

Web Authoring



NCVA C20148 Web Authoring
Draft Version
2000

Notes

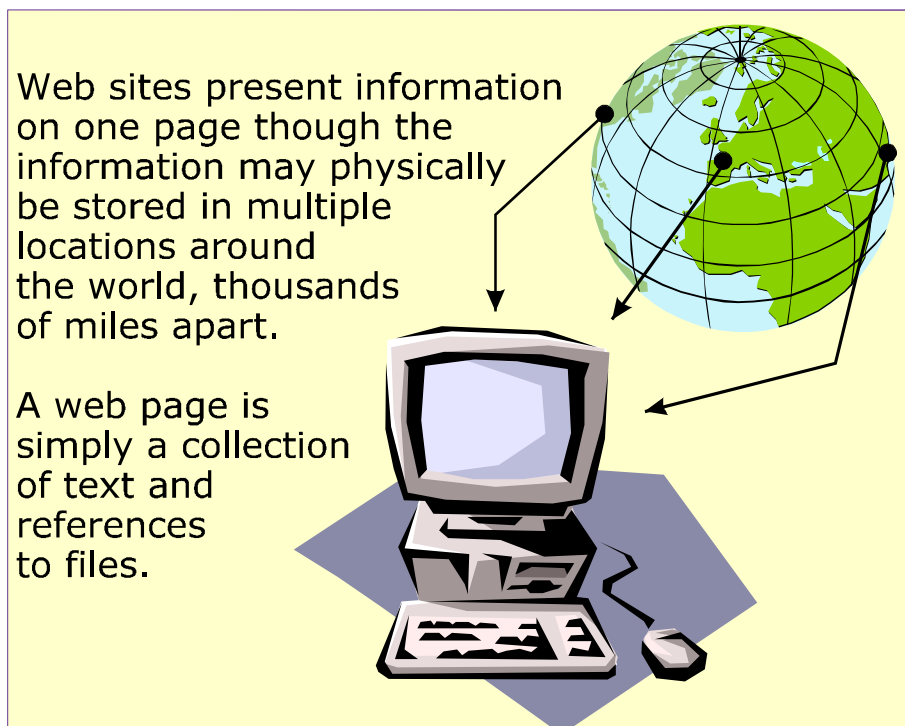
What is HTML?

HTML stands for **H**yper**T**ext **M**ark-up **L**anguage. This is the language of the **W**orld **W**ide **W**eb. The Internet is such a chaotic and confusing place that a user-friendly face had to be put on it. This friendly face is the WWW – which is simply a means of making the highly technical and complicated nature of the Internet seem easy and fun to use.

It's rather like turning the key of a car. Turning the key of a car is the user-friendly way of getting the engine to start. A rather less friendly way would be to use an old-style crank handle. Or to delve inside the engine and manipulate the spark plugs and engine valves manually. However, using a key is a more approachable way to start the engine. Nobody believes that turning the key starts the engine; turning the key starts the process that starts the engine – but your hands stay clean.

So it is also with the World Wide Web. It's worth noting that the web doesn't actually exist! There is only the Internet. But when we interact with the Internet using HTML we create the impression of a seamless easy to navigate collection of pages spread across the world.

Using HTML we can create documents that hide from the user the location of the data that they're accessing. HTML enables us to produce documents that can consist of multiple pages stored either on one location or in many locations. It isn't necessary for the user to know where the individual pages are located. They only need to know that they want the page – it's our job, using HTML, to enable them to get the document as easily as possible.



What's a HTML file?

A HTML document is what's called a plain text file. It contains nothing but plain alphanumeric ASCII characters stored in a commonly readable format. When the HTML file is loaded into a web **browser** the browser analyses the formatting information contained in the file and displays the content appropriately. For example, a web page that contains a picture doesn't have a corresponding HTML file that contains a picture. Instead a web page that contains a picture has a corresponding HTML file that contains a *reference* to a picture.

A few points are worth noting here. The Internet was originally largely a UNIX environment. In terms of the servers that make the Internet work, it still is. However, the end-users of the Internet may be using LINUX computers, or IBM mainframes or DEC Alpha workstations, or SUN SPARC stations, or Apple MAC computers, or AMIGA computers, or Windows based PCs. As such ASCII based documents are the only appropriate documents to use – since that's what ASCII is all about. ASCII stands for the American Standard Code for Information Interchange – it's the international standard for transferring data between different computer systems.

After the HTML file is downloaded it is loaded into a browser which looks at the formatting information contained in the file and then displays the file using it's own system specific steps.

For this reason, for example, HTML has options to display text using general instructions such as <BIG> where the browser decides what this instruction will display as, depending on local considerations. As such HTML is a truly cross-platform formatting and display language.

On most computer systems two parts, a filename and a filetype, identify a file. Written side-by-side both parts are separated by a full stop. For example, the file that contains the Microsoft Word program is identified by the file name WINWORD.EXE – the . EXE identifies the file as an EXEcutable computer program.

On PCs it's worth noting that how a downloaded file is handled is determined by the file type (also known as the file extension) associated with the file.

This partial directory listing lists file names and sizes, as well as data of last modification. Can you see the Excel program files? Can you spot the document template, or the database file?	10/29/97	12:00a	60,000	EFDFORM.HLP
	08/26/97	12:00a	49,152	EMAIL.DOT
	08/26/97	12:00a	529	EMPLOYEE.DBF
	10/29/97	12:00a	288	EULA8.CNT
	10/29/97	12:00a	39,427	EULA8.HLP
	08/26/97	12:00a	5,604,624	EXCEL.EXE
	08/26/97	12:00a	584,704	EXCEL8.OLB
	08/26/97	12:00a	37,048	EXCEL8.SRG
	10/29/97	12:00a	6,001	FATMS.SRG
	10/29/97	12:00a	15,120	FINDER.EXE
	08/26/97	12:00a	934	FINDFAST.CNT
	08/26/97	12:00a	111,376	FINDFAST.EXE
	08/26/97	12:00a	28,183	FINDFAST.HLP

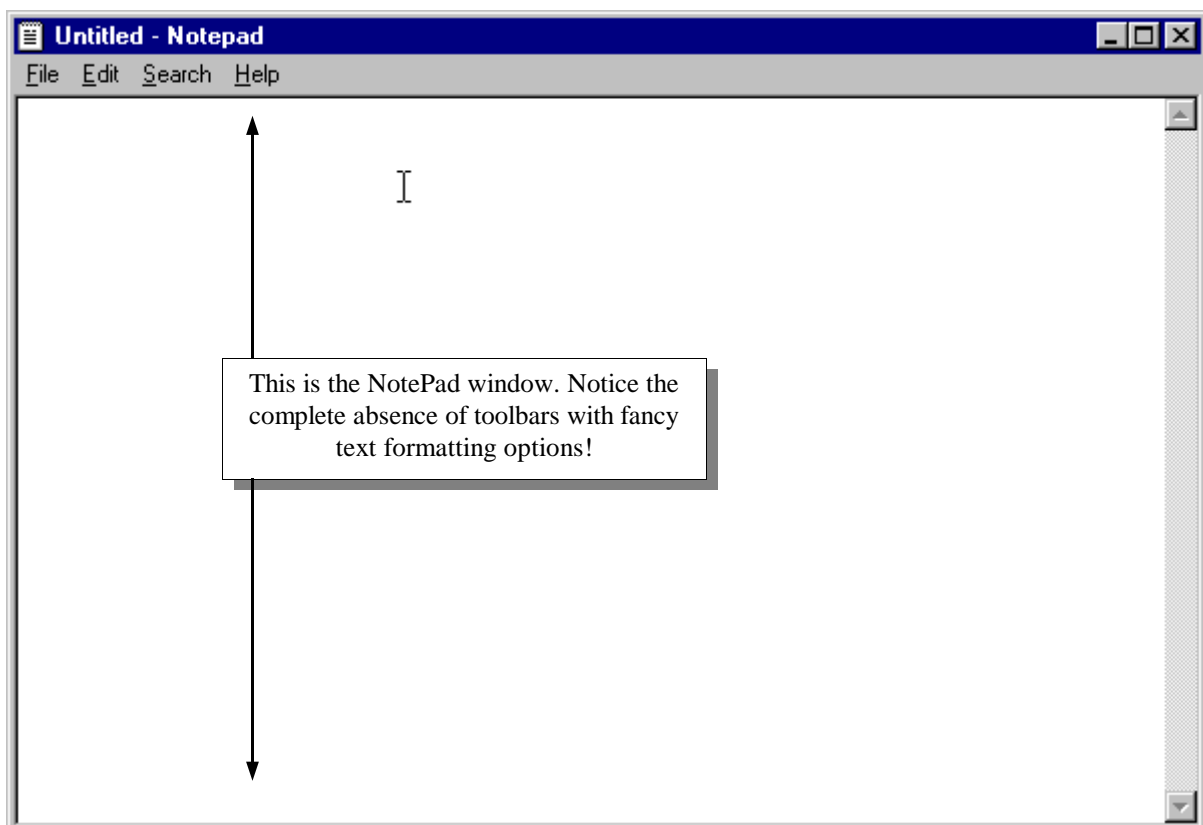
For example .TXT is understood on PCs to refer to plain-text files which are associated with the Notepad application. Files with the .DOC extension are associated with Microsoft Word. And as you saw earlier .EXE is one of the file extension used for computer programs on PCs.

