## File Transfer Protocol (FTP)

- Another High Level Protocol
- Is used to facilitate the transfer of files from one host to another
- Users can use FTP to transfer files to/from two hosts when they possess an account (**Username/Password**) on each host
- Internet users may receive files from hosts which have been set up with *anonymous FTP*

## Anonymous FTP

- *Anonymous FTP* accounts have been set up on hosts which have been designated *archive* sites
- These accounts have limited access rights, as well as some operating restrictions
- Internet users log in with username *anonymous* and a password with their email address
- Using email addresses allows the administrators to monitor who is using their services
- To retrieve a file, users need to know the host to connect to and the pathname of the file
- Note that there are some variations on how users connect and use specific hosts, i.e. don't assume all are set up the same
- There are differences in the implementation of FTP commands at sites

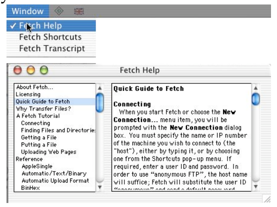
## Downloading/Uplading files — Using Web Browser and FTP Client

- FTP file downloads (*server-to-client*) may be performed either
  - using a Web Browser, Safari/Explorer/Netscape, with `ftp://ftp.site.com` -type URL, or
  - with a specialized FTP client, such as **Fetch (Macintosh)**, WS_FTP (for Windows), ftp (DOS), xftp,ftp (UNIX).
- FTP file uploads (*server-to-client*) best performed
  - with a specialized FTP client, such as **Fetch (Macintosh)**, WS_FTP (for Windows), ftp (DOS), xftp,ftp (UNIX).
  - Need **correct access privileges** for uploads to work.
  - Servers need to be configured correctly for **Web Browser Uploads** — Not Guaranteed.

## Mac FTP Client — Fetch

The preferred method of FTP for this course is to use **Fetch** on the Macintosh Computers.

- Fetch is a relatively easy to use package
- **Basically Drag'nDrop or Menu Driven**
- See help facility available from Fetch Window Menu for full info.



**Quick Guide to Fetch**

**Connecting**
When you start Fetch or choose the **New Connection...** menu item, you will be prompted with the **New Connection** dialog box. You must specify the name or IP number of the machine you wish to connect to (the "host"), either by typing it, or by choosing one from the Shortcuts pop-up menu. If required, enter a user ID and password. In order to use "anonymous FTP", the host name will suffice; Fetch will substitute the user ID

- Also see the online fetch documents (FAQs,...) at http://www.fetchsoftworks.com

## How Does FTP Operate

An FTP client and server actually make two connections:

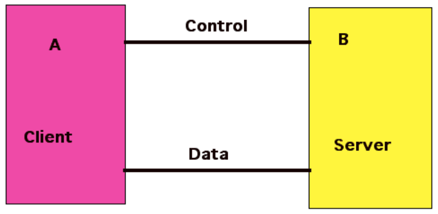
- Control Connection — FTP Protocol Dialogue
- Data Connection — File Transfer

FTP Control Connection

- Must be functioning for Data transfer to occur
- Control connection utilised the TELNET protocol (see later)
- Special FTP commands and responses — **the FTP Protocol**
- Text (ASCI) Command line oriented.

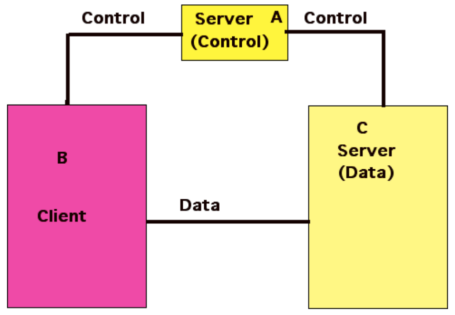
## FTP Standard Control Connection

FTP Connection is usually between 2 machines (A (Client) and B (Server)):



## FTP Alternative Control Connection

FTP Can support a more distributed model (Computers A (Control), B (Client) and C (Data)):



## A Command Line FTP Client

Let is study how we use such simple FTP applications:

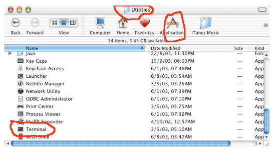
- first as they are sometimes very useful to use (over a Telnet client for example) and
- **we can gain a good appreciation of what is sort of communication is happening before we look at the actual FTP PROTOCOL next**

Note: You can actually see the PROTOCOL in operation of you use the `debug` command option (see below).

