

Notes from Well House Consultants

These notes are written by Well House Consultants and distributed under their Open Training Notes License. If a copy of this license is not supplied at the end of these notes, please visit

*<http://www.wellho.net/net/whcotnl.html>
for details.*

1.1 Well House Consultants

Well House Consultants provides niche training, primarily but not exclusively in Open Source programming languages. We offer public courses at our training centre and private courses at your offices. We also make some of our training notes available under our "Open Training Notes" license, such as we're doing in this document here.

1.2 Open Training Notes License

With an "Open Training Notes License", for which we make no charge, you're allowed to print, use and distribute these notes provided that you retain the complete and unaltered license agreement with them, including our copyright statement. This means that you can learn from the notes, and have others learn from them too.

You are NOT allowed to charge (directly or indirectly) for the copying or distribution of these notes, nor are you allowed to charge for presentations making any use of them.

1.3 Courses presented by the author

If you would like us to attend a course (Java, Perl, Python, PHP, Tcl/Tk, MySQL or Linux) presented by the author of these notes, please see our public course schedule at

<http://www.wellho.net/course/index.html>

If you have a group of 4 or more trainees who require the same course at the same time, it will cost you less to have us run a private course for you. Please visit our onsite training page at

<http://www.wellho.net/course/otc.html>

which will give you details and costing information

1.4 Contact Details

Well House Consultants may be found online at

<http://www.wellho.net>

graham@wellho.net

technical contact

lisa@wellho.net

administration contact

Our full postal address is

404 The Spa

Melksham

Wiltshire

UK SN12 6QL

Phone +44 (0) 1225 708225

Fax +44 (0) 1225 707126

FTP and Telnet Modules

FTP and telnet are excellent ways to work with remote computers, but they're designed around a human user with keyboard input and screen output. Using Perl modules, you can automate FTP and telnet processes that you run regularly, and even use FTP and telnet connections to access remote data from within a larger Perl application.

<i>Net::Telnet</i>	7
<i>Net::DNS</i>	9

The File Transfer Protocol (FTP) has become an industry standard for copying files from one location to another. Most commonly, files are "pulled" by a user from a server, but FTP can also be used to "push" data to a server. In this latter case, you will typically require an account or be limited to a very small area of the server's discs!

The original FTP client program is command-based, and Perl's Net::FTP module follows the same principle. Whilst users may be familiar with using a browser or graphic client to access FTP, that is not the approach that the module takes.

FTP actually uses two sockets rather than one; one channel is used for data, the other for control messages. For this reason, we recommend you use the CPAN module for FTP work rather than writing your own, unless your requirement is particularly specialist!

Most methods in the Net::FTP module return a true value if they succeed, and false if they fail, but there are exceptions noted later in the section.

Simple example

Net::FTP is object oriented; use the new method to create an instance which defines a connection to a particular host. Thus:

```
$ftpobj = Net::FTP -> new ("ftp.wellho.co.uk");
```

To connect and login, use the login method:

```
$ftpobj->login();
```

Note that the login method defaults to anonymous FTP, sending the user's email address as the password. To login as a user, you may give a user name and password, thus:

```
$ftpobj->login("p1","abc123");
```

(Recall that it's a security risk to include passwords in programs like this, and that the program will now have to be altered if the password is changed!)

To change to a particular directory on the FTP server, use the `cwd` method:

```
$ftpobj->cwd("/export/home/www/pub")
```

and then to retrieve a file:

```
$ftpobj->get("index.html");
```

Finally in this first example ... to disconnect:

```
$ftpobj -> quit();
```

Let's see that in use:

```
vole% ./ftpdemo
retrieving file from seal
file size 4086 bytes
vole%
```

```
#!/usr/bin/perl
```

```
use Net::FTP;
```

```
print "retrieving file from seal\n";
```

```
$ftpobj = Net::FTP -> new ("seal");
```

```
$ftpobj -> login("p1","abc123");
```

```
$ftpobj -> cwd ("/export/home/www/cosite/about");
```

```
$ftpobj -> get ("melksham.html");
```

```
$ftpobj -> quit;
```

```
print "file size ",-s "melksham.html"," bytes\n";
```

Figure 1 Example of use Net::FTP;

More flexible transfers

By default, FTP transfers files in ASCII mode. This is fine for text files (and can even help change end-of-line delimiters) but if binary files are being transferred they will be corrupted unless you specify:

```
$ftpobj -> type("binary");
```

Note that binary transfers are a little less efficient than ASCII ones, and that the type set applies to subsequent gets and puts.