.HTM and .HTML are the file extension associated with HTML files. When you double-click on a file that has either of the HTML extensions your operating system knows to open your Internet browser *because* of the extension.

This will cause us some little difficulty as we need to use the browser to view completed HTML files, but we also need to be able to edit the same files. But since double clicking to open a file calls up the browser we'll have to gain access to the file for editing by some other means. To further confuse matters we'll also want to be able to have the file open for editing and browsing *at the same time*.

To edit the files we need to use either a dedicated HTML manipulation program or a plain-text editor. In due time you'll use the dedicated web-editing programs, but we'll start with the plain-text editor called NotePad. NotePad is provided as part of the Windows operating system since every operating needs at least one plain-text editor.

To start NotePad click on Start/Programs/Accessories/Notepad.



What's a plain text editor?

You might not be aware of it but most of the files you create contain large amounts of formatting information. When you add such effects as bold and italic, or specify a different font and size, the computer application that you're using must store in your file not just the text that you've typed, but also information on the formatting of the text.

Plain text editors store none of this formatting information in a file because they don't support such formatting.

Plain text files can be opened by almost any application, as they contain no application specific data. It's for this reason that we can use NotePad to prepare HTML files.

What's a browser?

Browsers are what make the WWW work. The HTML files that are the content of the web mean little without dedicated programs to display their content. A browser is simply a program that knows how to correctly display HTML files and any other web content, as well as allowing you to interact with the content of those files, as necessary.

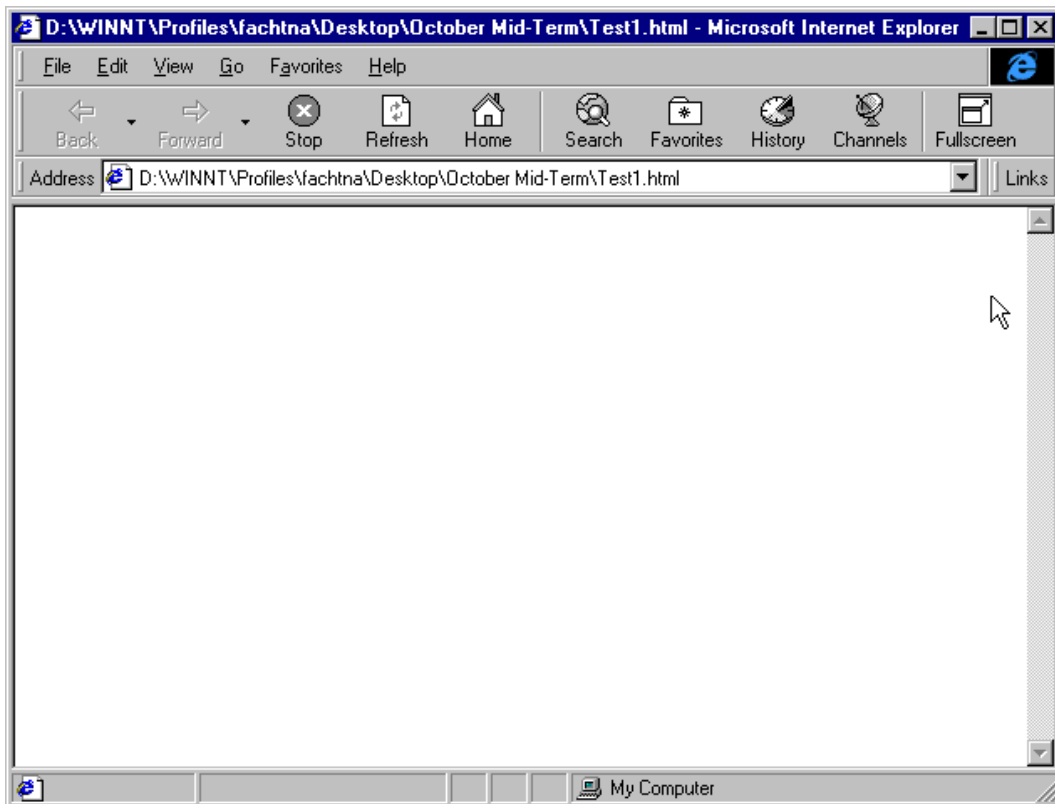
There are two main browsers in the world of IBM compatible PCs. These are Microsoft's Internet Explorer and Netscape's Navigator. Both of these browsers also exist on other platforms, such as Mac OS and UNIX. There are other PC browsers, such as Opera. Some of the 'original' browsers such as Lynx and Mosaic are commonly used on UNIX systems, though PC versions might also exist.

The choice of browsers is generally considered to be restricted to the offerings of the main two competitors in the field, Netscape and Microsoft. Both companies provide free versions of their software. There is much to be gained by these companies in defining standards on the Internet, and to be the company whose browser is most used is a target for both. Indeed, so valuable a prize is this that Microsoft give their Internet Explorer away free *as part of their operating system* Windows. This has led to a major legal battle in the United States of America about Microsoft's alleged abuse of its predominant position in the market place.

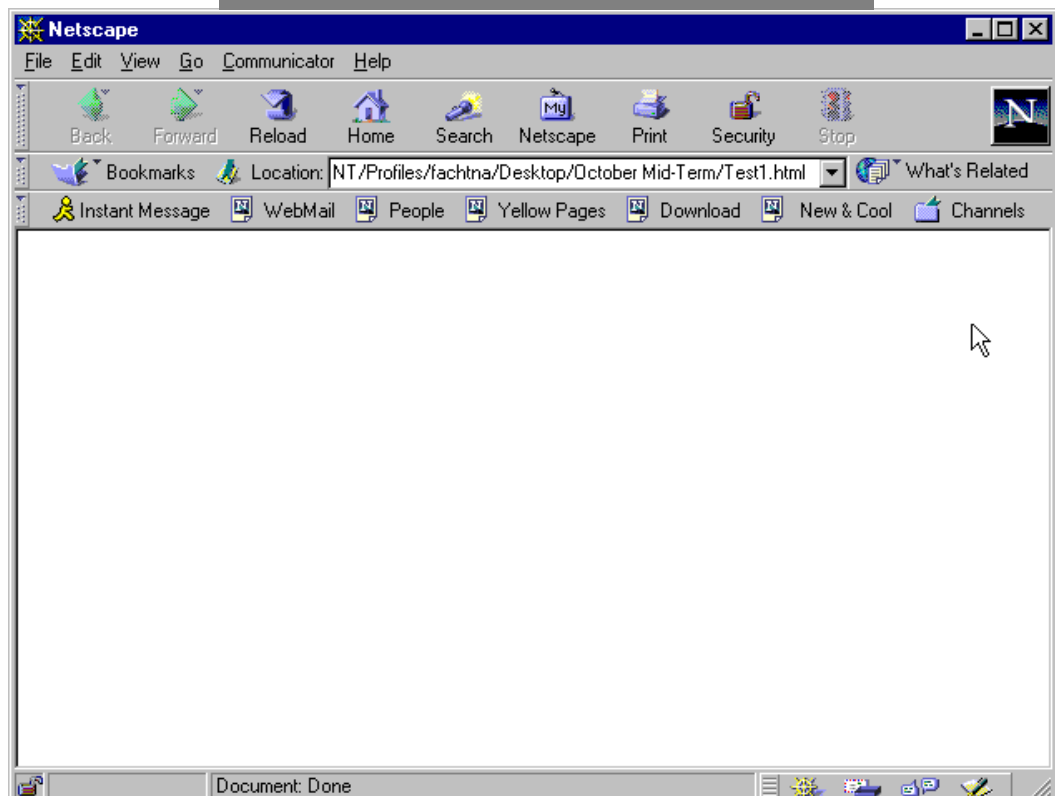
Since Netscape were in the browser market before Microsoft some of their products are far better. They're also a more innovative company. You should form your opinions as to which is the better browser – which you can only do by using both at some stage. If they're not already installed on your system you'll find the setup files on the **U:** drive, probably at **U:\Installation Files\Browsers**

You should also be aware that the Internet is an ever-changing place. The standards in use on the Internet are under constant review. The HTML standard is also being

changed. New browser versions also add new features, which can become de-facto standards overnight. Other features, such as Java Script become popular more slowly but succeed because of their usefulness.



Above: **Microsoft Internet Explorer** version 4.00
Below: **Netscape Navigator** version 4.07



What's in a HTML file?

As stated, a HTML file is just a plain-text file. Inside this file are a series of what are called **tags**. A HTML tag is always enclosed in angular brackets. Most tags exists in pairs, one tag to turn on an effect, one to turn off the same effect.

E.g. will start the use of bold text, will stop the use of bold.

E.g. <H1> will start the use of heading size 1 text, </H1> will stop its use.