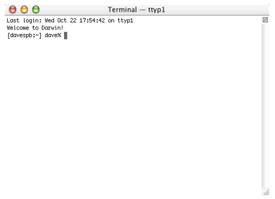
## First We Need a Terminal Window

Before we run the FTP client we need to open a Mac Teminal window:

- The Mac Terminal Application is located in the Applications/Utilities sub-folder on the Macs
- Double-click on application to get a Terminal Window
- Similar Process to get a UNIX/DOS Terminal Window



| Name | Date Modified | Size | Kind |
|---|---|---|---|
| Java | 22/8/03, 11:30PM | -- | Fold |
| Key Caps | 15/9/03, 06:03PM | -- | App |
| Keychain Access | 6/1/03, 07:48PM | -- | App |
| Launcher | 6/8/03, 03:54AM | -- | App |
| NetInfo Manager | 3/5/03, 05:26AM | -- | App |
| Network Utility | 6/1/03, 07:39PM | -- | App |
| ODBC Administrator | 6/1/03, 07:10PM | -- | App |
| Print Center | 3/5/03, 05:25AM | -- | App |
| Process Viewer | 6/1/03, 07:32PM | -- | App |
| StuffIt Expander | 4/10/02, 12:57AM | -- | App |
| Terminal | 3/5/03, 05:10AM | -- | App |
| Wish Shell | 6/8/03, 03:47AM | -- | App |



```
Last login: Wed Oct 22 17:54:42 on ttyp1
Welcome to Darwin!
[davespb:~] dave%
```

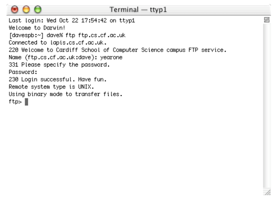
## Running a Simple FTP Client

To run such clients simply type `ftp` from the Mac/UNIX Terminal command line:

```
> ftp
```

or to access server more quickly give the FTP address as well:

```
> ftp ftp.cs.cf.ac.uk
```



```
Last login: Wed Oct 22 17:54:42 on ttyp1
Welcome to Darwin!
[davespb:~] dave% ftp ftp.cs.cf.ac.uk
Connected to lapis.cs.cf.ac.uk.
220 Welcome to Cardiff School of Computer Science campus FTP service.
Name (ftp.cs.cf.ac.uk:dave): yearone
331 Please specify the password.
Password:
230 Login successful. Have fun.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Next, You will usually be asked to login

- Username
- Password

## FTP Client Application Commands

**Note: These commands ARE NOT Part of the PROTOCOL**
(Although some are very closely related)

All the FTP client session commands may be abbreviated to their first three charcters, *e.g.*

```
ftp> hel
```

`help` **or** `?` `[ command-name ]` — list of all commands

```
ftp> rem  ***
```

`remotehelp [ command-name ]` — Request help from the remote FTP server.

## Seeing the actual FTP Protocol in Action

This is an excellent way for INVESTIGATION of ACTUAL PROTOCOL COMMANDS AND The SERVER REPSONSE

See **Internet Computing Lab Worksheet 3**.

To Turn ON/OFF `ftp` client protocol command listing:

`debug` — Toggle debugging mode.

`verbose` — Toggle verbose mode.

## Conversing with the Server in the actual FTP Protocol

Again an excellent way for INVESTIGATION of ACTUAL PROTOCOL COMMANDS AND The SERVER REPSONSE

To actually converse with the server in **Actual protocol commands** use:

