

HTTP

In the News -- 802.11a wireless

Yet another typical network effect / standards battle...

We'll all benefit if there's a single, non-proprietary standard for, in this case, wireless data communication. The vendors, as usual, would love to "own" the standard if possible...

<http://www.nytimes.com/2001/04/30/technology/30WIRE.html>

HTTP Request

port 80

Request sent from client to server

Server sends back one response with header and body parts

<http://www.w3.org/Protocols/> -- all sorts of HTTP info

HTTP "One shot" Transaction --

"stateless"

"Stateless" -- each request/reply pair (HTTP 1.0) uses its own connection. The dialog does not group logically related collections of requests.

The HTTP request/response protocol is "one shot" -- the server fulfills the request and closes the connection.

This keeps the server's job simple, but it makes it harder to design a sequence of pages that appear logically connected, since each HTTP request is separate.

The lack of continuity between one HTTP request and the next is one of the harder things to absorb when designing HTTP client/server systems.

GET Request

Request a resource

```
GET request HTTP\r\n\r\n
```

```
GET /foo.html HTTP/1.0\r\n\r\n
```

Actually use `\015\012` in your code in case `"\n"` has been mapped to the local end-of-line.

Complex URLs -- path/file/suffix

The simple description is that everything after the host is the "path"

The more complex description divides things after the host into the path, file, and suffix. That's what we'll use. See the RFC for an even more complex

decomposition of the URL. <http://www.w3.org/Protocols/rfc2616/rfc2616.txt>

```
http://foo.com/a/b/bar.html?a=b
```

scheme

http

host
 foo.com
path
 /a/b/
file
 after the last /
 bar.html
suffix
 after the file, starting with # or ?
 may contain almost anything

Request

The request is basically the path part of the URL `http://foo.com/dir/b.html` :
 request = `/dir/b.html`

Suffix

A "#foo" suffix is **not** sent as part of the request
 The "#" part of the url is handled on the client side.
`http://foo.com/dir/b.html#bar` : request = `/dir/b.html`

? Suffix

A "?a=b&c=13" suffix is sent as the end part of the request -- the server
 (typically a CGI) does use it as part of the request.
`http://foo.com/dir/b.html?bar=http://www.yahoo.com/`
 request = `/dir/b.html?bar=http://www.yahoo.com/`
 The suffixes may contain weird characters, so identify the "#" or "?" scanning
 from the left.

Get

telnet 80

Lab exercise: try telneting to port 80 of your favorite web server. Type the
 GET directive in by hand and watch what happens. On unix, the command
 looks like...

```
telnet www.stanford.edu 80
```

Get Options

Some servers are confused if the "GET" is in lowercase
 Usually the GET includes optional information on subsequent lines...

```

GET request HTTP/1.0
User-agent: Mozilla/2.02      -- the version of client software
Accept: image/gif, image/jpeg, */* -- return type preferences of the client
Referer: url      -- page where the client was just before making this request. The
server could use this to figure out who's pointing to it.
If-Modified-Since: date -- use to only retrieve a document if it is newer than the
one the client has in cache.
  
```

Blank line

A final blank line after all the options marks the end of the request. So the end
 of the request reads "\015\012\015\012".

HEAD

Similar to GET, but retrieves the header but not the data. Use to test for the header of a page but without actually downloading all the HTML. Some servers don't implement HEAD.

The Empty Request

What about URLs with no path -- e.g. `http://www.cnn.com`

The directory path and file are the empty string.

Could just send the empty string as the request in the HTTP protocol -- many servers interpret that as `"/`, but some treat it as an error.

Here's the new (as of HTTP 1.1 anyway) correct way for the client to deal with this: If the request would be the empty string, the client should think of the request as being `"/` and send that instead. Relative URLs in the response will be relative to `"/`.

HTTP Response

The server responds with a header section which describes the document, followed by a blank line, followed by the document data.

`HTTP/1.0 200 OK`

The very first line of the response has the form `VERSION CODE REASON` -- e.g. `HTTP/1.0 200 OK`

`VERSION` = the version of HTTP which the server is speaking

`CODE` = a numeric code which compactly indicates how the request is going

`REASON` = a human-readable statement of the `CODE`

Other Fields

The rest of the HTTP header contains a variety of information, much of it optional, about the document to be returned. The two most important fields are...

`Content-Type: MIME-type` -- type of the document returned

`Content-Length: length` -- the size in bytes of the document

HTTP Example

e.g. retrieve the HTML at

`http://www.stanford.edu/class/cs193i/test/basic.html`

There's a blank line after the request, and a blank line that separates the header from the body in the response.

```
[localhost:~] nick% telnet www.stanford.edu 80
Trying 171.64.14.237...
Connected to www.lb-a.stanford.edu.
Escape character is '^]'.
GET /class/cs193i/test/basic.html HTTP/1.0

HTTP/1.1 200 OK
Date: Mon, 30 Apr 2001 18:36:56 GMT
Server: Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 (Unix) mod_fastcgi/2.2.4
Last-Modified: Sat, 17 Mar 2001 00:17:11 GMT
ETag: "185f016a-15c-3ab2ad07"
Accept-Ranges: bytes
Content-Length: 348
Connection: close
Content-Type: text/html
```

```
<html>
<head>
<title>Basic</title>
</head>
```

```
<body>
<h1>Basic</h1>
```

A few absolute URLs and one relative URL...

```
<ul>
<li><A HREF="http://www.yahoo.com/">Yahoo</a>
<li><a href=http://www.theonion.com/>The Onion</a>
<li><A HREF=http://www.stanford.edu/class/cs193i/test/a.html>a absolute</A>
<li><a href=a.html>a relative</a>
</ul>

</body>
</html>
```

Connection closed by foreign host.

