows down the initial display of a document. Carefully select your images and the number of images in a document. To include an in-line image, use

<IMG SRC=*image_URL*>

where *image_URL* is the URL of the image file. The syntax for `IMG SRC` URLs is identical to that used in an anchor `HREF`. If the image file is a GIF file, then the filename part of *image_URL* must end with `.gif`. Filenames of X Bitmap images must end with `.xbm`; JPEG image files must end with `.jpg` or `.jpeg`; and Portable Network Graphic files must end with `.png`.

1.9.5.1 Image Size Attributes

You should include two other attributes on <IMG> tags to tell your browser the size of the images it is downloading with the text. The `HEIGHT` and `WIDTH` attributes let your browser set aside the appropriate space (in pixels) for the images as it downloads the rest of the file. (Get the pixel size from your image-processing software, such as Adobe Photoshop.)

For example, to include a self portrait image in a file along with the portrait's dimensions, enter:

```
<IMG SRC="SelfPortrait.gif" HEIGHT=100 WIDTH=65>
```

NOTE: Some browsers use the `HEIGHT` and `WIDTH` attributes to stretch or shrink an image to fit into the allotted space when the image does not exactly match the attribute numbers. Not all browser developers think stretching/shrinking is a good idea. So don't plan on your readers having access to this feature. Check your dimensions and use the correct ones.

1.9.5.2 Aligning Images

You have some flexibility when displaying images. You can have images separated from text and aligned to the left or right or centered. Or you can have an image aligned with text. Try several possibilities to see how your information looks best.

Aligning Text with an Image By default the bottom of an image is aligned with the following text. You can align images to the top or center of a paragraph using the `ALIGN=` attributes `TOP` and `CENTER`.

Be aware that an image will be aligned to one line of text. A large image will therefore create a large gap in your document.

1.9.5.3 Images without Text

To display an image without any associated text (e.g., your organization's logo), make it a separate paragraph. Use the paragraph ALIGN= attribute to center the image or adjust it to the right side of the window as shown below:

```
<P ALIGN=CENTER> <IMG SRC="BarHotlist.gif"> </P>
```

1.9.5.4 Alternate Text for Images

Some World Wide Web browsers–primarily those that run on VT100 terminals–cannot display images. Some users turn off image loading even if their software can display images (especially if they are using a modem or have a slow connection). HTML provides a mechanism to tell readers what they are missing on your pages.

The `ALT` attribute lets you specify text to be displayed instead of an image. For example:

```
<IMG SRC="UpArrow.gif" ALT="Up">
```

where `UpArrow.gif` is the picture of an upward pointing arrow. With graphics-capable viewers that have image-loading turned on, you see the up arrow graphic. With a VT100 browser or if image-loading is turned off, the word Up is shown in your window.

You should try to include alternate text for each image you use in your document, which is a courtesy for your readers.

1.9.5.5 Background Graphics

Newer versions of Web browsers can load an image and use it as a background when displaying a page. Some people like background images and some don't. In general, if you want to include a background, make sure your text can be read easily when displayed on top of the image.

Background images can be a texture (linen finished paper, for example) or an image of an object (a logo possibly). You create the background image as you do any image. However you only have to create a small piece of the image. Using a feature called tiling, a browser takes the image and repeats it across and down to fill your browser window. In sum you generate one image, and the browser replicates it enough times to fill your window. This action is automatic when you use the background tag shown below.

The tag to include a background image is included in the `<BODY>` statement as an attribute:

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

1.9.6 Colours

By default browsers display text in black on a grey background. However, you can change both elements if you want. Some HTML authors select a background colour and coordinate it with a change in the colour of the text.

Always preview changes like this to make sure your pages are readable. (For example, many people find red text on a black background difficult to read!) Also try previewing your pages on different computers as the web does require colours to be displayed accurately. Your nice black test on a red background might turn out as black on a dark purple background.

You change the colour of text, links, visited links, and active links using attributes of the `<BODY>` tag.

For example, enter:

```
<BODY BGCOLOR="#000000" TEXT="#FFFFFF" LINK="#9690CC">
```

This creates a window with a black background (`BGCOLOR`), white text (`TEXT`), and silvery hyperlinks (`LINK`).

