Tim Berners-Lee, CERN

WWW Server Guide

A Guide to WWW Servers

1. W3 Server Software

A W3 server, like the ftp daemon , is a program which responds to an incoming tcp connection and provides a service to the caller. There are many varieties of W3 server software to serve different forms of data.

1.1 Basic W3 servers

CERN server The basic W3 daemon program serves files already in hypertext or plain text. This daemon then is used as a basis for many other types of server and gateways .

NCSA server A server for files, written in C, public domain. Runs on top of a gopher-style database just like "gopherd".

Perl server from Marc VanHeyningen at Indiana University. Wriiten in perl .

Plexus Tony Sander's server originally based on Marc VH's, but incorporating lots more stuff, including an Archie gateway, etc etc

MacHTTP Server for the Macintosh.

REXX for VM A server consisting of a amall C program which passes control to a server written in REXX.

Whatever server you are running, you will probably be interested in:

- Tools for information providers
- Syle Guide for Online Hypertext

1.2 Making a new server

This daemon is often used as a basis for a more specific server for a given application. A server which allows a world of data to be seen as part of the W3 universe is known as a gateway. (Most servers could therefore be regarded as gateways, but the term implies some conversion or mapping between dissimilar worlds) . For short tutorials with examples, see:

- Writing a server in C
- Writing a server as a script

It is a good idea to pick the basic daemon or one of the servers in the list as a starting point when making a new server.

1.3 Other servers and Gateways

These are servers which provide data extracted from other systems. they are built using code from the basic daemon, or scripts. See

- List of Gateways available .

Tim BL

1.4 About documents generated from hypertext

Paper manuals generated from hypertext are made for convenience, for example for reading when one has no computer to turn to. We have tried to make the hypertext into fairly conventional paper documents, but they may seem a little strange in some ways.

All the links have been removed. Therefore, it is worth looking at the table of contents to see what there is in the manual. Something which is not explained in place may be explained in detail elsewhere.

We have tried to keep related matter together, but sometimes necessarily you might have to check the table of contents to find it.

Please remember that these are for the most part "living documents". That is, they are constantly changing to reflect current knowledge. If you see a statement such as "Product xxx does not support this feature", remember that it was the case when the document was generated, and may not be the same now. So if in doubt, check the online version. Of course, the living document may be out of date too, in which case it is helpful to mail its author. Tim BL

2. WWW Server user guide

The basic WWW server allows files and directories in a file system to be server to the world as menu trees, multimedia, and/or hypertext.
The http daemon, httpd , is a general server program which runs a w3 protocol, " HTTP ". This is a TCP/IP based protocol running by convention on port 80.

2.1 In this guide

Distribution	How to get the code.
Compilation	The daemon is compiled in the same way as the library and line mode browser –– see WWW distributed code .
Installation	How to install a server under unix internet daemon
Options	Command line options at run time
Rule File	The format of a rule file. By default, /etc/httpd.conf
Etiquette	Conventions you should follow to make life smoother
Debugging	If it doesn't seem to work
Known bugs	and improvements desired
Change History	change list of improvements made and bug fixes.

2.2 Related documents

HTML specification	A description of the hypertext markup language used for representing menus, etc
HTTP specification	A desription of the protocol used by the server.

2.3 Status of basic WWW server

A basic fast information server for files.

Author	TBL
Status:	Version 2 available by anonymous FTP, with no index search but file access, name mapping and security filter, ability to act as gateway for anything in the WWW library's repertoire, including WAIS.
Plans:	A version which will allow general unix users to set up an index search daemon. As index search tools are not generally available, we may use the NeXT digital Librarian or WAIS as an basis.
Platforms	Unix, VMS, VM/CMS (VM/XA).
Next Milestone:	Run shell scripts to implement virtual documents and searches.
More information:	User guide , Bug list , Internals , Change history .
Wider scope:	W3 servers , Other WWW software

Features include

- Installation under inetd or run stand-alone
- Can be run stand-alone by normal user
- Automatically generates hypertext view of directory tree
- Uses "README" files to document directory listings
- Handles multimple formats of same file, selects format apropriate for client capabilities
- Document name to filename mapping for longer-lived document names
- Can act as gateway for WAIS, news, etc if needed

2.4 WorldWideWeb CERN-distributed code

See the CERN copyright . This is the README file which you get when you unwrap one of our tar files. These files contain information about hypertext, hypertext systems, and the WorldWideWeb project. If you have taken this with a .tar file, you will have only a subset of the files.

THIS FILE IS A VERY ABRIDGED VERSION OF THE INFORMATION AVAILABLE ON THE WEB. IF IN DOUBT, READ THE WEB DIRECTLY. If you have not got ANY browser installed yet, do this by telnet to info.cern.ch (no username or password).

Files from info.cern.ch are also mirrored on ftp.ripe.net.

2.4.1 Archive Directory structure

Under /pub/www , besides this README file, you'll find bin , src and doc directories. The main archives are as follows:

bin/xxx/bbbb Executable binaries of program bbbb for system xxx. Check what's there before you bother compiling. (Note HP700/8800 series is "snake")

bin/next/WorldWideWeb_v.vv.tar.Z The Hypertext Browser/editor for the NeXT –– binary.

src/WWWLibrary_v.vv.tar.Z The W3 Library. All source, and Makefiles for selected systems.

src/WWWLineMode_v.vv.tar.Z The Line mode browser - all source, and Makefiles for selected systems. Requires the Library .

src/WWWDaemon_v.vv.tar.Z The HTTP daemon, and WWW-WAIS gateway programs. Source. Requires the Library.

src/WWWMailRobot_v.vv.tar.Z The Mail Robot.

doc/WWWBook.tar.Z A snapshot of our internal documentation - we prefer you to access this on line –– see warnings below.

2.4.2 Basic WWW software installation from source

This applies to the line mode client and the server. Below, $prod means LineMode or Daemon depending on which you are building.

Generated Directory structure

The tar files are all designed to be unwrapped in the same (this) directory. They create different parts of a common directory tree under that directory. There may be some duplication. They also generate a few files in this directory: README.*, Copyright.*, and some installation instructions (.txt).

The directory structure is, for product $prod and machine $WWW_MACH

WWW/$prod/Implementation Source files for a given product

WWW/$prod/Implementation/CommonMakefile The machine-independent parts of the Makefile for this product

WWW/$prod/$WWW_MACH/ Area for compiling for a given system

WWW/All/$WWW_MACH/Makefile.include The machine-dependent parts of the makefile for any product

WWW/All/Implementation/Makefile.product A makefile which includes both parts above and so can be used from any product, any machine.

Compilation on already supported platforms

You must get the WWWLibrary tar file as well as the products you want and unwrap them all from the same directory.

You must define the environmant variable WWW_MACH to be the architecure of your machine (sun4, decstation, rs6000, sgi, snake, etc)

In directory WWW, type BUILD.

Compilation on new platforms

If your machine is not on the list:

- Make up a new subdirectory of that name under WWW/$prod and WWW/All, copying the contents of a basically similar architecture's directory.
- Check the WWW/All/$WWW_MACH/Makefile.include for suitable directory and flag definitions.
- Check the file tcp.h for the system-specific include file coordinates, etc.
- Send any changes you have to make back to www-request@info.cern.ch for inclusion into future releases.
- Once you have this set up, type BUILD.

2.4.3 NeXTStep Browser/Editor

The browser for the NeXT is those files contained in the application directory WWW/Next/Implementation/WorldWideWeb.app and is compiled. When you install the app, you may want to configure the default page, WorldWideWeb.app/default.html. These must point to some useful information! You should keep it up to date with pointers to info on your site and elsewhere. If you use the CERN home page note there is a link at the bottom to the master copy on our server. You should set up the address of your local news server with

```
dwrite WorldWideWeb NewsHost  news
```

replacing the last word with the actual address of your news host. See Installation instructions .

2.4.4 Line Mode browser

Binaries of this for some systems are available in /pub/www/bin/ . The binaries can be picked up, set executable, and run immediately.

If there is no binary, see "Installation from source" above.

(See Installation notes). Do the same thing (in the same directory) to the WWWLibrary_v.cc.tar.Z file to get the common library.

You will have an ASCII printable manual in the file WWW/LineMode/Defaults/line-mode-guide.txt which you can print out at this stage. This is a frozen copy of some of the online documentation.

Whe you install the browser, you may configure a default page. This is /usr/local/lib/WWW/default.html for the line mode browser. This must point to some useful information! You should keep it up to date with pointers to info on your site and elsewhere. If you use the CERN home page note there is a link at the bottom to the master copy on our server.

Some basic documentation on the browser is delivered with the home page in the directory WWW/LineMode/Defaults. A separate tar file of that directory (WWWLineModeDefaults.tar.Z) is available if you just want to update that.