In order to push a file from the client to the server, a put method is provided. It takes either a single parameter (the name of the file is to be the same at each end) or two file names (from and to). The get command can also take a second parameter – the "to" file.

Enquiries

The ls method returns a pointer to a list of objects in the current directory, or a named directory if a name is given as the parameter.

The dir method returns a pointer to a similar directory listing but in "long" format, allowing you to work out what's a subdirectory, how large a download would be, etc.

Methods mdtm and size return the modification time and size of a particular file.

Let's use some of these enquiry methods to scout around part of an FTP site looking for all files with names ending in "html".

Figure 2 Searching an FTP site

```
vole% ./ftp2
Search an FTP site!
Site name: seal
Login name: pl
Password:
Directory to search from: /extra/disc0.slice7/pan
/extra/disc0.slice7/pan/norobots-rfc.html
/extra/disc0.slice7/pan/CGI.pm-2.49/cgi-lib_porting.html
/extra/disc0.slice7/pan/CGI.pm-2.49/cgi_docs.html
/extra/disc0.slice7/pan/docs/INDEX.rfc.html
/extra/disc0.slice7/pan/docs/rfc1305.html
/extra/disc0.slice7/pan/docs/rfc1769.html
/extra/disc0.slice7/pan/docs/rfc868.html
/extra/disc0.slice7/pan/docs/time.html
/extra/disc0.slice7/pan/CGI.pm-2.49/examples/index.html
9 files, total size 1310693 bytes
vole%
```

```
#!/usr/bin/perl

use Net::FTP;

# searches for files
# matching a certain
# pattern within or below a
# named directory on an FTP
# server

print "Search FTP site!\n";

# timeout set to 20 seconds
```

```

# in case a system isn't
# reachable, even though
# it's known!

print "Site name: ";

chop ($sname = <STDIN>);
$ftpobj = Net::FTP -> new ($sname,Timeout=>20) or
    die "Cannot access $sname via FTP\n";

# Ensure that password is not echoed!

print "Login name: ";
chop ($sname = <STDIN>);
`stty -echo`;
print "Password: ";
chop ($pwd = <STDIN>);
`stty echo`;
print "\n";

$ftpobj -> login($sname,$pwd) or
    die "Invalid user name and/or password\n";

# Check that given root really exists!

print "Directory to search from: ";
chop ($sdir = <STDIN>);
$ftpobj -> cwd ("$sdir") or
    die "Cannot access this directory\n";

# @dirstack is a queue of directories to be
# visited

@dirstack = ($sdir) ;
while ($this = shift @dirstack)
{
    $ftpobj -> cwd ("$this");
    $jpdire = $ftpobj -> dir;
    foreach (@{$jpdire}) {
        @ls = split;
        # skip hidden, parent and self!
        next if ($ls[8] =~ /^\.\/);
        if ($ls[0] =~ /^d/) {
            push @dirstack,$this."/".$ls[8];
            next;
        }
        if ($ls[0] =~ /^-\/ and $ls[8] =~ /html$/ )
        {
            print $this."/".$ls[8],"\n";
            $tsize += $ls[4];
            $count ++;
        }
    }
}

$ftpobj -> quit;
print "$count files, total size $tsize bytes\n";

```

Other facilities

There's a wide range of other methods and facilities available, many of which are likely to be unavailable to you because of aspects of server security (e.g. `rmdir`). Have a look in the manual pages for the module for further information.¹

2.1 Net::Telnet

At its simplest, Telnet is a protocol used for remote logins across a network, and you may have used it (giving a port number) to access other services. If you're familiar with what you're doing, you'll probably want to use sockets and a direct connection yourself rather than using `Net::Telnet`.