The six-digit number and letter combinations represent colours by giving their RGB (red, green, blue) value. The six digits are actually three two-digit numbers in sequence, representing the amount of red, green, or blue as a hexadecimal value in the range 00-FF. For example, 000000 is black (no colour at all), FF0000 is bright red, and FFFFFF is white (fully saturated with all three colours). These number and letter combinations are cryptic. Fortunately an online resource is available to help you track down the combinations that map to specific colours:

`http://www.biola.edu/cgi-bin/colorpro`

Exercises

1. Include the image `imap_test.gif` in your web page.
2. Give your web page a background. You can use either the previous image or try `/graphics/rhultile.gif`.

1.9.7 External Images, Sounds, and Animations

You may want to have an image open as a separate document when a user activates a link on either a word or a smaller, in-line version of the image included in your document. This is considered an external image and is useful if you do not wish to slow down the loading of the main document with large in-line images.

To include a reference to an external image, use

```
<A HREF = image_URL>link anchor</A>
```

You can also use a smaller image as a link to a larger image. Enter:

```
<A HREF="LargerImage.gif"><IMG SRC="SmallImage.gif"></A>
```

The reader sees the SmallImage.gif image and clicks on it to open the LargerImage.gif file.

Use the same syntax for links to external animations and sounds. The only difference is the file extension of the linked file. For example,

```
<A HREF="QuickTimeMovie.mov">link anchor</A>
```

specifies a link to a QuickTime movie. Some common file types and their extensions are:

File Type **Extension**

Plain text `.txt`

HTML document `.html`

GIF image `.gif`

TIFF image `.tiff`

XBM bitmap image `.xbm`

JPEG image `.jpg` or `.jpeg`

PostScript file `.ps`

AIFF sound `.aiff`

AU sound `.au`

QuickTime movie `.mov`

MPEG movie `.mpeg` or `.mpg`

Make sure your intended audience has the necessary viewers. Most UNIX workstations, for instance, cannot view QuickTime movies.

1.9.8 Commenting Your Files

You might want to include comments in your HTML files. Comments in HTML are like comments in a computer program–the text you enter is not used by the browser in any formatting and is not directly viewable by the reader just as computer program comments are not used and are not viewable. The comments are accessible if a reader views the source file, however.

Comments such as the name of the person updating a file, the software and version used in creating a file, or the date that a minor edit was made are the norm.

To include a comment, enter:

```
<!-- your comments here -->
```

You must include the exclamation mark and the hyphens as shown.

1.9.9 Troubleshooting

1.9.9.1 Avoid Overlapping Tags

Consider this snippet of HTML:

```
<B>This is an example of <DFN>overlapping</B> HTML tags.</DFN>
```

The word “overlapping” is contained within both the `<B>` and `<DFN>` tags. How does the browser format it? You won’t know until you look, and different browsers will likely react differently. In general, avoid overlapping tags.

1.9.9.2 Embed Anchors and Character Tags, But Nothing Else

It is acceptable to embed anchors within another HTML element:

```
<H1><A HREF="Destination.html''>My heading</A></H1>
```

Do not embed a heading or another HTML element within an anchor:

```
<A HREF"Destination.html">

<H1>My heading</H1>

</A>
```

Although most browsers currently handle this example, it is forbidden by the official HTML specification, and will not work with future browsers.

Character tags modify the appearance of other tags:

```
<UL><LI><B>A bold list item</B>

<UL>

<LI><I>An italic list item</I>

</UL>
```

However, avoid embedding other types of HTML element tags. For example, it is tempting to embed a heading within a list, in order to make the font size larger:

```
<UL><LI><H1>A large heading</H1>

<UL>

<LI><H2>Something slightly smaller</H2>

</UL>
```

Although some browsers, such as NCSA Mosaic for the X Window System, format this construct quite nicely, it is unpredictable (because it is undefined) what other browsers will do. For compatibility with all browsers, avoid these kinds of constructs.

What's the difference between embedding a <B> within a <LI> tag as opposed to embedding a <H1> within a <LI>? This is again a question of SGML. The semantic meaning of <H1> is that it's the main heading of a document and that it should be followed by the content of the document.Thus it doesn't make sense to find a <H1> within a list.

Character formatting tags also are generally not additive. You might expect that

```
<B><I>some text</I></B>
```

would produce bold-italic text. On some browsers it does; other browsers interpret only the innermost tag (here, the italics).