As you can see, these example tags are paired. The **terminating tag** also contains a forward slash. Because these tags exist in pairs and surround the text in a page, they are also called **containers**.

There are certain tags that are always mean to be present in a HTML document. Every file should start with <HTML> to indicate the start of the HTML within the file and </HTML> at the end. The HTML is broken up into two main sections, the **head** and the **body**. These are indicated by <HEAD> and by </HEAD> tags for the head and <BODY> and </BODY> tags for the body.

The head of the document is mean to contain such details as the **title** of the page. The title is the title which will appear on the **titlebar** of the browser as the page is being viewed. There may be other items in the head, such as search terms for the Internet search engines, or details of the document author etc.

The body contains the actual content of the document. Here is where you'll say what you want to say, and display any pictures, graphs, tables, text which will comprise the page.

So far, every HTML file should contain these items:

```
<HTML>
<HEAD>
<TITLE>
Put the name of the page here
</TITLE>
</HEAD>

<BODY>
Say what you want to say in here
</BODY>
</HTML>
```

You can get away with leaving out some 'essential' tags and by taking many shortcuts. But as a general rule you should take the time to carry out every necessary step to create web pages in accordance with the standards of the day (and with reference to

previous standards) as some viewers of your pages will have older browsers which may not display your pages properly if you break too many rules.

Connecting to the Internet

To connect to the Internet you need a functioning **TCP/IP** network connection. On your network you should already have a functioning connection, courtesy of your hard-working sysop. ☞

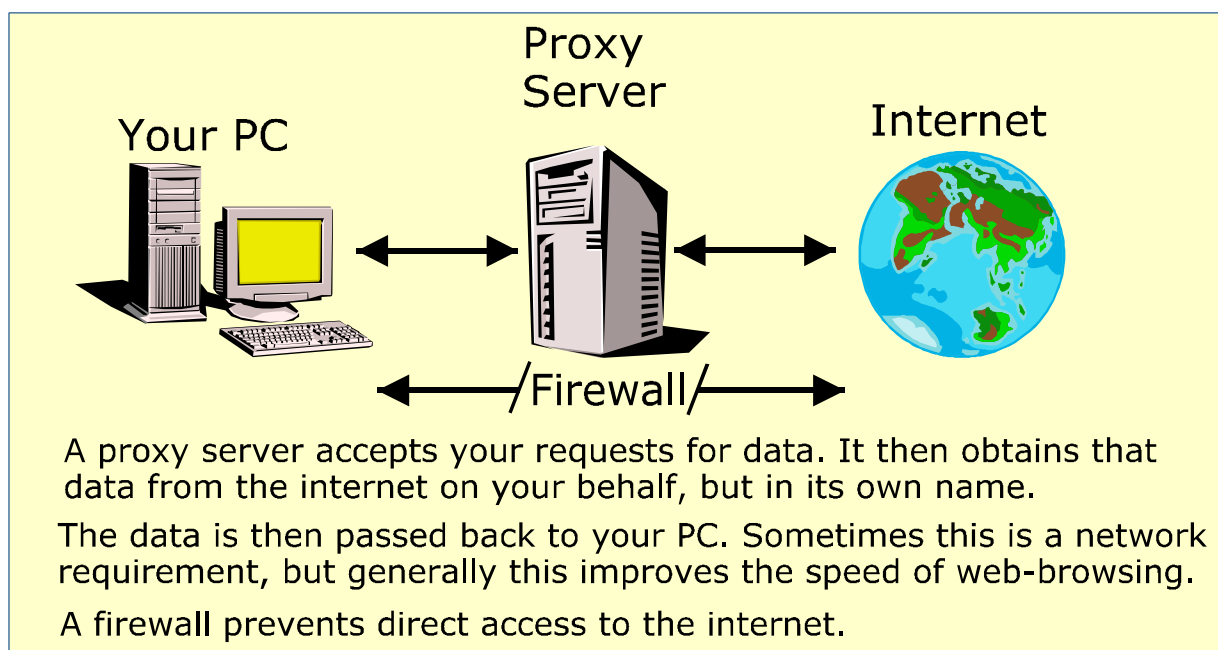
However, some networks feature **proxy-servers** and **firewalls** which prevent you from accessing the Internet directly. You can only gain access via the firewall. Our network uses a proxy server to provide Internet content. You need to provide the necessary details to your browser to enable it to communicate with Internet **hosts** via the proxy-server. If your browser tries to access remote hosts directly it will fail.

The necessary settings can be entered into Internet Explorer or Netscape Navigator. The place where you enter these details will vary. If you use Internet Explorer you must enter these values once on every machine that you use. Other persons using a machine after you will have to re-enter the settings. If you use Netscape Navigator the settings need only be entered once on every machine. So if someone has setup a computer once for Navigator you should not need to enter the settings again.

The necessary settings (beware! These are very open to change) are:

Proxy Type	Server	Port
*HTTP	proxy-server.cti-clonmel.ie	3128
FTP	proxy-server.cti-clonmel.ie	3128
SOCKS	proxy-server.cti-clonmel.ie	1080

*Mandatory entry



Some basic HTML tags

While there are many HTML tags that can be used, there are some simple tags which you'll use quite a lot. Some of them are given in the following table.

One thing, which adds to the power (as well as potential complexity) of the HTML language, is the **attributes** which can be used to further modify the tags that you'll use.

Tag	Terminating Tag	Function	Some Attributes	Comments
		Bold text		
<BIG>	</BIG>	Enlarges text to bigger than		
<BLINK>	</BLINK>	Causes text to flash		Don't over use!
<BODY>	</BODY>	Encapsulates body of HTML file	BACKGROUND BGCOLOR TEXT LINK VLINK	Mandatory
<BQ>	</BQ>	Quoted text	CLEAR	
 	None!	Break for a new line	CLEAR	Use to end paragraphs
<CENTER>	</CENTER>	Centre align text		
		Emphasise text		
	None!	Specify font to use	SIZE COLOR	Don't specify non-standard fonts
<Hx>	</Hx>	Specify text heading size	ALIGN CLEAR SKIP SRC	Valid values for x are 1-6
<HEAD>	</HEAD>	Encapsulate heading section		Mandatory
<HR>	None!	Draws a horizontal line	DIR ALIGN SIZE WIDTH	Useful to separate paragraphs, but don't overuse
<HTML>	</HTML>	Encapsulates whole document		Mandatory
<I>	</I>	Italics		
	None!	Insert an image file	SRC ALT ALIGN BORDER HEIGHT	
<MARQUEE>	</MARQUEE>	Creates an area of scrolling text	ALIGN BEHAVIOR BGCOLOR DIRECTION	Think before using
<P>	</P>	Paragraph break		</P> is optional
<PRE>	</PRE>	Defines an area of pre-	WIDTH	
<SMALL>	</SMALL>	Makes text smaller than normal		
		Makes text appear stronger		
_		Subscripted text		
[]	Superscripted text		
<TAB>	None!	Inserts a tab	ALIGN	
<TITLE>	</TITLE>	Defines title of the web page		Think about the ti-
<U>	</U>	Underlining		Don't over use
		An unordered list of elements	ALIGN PLAIN SRC WRAP	

The above table doesn't contain *all* the HTML tags, just some of them.

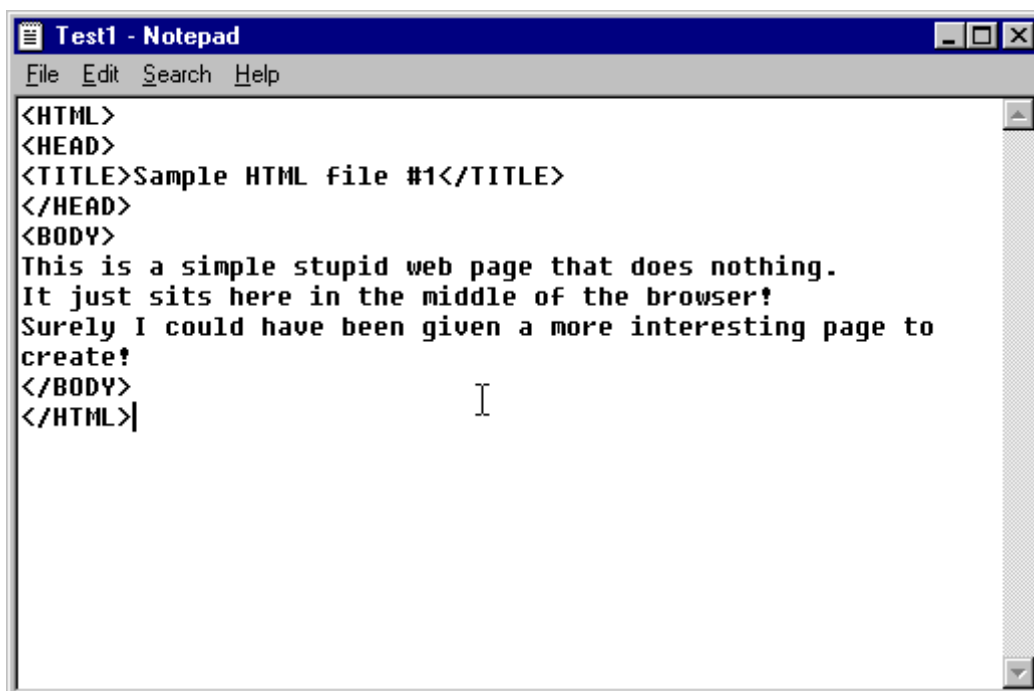
What you should know is that it's your own creativity that will produce impressive web pages. The HTML standard actually doesn't define a large number of page-building tools. Instead it defines a small set which you must use creatively.