```
quote arg1 arg2 ...
```

where `arg1 arg2 ...` are actual protocol commands.

**More on this later**

See **Internet Computing Lab Worksheet 3**.

## Connecting and Login On Inside `ftp`

If you have not already done so at `ftp` start up
(or if you quit and wish to start another session inside `ftp`)
use the following commands:

`open host [ port ]` — Establish a connection

`user user-name [ password ] [ account ]` — Identify yourself to the remote FTP server.

`account [ passwd ]` — Supply a (supplementary) password

## File Transfer Type

**IMPORTANT**: **GET THIS RIGHT FOR FILE TYPE**:

- (Raw) text files —- ASCII endcoded
- All audio, image, video files — binary encoded
- (Almost) All application files (e.g. Word Docs, excel files) — binary encoded
- all compressed files (e.g. zip, binhex) — binary encoded
- Uuencoded files (`.uu`) — ASCII endcoded

## Setting the File Transfer Type

You must set file correct file type, Before Transfer:

**ascii** — transfer of text only files

**binary** — transfer of files that contain binary data,

`type [ type-name ]` Set the "representation type" to `type-name` (ascii/binary).

Note: Most GUI FTP clients allow for automatic detection of transfer type
Dont trust them to get this right!!!

## Directory Traversal: Remote (Server) Side

It is useful to know which directory you are currently working in and also change this if you are not in the right one. You may also wish to make new ones (if you are allowed)

The follow commands achieve this:

`pwd` — List the name of the current working directory on the remote machine. For example:

`cd remote-directory` — Change the working directory.

`cdup` — Change the remote machine working directory to the parent.

`dir [ remote-directory ] [ local-file ]` — Supply a listing of the directory contents

`ls [ remote-directory | -al ] [ local-file ]` — Supply an abbreviated listing of the contents of a directory

`mdir remote-files local-file` — Like dir, except multiple remote files may be specified.

`mls remote-files local-file` — Like `ls`, except multiple remote files may be specified.

`mkdir directory-name` — Make a directory on the remote machine.

## Local (Client side) FTP commands:

You may wish to change where you download files to or upload files from on your computer:

`lcd [ directory ]` — Change the working directory on the local machine. If no `directory` is specified, the user's home directory is used.

**Note**: that new local directories can be created with

`!mkdir directory` and listed with `!ls` or `!dir`.

The `!` can in general be used to force many commands to work on the local side rather than the server side — it forces a local "command shell" to be created.

Commands executed after `!` are dependent on OS of client — Usually UNIX shell commands or perhaps DOS.

## FTP wildcards

**How can I list multiple (perhaps not all) files or down/upload multiple files (we'll see how very soon)?**

Traditional UNIX FTP servers allow the `ls, dir ....` commands to name several files. For example, the command

```
dir *.ps *.ps.gz
```

which refers to every file in the current directory whose name ends with `.ps` or `.ps.gz`.

The wild cards can also be used for multiple file downloading and uploading (which we now go on to discuss).

## Retrieving Files

### The Main Purpose of FTP

Use one of the following commands:

`get remote-file [ local-file ]` — Retrieve the `remote-file`

`recv remote-file [ local-file ]` — The same as for `get` .

`mget remote-files` — **Multiple Get (Next Slide)**

## Multiple Get

The `mget` command is typically used with wild cards (see `FTP` wild cards above). E.g.