The rest of the documentation is in hypertext, and so wil be readable most easily with a browser. We suggest that after installing the browser, you browse through the basic documentation so that you are aware of the options and customisation possibilities for example.

2.4.5 Server

The server can be run very simply under the internet daemon, to export a file directory tree as a browsable hypertext tree. Binaries are avilable for some platofrms, otherwise follow instructions above for compiling and then go on to " Installing the basic W3 server ".

2.4.6 XMosaic

XMosaic is an X11/Motif W3 browser.

The sources and binaries are distributed separately from FTP.NCSA.UIUC.EDU , in /Web/xmosaic . Binaries are available for some platforms. If you have to build from source, check the README in the distribution.

The binaries can be picked up, uncompressed, set "executable" and run immediately.

2.4.7 Viola browser for X11

Viola is an X11 application for reading global hypertext. If a binary is available from your machine, in /pub/www/bin/.../viola*, then take that and also the Viola "apps" tar file which contains the scripts you will need.

To generate this from source, you will need both the W3 library and the Viola source files. There is an Imakefile with the viola source directory. You will need to generate the XPA and XPM libraries and the W3 library befere you make viola itself.

2.4.8 Documentation

In the /pub/www/doc directory are a number articles, preprints and guides on the web.

See the online WWW bibliography for a list of these and other articles, books, etc. and also the list of WWW Manuals available in text and postscript form.

2.4.9 General

Your comments will of course be most appreciated, on code, or information on the web which is out of date or misleading. If you write your own hypertext and make it available by anonymous ftp or using a server, tell us and we'll put some pointers to it in ours. Thus spreads the web... Tim Berners-Lee

WorldWideWeb project

CERN, 1211 Geneva 23, Switzerland

Tel: +41 22 767 3755; Fax: +41 22 767 7155; email: timbl@info.cern.ch

2.5 Installing the basic WWW server

IIf using unix, for the simplest method see Installation under the Internet Daemon.

There are special instructions if you are installing under VMS .

The usual way to install a daemon is to either run it from the bootstrap command file (for example /etc/rc) so that it runs continuously, or to set up the internet daemon (inetd) to run it when a call comes in.

See a csh script which does everything below for unix BSD systems but which you should modify with care for your own system.

Note: With version 2.0 on, a rule file is no longer essential if you want to just export a directory tree.

The installation normally requires superuser status, but it is poosible to run httpd from a terminal session as a normal user.

2.5.1 Log file

If a log file is required, make sure that the user name under which the daemon is run has the right to write the file
Tim BL

2.5.2 Priviliged ports

The TCP/IP port numbers below 1024 are special in that normal users are not allowed to run servers on them. This is a security feaure, in that if you connect to a service on one of these ports you are fairly sure that you have the real thing, and not a fake which some hacker has put up for you.

The normal port number for W3 servers is port 80, which is such a port. (This number is assigned by the Internet Assigned Numbers Authority, IANA).

When you run a server as a test from a non-priviliged account, you will normally test it on other ports, such as 2784 or 5000 typically.

Under unix

The inet daemon (running as root) can listen for incomming conections on port 80 and pass them down to a process with a safer uid for the server itself. Of course, you have to be root to set up the inet daemon.

Under VMS

Under UCX, The process running as a server needs BYPASS privilege to listen to ports below 1024. This might mean you have to install the server. With other TCP/IP packages, privilege of some sort is similarly required.

Tim BL

2.5.3 Installing a daemon under inetd

This is how to to set up the internet daemon (inetd) to run your HTTPD server whenever a request comes in. (These steps are the same for any daemon under unix: you will probably find a similar thing has been done for the FTP daemon, ftpd, for example.)

Step1

Copy the daemon program or shell script (httpd in this example) into a suitable directory such as /usr/etc. Protect it from anyone writing to it except root.

Step2

Put "http" in the /etc/services file, or use the name of a specific service of your own if you want to use have a special port number.

(Exceptions: on a NeXT, see using the NetInfomanager . On any machine running NIS (yellow pages), see specicial instructions).

For example,

```
http 80/tcp # WorldWideWeb server
```

Step3

Put a line in the internet daemon configuration file, /etc/inetd.conf. For example,

```
http stream tcp nowait nobody /usr/etc/httpd httpd /Public
```

(That was all one line.) Here "http" is used as a link between the services file and inetd.conf: it could have been any identifier. "nobody" is the user name under which you want the daemon to run, which determines what privileges it has for example to read data. "/usr/etc/httpd" is the actual file name of the server. The rest of the line is the arguments passed to httpd: arg0 is the program name, "httpd", by convention. Here the argument "/Public" is the directory tree to be exported. This is in fact the default if no directory is given. See command line syntax for more details.

Note: The inted.conf format varies from system to system. If in doubt, copy the format of other lines in your existing inted.conf. For example, under ultrix there is no user name field –– everything runs as root.

Note: there seem to be, on the NeXT at least, a limit of 4 arguments passed across by inetd!

Step 4

When you have updated inted.conf, find out which process is running inetd, and send it a "HUP" signal. On BSD unix (For system V, use ps-el for ps aux) this looks like:

```
> ps aux | grep inetd | grep -v grep
root        85   0.0  0.9 1.24M  304K ?  S     0:01 /usr/etc/inetd
> kill -HUP 85
>
```

Test it

Test the server with the line mode browser by giving its address explicitly:

```
www http://myhost.dom.ain/welcome.html
```

This assumes that you have a file "welcome.html" in your exported directory. If it doesn't work, you have probably missed something. See notes on debugging . Tim BL

2.5.4 Using NIS (Yellow pages)

If your machine is running Sun's "Network Information Service", originally know as 'yellow pages", read this.

You must:

- First make an addition to the /etc/services file just as for a normal unix system.
- Then, change directory to /var/yp and type "make".

This will load the /etc/services file info the yellow pages information system.
Some peopl ehave found that they needed to reboot he system afterward for the change to take effect. Tim BL

2.5.5 Adding a service on the NeXT

The NeXT uses the the "netinfo" database instead of the /etc/services file. This is managed with the /NextAdmin/NetInforManager application. Here's how to add the service "www":

- Start the NetInfomanager by double-clicking on its icon.
- If you are operating in a cluster, open either your local domain (/hostname) or if you have authority, the whole cluster domain (/). If you're not in a cluster, just use the domain you are presented with.
- Select "services" from the browser tree.

- Select "ftp" from the list of services
- Select "dupliacte" from the edit menu.
- Select "copy of ftp" and double-click on its icon to get theproperty editor.
- Click on "name" and then on the value "copy of ftp". Change this to "www" by typing "www" in the window at the botton, and hitting return.
- Click on "port", and then on the value "21". Change it to "80".
- Use "Directory:Save" menu (Command/s) to save the result. You will have to give a root password or netinfo manager password.

Tim BL

2.6 The Rule File

The rule file (configuration file) defines how the WWW software will translate a request into a document name. For a server, it allows one to provide an extra level of name mapping above that given by links in the file system. It allows, for example, out of date names to mapped onto their more recent counterparts.

For the client, it allows access to certain servers to be remapped for example caching servers, or to local copies of the same information.

The rule file also allows access to be restricted. This is essential, to prevent, for example, unauthorized access to your password file.

By default, the rule file /etc/httpd.conf is loaded, unless specified otherwise with the -R or -r options .

See also: example rule files , Old format for software before 2.0 , Setting up gateways , Firewall gateways .

2.6.1 Mapping and filtering

Each line consists of an operation code and one or two parameters, referred to as the template and the result. Anything on a line after and including a hash sign (#) is ignored, as are empty lines.

The server uses the top rule first, then EACH SUCCESSIVE RULE unless told otherwise by PASS or FAIL. The operation codes are as follows

map template result	If the address matches the template, use the result string from now on for future rules.
pass template	If the address maches the template, use it as it is, porocessing no further rules.
pass template result	If the string matches the template, use the result string as it is, processing no futher rules.
fail template	If the address matches the template, prohibit access, processing no futher rules.

The template string may contain at most one wildcard asterisk ("*"). The result string may have one wildcard only if the template has one.

When matching,

- Rules are scanned from the top of the file to the bottom.
- If a request matches a "map" template exactly, the result string is used instead of the original string and applied to successive rules.
- If the request maches a "map" template with wildcard, then the text of the request which matches the wildcard is inserted in place of the wildcard in the result string to form the translated request. If the result string has no wildcard, it is used as it is.
- When a map substitution takes place, the rule scan continues with the next rule using the new string in place of the request. This is not the case if a pass ro fail is matched: they terminate the rule scan.

2.6.2 Suffix definitions

As well as any mapping lines in the rule file, the rule file may be used to define the data types of files with particular suffixes. The syntax

```
suffix  <suffix>  <representation> <encoding> [ <quality> ]
```

for example:

```
suffix  .pc     text/plain          7bit 1.0
suffix *.*     application/binary  binary 0.1
suffix * text/plain     7bit
```

The parameters are as follows:

<**suffix**> The last part of the filename. There are two special cases. "*.*" matches to all files which have not been matched by any explicit suffixes but do contain a dot. "*" by itself matches to any file which does not match any other suffix.