`Net::Telnet` allows you to connect to (and then log in to) a remote host, and have a shell exchange with that host. It allows (for example) you to handle timeouts easily and read back up to and including a shell prompt.

Running a remote shell and commands

This example connects to a remote host – logging in as a user and returning the results from running a Unix command.

`Net::Telnet` is object oriented; firstly create your Telnet object:

```
$telobj = Net::Telnet->new(Host => "seal",
                          Prompt => '/% $/');
```

Then log in to the server:

```
$telobj -> login ("p1","abc123");
```

(see security note under FTP about this!)

And then issue a command and retrieve the results:

```
@response = $telobj->cmd("grep $myhost /usr/etc/httpd/logs/error_log");
```

You can run a whole series of commands in a single session; when you're done, disconnect using:

```
$telobj -> close;
```

¹ If you want to install `Net::FTP`, it's a part of the `libnet` bundle on the CPAN.

```

vole% ./telnetdemo
reporting end of web server log files:

co-access_log
=====
sealion - - [26/Jul/1999:17:59:21 +0100] "GET /cgi-bin/wellho/ford_data.pl HTTP/1.0" 200 1623
lecht - - [26/Jul/1999:19:54:45 +0100] "GET / HTTP/1.0" 304 0
seal - - [26/Jul/1999:19:54:46 +0100] "GET /welling/space.gif HTTP/1.0" 304 -
walrus - - [26/Jul/1999:19:54:46 +0100] "GET /welling/space.gif HTTP/1.0" 304 -
sealion - - [26/Jul/1999:19:54:49 +0100] "GET /welling/index.jpg HTTP/1.0" 304 -
flipper - - [26/Jul/1999:19:54:49 +0100] "GET /welling/WHhead.gif HTTP/1.0" 304 -
magnet - - [26/Jul/1999:19:54:53 +0100] "GET /about/modules.html HTTP/1.0" 304 -
seal - - [27/Jul/1999:07:02:13 +0100] "GET /welling/connect.jpg HTTP/1.0" 200 12356
lecht - - [27/Jul/1999:07:02:21 +0100] "GET /support/connect.html HTTP/1.0" 200 30312

firstalt-access_log
=====
seal - - [26/Jul/1999:17:28:52 +0100] "GET /java/JA/folder/rook.html HTTP/1.0" 200 1250
walrus - - [26/Jul/1999:17:28:54 +0100] "GET /java/JA/folder/rook.class HTTP/1.0" 200 4636
sealion - - [26/Jul/1999:17:30:57 +0100] "GET /java/JA/folder/jackdaw.html HTTP/1.0" 200 1490
flipper - - [26/Jul/1999:17:30:59 +0100] "GET /java/JA/folder/jackdaw.class HTTP/1.0" 200 3469
magnet - - [26/Jul/1999:17:31:10 +0100] "GET /java/JA/folder/jackdaw2.html HTTP/1.0" 200 268
aviemore - - [26/Jul/1999:17:31:48 +0100] "GET /java/JA/folder/jackdaw.java HTTP/1.0" 200 4724
lecht - - [26/Jul/1999:17:38:52 +0100] "GET /java/JA/graham/ftest.html HTTP/1.0" 200 1488
seal - - [26/Jul/1999:17:39:00 +0100] "GET /java/JA/graham/ford.class HTTP/1.0" 200 3462
walrus - - [26/Jul/1999:17:40:25 +0100] "GET /java/JA/graham/ftest.html HTTP/1.0" 200 1519

wellho-access_log
=====
walrus - - [25/Jul/1999:07:40:46 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 4366
sealion - - [25/Jul/1999:07:41:10 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 6979
flipper - - [25/Jul/1999:07:41:12 +0100] "GET /entry.gif HTTP/1.0" 200 3392
magnet - - [25/Jul/1999:07:41:47 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 5137
aviemore - - [25/Jul/1999:07:41:48 +0100] "GET /validate.gif HTTP/1.0" 200 3199
lecht - - [25/Jul/1999:07:42:05 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 4801
seal - - [25/Jul/1999:07:42:06 +0100] "GET /correct.gif HTTP/1.0" 200 3907
walrus - - [25/Jul/1999:07:42:17 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 5142
sealion - - [25/Jul/1999:07:42:25 +0100] "POST /cgi-bin/avair/avair.pl HTTP/1.0" 200 6979

vole%