```
mget *.ps *.ps.gz
```

which downloads to every file in the current directory whose name ends with `.ps` or `.ps.gz`.

## The `prompt` Command

The setting of the `prompt` is also useful for use with `mget` (and `mput` which we see below)

`prompt` — Toggle interactive prompting. By default, prompting is turned on.

If prompting is :

**Turned on** — you have to repsond yes (`y`) or no (`n`) to multiple down/upload request for each file

- sometimes useful — to select files more finely than a wildcard allows
- can be tedious if wildcard can select all files already

**Turned off** — any `mget` or `mput` will transfer all files, and any `mdelete` (see below also) will *delete all files*.

## Uploading Files

The other primary use of FTP

The following commands maybe used:

`put local-file [ remote-file ]` — Store a local file on the remote machine.

`send local-file [ remote-file ]` — The same as for `put`.

`append local-file [ remote-file ]` Append a local file to a file on the remote machine.

`mput local-files` — *Multiple put*, similar to `mget`

## Terminating the FTP session and Quitting `ftp`

**When have finished all file transfers you will have to**:

`close` /`disconnect` — Terminate the FTP session with the remote server, and return to the command interpreter.
You could connect (`open`) to another FTP server, OR

`bye`/`quit` — Terminate the FTP session with the remote server and exit `ftp`.

## The FTP Protocol

Like the email protocols SMTP and POP, you enter into a dialog with a (FTP) server

- Fetch (menu, mouse driven GUI FTP Clients): the Protocol *converstaion* is taken of for you.
- Command line based (UNIX/DOS) `ftp` clients
  - you can turn on `debug` to see the Protocol command conversation and
  - use `quote` to converse in the protocol yourself
- All protocols are 4 ASCII Characters long
- You connect to FTP Port Numbers 21 (Data) and 20 (Commands) on the server.

## FTP Protocol Commands

### Login on

- The client normally begins each FTP connection with a `USER` request;
- then, depending on the server's response, a `PASS` request;
- and then, depending on the server's response, an `ACCT` request.

## The `TYPE` command — Setting File Transmission Type

- A `TYPE` request controls the binary flag.
- It requiures a parameter after the command.
  There are four possibilities for the parameter:

  **A** — Turn the ASCI Text flag on.
  (A for Ascii)
  **A N** — Turn the ASCI Text flag off.
  (N for non-print, some other flags)
  **I** — Turn the binary flag on.
  (I for Image)
  **L 8** — Turn the binary flag off.
  (L for Local Byte Size, can use other numbers (not common)

  Example: `type I`

- The server accepts the `TYPE` request with code 200.

## Navigating Directories

- A `PWD` request asks the server to list the Present Working Directory.
- A `CWD` (Change Working Directory) request has a single parameter giving a pathname for a directory to change to. I
- A `CDUP` request asks the server to remove the last slash, and everything following it, from the name prefix.
  If this produces an empty name prefix, the new name prefix is a single slash.

## Listing Files: The `LIST` and `NLST` commands

A `LIST` or `NLST` request asks the server to send the contents of the Current Working Directory over the data connection already established by the client.

Back Close

## Connecting to new Server: `PASV` and `PORT` Commands

A `PASV` request asks the server to accept a data connection on a new TCP port selected by the server. `PASV` parameters are prohibited.

A `PORT` request asks the server to use a different mechanism of creating a data connection: the server makes a TCP connection to the client.

Back Close

## Retrieving Files: `RETR` and `REST` Commands

- A `RETR` request asks the server to send the contents of a file over the data connection already established by the client.
- **The `REST N` command — *Restart download***
  The server keeps track of a start position for the client.
  - The start position is a nonnegative integer (`N`).
  - At the beginning of the FTP connection, the start position is clearly 0.
  - Most Modern FTP clients can use this feature.
  - Useful for restarting partial downloads

Back Close

## Uploading/Storing files

- A `STOR` request asks the server to read the contents of a file from the data connection already established by the client.
- `APPE` is just like `STOR` except that, if the file already exists, the server appends the client's data to the file.
- `STOU` is just like `STOR` except that it asks the server to create a file under a new pathname selected by the server.

Back Close

## Directory Commands and Deleting Files

- A `MKD pathname` request asks the server to create a new directory.
  The `MKD` parameter `pathname` specifies the directory name.
- An `RMD pathname` request asks the server to remove a directory.
- A `DELE filename` request asks the server to remove a regular file.
- A `RNFR filename1` request asks the server to begin renaming a file.
- A `RNTO filename2` request asks the server to finish renaming a file. must immediately follow `RNFR filename1`.
  - Together the rename `filename1` to `filename2`

## Terminating the FTP Session: The `QUIT` command

A `QUIT` request asks the server to close the connection:

```
>     QUIT
<      221 Bye.
```

## The Example FTP Session with Protocols

```
ftp> debug
Debugging on (debug=1).
ftp> open ftp.cs.cf.ac.uk
Connected to thrall.cs.cf.ac.uk.
220-*************************************************************************
220- Cardiff Computer Science campus ftp access. Access is available
220- here as anonymous, by ftp group or by username/password.
220-
220- The programs and data held on this system are the property of the
220- Department of Computer Science in the University of Wales, Cardiff.
220- They are lawfully available to authorised Departmental users only.
220- Access to any data or program must be authorised by the Department
220- of Computer Science.
220-
220- It is a criminal offence to secure unauthorised access to any programs
220- or data on this computer system or to make any unauthorised
220- modification to its contents.
220-
220- Offenders are liable to criminal prosecution.  If you are not an
220- authorised user do not log in.
220-*************************************************************************
220-
220-Cardiff University.  Department of Computer Science.
220-This is the WUSL ftp daemon.  Please report problems to
220-Robert.Evans@cs.cf.ac.uk.
220-
220 thrall.cs.cf.ac.uk FTP server (Version wu-2.6.1(1) Mon Sep 18 12:45:30 BST 2000) ready.
Name (ftp.cs.cf.ac.uk:dave): year1
---> USER year1
331 Password required for year1.
Password:
---> PASS year1ic
230-
230-Welcome to the guest ftp server for Year 1 Internet Computing
```