1.9.9.3 Check Your Links

When an <IMG> tag points at an image that does not exist, a dummy image is substituted. When this happens, make sure that the referenced image does in fact exist, that the hyperlink has the correct information in the URL, and that the file permission is set appropriately (world-readable).

1.9.10 A Longer Example

Here is a longer example of an HTML document: <HTML>

<HEAD>

<TITLE>A Longer Example</TITLE>

</HEAD>

<BODY>

<H1>A Longer Example</H1>

This is a simple HTML document. This is the first paragraph. <P>

This is the second paragraph, which shows special effects. This is a word in <I>italics</I>. This is a word in <B>bold</B>. Here is an in-lined GIF image: <IMG SRC="myimage.gif">.

<P>

```
This is the third paragraph, which demonstrates links.  Here is a hypertext
link from the word <A HREF="subdir/myfile.html"> foo </A> to a document
called "subdir/myfile.html".  (If you try to follow this link, you
will get an error screen.)  <P>

<H2>A second-level header</H2>

Here is a section of text that should display as a fixed-width font:
<P>

<PRE>

On the stiff twig up there

Hunches a wet black rook

Arranging and rearranging its feathers in the rain ...

<PRE>

This is a unordered list with two items:  <P>

<UL> <LI> cranberries <LI> blueberries </UL>

This is the end of my example document.  <P>

<ADDRESS>Me (me@@mycomputer.univ.edu)</ADDRESS>

</BODY>

</HTML>
```

The formatted version looks like this:

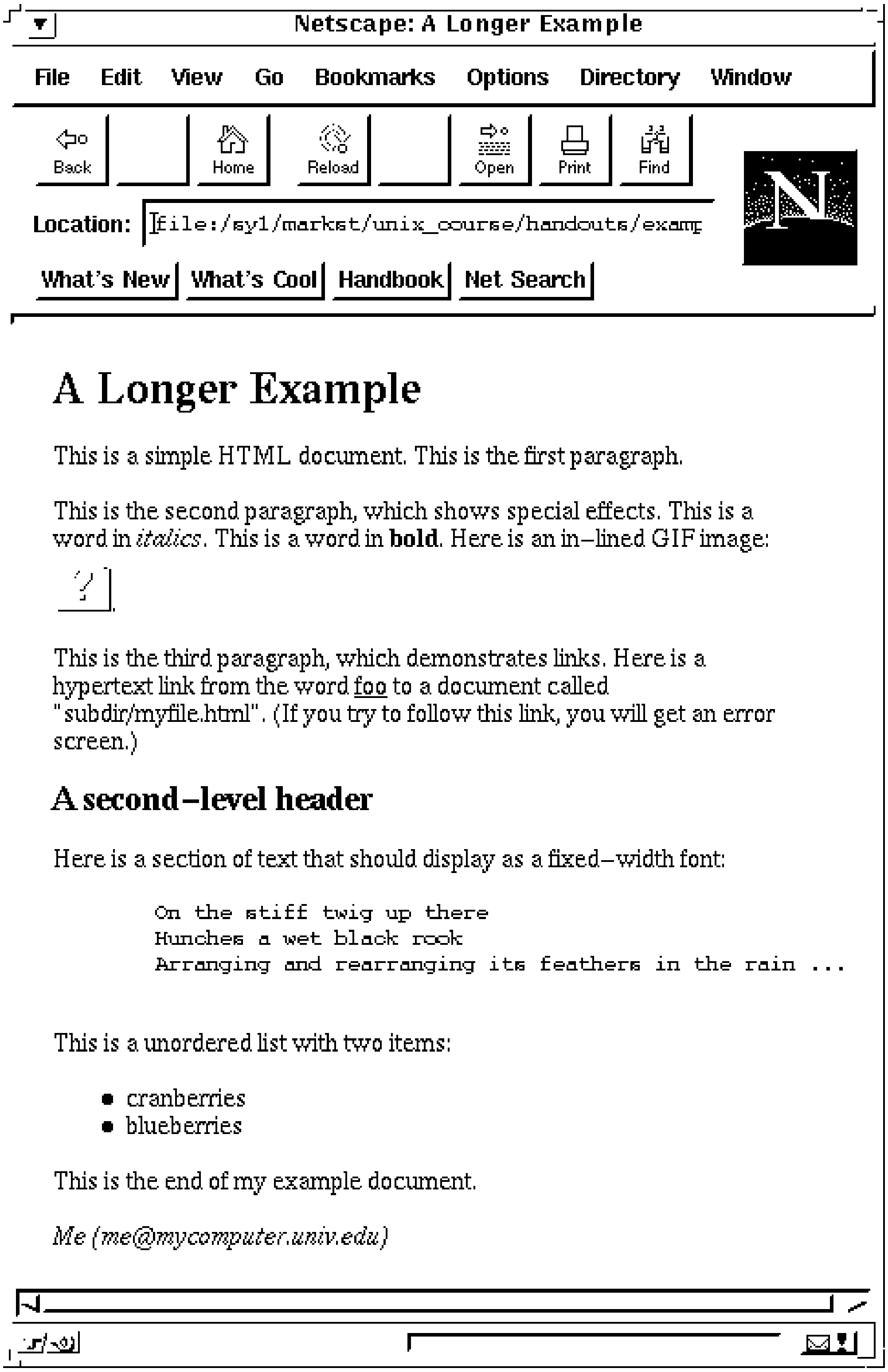


Figure 1.5 A longer HTML example

In addition to tags already discussed, this example also uses the `<HEAD>` ... `</HEAD>` and `<BODY>` ... `</BODY>` tags, which separate the document into introductory information about the document and the main text of the document. These tags don't change the appearance of the formatted document at all, but are useful for several purposes (for example, NCSA Mosaic for Macintosh 2.0, for example, allows you to browse just the header portion of document before deciding whether to download the rest), and it is recommended that you use these tags.

1.9.11 Redirects

Occasionally HTML documents have to be moved or their name changed. This means that web users who have saved this address will not be able to access this document any more.

Occasionally URLs will be wrongly advertised and web users who access the wrong address will have to be redirected.

Redirection is not supported by HTML directly. HTML itself just allows setting up a page named with the wrong URL with a link to the right address. Many browsers however allow embedding `meta` tags into the HTML which are then interpreted by the browser. One such `meta` tag implements redirects. This has to be used with caution as not all browsers support this tag.

Assume we have a URL `http://www.dcs.rhbnc.ac.uk/cpd` but we wrongly told all our friends to use `http://wwwnew.dcs.rhbnc.ac.uk/cpd`. We now have to set up a little web page with the URL
`http://wwwnew.dcs.rhbnc.ac.uk/cpd/index.html`
that will redirect users to `http://www.dcs.rhbnc.ac.uk/cpd`:

```
<meta http-equiv="refresh" content="6; url=http://www.dcs.rhbnc.ac.uk/cpd"
>

<H1>Royal Holloway:  CPD</H1> The URL for this document has changed;
the correct URL is
<A HREF="http://www.dcs.rhbnc.ac.uk/cpd"> http://www.dcs.rhbnc.ac.uk/cpd
</A>,
but many browsers will automatically route you there in a few more
seconds...  If nothing has happened after about thirty seconds, please
use the manual link above.
```

The `meta` tag is not displayed and users will usually not be aware of a redirect taking place.

1.9.12 Server Side Includes

Server side includes allow inclusion of documents in the current document in a way the user cannot detect. This allows for modularisation of web pages, e.g. the author and return address can be held in on file for all documents and just be included on the fly. A simple change to the address file will update all documents.

Server side includes are useful, but put an extra load on the server which is why many servers disable this feature. If a document includes SSIs it must have the `.shtml` extension[18].

To include a document `include.html` in our main document `main.shtml` we have to use the following syntax:

```
<!--#include virtual="include.html" -->
```

The file will be included as if the contents of the include file had been at that position in the main file. It is also possible to include files that include SSIs if their extension is `.shtml`.

Another useful thing is to include a last modification date for a document:

```
<P>

This document was last modified on

<!--#echo var="LAST_MODIFIED" -->

</P>
```

It is also possible to check the modification time of another document:

```
<!--#flastmod virtual="include.html" -->
```

For more information on SSIs see the NCSA tutorial at :

`http://hoohoo.ncsa.uiuc.edu/docs/tutorials/includes.html`

Exercises

1. Add a line to your home page that automatically shows when it was last modified.

2. Now try to put the footer into a separate file and let the server automatically include it.

[18] Other servers might use different extensions although this is the default.

1.9.13 Imagemaps

Imagemaps are clickable images. Each section of an image represents a link to another document. Imagemaps are unfriendly to users of browsers which do not display images. These users are just given a list of URLs.