<**representation**> A MIME "content-type" style description of the repreentation in fact in use in the file. See the HTTP spec. This need not be a real MIME type –– it will only be used if it matches a type given by a client.

<**encoding**> A MIME content transfer encoding type. Much more limited in variety than representations, basically whether the file is ASCII (7bit or 8bit) or binary. A few other encodings are allowed, and maybe extension to compression.

<**quality**> Optional. A floating point number between 0.0 and 1.0 which determines the relative merits of files xxx.* which differ in their suffix only, when a link to xxx.multi is being resolved. Defaults to 1.0.

2.6.3 Presentation definitions

In the rule file for a client, you can define the presentation of a given data type. The syntax is

```
presentation   <representation>  <command-string>
```

where the parameters are

<**representation**> A MIME-style content type. You can use regulare MIME types, such as image/jpeg, or your own extensions which start with x-, such as image/x-tiff, application/x-my-app. See also above .

<**command string**> The command needed to display a temporary file of this type. A "%s" within this string will be replaces with the name of the temporary file. Note that is any file suffix has been specified as corresenponding to this representation, then the temporarty file will be give that (or the first if there is a choice) suitable suffix.

Tim BL

2.6.4 Rule file examples

A basic rule file for the http daemon might look like this (it looked different before version 2.0):

```
pass    /          file:/u/john/welcome.html
pass    /*         file:/u/john/public/*
fail   *
```

The first line maps the root document onto a specific document about the server, and accepts it. (see etiquette about the welcome page)

The second line maps all document names onto filenames in a particular directory and accepts them.

The third line disallows access to all other documents. (There won't be in any in this case because of the mapping, but its wise to put in for later).

Second example

```
map    /            /tnotes/welcome.html
map    /tnotes/*    file:/u/john/public/*
map    /seminars/*  file:/u/jane/seminars/*
pass   file:/u/john/public/*
pass   file:/u/jane/seminars/*.html
fail   *
```

The first line maps the root document onto a specific document about the server. Because it is "map and not "pass", it DOESN'T accept it but passes it on for futher mapping by lines futher down.

The second line maps all document names starting with /tnote/ onto filenames in a particular directory where john maintains the technical notes. If someone else takes over the technical notes, we can change this. Here we are starting to distinguish between document names and file names. This can be carried much further if necessary, but one level of mapping is enough to allow for changes of administration of different areas.

The third line separately maps the seminar information into Jane's directory.

The fourth and fifth line enable access to anything in John's "public" directory, and any .html file in Jane's "seminar" directory tree. Note here that the * maps to any sequence INCLUDING SLASHES so all files in any subdirectory of /u/jane/seminars will be enabled so long as they end in .html.

The bottom line will pick up for example any attempt to use the server to access non-html files in Jane's seminars directory.

Configuration file for a WAIS gateway

The httpd daemon can be used as a WAIS gateay if it has been compiled with the necessary options and linked with the freeWAIS software. A suitable configuration file is

```
map     /*    wais://*
pass wais://*
fail *
```

2.7 Server Command Line

The command line syntax for the basic www server allows a number of options and an optional directory argument.

```
httpd  [options] [directory]
```

The directory argument, if present, indicates the directory to be exported. (Version 2.0 and later only.) If not present, either a rule file is be used, to export combinations of directories, or else the default is to export the "/Public" directory tree.

2.7.1 Examples

```
httpd -p 80  -dyt /ftp/pub
```

This exports the entire /ftp/pub tree with browsable directories and README files included at the top of directory listings.

```
httpd
```

This comamnd in the inetd configuration file inetd.conf exports the /Public directory tree. This tree may contain soft links to other directory trees.

-dn Disable directory browsing. An attempt to access a directory will generate an error response.

-dy Enable direcory browsing. Directories are returned as hypertext documents. See browsing directories . This is the default.

-ds Enable directory browsing only for directories containing a file named ".www_browsable".

-dt For any browsable directory which contains a README file, include the text of the README file at the top of the document before the listing. This is the default.

-db As -dt but put the README at the bottom, after the listing. The -db and -dt options may be combined with -dy as -dyb, -dty etc.

-dr Disables the README inclusion feature .

-l file Log all calls to the given file. The file is appended to if it already exists.

-p port Specify the port number. If this option is not given, the daemon assumes that it has been run by inetd, and uses stdin and stdout as its communication channel . Note that port numbers under 1024 are privileged .

-v Verbose mode. Copious trace messages are written to the standard output stream. Mainly for debugging.

-r file Load a rule file . The rules are added after any rules already loaded. Inhibits the loading of the default rule file.

-R Do not use. Inhibit the loading of the default rule file. Warning: running without a rule file normally poses a security problem. It won't work in general as only the path part of a URL is input into the rule file, and a fully qualifiue URL (with file: in front for example) is required on output.

Tim BL

2.8 Debugging the daemon

Suppose you think you have installed a W3 server but it doesn't work. That is, you have followed the installation instructions and the test at the end fails. Here we assume you have used port 80. If you have a situation not handled by this problem-solving guide, please mail me.

Type

```
www http://myhost.domain:80/
```

What happens?

- "Cannot connect to information server" message, "Unable to access document" or some other generic-sounding error message

- An empty document is displayed
- A document containing the words "Document address invalid or access not authorised", or some "Error 500" message is displayed
- A document is displayed, but not what you wanted the server to give in response to that document name (/)

Tim BL

2.8.1 Document address invalid

You have accessed a W3 server and you get back a message "Document address invalid or access not authorized", or some other error message from the server.

The 1.x server does not (originally for security reasons) distringuish between a document which does not exist, and one to which you are not allowed access. However, most server are public servers which allow access to anyone, so if you are following a bona fide link, this could mean

- You have been passed a bad document address. If you are following a link, check with the author of the document which contained the link.
- The document has been moved. Check with the server administrator. You should be able to find out who runs the server by going to the welcome page (type "g /" with the line mode browser) and seeing a link to information about the maintainers.

If you are the server administrator, and you can't understand why the daemon refuses to deliver the file,

- Check the rule file if you have one. Think out way the document name will be mapped successively by each line, and what the result will be. Checking the trace below may help clarify this.
- Run the daemon with trace from a terminal session to get trace information

Tim BL

2.8.2 Can't connect to server

There is more information you can get. use the "verbose" option on the browser to find out what went wrong:

```
www -v http://myhost.domain:80/
```

What do you get? A load of trace messages. There are several cases.

- The browser can't look up the name of the host. If it can, it will display "Parsed address as" message. If not, try fixing your name server or /etc/hosts file, or quoting the IP number of the host in decimal notation (like 128.141.77.45) instead.
- The browser can get to the host but gets "Connection refused" status back .
- Your browser gets an error number but prints "error message not translated". This is because when it was compiled on your platform it didn't know what form the error message table took. Try the same thing form a unix platform for example.
- You get some network error like "network unreachable". Depending on whether the IP network is your responsibility or not, and your attitude to life, either fix it, try again in an hour's time, or complain to someone.

Tim BL

2.8.3 "Connection Refused"

The browser tries to connect to the daemon but gets this status in the trace.

This means that noone was listening on that port number. Check the port numbers match btween server and client. Make sure you specify the port number explicitly in the document address for www.

If you are running the daemon without the inet daemon, (with the -a option) then try running it from the terminal with -v as well. The trace for the server should say "socket, bind and listen all ok". If it does, and you still get "connection refused", then you must be talking to the wrong host (or, conceivably, different ethernet adapters on the same host)

If you are running with the inet daemon, then check both the services file (/etc/service) or database (yellow pages, netinfo) if your system uses it, and the /etc/inetd.conf file. Check the service name matches between these two.

Did you remember to kill -HUP the inet daemon when you changed the inted.conf file?

Try running the deamon from a shell window to see what happens better. Tim BL

2.8.4 You get an Empty document

The document sent back is empty, but there is no error message.

The inet daemon has started a process to run your server but it immediately failed. Possibilities include:

- The daemon may not be in the file specified, or may not be executable by the specified user (or, if a user id is not specified in your variety of inetd.conf, root)
- You have written your own daemon and it crashes.
- You are using ours and it crashes (mail us!)

Try running the daemon from a terminal window to see what happens. Tim BL

2.8.5 Bad output from the daemon

These are some ideas:

- Try running the server from the terminal .
- Check the HTML source the daemon produces with

```
www -source http://myost.domain:80/
```

- Try telnetting to the daemon and simulating the client:

```
> telnet myhost.domain 80
Connected to myhost.domain on port 80
Escape is ^[
GET /documentname
```

Tim BL

2.8.6 Telnetting to a server

Most implementations of telnet allow you to specify a port number. Under unix this is often just a second parameter, under VMS a /PORT option.