```
230-in the Department of Computer Science at the University of Wales, Cardiff.
230-
230-Please note that all commands and transfers from this ftp account
230-are logged and kept in an audit file.
230-
230-
230 User year1 logged in.  Access restrictions apply.
ftp> dir
---> PORT 131,251,42,151,155,230
200 PORT command successful.
---> LIST
150 Opening ASCII mode data connection for /bin/ls.
total 32
drwxrwxrwx   2 y1ftp       2048 Nov  8  1999 ex_gif
drwxrwxrwx   2 y1ftp       2048 Nov  8  1999 ex_hqx
drwxrwxrwx   2 y1ftp       2048 Nov  8  1999 ex_text
drwxrwxrwx   2 y1ftp       2048 Nov  8  1999 ex_uu
drwxrwxrwx   2 y1ftp       2048 Nov  8  1999 ex_zip
drwxr-xr-x   2 y1ftp        512 Oct 18  1999 exercise
drwxrwxr-x   2 gueftp      2048 Nov  5  1999 incoming
drwx--x--x   2 staff       1024 Nov 11  1999 marker
drwxrwxr-x   2 gueftp      2048 Nov 10  1999 test
226 Transfer complete.
489 bytes received in 0.0064 seconds (75.12 Kbytes/s)
ftp> cd exercise
---> CWD exercise
250 CWD command successful.
ftp> dir
---> PORT 131,251,42,151,155,231
200 PORT command successful.
---> LIST
150 Opening ASCII mode data connection for /bin/ls.
total 156
-rw-rw-r--   1 staff      25943 Dec  8  1997 ex.gif
-rw-rw-r--   1 staff      53104 Oct 18  1999 ex.txt
226 Transfer complete.
```

```
117 bytes received in 0.0022 seconds (52.60 Kbytes/s)
ftp> bin
---> TYPE I
200 Type set to I.
ftp> get ex.gif
---> PORT 131,251,42,151,155,232
200 PORT command successful.
---> RETR ex.gif
150 Opening BINARY mode data connection for ex.gif (25943 bytes).
226 Transfer complete.
local: ex.gif remote: ex.gif
25943 bytes received in 0.092 seconds (275.08 Kbytes/s)
ftp> prompt
Interactive mode off.
ftp> prompt
Interactive mode on.
ftp> mget *.*
---> PORT 131,251,42,151,155,233
---> TYPE A
---> NLST *.*
---> TYPE I
mget ex.gif? y
---> PORT 131,251,42,151,155,234
200 PORT command successful.
---> RETR ex.gif
150 Opening BINARY mode data connection for ex.gif (25943 bytes).
226 Transfer complete.
local: ex.gif remote: ex.gif
25943 bytes received in 0.1 seconds (252.73 Kbytes/s)
mget ex.txt? y
---> PORT 131,251,42,151,155,235
200 PORT command successful.
---> RETR ex.txt
150 Opening BINARY mode data connection for ex.txt (53104 bytes).
226 Transfer complete.
local: ex.txt remote: ex.txt
```