However as they are immensely popular here is a small example from the following URL:

`http://www.ee.ed.ac.uk/online/imagemaps.html`

There are two ways of implenting imagemaps one is to use server-side imagemaps where the server does the processing. This is neater because the HTML does not include non-displayed information, however it is harder to set up and might be disabled on some servers. Client-side imagemaps are easier to program and maintain, but require the browsers support.

Client-side imagemaps are also less wasteful of bandwidth as server-side clientmaps require the browser to send the coordinates to the server, which reads a map file and sends a redirect back to the client to tell it which file to retrieve.

Client-side imagemaps allow all of this processing to be done in the browser. The format of the data is very similar to that used by the server-side maps, but it appears as HTML tags in the file (or possibly in another file).

Generally it is best to provide both methods simultaneously.

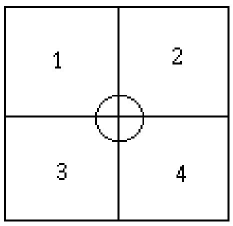


Figure 1.6 An example imagemap

For the image 1.6, a client-side image map would look like

```
<MAP NAME="test">

<AREA SHAPE="circle" COORDS="95,100,10" HREF="bull.html">

<AREA SHAPE="rect" COORDS="35,45 95,100" HREF="one.html">
```

```
<AREA SHAPE="rect" COORDS="95,45 155,100" HREF="two.html">
<AREA SHAPE="rect" COORDS="35,100 95,155" HREF="three.html">
<AREA SHAPE="rect" COORDS="95,100 155,155" HREF="four.html">
<AREA SHAPE="rect" COORDS="0,0 200,200" HREF="nothing.html">
</MAP>
```

and the image would be included with

```
<IMG SRC="test.gif" ISMAP USEMAP="#test">
```

When using an interactive program to produce imagemaps these tags can be automatically created. If server side imagemaps are used, a separate map file has to be generated, e.g. `test.map`. This is read by the server to determine the clickable areas. The imagemap would then be included as:

```
<A HREF="test.map"><IMG SRC="test.gif" ISMAP></A>
```

Both methods can be specified and the browser decides which to use by combining the two formats:

```
<A HREF="test.map"><IMG SRC="test.gif" ISMAP USEMAP="#test"></A>
```

Exercises

1. Go to the URL mentioned above and test the imagemap.
2. Now write your own web page with the above coordinates and the image from the page. The image can be found in your public HTML directory.
3. Run mapedit using the image `test.gif` and create your own imagemap.

1.9.14 Java

Java is a object oriented programming language. Once a Java program has been compiled it can be run on any computer that supports Java. This means that a program only has to be compiled once, rather than for each type of computer the program is supposed to run on. The main application of Java lies in writing applications for the web, where the programmer does not know in advance which computer is going to run the program.

1.9.14.1 What is JavaScript

JavaScript is a scripting language which uses similar concepts to Java. It is included in the HTML and in contrast to Java it is not compiled and can be copied and modified by anybody.

Appendix A

Manual Pages

A.1 chmod - Changing File Access Permissions

NAME

chmod - change the access permissions of files

SYNOPSIS

chmod [-Rcfv] [--recursive] [--changes] [--silent] [--quiet] [--verbose] [--help] [--version] mode file...

DESCRIPTION

This manual page documents the GNU version of chmod. chmod changes the permissions of each given file according to mode, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new permissions.

The format of a symbolic mode is '[ugoa...][[+-=][rwxXstugo...]...][,...]'. Multiple symbolic operations can be given, separated by commas.

A combination of the letters 'ugoa' controls which users' access to the file will be changed: the user who owns it (u), other users in the file's group (g), other users not in the file's group (o), or all users (a). If none of these are given, the effect is as if 'a' were given, but bits that are set in the umask are not affected.

The operator '+' causes the permissions selected to be added to the existing permissions of each file; '-' causes them to be removed; and '=' causes them to be the only permissions that the file has.

The letters 'rwxXstugo' select the new permissions for the affected users: read (r), write (w), execute (or access for directories) (x), execute only if the file is a directory or already has execute permission for some user (X), set user or group ID on execution (s), save program text on swap device (t), the permissions that the user who owns the file currently has for it (u), the permissions that other users in the file's group have for it (g), and the permissions that other users not in the file's group have for it (o).