Entering HTML code

? You can try to create a simple HTML document using NotePad and the browser of your choice. Start NotePad and enter the following HTML code:

```
<HTML>
<HEAD>
<TITLE>Sample HTML file #1</TITLE>
</HEAD>
<BODY>
This is a simple stupid web page that does nothing.
It just sits here in the middle of the browser!
Surely I could have been given a more interesting page to
create!
</BODY>
</HTML>
```

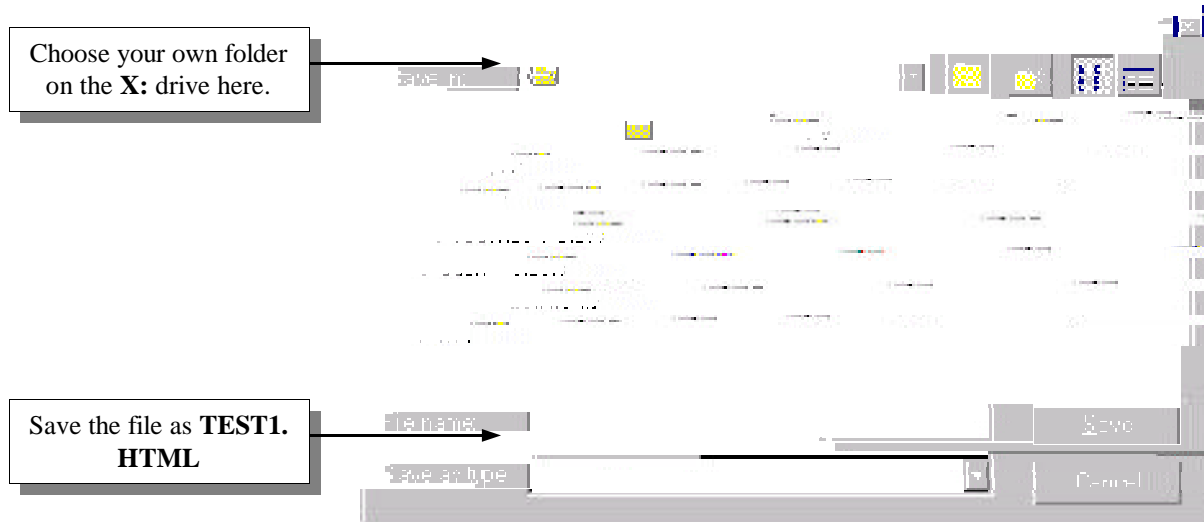
When the above code is entered, it should look as shown here:



The next stage is to save this file to disk. You can't just click on File and Save on this occasion. We first need to tell NotePad *how* to save this file. In reality this is only a small variation - you're not going to just provide a filename, you'll also provide the . HTML file type.

Saving the file as .HTML

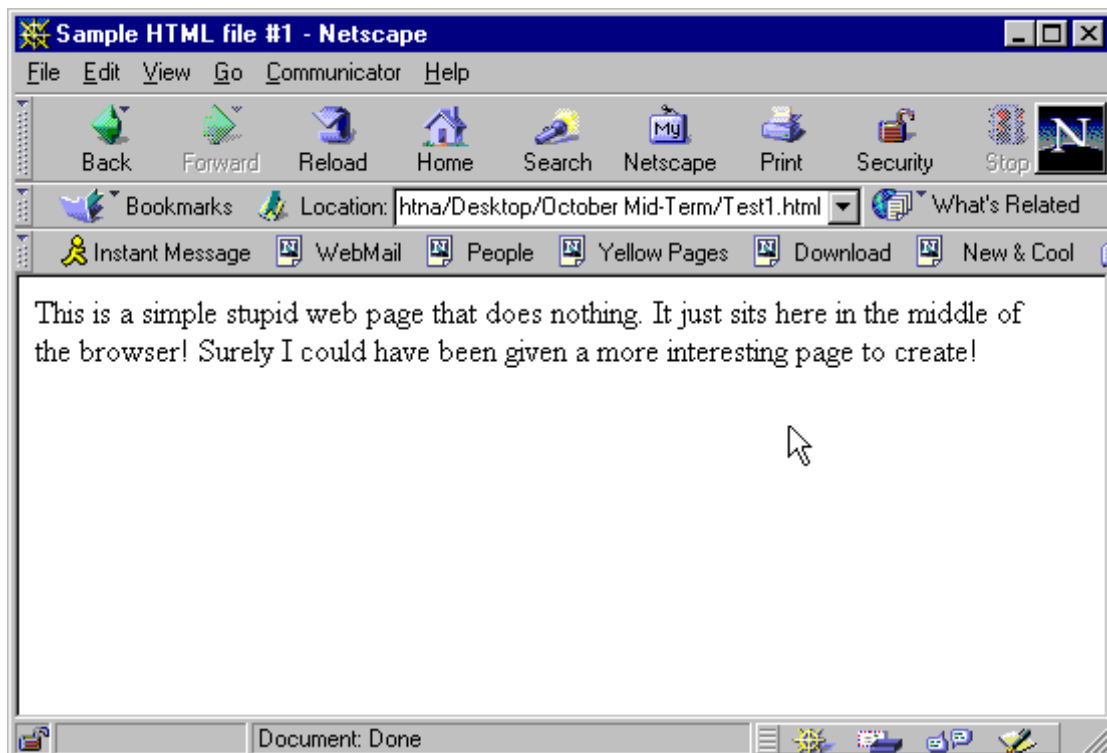
NB When you go to save the text ensure that you save it in your own folder on the **H:** drive. Also ensure that you add the letters **.HTML** to the name of the file before clicking on the Save button. The name of the file will be **TEST1.HTML**



By adding **.HTML** to the file name you make the file of type HTML and as such it will be recognised by the operating system as a web page.

Next start your browser. **Do Not Close Or Exit From Notepad.** Open the file you've created using File and Open from the browser menu.

The file when viewed in the browser should appear as shown below. If you cannot open the file or it looks *significantly* different then re-check the earlier steps you took.

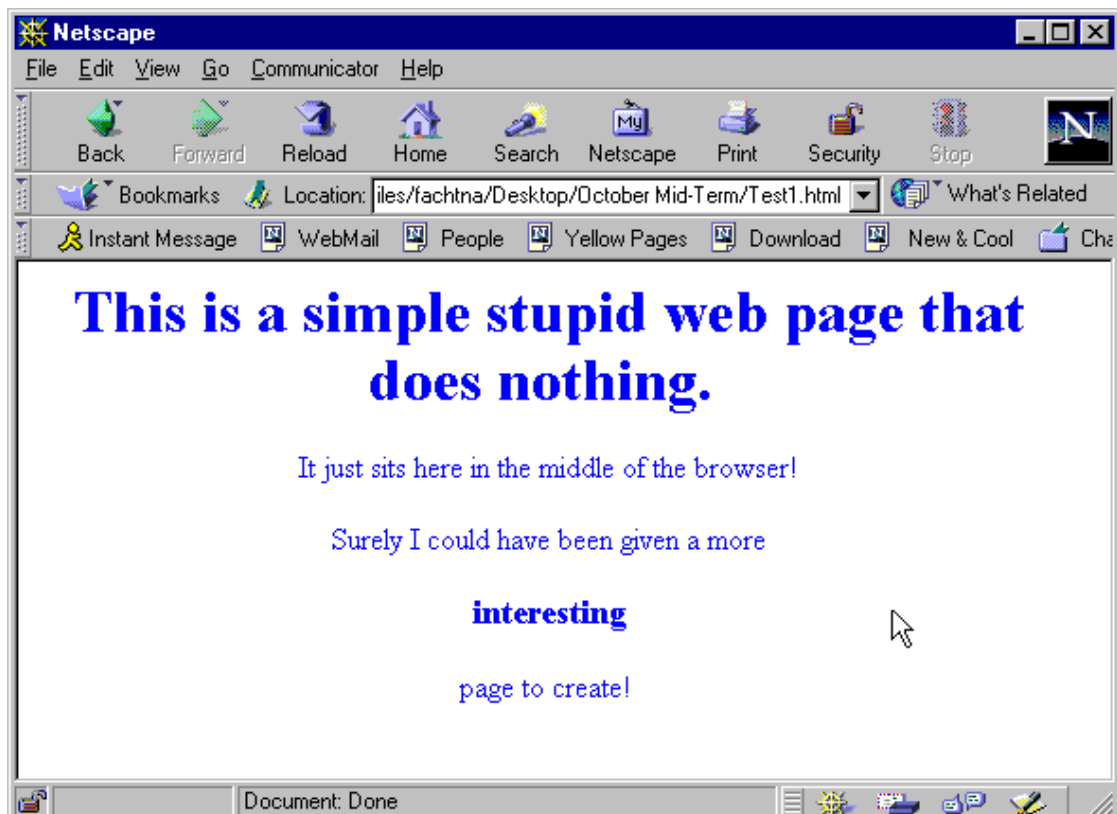


You've just created about the most simple HTML file that's possible. You should notice the difference in appearance between the text that you entered and the appearance of the page in the browser window. The difference is that the browser tags you entered aren't *displayed* by the browser. Instead they're acted upon.