```

Figure 3 Reporting end of web server log files

```

#!/usr/bin/perl

use Net::Telnet;

print "reporting end of web server log files:\n";

$telobj = Net::Telnet->new(
    Host => "seal",
    Prompt => '/% $/');

$telobj -> login("p1","abc123");

```

```

$rad = $telobj -> cmd ("cd /usr/local/etc/httpd/logs");
@rad = $telobj -> cmd ("ls -l *access*");
pop @rad;
chomp @rad;
print "\n";

$="" ;
foreach (@rad) {
@recent = $telobj -> cmd ("tail $_");
pop @recent;
print;
print "\n", "="x length($_), "\n";
print "@recent\n";
}

$telobj -> close;

```

Using Net::Telnet to hold a remote conversation

This is an alternative to using the standard Socket module. With this approach, you are required to understand the underlying protocol.

Use the constructor to establish the link, specifying host and port number in the parameters:

```

$stob = Net::Telnet -> new(
    -host => "seal",
    -port => 80,
    -timeout = 5 );

```

The timeout specified here is for the time taken to establish the connection.

Write to the port opened:

```

$stob -> print "HEADER /index.html HTTP/1.0\n\n";

```

Careful – this is not Perl's `print` function, but the one that's in this module!

Read back a response, allowing only a certain amount of time:

```

@response = $stob -> getlines(-timeout => 10);

```

And when you're done, you may wish to close the connection (in practice, the server will often have done so as your exchange is completed):

```

$stob -> close;

```

2.2 Net::DNS

The Domain Name Service (DNS) is the mechanism by which host computer names, mail forwarder addresses, etc. are resolved on the internet.

Perl has built-in functions `gethostbyname` and `gethostbyaddr` that suffice for many programmers; the DNS modules provide an extended capability.

Name setup and resolution is a complex topic; there's a complete book on DNS in the O'Reilly series, and `Net::DNS` isn't a single module; rather it's a series of linked classes and modules. If you need further training on DNS administration, please ask; we have a half day course available.



Exercise

License

*These notes are distributed under the **Well House Consultants Open Training Notes License**. Basically, if you distribute it and use it for free, we'll let you have it for free. If you charge for its distribution of use, we'll charge.*

3.1 Open Training Notes License

Training notes distributed under the **Well House Consultants Open Training Notes License** (WHCOTNL) may be reproduced for any purpose PROVIDE THAT:

- This License statement is retained, unaltered (save for additions to the change log) and complete.
- No charge is made for the distribution, nor for the use or application thereof. This means that you can use them to run training sessions or as support material for those sessions, but you cannot then make a charge for those training sessions.
- Alterations to the content of the document are clearly marked as being such, and a log of amendments is added below this notice.
- These notes are provided "as is" with no warranty of fitness for purpose. Whilst every attempt has been made to ensure their accuracy, no liability can be accepted for any errors of the consequences thereof.

Copyright is retained by Well House Consultants Ltd, of 404, The Spa, Melksham, Wiltshire, UK, SN12 6QL - phone number +44 (1) 1225 708225. Email contact - Graham Ellis (graham@wellho.net).

Please send any amendments and corrections to these notes to the Copyright holder - under the spirit of the Open Distribution license, we will incorporate suitable changes into future releases for the use of the community.

If you are charged for this material, or for presentation of a course (Other than by Well House Consultants) using this material, please let us know. It is a violation of the license under which this notes are distributed for such a charge to be made, except by the Copyright Holder.

If you would like Well House Consultants to use this material to present a training course for your organisation, or if you wish to attend a public course is one is available, please contact us or see our web site - <http://www.wellho.net> - for further details.

Change log
Original Version, Well House Consultants, 2004

Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____
Updated by: _____ on _____

License Ends.