The HTTP protocol is a telnet protocol, so you can simulate it just by typing things in. This will help you to see exactly what a sending back, and it will check you that it really is the server not the browser which has a problem.

Here is an example. (You type "telnet..." and "GET ...").

```
> telnet myhost.domain 80
Connected to myhost.domain on port 80
Escape is ^[
GET /documentname
<PLAINTEXT>
Document name "/documentname" invalid.
```

2.8.7 Running under shell

You don't have to run the daemon under the inted if it doesn't work. You can run it from a shell session.

If the daemon is httpd, then run it from your terminal, with a different port number like 8000. You use the -p option .

```
httpd -p 8000
```

Note: You must be root (under VMS, have some privilege) to run with a port number below 1024.

If you select a port above 1024, then you can run as a normal user. This way, anyone can publish files on the net. Howeever, it isn't very reliable, as your server will not automatically come back up if the machine is rebooted. In the long term it is best to install it under "inetd".

You can't use a port number which has been used by a daemon process recently, so you may have to switch port number if you ˆC and restart the daemon. When it is running like this, you can read the trace messages and use a debugger on it if necessary. (See also: telnetting to the server)

Debugging using Trace

If you can't understand why a server refuses to give back a document, then run wiith the -v option to get trace. You will see the daemon setting up the rules for translating requests into local URLs, and you will see its attept to access the file (assuming you map requests onto files).

```
httpd -v -p 8000
```

Try to access the document from a client using another terminal window. Look at the trace printout. It will probably explain what is happening. If it includes specific messages below, follow them to detailed help.

- Can't find internet hostname ‘’

If you still can't figure out the problem, mail your local guru help desk or if desperate www-request@info.cern.ch ENCLOSING a copy of that trace.

Even simpler

For testing a daemon very simply, without using a client, you can make the terminal be the client. With httpd, or if the server is a shell script "myserver", try just running it with the terminal and typing GET /documentname into its input:

```
> httpd
GET /
```

Try it with the -v option if what comes back isn't a formatted document. Tim BL

2.9 The basic W3 server: Internals

This describes the generic hypertext daemon (server) program. The daemon is part of the WWW project. See also:

- User guide .
- Bugs and Features
- Other servers

The hypertext daemon, like the ftp daemon, is a program which responds to an incomming tcp connection and provides a service to the caller.

2.9.1 Sources

A compilation option (SELECT) controls whether more than one connection can be handled at a time. This is a function of whether the TCP/IP implementation beneath the application has a working "select()" routine. If it is not true, this implementation services one connection, then drops it before accepting another one. In neither case does the daemon concurrently serve two clients, nor does it fork off a process to do that.

The basic server loop is in the file HTDaemon.c . A separate module (for example HTRetrieve.c) contains the code to handle one request. Various specific versions of this may be written for different flavours of server. Also used are various modules of WWW common code. The httpd released from CERN uses almost the entire W3 library and can therefore access any object which a browser running on that machine can access, and return it as HTML or some other format. Tim BL

2.10 Bugs and Improvements needed

Improvements to be made in the HTTP daemon program are as follows. (Se also Features)

- Call shell scripts to perform searches on directory trees or documents.
- The HTRetrieve() routine ought to be able to pick up the user node and userid, etc...
- Ought to have chroot option. (wwwww July 93)

Tim BL

2.11 Daemon features: Update history

History list for the WWW daemon . (See also bugs). Many other changes to the daemon are in fact changes to the common code library.

2.11.1 2.06 7 June 93

- Bug fix: Load error 500 returned as proper HTTP status, not as simple document.
- WAIS gateway now caches source files again.
- Bug fix: Daemon used to try to display graphics file locally on the server when the client couldn't display them! Cause of much confusion :-)

2.11.2 2.05

- Big bug fix in local file directory handling .. didn't work in 2.04!

2.11.3 2.04 28 April 93

- With the properly compiled libwww library, this daemon will operate as a WAIS, news etc gaetway if so configured.
- WAIS gateway operation bug fix.

2.11.4 2.03-beta: unreleased

- Bug fix: operation with no rule file didn't work as expected.

2.11.5 2.02-beta: 17 March 93

- Misleading error trace removed.
- Compiled on HP, SGI, Sun, DEC, NeXT and binaries available
- Binary handling fixed in library.
- Reference to missing HTDirRead.h removed.
- Assumes that user can handle files of unknown format (application/binary).

2.11.6 2.00-alpha 15 Mar 93

- Simple command line –– with no parameters, exports the /Public directory.
- Multiformat handling –– see library changes for 2.0. Links to .multi filenames resolve to any file with same root, any recognised extension.

2.11.7 unrealeased 0.9b

- Bug fix: If a PASS or FAIL line in the configuration file acted on a single document id (ie no wildcard) then it crashed the daemon. (HTRules.c, 17-Jun-92, TBL).

2.11.8 Sept 1991 v0.3

- Bug fix: Plain text files were returned to be parsed as SGML, causing them to come out as garbage. (Mike Sendall)

2.11.9 August 1991 v 0.2

- -R option now suppresses default rule file.
- Rule file format changed completely. Now allows authorisation of specific paths only.

2.11.10 June 1991 version 0.1

- -r and -R options for rules
- Default address is now for Inet daemon working. (29 June)
- -l option to log to a file.
- -a option for address other than default

Tim BL

3. A Shell Server for HTTP

The HTTP protocol is very simple. The following is an example of a server program written in sh:

```
#! /bin/sh
read get docid
echo "<TITLE>$docid</TITLE>"
echo Here is the data
```

The docid may have a trailing carriage return to be stripped off on some systems. You can modify that script to produce the data you actually want. The HTML syntax for marked-up text is fairly simple, but if you want just to send plain text, then just send the .PLAINTEXT.tag first:

```
#! /bin/sh
read get docid
sed -f txt2html.sed $docid
```

or in csh

```
#! /bin/csh
request = ( `echo $<`)
if ($#request <2) exit
sed -f txt2html.sed $request[2]
```

When you have written your script, set the execute bit and then configure the inet daemon to run it . A few more examples:

- A sh script to generate a menu for files in a directory
- An awk script to generate menu from a list of files .
- A perl script for all kinds of stuff on the ASIS server
- The shell script of the Hytelnet gateway

If you know the perl language, then that is a powerful (if otherwise incomprehensible) language with which to hack together a server.
See also a case study of mapping a database onto the web .
All contributions to these examples welcome! Tim BL

3.1 Making a server

Here is a run-through of what is needed to make a www server , with examples from a suggested server for the HEPDATA base of Mike Whalley . See also etiquette .

Basically, to make the data available, you make a server which is a modified version of your program. When a user follows a link to HEPDATA (or runs a command to jump straight there), the client program opens a connection to a server program on a VM machine (say, but could be VMS or unix). The server in turn runs your program.

Let me just describe the essence of the changes needed so that you can get an idea of how much effort would be involved.

The first thing you do is to make up an arbitrary naming method for anything which HEPDATA can display. In this I include the welcome page, any menu, any article, any help text. Typically one invents a hierarchical naming scheme, like

```
/HEPDATA The first "welcome" menu
/HEPDATA/HELP The top-level help

/HEPDATA/HELP/REAC The help on the reaction database.

/HEPDATA/REAC The reaction database itself

/HEPDATA/REAC?P+PBAR list of reactions involving p and pbar (?)

/HEPDATA/DATA/RD125V687 Some article (say).
```

You do this because, whereas an interactive user follows a path through the program, the W3 user calls the program once for each thing. There is no "state" information. This allows one to make a hypertext link to any part of the scheme and jump back in again later. For example, one might want to quote an article, or the reaction database, or a particular list of reactions.

Now all you do is modify the program so that, given a name above, it will

return the required document. This means basically turning it from a sequence the user goes through into a set of conditionals to isolate each of the individual cases above. Apart from that, the data retrieval code is unchanged apart from the output formatting. Many of the options in fact mean mapping the name onto a fixed

file's name its the searches which have to activate real code.

The hypertext trick you need to use in the menus. Where an option is normally output to the screen, you have to tell the client what to ask for is the user selects that option. For example, in the main menu /HEPDATA you have an option which gives the help. You would represnt this "anchor" as

```
<A NAME=4 HREF=/HEPDATA/HELP> Help </A>
```

"Help" is all that is displayed, with some indication that it is an option. If the user choses (clicks a mouse on, choses by number depending on which client he has) then the client asks the server for /HEPDATA/HELP. ("A" is for "anchor", "HREF" is for "hypertext reference")

For the index searches, it's as simple. When the server sends the text called /HEPDATA/REAC it also sends a special tag . This tells the client to enable a FIND command, or find panel etc (depending on the client). You don't have to do any human interface work. The client automatically comes back with a search coded up in the form /HEPDATA/REAC?P+PBAR etc. Your server in turn returns a menu (say) with pointers to the data which has been found.