Without closing the browser go back to NotePad. To do this use either the taskbar or ALT and TAB. When you're back in NotePad *modify* the text file to look as shown here:

```
<HTML>
<HEAD>
<TITLE>Sample HTML file #1</TITLE>
</HEAD>
<CENTER>
<BODY BGCOLOR=WHITE TEXT=BLUE>
<H1> This is a simple stupid web page that does nothing.</H1>
It just sits here in the middle of the browser!<P>
Surely I could have been given a more<P>
<B><BIG>interesting</BIG></B><P> page to create!
</CENTER></BODY>
</HTML>
```

When this change has been made save the changes. There is no need to use Save As... Now go back to your browser and click on either the reload button (Netscape) or the refresh button (Internet Explorer) as appropriate. Your reloaded page should now look as shown here:



Look closely at the changes you've made. Notice how we've added some attributes to the <BODY> tag.

Colours

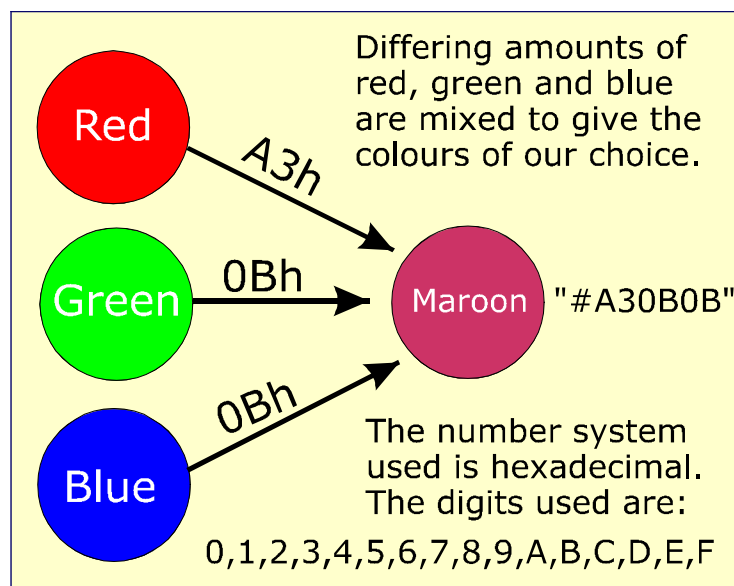
We specified a colour for the text and for the background. Some colours are defined namely, Black, Maroon, Green, Olive, Navy, Purple, Teal, Gray, Silver, Red, Lime, Yellow, Blue, Fuchsia, Aqua, White. These are not the only defined colours but cover the most popular options.

We can, of course, define our own colours. Before doing this it helps to know that computers use two colour systems. **RGB** is the **R**ed **G**reen **B**lue system and is used for monitors. **CMYK** is the **C**yan **M**agenta **Y**ellow **B**lack system and is used for many colour printers.

RGB is an additive system based on black. That is, the screen is black; as you add elements of red, green and blue other colours will form until white is the colour produced.

CMYK is an additive system based on white. That is, the paper in the printer is white; as amounts of cyan, magenta and yellow are added other colours will form. This method doesn't produce black so an extra print cartridge is used for that.

In the RGB system colours can be defined using "#RRGGBB" strings. We indicate how much red (RR), green (GG) and blue (BB) we want in the colour of our choice. Instead of RR we write a two-digit hexadecimal number representing how much red we wish to be present. Instead of GG we write a two-digit hexadecimal number representing how much green we wish to be present, *etc.* Since the numbering system used is hexadecimal, two digits mean 256 possible values between 00 and FF.



So instead of typing `BGCOLOR=WHITE` we could have defined the colour manually.

Since white is produced by combining all colours equally we could define white as “#FFFFFF”. Black is the absence of colour and is equivalent to “#000000”. Red is “#FF0000”. Yellow is “#FFFF00”. Green is “#00FF00”. Olive is “#A1A100”. Maroon is “#A30B0B”. Light purple is “#AAAADD”. Orange is “#FF9900”.

Try experimenting with text and background colours to see how the systems works. Enter different values, and go through the Save/Reload cycle to see the effect.

Since you're using hexadecimal the valid digits are: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F - use of any other digits will be seen as an error and 00 will be substituted. Experiment also with the background colours available. Watch out for disappearing text. This will happen if your text and background colours are the same.

Since there are 256 possible values each for red, green and blue it means that there are 16,777,216 possible colours that can be created. But don't get too carried away, most users of the Internet will have their systems set to have only 256 simultaneously displayable colours; and it's not the computer on which you generate the HTML that's important - it's the computer on which it's browsed that you need to think about. The golden rule of Internet publishing is: ***Remember the end user!***

Text alignment

Another change you made was to specify an alignment for your text using the `<CENTER>` tag. The other alignments available are `<RIGHT>` and `<LEFT>`. If you don't specify a text alignment `<LEFT>` is assumed.

When you specify a text alignment it remains in effect until you end it with a terminating tag, at which point left alignment will come back into effect. Specifying a new alignment will also end a previously specified alignment.

Under some other circumstances alignment effects will revert also. You'll see these later.

Breaking up text

You also used the `<P>` tag. `<P>` is used to indicate the start of a new paragraph of text. The `</P>` terminating tag is actually optional at the end of paragraphs, since by defining the start of another paragraph we implicitly terminate the previous paragraph.

You could have used the `
` line break tag instead. As you go on you'll see that both tags aren't actually the same. However, as this point in time we can use either.

Text emphasis

Notice how the word 'interesting' stands out in the page. This isn't just because of being on a line by itself. You used the tag to make the text bold. Other available tags are <U> for underlining and <I> for italic. Try changing the emphasis used to see how these tags work.

Text size

You also specified a different size for your text. It's enlarged by the <BIG> tag. This enlarges the text, to a degree determined by the browser, to become bigger than the current sized text.

The heading 'This is a simple stupid web page that does nothing.' is also larger than normal, through the use of the <H1> tag. This is a more definitely defined size. The available sizes are <H1> to <H6> - try changing the sizes used. Go through the Save/Reload cycle to see the effect. Remember to change the terminating tags as well as the opening tags!

Note that you can't use numbers other than 1-6 inside a <Hx> tag. Use of other values will be seen as an error and normal sized text will be displayed instead. See what happens if you use <BIG> *inside* the <H6> container. Remember to go through the Save/Reload cycle to see the effect of any changes you make.

? Create a new file, called TEST2.HTML and enter this code:

```
<HTML>
<HEAD>
<TITLE>Sample HTML file #2</TITLE>
</HEAD>
<BODY BGCOLOR="#0EAC00" TEXT="#CCCCFF">
<CENTER>
<H2>This is a simple heading</H2>
</CENTER>
<HR>
This is simple text placed <I>between</I> two horizontal
rules. This isn't <U>very</U> <U>very</U> important. If it
was, it would be in <B>bold.</B><BR>
Of course, anything <I>really</I> important would be made
<BIG><BIG>bigger</BIG></BIG> and <FONT
COLOR=YELLOW>brighter<FONT COLOR="#CCCCFF"> to make it
<BLINK>stand out.</BLINK>
<HR><SMALL><SMALL>On the other hand, <I>cosmically
speaking</I> does it <I></SMALL></SMALL></SMALL><FONT
COLOR=WHITE>matter<FONT
```

(Continued on page 17)