```
53104 bytes received in 0.17 seconds (297.91 Kbytes/s)
ftp> close
---> QUIT
221-You have transferred 104990 bytes in 3 files.
221-Total traffic for this session was 107294 bytes in 6 transfers.
221-Thank you for using the FTP service on thrall.cs.cf.ac.uk.
221 Goodbye.
ftp>
```

Make sure that you can pick out the different `ftp` PROTOCOL commands and responses and see how they relate to the ftp client commands in this output.

Notice that the `ftp` PROTOCOL commands and responses are only displayed when the debugging feature is turned on.

## Secure FTP (SFTP)

### Why Do I need SFTP

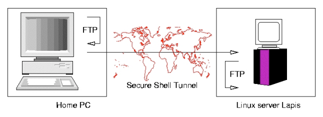
- If you wish to access our School's Computers from outside of School
  - If you upload files to your account
  - If wish to download files from your account to home computer
- You may wish to protect your file transactions to other servers too (if they support SFTP)
- **If you wish to connect this way make sure you have an FTP client that supports SFTP**
  `sftp` on Mac OS X/UNIX command line, Fetch does not support SFTP yet.
  There are plenty Freely available on the Web for all platforms.

For more information of SFTP on Schools computers see Web or PDF files online.

## How does SFTP Work?

Very Simply:

- Recall FTP uses TELNET for command connection
- As we will see in next section, Secure Shell (SSH) is an encrypted version of TELNET
- SFTP essentially creates create an SSH tunnel from your workstation to the server such that:
  - the tunnel's entrance listens for FTP connections on your own workstation
  - the tunnel encrypts the FTP traffic and sends it to server
  - the tunnel's exit decrypts the traffic and connects to an FTP daemon on server



## Telnet

### What is Telnet?

- The Telnet protocol is often thought of as simply providing a facility for remote logins to computer via the Internet.
- A simple terminal using the local telnet program (known as the client program)
- A Protocol (and a program) that lets you use the power of the Internet to connect you to databases, library catalogs, and other information resources around the world.
- Used in other protocols
  - **FTP Control Connection uses Telnet**

## Basic Uses of Telnet

- Telnet lets you connect directly to another computer on the Internet and run programs on the computer (provided you have access permission
  - Username and
  - Password
- Theoretically you have a lot of power at your fingertips.
  - Your dumb terminal or
  - very old PC

  Can (potentially) run programs on the worlds most powerful computers.

## Problems Using of Telnet?

- Life is not that simple though:
  - Telnet is old. It is text based only.
  - You have to know how to run programs at the other end.
    - ∗ Different Computer System — E.g. Unix
    - ∗ Different Commands, Directory Structures
    - ∗ Different Language — Programming and "spoken" — E.g. Unix Shell Scripts

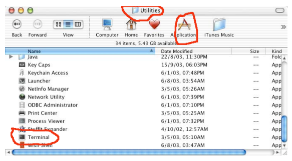
## Using Mac Terminal

The preferred method of TELNET for this course is to use **Mac Terminal Client** on the Macintosh Computers.

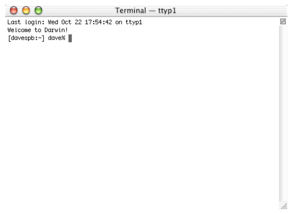
The application is found the in **Applications/Utilities** sub folder.

It essentially fires up a simple command line terminal window.

As far as a user is concerned you do not see any of TELNET protocol communication.



| Name | Date Modified | Size | Kind |
|---|---|---|---|
| Java | 22/8/03, 11:30PM | -- | Fold |
| Key Caps | 15/9/03, 06:03PM | -- | App |
| Keychain Access | 6/1/03, 07:48PM | -- | App |
| Launcher | 6/8/03, 03:54AM | -- | App |
| NetInfo Manager | 3/5/03, 05:26AM | -- | App |
| Network Utility | 6/1/03, 07:39PM | -- | App |
| ODBC Administrator | 6/1/03, 07:10PM | -- | App |
| Print Center | 3/5/03, 05:25AM | -- | App |
| Process Viewer | 6/1/03, 07:32PM | -- | App |
| StuffIt Expander | 4/10/02, 12:57AM | -- | App |
| Terminal | 3/5/03, 05:10AM | -- | App |
| Wish Shell | 6/8/03, 03:47AM | -- | App |