You can also put some formatting tags (like headings) which will make the data look really nice on a window system.

Tim BL

4. W3 and HTMLTools

These tools, part of the available WWW software , are managements of W3 servers, generation of hypertext, etc.

4.1 Generating HTML

List of filters and converters between various formats and HTML, collected by Richard Brandwein.

Mail Archive to HTML Make that mail archive available on the web. Markus.Stumpf@Informatik.TU-Muenchen.DE

Framemaker interface There are some tar files on the anonymous FTP archive on file://info.cern.ch/www/src which allow FRAMEmaker to be used as a W3 tool. Dan Conolly, Convex. Incldues MIF HTML translation.

Making HTML into TeX We did this with the "WWW Book" to print it. See the Makefile for example, and the scripts html2latex.sed and sub1.sed . We wrote a special introduction, but otherwise all the text was hypertext from the W3 project.

Generating HTML These are scripts for generating SGML hypertext from things like directory listings, etc. Also, for checking and correcting dubious HTML.

WP5.1 to HTML WordPerfect 5.1 to HTML conversion

LaTex to HTML Code from Nikos Drakos, Computer Based Learning Unit, University of Leeds.

4.2 Editing HTML

BBEdit Extensions Allow easier edit of HTML files with BBedit on the Mac.

NeXTStep editor WYSIWYG hypertext.

html-mode for Emacs Not wysiwyg but useful.

4.3 Analysing Log Files

Server log analysis Analysing server logs requires first of all changing the numeric internet node numbers into domain names. httpd-analyse.c is a program to do that. Feed the results through awk and grep of your choice! Some documentation on the program.

Server log analysis Getsites .c is a program which generates reports on a weekly or monthly basis.

4.4 Web Wanderers

Web-roaming robot etc Guido van Rossum's knobot code in "Python" language.

Web Checker James Pitkow's web checking robot

4.5 Public WWW Access Services

Telnet server Setting up a service machine for anonymous users to log in to a www client.

Mail Robot A program to return any information in the web information by electronic mail

Tim BL

4.6 HTMLGeneration

Here are some example files you can use for generating HTML from lists of files and other things.

RTF to HTML Convert RTF (using specific styles) into HTML.

fix-html.pl written by Dan Connolly, is a perl script to legitimize old HTML files into SGML-abiding HTML (as per the DTD that Dan created).

text2html.sed A sed script to turn plain text into plain-looking valid HTML markup so that it will be rendered just as it was.

ls2html.awk is an awk script which will just take a list of names and generate a menu.

dir2html is a shell script which generates a menu of pointers to files with particular suffixes in a set of directories. It also includes a README file at the head of the hypertext list if one exists.

htn2html.c See the Hytelnet gateway for the program to convert hytelnet data into HTML.

findrefs.pl Written by Ari Lemmke, finds references http:... in plain text files and generates anchors out of them.

LaTeX to HTML Latex to HTML converter program by Nikos Drakos - not only does it successfully show the more complex Latex formatting, for example for mathematics, but it also has a set of iconic images, which are included for navigation, and to mark footnotes and references.

You can make any variations on these you like of course. [CERN does not accept any responsability for things quoted in these lists].

4.7 Updating the Newsgroup lists

To update some of the news pages automatically you must be logged on to the news server or have the news directories mounted.

Carl mentioned that you must be a member of the UNIX group news (otherwise you won't have permission to read the news directories) but that doesn't seem to be necessary for these functions.

4.7.1 UpdateGroups

This script updates the list of newsgroups. For the overview list , it saves everything before the "Others" heading, and adds on a list of pointers to newsgroup stems not already mentioned in the saved hypertext.

For each stem, it saves any command before the glossary list of groups, and then regenerates that list of groups.

4.7.2 NewsPage_Update (old)

The script NewsPage_Update creates complete lists of active groups for the following groups: alt, bionet, bit, biz, cern, ch, comp, eunet, gnu, news, rec, sci, soc, talk, vmsnet. It does this by writing the header in explicitly for each group, and then generating a list of of subgroups using FindGroups

For comp and news, a full list is placed in fullcomp.html and fullnews.html. The files comp.html and news.html are formatted by hand already, and so are not touched by the script.

NewsPage_Update works by writing some HTML text into a file for each group to be updated, called [newsgroup_name].html.new, then calling the script FindNewsGroups. This checks the file /usr/local/lib/news/newsgroups for the groups within the current group which are active. Finally the new file is renamed to remove the .new.

The list of stems to search, and their titles and any other comment is hardcoded into the NewsPage_Update script, and the list is DUPLICATED in Others_Update.

4.7.3 Others_Update

The Others_Update script finds stems which are not included in the Overview.html file, but which are active. This list of which groups not to include is hardcoded into the script. For each group, it calls GrpCreate. This adds the name to OtherGroups/Overview. It then runs FindNewsGroups for each group.

NOTE

Once the script has completed all the .new groups must be renamed manually to remove the .new extension.

4.7.4 GrpCreate

This reads a newsgroup stem name from stdin.

It then creates the top of a file for the list of groups with that stem. This will be called ${nn}.html.new. where ${nn} is the stem name. Unfortunately there is no way to get a description of the stem to include in this file. However, if the .html file already exists, it will use everything up to an excluding the first DL tag from the .html file for the .html.new file. Therefore, everything above the DL tag may be hand edited.

GrpCreate adds a pointer from OtherGroups/Overview.html.new to the .html file.

The .html file is renamed .html.old, and teh .html.new becomes .html, with diffs being stored in a .diffs file under the date. .\" Macros for HTML .\" Jim Davis 6 Nov 92 .ps 12 .in 5 .de B .. .de R .. .de H1 .ti -5 .ps 18 \fB\\$1\fR .ps 12 .br .. .de H2 .ti -3 .ps 14 \fB\\$1\fR .ps 12 .br .. .de H3 \\$1 .br .. .de H4 \\$1 .. .de H5 \\$1 .. .de H6 \\$1 .. .de H7 \\$1 .. .de H8 \\$1 .. .de H9 \\$1 .. .de DL .in +5 .. .de DE .in -5 .. .de DT .ti -3 * \\$1 .. .de DD .br ..

```
Date: Wed, 4 Nov 1992 16:48:34 -0500
From: Jim Davis <davis@dri.cornell.edu>
To: wei@xcf.berkeley.edu, www-talk@nxoc01.cern.ch
Subject: improved printing of WWW files
```

If you can't quite manage to live without hardcopy, you may wish sometimes to print WWW files. I have written a couple of scripts to do this. They are particularly useful with Pei Wei's excellent Viola WWW browser.

A tar archive is available for anonymous FTP:

dri.cornell.edu/pub/davis/print-www.tar

It contains:

```
README
print-www
print-www.l
html-to-latex
html2latex.sed (modified version of original CERN version)
```

The hardest part was writing the perl script to obtain documents via http protocol - turns out you cant just run pipes through telnet.

The conversion from HTML to LaTex is not really robust yet - this is doubly hard since there is no guarentee that the HTML is legal. But at least it works for my test cases. No doubt it will be improved in time.

best wishes

5. Gateway Software

See also: W3 server software , W3 client software

These are servers which provide data extracted from other systems. they are built using code from the basic daemon, or scripts.

ACEDB gateway (see also the french version) ACEDB is the database program written for the nematode genome project.

FIND gateway for CERN/VM XFIND which calls a REXX exec to get the information from the XFIND system running on the CERNVM mainframe.

Hytelnet gateway A gateway to Peter Scott's list of telnet sites

VMS Help gateway This allows any VMS help files to be made available to WWW clients. Runs on VAX/VMS.

WAISGate A gateway to information available using the W.A.I.S. protocol.

DCLServer A server for VMS systems which allows you to write a gateway to your own favorite information system using DCL.

System33 A (big) csh script server providing data including Xerox System33 documents, man pages in plain text, phone numbers, etc. etc...!

Oracle A generic server to oracle. Could be used as a basis for gateways to specific Oracle databases.

Geography Gateway to the Geography server at U Michigan

TechInfo TechInfo is the CWIS from MIT. A gateway exists thanks to Linda Murphy/Upenn.

Tim BL

5.1 Geography gateway

Wed, 18 Nov 1992

Jim Davis <davis@dri.cornell.edu> Here is a quickly hacked up Gateway from WWW to the University of Michigan Geography server. It expects one argument, a WWW doc id. It ignores the "pathname", extracts the search words, then passes those to the server. It does NOT parse the data returned by the server (that is an improvment yet to be done) but you can understand the output.

To use this, you would need to have an HTTP server running someplace where you can attach this gateway. I can provide the very simple HTTP server I use here, but this subject is already documented in the WWW online documentation.