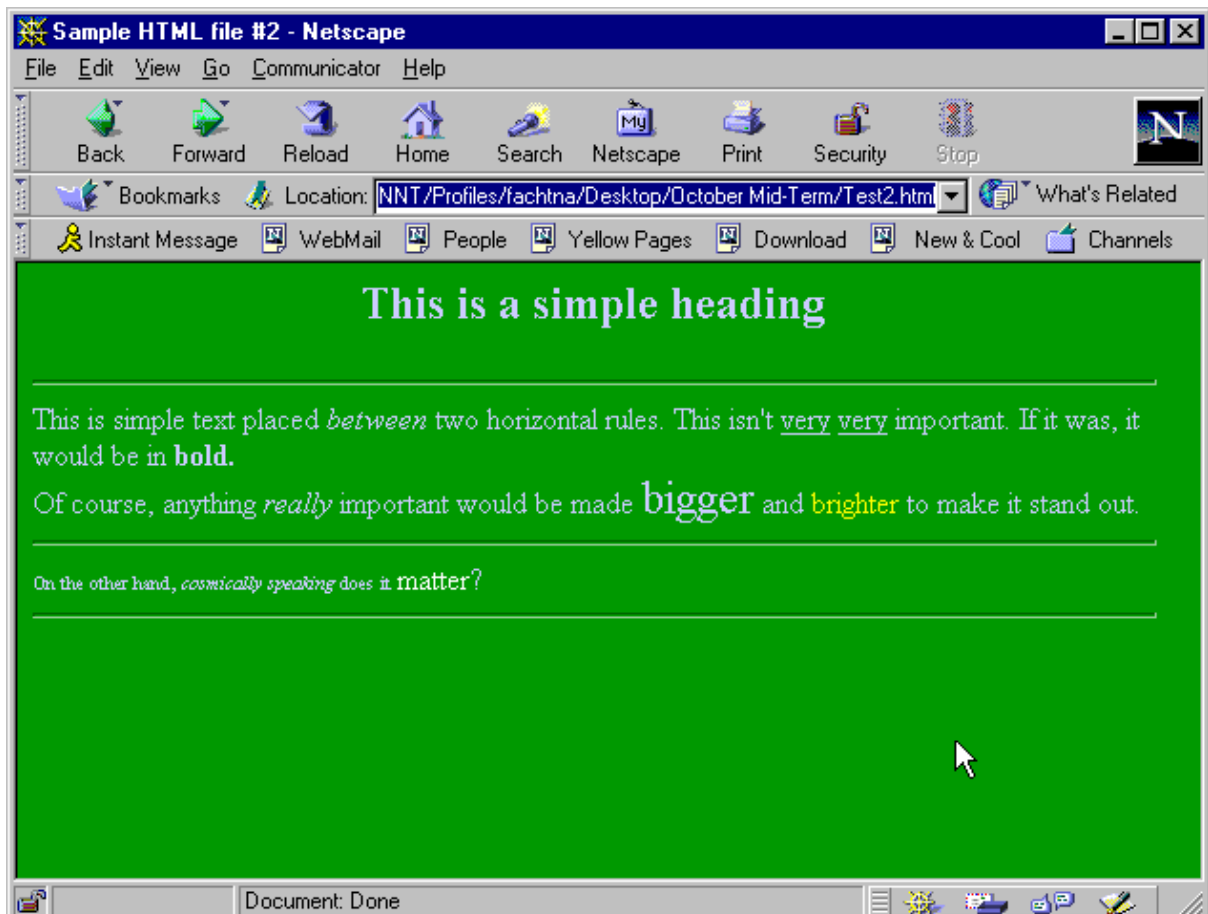
(Continued from page 16)

```
COLOR="#CCCCFF">?</I>
```

```
<HR>  
</BODY>  
</HTML>
```

The completed document should look more or less as shown below. If your completed file looks radically different, go back and check your work.

Before you go on you should be able to answer these questions:



What does <SMALL> do?

What colour is "#0EAC00"?

What does do?

What's the difference between <HR> and
?

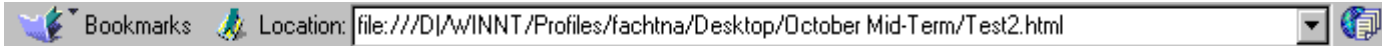
What's the difference between <HR> and <HR WIDTH=50%>?

Why does the font command read rather than ?

Why did we write <U>very</U> <U>very</U>? Why did we have <U> </U> in the middle of the code?

Document locations

If you look at the browser window again, you'll see beneath the menu and toolbars an **address** or **location** bar.



This line contains the address, or location, on the network of the file you're currently viewing. It should be something like:

file:///D:/WINNT/Profiles/fachtna/Desktop/October Mid-Term/Test2.html

This represents the location of your file, as well as the protocol used to access it. This known as a **URL** or **uniform resource locator** which uniquely specifies the location of the document. On the internet every object has it's own URL - which is why URL is also used to stand for **unique resource location**.

The general form of a URL is:

protocol://host/path/filename.filetype

The **protocol** is the rules used to access the file. The **host** is the name of the computer which contains the file. The **path** describes where on the host the file is stored. The filename.filetype specifies the file.

The path and filename are very often optional. If no protocol is specified the **http** protocol is general assumed. The protocol used defines *how* the data is to be transferred across the internet. Protocols are sets of rules. It's important that your computer and the computer that holds the data follow the same set of rules - otherwise the transfer probably won't work.

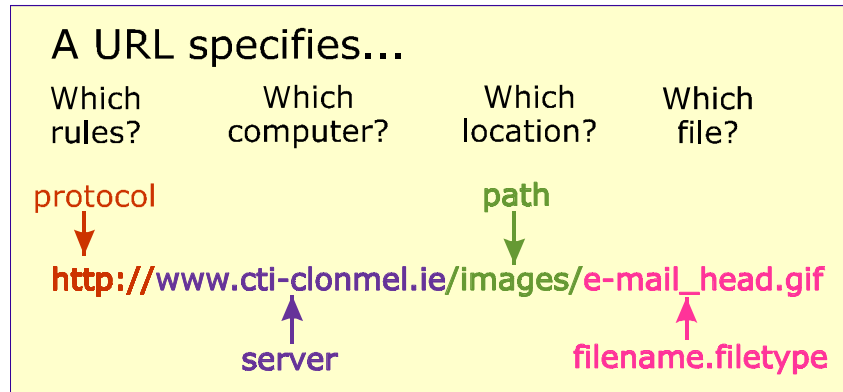
Part of the definition of a protocol is the **port** on which the **remote server** listens for requests for data. Web servers listen on port 80 - so requesting data from any other port won't yield the correct result.

Protocol	Means . . .	Port
http://	Hyper-Text Transfer Protocol	80
ftp://	File Transfer Protocol	21
file://	Gets files from your own system	n/a
telnet://	Remote CLI logins using telnet protocol	23
mailto://	Sends e-mail	n/a
news://	Connect to newsgroups	

Because we're working with web-pages, and **http://** is the default protocol used by web-browsers you'll probably not have to worry too much on a day-by-day basis about protocols.

However knowledge of protocols and URLs is considered to be basic in the world of

the Internet, as well as being essential to the creation of anything other than the most primitive of web-pages. The example shown here refers to a picture file which is part of the CTI web-site. The



protocol is specified as `http://`; the name of the computer is `www.cti-clonmel.ie`; the file is stored in a folder called `/images/`; the file itself is called `e-mail_head.gif`

Picture formats

The **.gif** means that this file is a picture in CompuServe **G**raphics **I**nterchange **F**ormat - one of the three graphics formats supported on the Internet. The two other acceptable formats are **.jpg** - which is the standard defined by the **J**oint **P**hotography **E**xperts **G**roup and **.png** - the **P**ortable **N**etwork **G**raphic.

These formats are supported by all browsers. These formats are acceptable because no license fee is required to use them, but also because they both give excellent **compression** of pictures.

By compressing pictures we reduce their file storage size, often without noticeably affecting the quality of the picture itself. This is advantageous as smaller files transfer across the Internet faster, since they require less **bandwidth**. Therefore the **download** time of the web pages is kept to a minimum.

The formats are pronounced as “dot gif” and “dot j-peg” - or just “gif” and “j-peg” as appropriate.

JPG has the added advantage of allowing us to specify a compression ratio, which means we can choose how much detail to lose from the picture prior to saving it.