A numeric mode is from one to four octal digits (0-7), derived by adding up the bits with values 4, 2, and 1. Any omitted digits are assumed to be leading zeros. The first digit selects the set user ID (4) and set group ID (2) and save text image (1) attributes. The second digit selects permissions for the user who owns the file: read (4), write (2), and execute (1); the third selects permissions for other users in the file's group, with the same values; and the fourth for other users not in the file's group, with the same values.

chmod never changes the permissions of symbolic links; the chmod system call cannot change their permissions. This is not a problem since the permissions of symbolic links are never used. However, for each symbolic link listed on the command line, chmod changes the permissions of the pointed-to file. In contrast, chmod ignores symbolic links encountered during recursive directory traversals.

OPTIONS

-c, --changes
: Verbosely describe only files whose permissions actually change.

-f, --silent, --quiet
: Do not print error messages about files whose permissions cannot be changed.

-v, --verbose
: Verbosely describe changed permissions.

-R, --recursive
: Recursively change permissions of directories and their contents.

--help
: Print a usage message on standard output and exit successfully.

--version

Print version information on standard output then exit successfully.

GNU Emacs Reference Card

(for version 19)

Starting Emacs

To enter GNU Emacs 19, just type its name: `emacs`

To read in a file to edit, see Files, below.

Leaving Emacs

suspend Emacs (or iconify it under X)	`C-z`
exit Emacs permanently	`C-x C-c`

Files

read a file into Emacs	`C-x C-f`
save a file back to disk	`C-x C-s`
save **all** files	`C-x s`
insert contents of another file into this buffer	`C-x i`
replace this file with the file you really want	`C-x C-v`
write buffer to a specified file	`C-x C-w`

Getting Help

The Help system is simple. Type `C-h` and follow the directions. If you are a first-time user, type `C-h t` for a **tutorial**.

remove Help window	`C-x 1`
scroll Help window	`ESC C-v`
apropos: show commands matching a string	`C-h a`
show the function a key runs	`C-h c`
describe a function	`C-h f`
get mode-specific information	`C-h m`

Error Recovery

abort partially typed or executing command	`C-g`
recover a file lost by a system crash	`M-x recover-file`
undo an unwanted change	`C-x u` or `C-_`
restore a buffer to its original contents	`M-x revert-buffer`
redraw garbaged screen	`C-l`

Incremental Search

search forward	`C-s`
search backward	`C-r`
regular expression search	`C-M-s`
reverse regular expression search	`C-M-r`
select previous search string	`M-p`
select next later search string	`M-n`
exit incremental search	`RET`
undo effect of last character	`DEL`
abort current search	`C-g`

Use `C-s` or `C-r` again to repeat the search in either direction. If Emacs is still searching, `C-g` cancels only the part not done.

Motion

entity to move over	**backward**	**forward**
character	`C-b`	`C-f`
word	`M-b`	`M-f`
line	`C-p`	`C-n`
go to line beginning (or end)	`C-a`	`C-e`
sentence	`M-a`	`M-e`
paragraph	`M-{`	`M-}`
page	`C-x [`	`C-x ]`
sexp	`C-M-b`	`C-M-f`
function	`C-M-a`	`C-M-e`
go to buffer beginning (or end)	`M-<`	`M->`

scroll to next screen	`C-v`
scroll to previous screen	`M-v`
scroll left	`C-x <`
scroll right	`C-x >`
scroll current line to center of screen	`C-u C-l`

Killing and Deleting

entity to kill	**backward**	**forward**
character (delete, not kill)	`DEL`	`C-d`
word	`M-DEL`	`M-d`
line (to end of)	`M-0 C-k`	`C-k`
sentence	`C-x DEL`	`M-k`
sexp	`M-- C-M-k`	`C-M-k`

kill **region**	`C-w`
copy region to kill ring	`M-w`
kill through next occurrence of *char*	`M-z` *char*
yank back last thing killed	`C-y`
replace last yank with previous kill	`M-y`

Marking

set mark here	`C-@` or `C-SPC`
exchange point and mark	`C-x C-x`
set mark *arg* **words** away	`M-@`
mark **paragraph**	`M-h`
mark **page**	`C-x C-p`
mark **sexp**	`C-M-@`
mark **function**	`C-M-h`
mark entire **buffer**	`C-x h`