```
Last login: Wed Oct 22 17:54:42 on ttyp1
Welcome to Darwin!
[davespb:~] dave%
```

## Shells and Secure Shells

You may wish to run applications on remote machines from the command line

- All dialogs with remote machine use TELNET protocol.

For local remote connections to School Computers you can use

`rlogin` **localhost** — to remote login to a given localhost

For External remote connections to Dept. Computers you can use

`ssh` **host** — to secure shell to a given host. Secure shell uses encryption of all TELNET transactions.

For more information of TELNET/Secure Shell on Schools computers see Web or PDF files online.

## How Telnet Works:The Network Virtual Terminal

- Communication via TCP/IP protocols
- communication based on a set of facilities known as a Network Virtual Terminal (NVT).
- Telnet client program:
  - responsible for mapping incoming NVT codes to the actual codes needed for user's display device
  - responsible for mapping user generated keyboard sequences into NVT sequences.
- NVT uses 7 bit codes for characters (ASCII Codes),
- The display device, referred to as a printer, is only
- NVT ASCII is used by many other Internet protocols.

# The World Wide Web

### What is the World Wide Web?

One simple definition of the WWW is:

**The WWW is a Hypertext Information System**

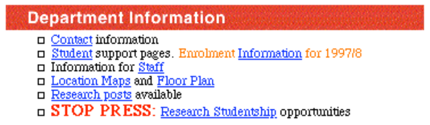
## What is Hypertext?

Hypertext is a method of presenting information where selected words in the text can be **expanded** at any time to provide other information about the word

*i.e.* these words are **links** to other documents which may be **text**:

- files, pictures, audio, video, anything really.

Links usually identified by highlighted (underlined) and coloured text



**Department Information**

- Contact information
- Student support pages. Enrolment Information for 1997/8
- Information for Staff
- Location Maps and Floor Plan
- Research posts available
- **STOP PRESS:** Research Studentship opportunities

Figure 10: Hypertext Links (marked in blue and underline)

## Hypertext browsing

- Non-linear structure (not a book)
- You read what you want next
- Click on Hypertext links to navigate the WWW.

## Hypermedia browsing

*Links* to other documents may not just be text can may be buttons, areas of a pictures, anything Flash.

## Hypertext Markup Language (HTML)

**HTML** (Hypertext Markup Language) is the language used to create WWW pages and associated links (*e.g.* URLs which we have already met).

## Features of the WWW

Other features of the WWW include:

**Graphical** — by definition text, graphics and other media can coexist on Web page.

**Easy to use** — hypertext and good WW browsers are intuitive tools to use.

**Cross-platform** — One big advantage is that WWW (html pages etc) can run on almost any computer.

**Distributed** — Information and resources shared globally.

**Dynamic** — Information on the WWW can be constantly updated (unlike a book or CD-ROM). Live information can assimilated.

**Interactive** — through HTML forms and other tools (*e.g. Flash, PHP, Perl, Java applets*) the WWW can be made interactive.

## Web Browsers

A Web browser is a special application that reads Web pages.

Common WWW browsers include/have included:

**Apple Safari** — Most popular Mac Browser now – very fast page rendering

**Internet Explorer** — Microsoft's Dominant Browser - but no longer developed for Macs

**Opera** — Fast page rendering, neat features, cross platform

**Mozilla** — Open source web browser code

**Netscape Navigator** — One the most popular early browsers (Now Dead?).

**Mosaic** — First graphical browser.

**Lynx** — Text only browser.

## Web Servers

To view and browse information on the web all you need is a **browser**.

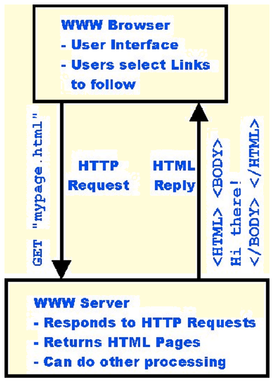
To publish and deliver information of the web you need a Web **server**.

**A web server is a program that runs on web sites and is responsible for replying to a web browser request for files**.

## Client (Browser) Server interaction

When a browser requests a Web page:

- the browser makes a connection to a server using **http protocol (More Later)**.
- the server then
  - accepts the connection,
  - sends the files contents back to browser, and
  - closes the connection.
- the browser then formats the (html) information received.


WWW Browser
- User Interface
- Users select Links to follow
GET "mypage.html"
HTTP Request
HTML Reply
<HTML> <BODY>
Hi there!
</BODY> </HTML>
WWW Server
- Responds to HTTP Requests
- Returns HTML Pages
- Can do other processing