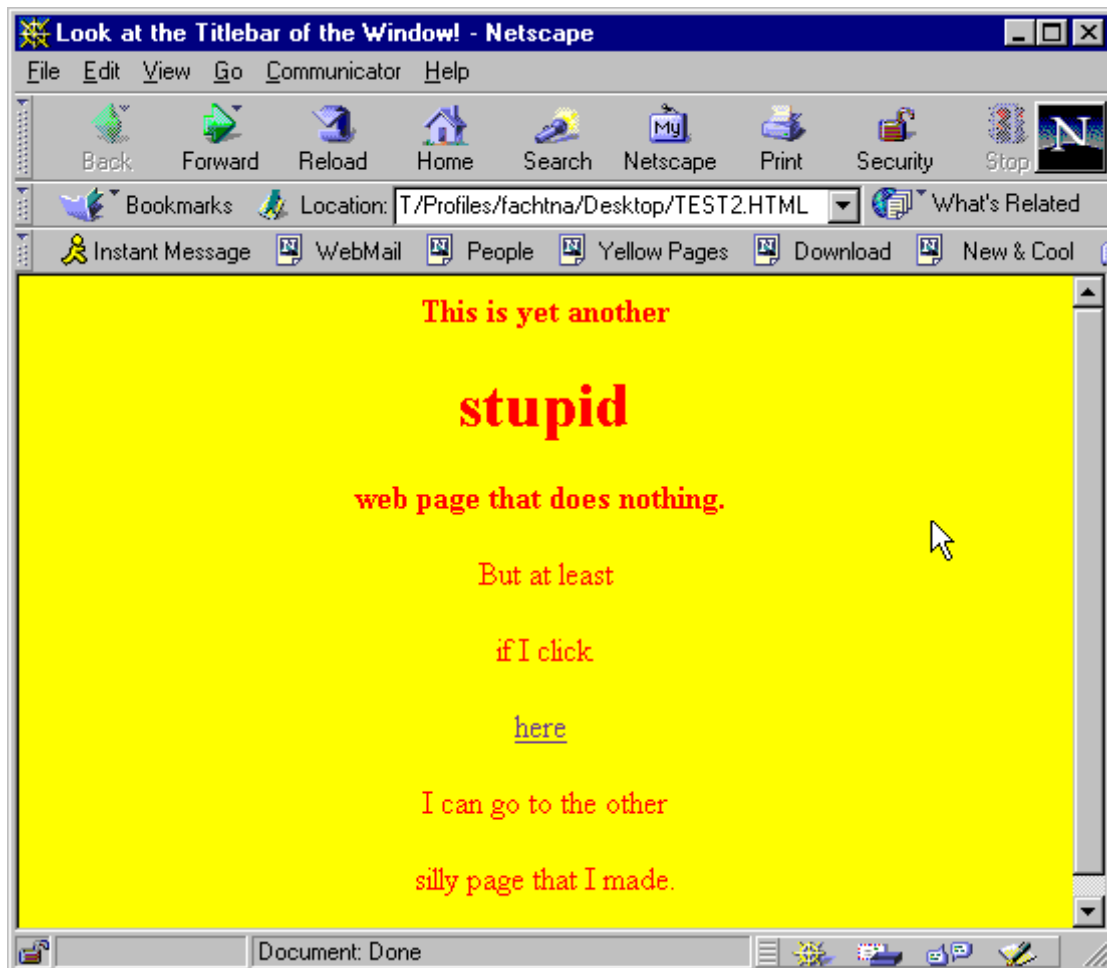
The use of other picture formats *is not supported and should not be considered*.


? Create a new HTML file and enter the code shown here:

```
<HTML>
<TITLE>Look at the Titlebar of the Window!</TITLE>
<BODY BGCOLOR=YELLOW TEXT=RED>
<CENTER>
<H4>This is yet another<H1>stupid</H1>
web page that does nothing.
</H4><P>But at least<P>if I click<P>
<A HREF="TEST1.HTML">here</A>
<P>I can go to the other<P>
silly page that I made.<P>
</CENTER>
</HTML>
```

Save this file using the name TEST3.HTML and then open it in your browser window. It should look as shown below.

Notice the word here in the middle of the page. Notice also how if you point at the



word that the mouse pointer changes to a pointing hand. This is because  through the use of the <A> tag we made the word here into what's called a **link**.

Links are what makes the Internet into the **World Wide Web**. They're

A pointing hand cursor signifies a

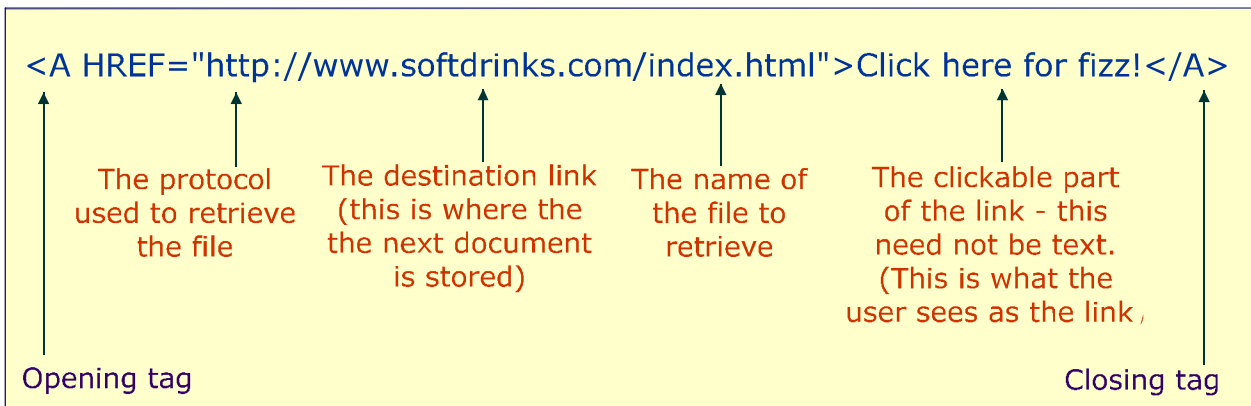
what makes the web navigable. Using them we enable users of our web pages to move from document to document with a single click of the mouse. The location of the documents is not necessarily of relevance, or even known, to the user. The location of the documents may be within the same system, or it may be on the other side of the world - it's still just a single click to get there.

Links are the basis of hypertext - which is the ability to move within a document or from document to document with great ease.

The <A> tag was used to create the clickable link. This tag takes two forms. It can be either when it defines a place to link *from*, or when it defines a place to link *to*. Normally you'll define places to link *from* as the other form of definition, though useful, is best used withing one document for the creation of **anchors**.

In this example the link was to another document that we had created. The command was here The <A > section defines where the browser is to locate the next document to display, if the user chooses the link. The tag terminates the link, but also terminates the text to be underlined as the clickable part of the link. The diagram below outlines the main parts of a link.

Note that while in the above diagram much detail was given about the location of the



document which was the link, in our example we only specified the name of the file to retrieve. We didn't specify the location of the file. We also didn't specify the protocol to use.

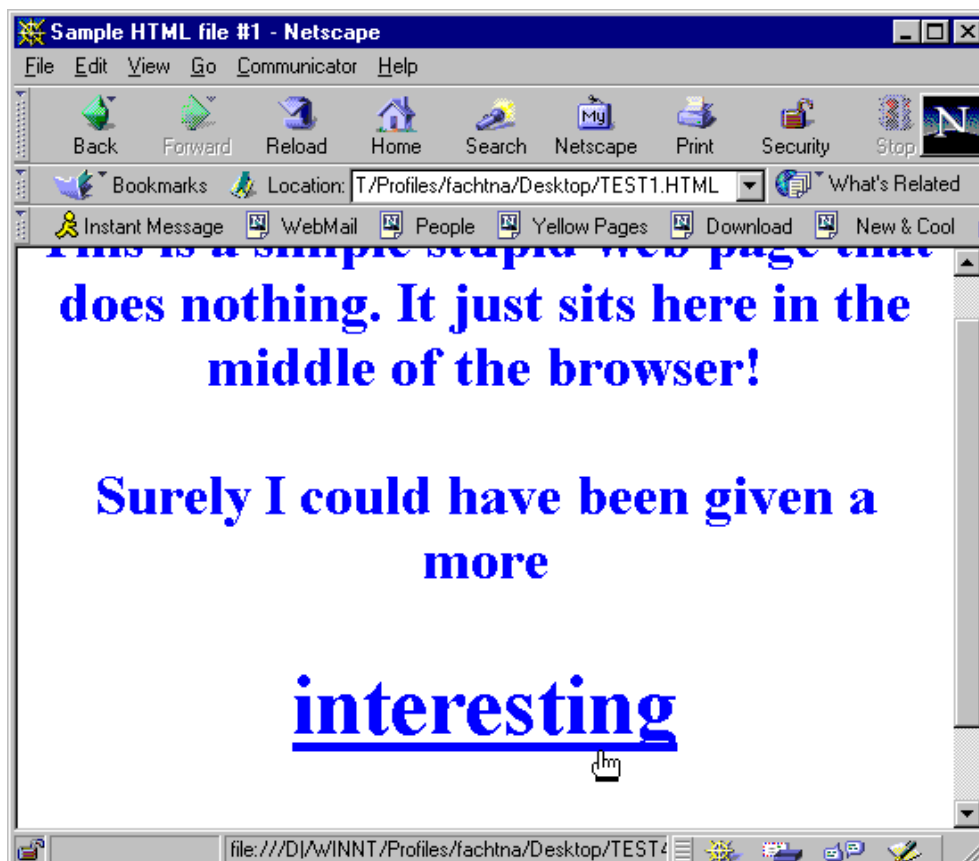
In the absence of a protocol being specified `http://` is used by the browser. In the absence of a computer name it's assumed that the next document will be loaded from the same computer as the referencing file. In the absence of path/directory/folder information it's assumed that the next document will be loaded from the same path as the referencing file. This means that since our link was just `HREF="TEST1.HTML"` that the browser will assume that both `TEST1.HTML` and `TEST3.HTML` are in the same folder. It therefore tries to load `TEST1.HTML` from the same folder in which

you created TEST3.HTML - if the document cannot be located, for any reason, you'll receive an error message.