Query Replace

interactively replace a text string	`M-%`
using regular expressions	`M-x query-replace-regexp`

Valid responses in query-replace mode are

replace this one, go on to next	`SPC`
replace this one, don't move	`,`
skip to next without replacing	`DEL`
replace all remaining matches	`!`
back up to the previous match	`^`
exit query-replace	`ESC`
enter recursive edit (`C-M-c` to exit)	`C-r`

Multiple Windows

delete all other windows	`C-x 1`
delete this window	`C-x 0`
split window in two vertically	`C-x 2`
split window in two horizontally	`C-x 3`
scroll other window	`C-M-v`
switch cursor to another window	`C-x o`
shrink window shorter	`M-x shrink-window`
grow window taller	`C-x ^`
shrink window narrower	`C-x {`
grow window wider	`C-x }`
select buffer in other window	`C-x 4 b`
display buffer in other window	`C-x 4 C-o`
find file in other window	`C-x 4 f`
find file read-only in other window	`C-x 4 r`
run Dired in other window	`C-x 4 d`
find tag in other window	`C-x 4 .`

Formatting

indent current **line** (mode-dependent)	`TAB`
indent **region** (mode-dependent)	`C-M-\`
indent **sexp** (mode-dependent)	`C-M-q`
indent region rigidly *arg* columns	`C-x TAB`
insert newline after point	`C-o`
move rest of line vertically down	`C-M-o`
delete blank lines around point	`C-x C-o`
join line with previous (with arg, next)	`M-^`
delete all white space around point	`M-\`
put exactly one space at point	`M-SPC`
fill paragraph	`M-q`
set fill column	`C-x f`
set prefix each line starts with	`C-x .`

Case Change

uppercase word	`M-u`
lowercase word	`M-l`
capitalize word	`M-c`
uppercase region	`C-x C-u`
lowercase region	`C-x C-l`
capitalize region	`M-x capitalize-region`

The Minibuffer

The following keys are defined in the minibuffer.

complete as much as possible	`TAB`
complete up to one word	`SPC`
complete and execute	`RET`
show possible completions	`?`
fetch previous minibuffer input	`M-p`
fetch next later minibuffer input	`M-n`
regexp search backward through history	`M-r`
regexp search forward through history	`M-s`
abort command	`C-g`

Type `C-x ESC ESC` to edit and repeat the last command that used the minibuffer. The following keys are then defined.

previous minibuffer command	`M-p`
next minibuffer command	`M-n`

GNU Emacs Reference Card

Buffers

select another buffer	C-x b
list all buffers	C-x C-b
kill a buffer	C-x k

Transposing

transpose **characters**	C-t
transpose **words**	M-t
transpose **lines**	C-x C-t
transpose **sexps**	C-M-t

Spelling Check

check spelling of current word	M-$
check spelling of all words in region	M-x ispell-region
check spelling of entire buffer	M-x ispell-buffer

Tags

find a tag (a definition)	M-.
find next occurrence of tag	C-u M-.
specify a new tags file	M-x visit-tags-table
regexp search on all files in tags table	M-x tags-search
run query-replace on all the files	M-x tags-query-replace
continue last tags search or query-replace	M-,

Shells

execute a shell command	M-!
run a shell command on the region	M-|
filter region through a shell command	C-u M-|
start a shell in window *shell*	M-x shell

Rectangles

copy rectangle to register	C-x r r
kill rectangle	C-x r k
yank rectangle	C-x r y
open rectangle, shifting text right	C-x r o
blank out rectangle	M-x clear-rectangle
prefix each line with a string	M-x string-rectangle

Abbrevs

add global abbrev	C-x a g
add mode-local abbrev	C-x a l
add global expansion for this abbrev	C-x a i g
add mode-local expansion for this abbrev	C-x a i l
explicitly expand abbrev	C-x a e
expand previous word dynamically	M-/

Regular Expressions

any single character except a newline	. (dot)
zero or more repeats	*
one or more repeats	+
zero or one repeat	?
any character in the set	[...]
any character not in the set	[^ ...]
beginning of line	^
end of line	$
quote a special character *c*	*c*
alternative ("or")	\\\|
grouping	\\(... \\)
*n*th group	*n*
beginning of buffer	\\`
end of buffer	\\'
word break	\\b
not beginning or end of word	\\B
beginning of word	\\<
end of word	\\>
any word-syntax character	\\w
any non-word-syntax character	\\W
character with syntax *c*	\\s*c*
character with syntax not *c*	\\S*c*