Status Codes

200 = OK

3xx = Minor client error— document in another location

301 = Moved permanently. The Location: field of the header gives the correct URL.

302 = Moved temporarily -- the server may suggest the correct url using a location: in the header

304 = Not Modified. The request had If-Modified-Since: field, but the document has not been modified, so the client should use their cached copy.

- 400 = Real client error
 - bad request, document not found, not allowed
 - 400 = syntax error
 - 401 = Unauthorized
 - 403 = Forbidden -- "permission denied"
 - 404 = Not found
- 500 = Server error
 - service not implemented, service not available
 - 503 = Service unavailable. The server is temporarily not able to provide the service. The header may contain a Retry-After: field to indicate when the client might give it another shot.

MIME Types

Multipurpose Internet Mail Extensions -- a standard which predates HTTP for identifying different types of data for inclusion in email messages.

content-type/sub-type [*; aux-info*]

text/html

text/plain

text/plain ; charset = us-ascii

multipart/mixed ; boundary = SpecialBoundaryString

application/pdf

application/postscript

audio/basic -- .au audio

image/jpeg

video/quicktime

"x-.." = experimental

Since the MIME type is in HTTP response header, the server is taking responsibility for identifying the type of the data. Servers may be programmable in this respect. Most servers use the file extension (.jpeg, .html .htm) to guess the MIME type to claim when they send the bytes back.

Not-Found Example

```
Sunburn:~ > telnet www-cs-faculty.stanford.edu 80
GET /~nick/foobar/ http/1.0
```

```
HTTP/1.0 404 Not Found
Date: Mon, 29 Apr 1996 16:33:53 GMT
Server: NCSA/1.4.1
Content-type: text/html
```

```
<HEAD><TITLE>404 Not Found</TITLE></HEAD>
<BODY><H1>404 Not Found</H1>
The requested URL /~nick/foobar was not found on this server.<P>
</BODY>
```

Trailing / Example

Omit the '/' URL forwarding -- is that trailing slash really necessary? Yes!
Here we'll try the url `http://www.stanford.edu/class/cs193i` -- no trailing /

```
[localhost:~] nick% telnet www.stanford.edu 80
```

```
Trying 171.64.14.237...
```

```
Connected to www.lb-a.stanford.edu.
```

```
Escape character is '^]'.  
GET /class/cs193i HTTP/1.0
```

```
HTTP/1.1 301 Moved Permanently
```

```
Date: Mon, 30 Apr 2001 18:40:17 GMT
```

```
Server: Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 (Unix) mod_fastcgi/2.2.4
```

```
Location: http://www.stanford.edu/class/cs193i/
```

```
Connection: close
```

```
Content-Type: text/html
```

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

```
<HTML><HEAD>
```

```
<TITLE>301 Moved Permanently</TITLE>
```

```
</HEAD><BODY>
```

```
<H1>Moved Permanently</H1>
```

```
The document has moved <A HREF="http://www.stanford.edu/class/cs193i/">here</A>.<P>
```

```
<HR>
```

```
<ADDRESS>Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 Server at <A
```

```
HREF="mailto:webmaster@www.stanford.edu">www.stanford.edu</A> Port 80</ADDRESS>
```

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

```
Connection closed by foreign host.
```

Location: url

The location: header field is used for some HTTP returns to give a new URL for the client to try. Most browsers re-try the url automatically. That's why you can omit the trailing / in many cases, and the browser/server will still get you to the right place (with twice as many transactions).

HTTP 1.1

The 1.1 revision to HTTP has a couple important extensions...

1. The host: field allows the request to indicate the DNS name of the host from the URL. This allows a single machine to serve pages for multiple name -- e.g. `www.foo.com`, `www.bar.com`, etc. are all actually hosted by one machine that looks at the host: field to know which doc root to use.
2. "persistent" connections -- the server does not need to close the connection after sending the response. The connection is kept up for a little while, and the client can make subsequent requests on it. e.g. the client can then start requesting the images. This is more efficient -- we don't have to do the whole TCP setup again and again. HTTP/1.0 had a primitive "keep-alive" feature to accomplish the same thing.

For simple web-robot type uses, HTTP/1.0 is fine.

A simple HTTP/1.1 connection...

```
[localhost:~] nick% telnet www.stanford.edu 80
Trying 171.64.14.237...
Connected to www.lb-a.stanford.edu.
Escape character is '^]'.
GET /class/cs193i/test/basic.html HTTP/1.1
Host: www.stanford.edu

HTTP/1.1 200 OK
Date: Mon, 30 Apr 2001 18:45:15 GMT
Server: Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 (Unix) mod_fastcgi/2.2.4
Last-Modified: Sat, 17 Mar 2001 00:17:11 GMT
ETag: "185f016a-15c-3ab2ad07"
Accept-Ranges: bytes
Content-Length: 348
Content-Type: text/html

<html>
<head>
<title>Basic</title>
...
## The connection is ready for another request
```

Here's an HTTP/1.1 example where the connection closes immediately as in HTTP/1.0

```
[localhost:~] nick% telnet www.stanford.edu 80
Trying 171.64.14.237...
Connected to www.lb-a.stanford.edu.
Escape character is '^]'.
GET /class/cs193i/test/basic.html HTTP/1.1
Host: www.stanford.edu
Connection: close

HTTP/1.1 200 OK
Date: Mon, 30 Apr 2001 19:10:19 GMT
Server: Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 (Unix) mod_fastcgi/2.2.4
Last-Modified: Sat, 17 Mar 2001 00:17:11 GMT
ETag: "185f016a-15c-3ab2ad07"
Accept-Ranges: bytes
Content-Length: 348
Connection: close
Content-Type: text/html

<html>
<head>
...
Connection closed by foreign host.
```