Source code in perl

5.2 The WWW TechInfo gateway

This is a gateway built using the basic server code, plus one source file in C. Thanks to Linda Murphy of Univerity of Pennsylvania for the etchinfo code.

- The gateway data as running at CERN
- The source file

Tim BL

5.3 The W.A.I.S. - WWW gateway

This is an example of a WWW server and a WAIS client. It is just the regular httpd daemon linked with:

- a version of the libwww library which was compiled with the DIRECT_WAIS option, and includes the HTWAIS module;
- the freeWAIS libraries from CNIDR .

See a summary of some data available through the gateway .

5.3.1 WSRC files

The gateway keeps a cache of WAIS "source" files. These are files describing WAIS servers. They are normally picked up automatically by searching a "directory of servers" index. Once the gateway has picked up a desciption of a server, it uses the description to describe the server to those who follow links to it. (See the HTWSRC module of libwww)

These source files are parsed, and are kept in the directory /usr/local/lib/WAIS under the server name, port, and database name. Tim BL

Warning: this is no longer working with http 1.0 . This is a known bug

5.4 VMS Help server

This server can provide WWW users with any information stored in VMS Help format.

Additional information available: :->

Try me !	An example server running at CERN
Status	The current state, pointers to more information

JFG

5.4.1 Gateway to VMS Help: Internals

These are technical and installation notes about the gateway to VMS Help . Please send bug reports and suggestions to Jean-Francois Groff (jfg@cernvax.cern.ch).

Sources

The program consists of the generic daemon HTDaemon.c , and a special function, stored in VMSHelpGate.c , to retrieve VMS Help data and convert it to HTML.

Installation

The files you need are as follows. You should customise them, putting in your own directory names.:

launchgate.com	Runs the server as a detached process. Put a call to this from your sys$startup procedure, wherever that is. This detaches a job to use www_server.com ans input, and a log file as output.
www_server.com	The server command file, a wrapper for the actual server executable. In this file, set the temporary directory for the storage of a cache of .HLP files. This file runs the executable.
test.com	Here is just an example of a file to build and test the server.

descrip.mms This is an MMS file to build the executable. If you don't have MMS, you may be able to figure out from loking at it which commands you should use. You can find a machine running MMS and generate the equivalent .com files. See comments at the top of this file on how to run it.

The source files and executable .EXE are currently (October 92) available on HEP decnet in vxcrna::disk$d1:[jfg.www...]. Note also you can pick up the master sources from dxcern:: automatically by running

MMS /MACRO=(U=DXCERN::).

If you are not in HEP decnet, you should find the sources in the WWWDaemon_v.vv.tar.Z file in the distribution. See the README file.

JFG

5.4.2 VMS Help server Bugs

This is a list of known bugs and desired improvements. Don't let it shrink too fast : send your bug reports and suggestions to Jean-Francois Groff (jfg@cernvax.cern.ch).

- The keyword search works fine on any number of levels down, but then the generic daemon doesn't know how deep the server went, so anchor names lack the intermediate levels. Solution : generate anchor names relative to the input path (before '?').
- DANGER : Attempts to access VMS topics with a weird name like ":=" will crash the server because VMS will try to create a .HLP file with an invalid file specification due to these special characters. Solution : Make a good escaping system (that works with VMS and Un*x styles as well). Crude and bulletproof solution : Ignore any offending topic name !
- Reference to another help library through @ will only search SYS$HELP for the corresponding .HLB file.
- We need an overview page that lists all help libraries available.

JFG

5.4.3 VMS Help server Features

This lists the main features of the VMS Help gateway, with improvements in reverse chronological order. Help make it grow fast : send your bug reports and suggestions to Jean-Francois Groff (jfg@cernvax.cern.ch).

Experimental gateway 0.4 −− 2 Oct 91

- Accepts user queries by number or by name. In the latter case, can go down several levels, for instance, from the main help page : "cc /lib" will go to topic CC, subtopic /LIBRARY.
- On invocation with only //node:port/HELP, displays the contents of the standard VMS Help library SYS$HELP:HELPLIB.HLB (function lis_to_html).
- Address format : //node:port/HELP/[@library/][topic[/subtopic]*]

JFG

6. Style Guide

This guide is designed to help you create a hypertext database effectively communicates your knowledge to the reader. It has been prepared in the light of comments by readers, and many demands by providers of online documentation. Some of the points made may be influenced by personal preference, and some may be common sense, but a collection of points has been demanded, and so here it is.
The guide is designed to be read sequentially, but feel free to depart from this. The sections are as follows:

- Introduction
- Overall structure of your work
- Within each document
- Test your document
- Background reading
- Reader comments

6.1 This document is open to comment

Suggestions are strongly invited, if you think of anything mail it to timbl@info.cern.ch, mentioning the Style Guide for Online Hypertext or its URL. Tim BL

6.2 Introduction

You are going to write (or generate) some online hypertext. Because hypertext is potentially unconstrained you are a little daunted. Do not be. You can write a document as simply as you like. In many ways, the simpler the better.
You will be writing a number of separate files. These files will be linked to each other, and to external documents, to make your final work.
You may think of your work as a "document", and if it were on paper, then you would call it that. In the online case though, we tend to refer to each individual file as a document. A document may correspond, in the book analogy, to a section or a subsection, or even a footnote. In this guide, we'll refer to the whole collection as a work.
The document is the unit by which information is picked up. At any one time, a document is completely loaded into the reader's computer. It is also normally the amount you edit at any one time, though with a good editor you will probably have a number of documents open at a time.
The section on structure discusses how you organize your material into documents. Another section discusses how to organise your material within a document .
(Up to overview , on to structure) Tim BL

6.3 Structure

If you have in mind a body of information to put across to your reader, you probably have a mental organisation for it. Normally this is a sort of hierarchical tree, like the chapters of a book if you were to write a book.
Keep this structure. It helps readers to have a tree structure as a basis for the book: it gives them a feeling of knowing where they are. You can also us this structure for oganising your files in directories.
You should also bear in mind:

- The reader's preconceived structure
- The idea of overlapping trees
- How big to make each document

(Up to overview , back to Introduction, on to: writing each document) Tim BL

6.3.1 The reader's structure .

Remember always the audience for whom you are writing. If they are novices in the subject, it will normally help if you are firm about the structure of your work, so that they can learn the structure of the knowledge itself. For example, if you feel that the subject falls into three distinct areas, then that is an importnat thing to teach.

If, however, your readers will already have some knowledge in the subject, then they will already have formed their own structure for it. In this case they will conciously or subconsiouly know where they expect to find things. If your structure is different from theirs, enforcing it too strongly will confuse them and put them off.

You may in this case have resist a strong tendency to put across your own structure strongly and to the detriment of all others. There are two solutions.

If you have a single well-defined audience in mind, who will share a similar world view, then try to write excatly for that world view rather than yours.

If you are simultaneously writing for more than one group, then you must provide for both.

When you make a reference, qualify it with a clue to allow soime people to skip it. For example, "If you really want to know how it works inside, see the Internals guide", or "A step-by-step introduction is in the tutorial".

Provide links for both reader's views. Your work will be more connected than a simple tree, but with proper qualifiaction, noone should get lost.

Provide two sepate tree "roots". For example, you can write a step-by-step tutorial and a functionaly direct reference tree for the same data. Both will at the lowest level have the same data, but while the first will deal with the simple things first, the second may be functionnaly grouped. This is just like having several indexes to a book. The tutirial might also include information which the reference work does not.

(Up to overview , back to Introduction , on to: writing each document) Tim BL

6.3.2 Overlapping Trees

Here is an example of a work (describing some programming functions, say) with two separate structures:

```
Tutorial Reference
   |     |
  Let's do it togther        -----------------
from simple to difficult       | |
    | by Functional      Alphabetical
    |     group      by name
  Task oriented examples       | |
    |        -----------------
    |     |
  Examples of use of    Syntax definition for
  specific functions   <-------->    specific functions
```

The novice user starts at the top left, and works his way down. Where he needs specific details, he will get down to the examples and from them a link to the underlying definitive desctiptions of each. As far as he is concerned, he is reading a tree-strucured work. In fact, he is reading the same information as the expert who, coming in to check on one particular function, then looks up an example of its use. (Up to structure , back to user's structure , on to: document size) Tim BL

6.3.3 How big to make each document

The most important point here is that a document should put across a well-defined concept. It is not generally worth splitting one idea arbitrarily into two bits in order to make the bits smaller. Nor is it a good idea to put together ideas which area really separate just to make a bigger document.

A document can be as small as a footnote .

There are two upper limits on a document's size. One is that long documents will take longer to transfer, and so a reader will not be able to simply jump to it and back as fast as he or she can think. This depends a lot on the link speed of course.

The other limit is the difficulty for a reader to scroll through large documents. Readers with character based terminals don't general read more than a few screens. They often only absorb what is on the first screen, as if that is not interesting they won't be bothered to scroll down. Readers are also put off by being left at the top of a large document.