To go back to the previous document you can use the back button on your browser. However, you can also add a link in the original file to enable you to move directly between your two files. Open up and edit the file TEST1.HTML to read as shown here:

```
<HTML>
<HEAD>
<TITLE>Sample HTML file #1</TITLE>
</HEAD>
<CENTER>
<BODY BGCOLOR=WHITE TEXT=BLUE>
<H1> This is a simple stupid web page that does nothing.
</H1>
It just sits here in the middle of the browser!<P>
Surely I could have been given a more<P>
<B><A HREF="TEST3.HTML"><BIG>interesting</BIG></A></B><P>
page to create!
</CENTER>
</BODY>
</HTML>
```

Save and reload the file. It should now look as shown below. The word **interesting** is



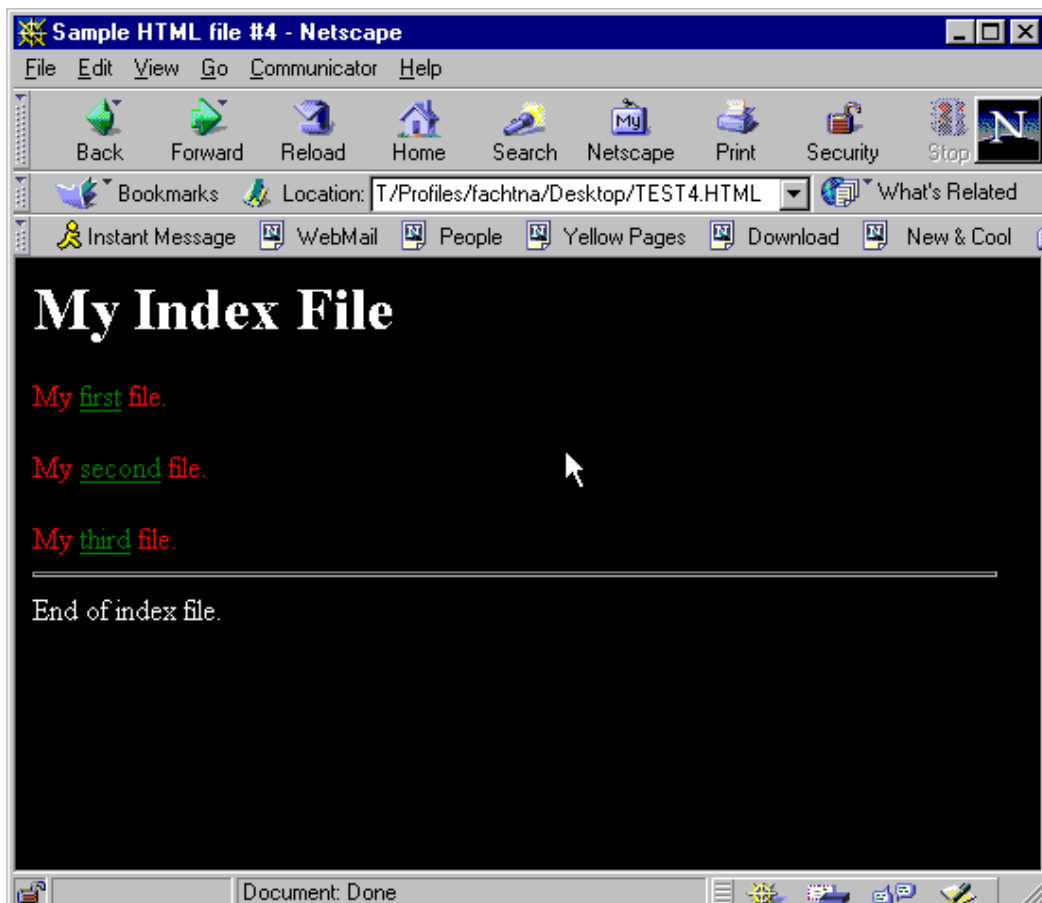
now a link to the file TEST3.HTML This means that you can move from file to file using links. You can still use the backward and forwards navigation buttons provided by the browser, but you can also use the links contained within the files to move from one to the other.

Try making an index file by entering this code, and saving it as TEST4.HTML

```
<HTML>
<HEAD>
<TITLE>Sample HTML file #4</TITLE>
</HEAD>
<BODY BGCOLOR=BLACK TEXT=WHITE VLINK=GREEN LINK=YELLOW>
<H1>My Index File</H1>
<FONT COLOR=RED>
<P>My <A HREF="TEST1.HTML">first</A> file.
<P>My <A HREF="TEST2.HTML">second</A> file.
<P>My <A HREF="TEST3.HTML">third</A> file.
<HR>
</FONT>
End of index file.
</BODY>
</HTML>
```

The completed file should look as shown here:

This sequence of files breaks what's considered to be a golden rule of the web. There

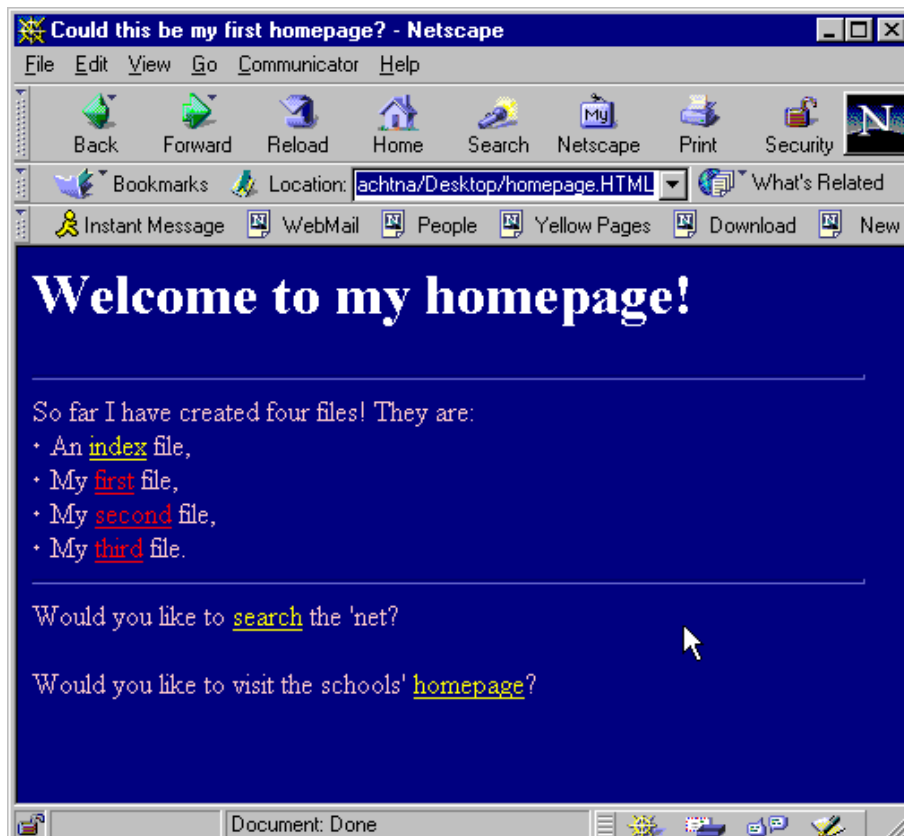


should be no dead ends on the web. Every file should have at least one link. Suggested links include a link back to the referring page, to an index file, to a search engine or to another page in a sequence.

For practice, modify each file you've created so far so that each page has a reference to at least one other page, for example to the index file you've just created.

Try creating a home page. Save the file as INDEX.HTML - enter and then modify the HTML shown here to suit your own requirements.

```
<HTML><HEAD>
<TITLE>Could this be my first homepage?</TITLE>
</HEAD><BODY BGCOLOR=NAVY TEXT=PINK VLINK=RED LINK=YELLOW>
<FONT COLOR=WHITE><H1>Welcome to my homepage!</H1></FONT>
<HR>So far I have created four files! They are:
<BR><LI><P>An <A HREF="TEST4.HTML">index</A> file,
<LI><P>My <A HREF="TEST1.HTML">first</A> file,
<LI><P>My <A HREF="TEST2.HTML">second</A> file,
<LI><P>My <A HREF="TEST3.HTML">third</A> file.
<HR></FONT> Would you like to <A HREF="http://altavista.
digital.com">search</A>
the 'net?
<P>Would you like to visit the schools' <A
HREF="http://www.cti-clonmel.ie/">homepage</A>?</BODY>
</HTML>
```



The completed page will look similar to this one:

```
<HTML><HEAD>
<TITLE>Could this be my first homepage?</TITLE>
</HEAD><BODY BGCOLOR=NAVY TEXT=PINK VLINK=RED LINK=YELLOW>
<FONT COLOR=WHITE><H1>Welcome to my homepage!</H1></FONT>
<HR>So far I have created four files! They are:
<BR><LI><P>An <A HREF="TEST4.HTML">index</A> file,
<LI><P>My <A HREF="TEST1.HTML">first</A> file,
<LI><P>My <A HREF="TEST2.HTML">second</A> file,
<LI><P>My <A HREF="TEST3.HTML">third</A> file.
<HR></FONT> Would you like to <A HREF="http://altavista.
digital.com">search</A>
the 'net?
<P>Would you like to visit the schools' <A
HREF="http://www.cti-clonmel.ie/">homepage</A>?</BODY>
</HTML>
```