Registers

save region in register	C-x r s
insert register contents into buffer	C-x r i
save value of point in register	C-x r SPC
jump to point saved in register	C-x r j

Info

enter the Info documentation reader	C-h i
Moving within a node:	
scroll forward	SPC
scroll reverse	DEL
beginning of node	. (dot)
Moving between nodes:	
next node	n
previous node	p
move **up**	u
select menu item by name	m
select *n*th menu item by number (1–5)	*n*
follow cross reference (return with l)	f
return to last node you saw	l
return to directory node	d
go to any node by name	g
Other:	
run Info **tutorial**	h
list Info commands	?
quit Info	q
search nodes for regexp	s

Keyboard Macros

start defining a keyboard macro	C-x (
end keyboard macro definition	C-x)
execute last-defined keyboard macro	C-x e
append to last keyboard macro	C-u C-x (
name last keyboard macro	M-x name-last-kbd-macro
insert Lisp definition in buffer	M-x insert-kbd-macro

Commands Dealing with Emacs Lisp

eval **sexp** before point	C-x C-e
eval current **defun**	C-M-x
eval **region**	M-x eval-region
eval entire **buffer**	M-x eval-current-buffer
read and eval minibuffer	M-ESC
re-execute last minibuffer command	C-x ESC ESC
read and eval Emacs Lisp file	M-x load-file
load from standard system directory	M-x load-library

Simple Customization

Here are some examples of binding global keys in Emacs Lisp. Note that you cannot say "\M-#"; you must say "\e#".

```
(global-set-key "\C-cg" 'goto-line)
(global-set-key "\C-x\C-k" 'kill-region)
(global-set-key "\e#" 'query-replace-regexp)
```

An example of setting a variable in Emacs Lisp:

```
(setq backup-by-copying-when-linked t)
```

Writing Commands

```
(defun command-name (args)
  "documentation"
  (interactive "template")
  body)
```

An example:

```
(defun this-line-to-top-of-window (line)
  "Reposition line point is on to top of window.
With ARG, put point on line ARG.
Negative counts from bottom."
  (interactive "P")
  (recenter (if (null line)
                0
              (prefix-numeric-value line))))
```

The argument to `interactive` is a string specifying how to get the arguments when the function is called interactively. Type `C-h f interactive` for more information.

For copies of the GNU Emacs manual, write to the Free Software Foundation, Inc., 675 Massachusetts Ave, Cambridge MA 02139.

Appendix B

Some FTP and WWW Sites

Company	ftp	WWW
3COM	ftp.3com.com	www.3com.com
Adaptec	ftp.adaptec.com	www.adaptec.com
Adobe	ftp.adobe.com	www.adobe.com
Apple	ftp.austin.apple.com	www.apple.com
Ati	ftp.atitech.ca	www.atitech.ca
Borland	ftp.borland.com	www.borland.com
CA-Associates	mf.cai.com	www.cai.com
Compaq	ftp.compaq.com	www.compaq.com
Conner	ftp.conner.com	www.conner.com
Creative Labs	ftp.creaf.com	www.creaf.com
Dell	dell1.dell.com	www.dell.com
Diamond	ftp.diamondmm.com	www.diamondmm.com
Hewlett-Packard	ftp.cup.hp.com	www.hp.com
IBM	ftp.austin.ibm.com	www.ibm.com
Id_Software	ftp.idsoftware.com	www.idsoftware.com
Intel	ftp.intel.com	www.intel.com
Lotus	ftp.ccmail.com	www.ccmail.com
Maxtor	ftp.maxtor.com	www.maxtor.com
Micron	ftp.micron.com	www.micron.com
Microsoft	ftp.microsoft.com	www.microsoft.com
NEC	ftp.nec.com	www.nec.com
Novell	ftp.novell.com	www.novell.com
Oracle	ftp.oracle.com	www.oracle.com
Seagate	ftp.seagate.com	www.seagate.com
Supra	ftp.supra.com	www.supra.com
Symantec	ftp.symantec.com	www.symantec.com
Toshiba	ftp.tais.com	www.tais.com
US-Robotics	ftp.usr.com	www.usr.com
Western Digital	ftp.wdc.com	www.wdc.com