Readers with graphic interfaces generally scroll through long documents with a scroll bar. When the scroll bar is moved a small amount, the document should move a sufficiently small amount so that some of the original window-full is still left in the window. This allows the reader to scan the document. If the document is any bigger, then it is basically unreadable, in that any movement of the scroll bar will loses the place and leaves the reader disoriented.

Advantages with longer documents are that it is easier for readers with scrollbars to read through in an uninterrupted flow, if that is how the document is written.

Also, one doesn't have to go to the trouble of making (or generating) so many links and keeping them up to date if things are altered. If making the links is a problem, just settle for one link to a contents page. Some browsers have "next" and "previous" buttons to allow a document to be browsed serially according to a list.

(In fact, one can normally scroll up and down explicitly page by page, but this is gives the same feeling as the terminal interface.)

A rough guide, then, for the size of a document is:

- For online help, menus giving access to other things: small enough to fit on 24 lines. Check this by using a terminal browser.
- For textual documents, of the order of half a letter-sized (A4) page to 5 pages.

(Up to structure , back to overlapping trees , on to: within each document) Tim BL

6.4 Within each document

This section of the style guide deals with the layout of text within a "document", the unit of retrieval of information on the web.

To be completed.

You should try to:

- Sign your work
- Give its status
- Make links into context .
- Use context-free document titles
- Format device-independantly
- Write for the printed work too
- Write readable text despite the links
- Avoid talking about mechanics

(up to overview , back to structure , on to testing) Tim BL

6.4.1 Sign It!

An important aspect of information which helps keep it up to date is that one can trace its author. Doing this with hypertext is easy −− all you have to do is put a link to a page about the author (or simply to the author's phone book entry).

Make a page for yourself with your mail address and phone number. At the bottom of files for which you are responsible, put a small note −− say just your initials −− and link it to that page. The address style (typically right justified) is useful for this.

Your author page is also a convenient place to put and disclaimers, copyright noitices, etc which law or convention require. It saves cluttering up the mesages themselves with a long signature.

If you are using the NeXT hypertext editor, then you can put this link from your default blank page so that it turns up on the bottom of each new document.

(up , back to ..., on to giving your document's status)

6.4.2 The status of your document

Some information is definitive, some is hastily put together and incomplete. Both are useful to readers, so do not be shy to put information up which is incomplete or out of date −− it may be the best there is. However, do remember to state what the status is. When was it last updated? Is it complete? What is its scope? For a phone book for example, what set of people are in it?

Not every document needs a status declaration, if there is something in the overview page of the work which covers it.

You can of course also give a feel for the status of the text by its language ... bad spelling, missing capitals, and relaxed grammer all indicate informal notes. Careful use of verbs such as "shall" and "should", and the introduction of Long Capitalised Noun Phrases (LCNPs) will give at least the impression of an ISO standard. ;-)

Date it

In some cases it can be useful to put creation dates and last modified dates on your work. (Note that this is the sort of thing which one could make a server do automatically with a little programming).

Figure out whether putting one might later save the reader from following out of date information.

(back to Sign It, On to links into context)

6.4.3 Linking to context

A major difference between writing part of a serial text, and an online document, is that your readers may have jumped in from anywhere. Even though you have only made links to it from one place, any other person may want to refer to that particular point, and will so make a link to that particular part of your work from their own. So you can't rely on your reader having followed your path through your work.

Of course if you are writing a tutorial, it will be important to keep the flow from one document to the next in the order you intended for its primary audience. You may not wish to cater specially for those who jump in out of the blue, but it is wise to leave them with enough clues so as not to be hopelessly lost. Some ways of doing this are:

- Watch that your text and vocabulary stands by itself. Starting a document with "The next thing we we consider is..." or "The only solution to this problem is..." will certainly confuse.
- Sometimes the opening words refer to the context, and can be linked to background information. For example, in the WWW project documentation, the first occurence of the acronym WWW is often linked back to the central project document.
- The navigation hints at the top or bottom of the document can give explicit pointers. Examples are at the bottom of this document.

It can also be useful to imagine as you are writing that you yourself may wish to reuse the document. some day.

(Part of style guide for online hypertext . Up to Writing each document , on to Title tag) Tim BL

6.4.4 Device Independence

The hypertext you write is stored in HTML language, which does not contain information about the fonts and paragraph shapes and spacing which should be used for displaying the document.

This gives great advantages in that your document will be rendered successfully on whatever platform it is viewed, including a plain text terminal.

You should be aware that different clients do use different spacing and fonts. You should be careful to use the structuring elements such as headers and lists in the way in which they were intended. If you don't like the rendering on your particular client, don't try to fix it by using inappropriate elements, or trying for example to force extra spacing with empty elements. This may well end up being interpreted differently by other clients and looking very strange. You can in many cases configure the client displays each element.

For example:

- Always use heading levels in order, with one heading level 1 at the top of the document, and if necessary several level 2 headings, and then if necessary several level 3 headings under each level 2 heading. If you don't like the way heading level 2 is formatted, fix it on your client, don't just skip to heading level 3.
- Don't put extra spaces or blank lines into your text to pad it out, except in preformatted (PRE) sections.
- Don't refer in your text to facets of particular browesrs. Asking someone to "click here" won't make sense without a mouse, just as asking someone to "select a link by number" will betray the fact that you were using the line mode browser. Just leave a link. The instructions get boring as the user will normally know how to select a link.

See also: testing your document .

Following these guidelines you may find that the end result does not appear on your screen exactly as you would like, but your readers will probably be happier.

(Part of the Style Guide for Online Hypertext . Up to within each document , back to , on to printable hypertext) Tim BL

6.4.5 Printable hypertext

In an ideal world, paper might not be necessary. In a next to ideal world, one would have enough time to write a hypertext version of a document and also a completely reauthor a paper version. In the real world, you wilkl probably want to generate any printed documents and online documents from the same file.

Suppose the HTML files will be the master, and you will generate the printable from this, by translation into TeX, etc.

If you might one day want to do this, try to avoid references in the text to online aspects. "See the section on device independence " is better than "For more on device independence, click here .". In fact we are talking about a form of device independence .

Unfortunately the recommended practices of signing each document and giving navigational links tend to mess up the printable copy, though one can of course develop ways of stripping them out if they follow a common format.

(Up to: within each document; back to device independece , on to .readable text) Tim BL

6.5 Test your document

In a way your hypertext is like a book, which you should have proof-read. In a way, it is like a program which you should have tested. At least get someone from the group for which you wrote the document to read it and give you some feedback. Other ideas are:

- Read the document several different client programs, to ensure that you have formatted it in a device independent way.

- Monitor the readership of your document. You can do this by analysing the server log files . You may find that some parts are not being read, perhaps because people are looking in the wrong place for them. You may see that people often follow a path and backtrack. If you can guess what they were looking for, you can make the clues around the link more helpful. (Remember to keep log information confidential until you have removed user information from it.)
- Make it clear whether your will accept criticism or suggestions from your readers, and how they should send it.
- Ask people to solve problems using the document, and report on their success. If they fail, find out what they were looking for, whether it was in the document at all,

6.5.1 How much testing?

Testing takes time. The decision of how much testing you do is based on the quality of the document you wish to provide. You are balancing your reader's time and effort against yours. If your document is "selling" an idea, or if you are selling the document or providing a service, you will want to make it as easy as possible for the reader. If many people will read your work, a little of your time will save a lot of theirs.

If however you are documenting some obscure part of a system in which no one other than yourself is likely to be interested, or if you feel that your readers are lucky to have anything available at all, there is no point wasting time testing it. In the event of someone needing the information, they might have to go to some extra trouble to follow several links to find what they want, and then to understand what you have written. This may be the most efficient way of working. I emphasize this because there is very much information which is for a fleeting moment in people's minds, or is hastily scribbled down on some file, and which may be important to posterity. It is better for this information to be available even in unpolished form than for it to be hidden out of embarrassment for its form. Before electronic technology, the effort of publishing was such that this information was never seen, and it was a waste, and and considered an insult to one's readers, to publish something which was not of high quality. Nowadays, there is "publishing" at all levels, and both high quality and hasty documents have their value. It is important, though, to make it clear what the quality of a document is when making a reference to it, to avoid disappointment.

Monitoring the server log files will tell you which documents are really being read. You can use your time most efficiently to improve the quality of those. Of course, analysing the server log files also takes time!

(Part of the Style Guide for Online Hypertext . Back to Within each doument, On to Background reading) Tim BL

6.6 Within each document

This section of the style guide deals with the layout of text within a "document", the unit of retrieval of information on the web.

To be completed.

