Networking using the HTTP protocol

Server-Side Web Languages

Uta Priss
School of Computing
Napier University, Edinburgh, UK

Outline

The HTTP protocol

Manual interaction with the HTTP protocol

Manual interaction with a webserver is possible by typing

```
telnet www.soc.napier.ac.uk 80
```

then typing

```
GET / HTTP/1.0
```

and then pressing enter twice.

Format of the first line of an HTTP request:

`METHOD space Request-URI space HTTP-Version \n\n`

Methods are GET, HEAD, OPTIONS, POST, DELETE, etc.

The Request-URI is usually the path and name of the document (eg. "/" or "/somedir/index.html").

The HTTP-Version is currently either 1.0 or 1.1.

A typical HTTP request with GET

The HTTP request is normally generated by the user's browser. A typical request with "GET" is

```
GET /cgi-bin/somefile.cgi HTTP/1.0
CONNECTION: Keep-Alive
USER-AGENT: Mozilla/4.0 (compatible; MSIE 5.22)
PRAGMA: no-cache
HOST: www.dcs.napier.ac.uk
ACCEPT: image/gif, image/x-xbitmap, image/jpeg, */*
QUERY_STRING: name=Mary&comments=Hello
```

A typical HTTP request with POST

```
POST /cgi-bin/somefile.cgi HTTP/1.0
REFERER: http://napier.ac.uk/cgi-bin/someform.html
CONNECTION: Keep-Alive
USER-AGENT: Mozilla/4.0 (compatible; MSIE 5.22)
HOST: www.dcs.napier.ac.uk
ACCEPT: image/gif, image/x-xbitmap, image/jpeg, */*
CONTENT-TYPE: application/x-www-form-urlencoded
CONTENT-LENGTH: 42
name=Mary&comments=Hello
```

A typical answer from a webserver

```
HTTP/1.1 200 OK
Date: Thu, 18 Nov 2004 15:41:04 GMT
Server: Apache/2.0.40 (Red Hat Linux)
Accept-Ranges: bytes
X-Powered-By: Open SoCks 1.2.0000
Last modified: Thu, 18 Nov 2004 15:41:04 GMT
Connection: close
Content-Type: text/html; charset=ISO-8859-1
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional/
"http://www.w3.org/TR/html4/loose.dtd">
<html>

...
```

Different possible first lines

```
HTTP/1.1 200 OK
HTTP/1.1 404 Not Found
HTTP/1.1 302 Found
HTTP/1.0 500 internal error
```

(302 indicates a redirection):

Low-level interaction with the HTTP protocol

Normally users need not interact with the HTTP protocol on a low level because browsers do all the work. But there are reasons why programmers may need more low level access to the web:

- Testing of websites from the command-line. For example, if a whole website is to be checked for dead links this can be conveniently done from the command-line.
- Testing of the security of one's own website (especially for scripts) by trying to break in.
- Web crawlers, spiders, robots: programs that navigate the web and collect information, for example, for creating search engines.

What tools to use

Most modern programming languages contain libraries that allow for direct interaction with the HTTP protocol (such as libwww or socket libraries).
In Perl different libraries are available for interaction at different levels of abstraction (e.g. IO::Socket is less abstract than LWP).