You should try to:

- Sign your work
- Give its status
- Make links into context .
- Use context-free document titles
- Format device-independantly
- Write for the printed work too
- Write readable text despite the links
- Avoid talking about mechanics

(up to overview , back to structure , on to testing) Tim BL

6.7 Background reading

Some other documents which may be of relevance, if you are reading the Style Guide for Online Hypertext :

- The HTML Specification and references from it
- A Beginner's Guide to writing HTML
- World-Wide Web server software - a list of pointers
- Web Ettiquette –– for Server Administrators

(Back to testing, on to ...)

7. Mail Robot

The mail robot is a program which will accept incoming mail and allow remote users to:

- Subscribe to mailing lists (and unsubscribe)
- Retrieve information given a W3 addresss (URL)

Originally from UC Berkeley, an enhanced robot is distributed as part of the world-wide web global information initiative . Futhur information available is:

Help	The help file for users of the robot service
Installation	Installation instructions for unix system managers
Bugs	Lists of improvements requested or needed.
Change history	A list of features introduced and bugs fixed.
See also	Other WWW software

7.1 Using the W3 mailing robot

This robot maintains the W3 mailing lists, and allows W3 documents to be retrieved on request.

You can subscribe or unsubscribe to any of the various WWW mailing lists by sending email to the robot "listserv@info.cern.ch" –– see the commands listed below.

If you have any problems, requests or questions for a human being, mail "www-request@info.cern.ch". Lists are:

www-announce	Anyone interested in WWW, who would like information about new releases or new online data available. Please refrain from posting administrivia to this large list !
www-talk	Developers of WWW code, or those interested in discussions of technical details

You can also find information on WWW (as well as many other things!) by telnetting to info.cern.ch (no username, no password).

If you want to pick up the WWW software, then use anonymous FTP to info.cern.ch and look in directory /pub/www. Subdirectories are src for the latest source packages, bin for executables for various machines, doc for "paper copies" of articles on WWW in PostScript and ASCII form. To read the latest documentation, use WWW !

7.1.1 Commands

The commands understood by the listserv program are:

HELP	lists this file. This is also sent whenever a message to listserv is received from which no valid command could be parsed.
HELP groupname	lists a brief description of the group requested.
ADD listname	Add yourself to the list
DELETE listname	take yourself off the list
ADD address listname	Add yourself with a given mail address to the given list. The address must not contain spaces!
DELETE address listname	Remove the given name from the given list. For all ADD/DELETE commands, mail is sent to the address given to confirm the add or delete operation.
SEND document-address	returns a document with the requested W3 address.
STOP	Stop processing requests: ignore the rest of the message. Needed if you send a signature on the end of your message (or if some gateway adds one). If in doubt, use it.

A command must be the first word on each line in the message. Lines which do not start with a command word are ignored. If no commands were found in the entire message, this help file will be returned to you. A single message may contain multiple commands; a separate response will be sent for each.

Examples

```
add www-announce

add me@host.uni.edu www-announce

delete me@host.uni.edu www-talk

send http://info.cern.ch/hypertext/DataSources/bySubject/Overview.html
```

7.1.2 Subscription

If you are not sending mail from your preferred mail address, then you can use the second form of the command to give your mail address. If you are not on the internet, please convert your address into arpa stye. (For example, UK users please use international ordering joe@host.ac.uk) Just speficy the mailbox, without any spaces.

If you omit the 'address' the command will assume the mailbox that is in the From: line of the message. Note that SUBSCRIBE is a synonym for ADD; UNSUBSCRIBE for DELETE.

Please note that is IS possible to add or delete someone else's subscription to a mailing list. This facility is provided so that subscribers may alter their own subscriptions from a new or different computer account. There is therefore some potential for abuse; we have chosen to limit this by mailing a confirmation notification of any addition or deletion to the address added or deleted including a copy of the message which requested the operation. At least you can find out who's doing it to you.

Note that although you would mail submissions to a mailing list by addressing mail to e.g., www-talk@info.cern.ch, in a subscription request you specify the name of the list simply (without the @hostname part) as in the first example above.

7.1.3 Retrieving documents

The SEND command (or the WWW command which is equivalent) returns the document with the given W3 address, subject to certain restrictions. Hypertext documents are formatted to 72 character width, with links numbered. A separate list at the end gives the document-addresses of the related documents.

If the document is hypertext, it links will be marked by numbers in brackets, and a list of document addresses by number will be appended to the message. In this way, you can navigate through the web, albeit only at mail speed.

If you don't know where to start, try asking for one of

```
http://info.cern.ch./hypertext/DataSources/bySubject/Overview.html
http://info.cern.ch./hypertext/DataSources/bySubject/Physics/HEP.html
http://info.cern.ch./hypertext/WWW/TheProject.html
```

for lists of futher pointers.

7.1.4 CAUTIONARY NOTE

As the robot gives potential mail access to a *vast* amount of information, we must emphasise that the service should not be abused. Examples of appropriate use would be:

- Accessing any information about W3 itself;
- Accessing any CERN and/or physics-related or network development related information;

Examples of INappropriate use would be:

- Attempting to retrieve binaries or .tar files or anything more than directory listsings or short ASCCII files from FTP archive sites;
- Reading internet newsgroups which your site doesn't take;
- Repeated automatic use;

There is currently a 1000 line limit on any returned file. We don't want to overload other people's mail relays or our server. We reserve the right to withdraw the service at any time. We are currently monitoring all use of the server, so your reading will not initially enjoy privacy. End of cautionary note.

Enjoy! The W3 team at CERN (www-bug@info.cern.ch)

7.2 Installation

Here are the steps necessary to install the Mail Robot product on your unix system.

7.2.1 Customisation

Set up the variables in listserv.h and CommonMakefile to suit your site.

POSTMASTER	The address from which messages appear to come. Why not listserv? Perhaps to prevent mail loops.
SECUREWWW	The executable W3 line mode browser (v1.3 or later, so as to have the -listrefs option). This is a separate product. For security, www should be writable only by root.
SERVERDIR	The directory in which you want to put your mailing lists and help about them.

7.2.2 Compile the programs

Everything compiled on AEM's MicroVax II running ULTRIX 3.0 then TBL's NeXT without any problem at all. Your results may vary.

7.2.3 Create your SERVDIR

wherever you specified in listserv.h. Install a HELP file, perhaps using the example-files/HELP in this directory as a template.

7.2.4 Set up an alias "listserv"

Make an alias in your /etc/aliases (or /etc/sendmail/aliases, whatever you have) that points to this program, for example:

```
listserv: "|/usr/local/mail/listserv"
robot: "|/usr/local/mail/listserv"
```

7.2.5 For each mailing list

Create a name.info file giving a bit of information about that mailing list. see the *.info files in the example-files subdirectory.

Create a name file in the same directory, consisting of email addresses one to a line of subscribers to a group. If it is for a brand-new group, create an empty file. Remember that this file must be writable by the mail daemon. The name of the file is just the name of the group.

Depending on how you have your mailing lists set up, you may need to add an alias to the /etc/aliases file for each of the mailing lists. For example:

```
real-recipes: :include:/usr/local/mail/maillists/recipes
```

So sending mail to real-recipes actually goes to each of the subscribers listed in /usr/local/mail/maillists/recipes

7.2.6 Install listserv

Install in the appropriate directory. Edit the CommonMakefile and then

```
make install
```

7.2.7 Run newaliases

This gets sendmail to read the changes in /etc/aliases.

```
newaliases
```

7.2.8 Try it out

Send mail to listserv with body

```
HELP
```

for example. You should get a plain text version of the help file.

7.3 Mail Robot

This is a "listserv" type program which maintains mailing lists, and allows W3 documents to be retrieved by electronic mail.

Author:	Various, modified by TBL.
Status:	Source available by anonymous FTP. (Oct 92)
Current version:	1.0
Platforms:	Unix only.
More information:	Overview , Bugs , change history .

7.4 Bugs

This is a list of bugs in or improvements desired in the Mail Robot. See also the list of bug fixes .

- The INDEX command ought to be implemented, but for some reason always returns an empty list. Occasionally it seems to work.

7.5 Change History

Changes to the Mail Robot , in reverse chronological order:

7.5.1 October 1992

TBL added information retrieval possibility using WWW. Release as an unsupported W3 product to those who ask for it.

7.5.2 1991

TBL rewrote str.c (used to overwrite its arguments).

7.5.3 AEM

A. E. Mossberg, aem@mthvax.cs.miami.edu made a couple minor changes, to make it slightly less UCSD-specific. He also added a README, and example files in the subdirectory example-files.

7.5.4 Origin

Note this is NOT the bitnet LISTSERV program. The term "mail robot" is yused to attempt to prevent confusion between these two products, which have different functionality although they do basically the same sort of thing.

This was the UCSD listserv program, which AEM retrieved from ucsd.edu by anonymous ftp, TBL retrieved from ftp.eff.org As retrieved, from file://ftp.eff.org/pub/listserv2.shar, it consisted of the following files:

```
README
Makefile
commands.c
listserv.h
main.c
str.c
subscribe.c
```