Working of HTTP

How does it work?

The basic principle is a client/server connection: a client (the web browser) connects to a server, sends a request and the server replies. The connection is a simple TCP/IP socket connection, generally on port 80, but we use 8080 from time to time (usually with web proxys).

HTTP is a (very) simple protocol. It is so simple that we, human, can directly speak with web servers. We just need to open a connection on a web server with a telnet client and to type HTTP commands by hand. Here is a short example (what we must type is written in bold characters; [CR] means that we must strike the return key to add a line break):

```
$ telnet www2.themanualpage.org 80[CR]
Trying...
Connected to web.pro.proxad.net.
Escape character is '^]'.
GET http://www2.themanualpage.org/http/hello.txt[CR]
Hello

Connection closed by foreign host.
```

By typing `GET http://www2.themanualpage.org/http/hello.txt` we ask for the /http/hello.txt document to the server called www2.themanualpage.org, and the server returned the content of this file. It is the very basement of HTTP: we make a request with simple words and the server replies immediately with simple words as well and sends the result of the request.

Internet Explorer, Netscape and other web browsers, want you ask for a web page using a URL, do exactly the same (but you do not see it).

The different HTTP versions

The latest HTTP version is **HTTP/1.1**. Before this version, there were 2 other versions: **HTTP/0.9** and **HTTP/1.0**. At the moment HTTP/1.0 is maybe the most commonly used version of the protocol. When we give the version of HTTP, we use "HTTP/" followed by the version of the protocol.

When we connect to a server, one must first tell it which version we use. By default, is nothing is specified, we use HTTP/0.9. That is what happened in the example above. When we ask for a version the server does not implement, it sends first the version of HTTP it is going to use in its reply. The client should then be able to understand this version. Example:

```
$ telnet www2.themanualpage.org 80
```

```
Trying...
Connected to web.pro.proxad.net.
Escape character is '^]'.
HEAD http://www2.themanualpage.org/http/hello.txt HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 14 Oct 1999 06:43:02 GMT
Server: Apache/1.3.6 LV/LM-1.3 (Unix) PHP/3.0.9
Last-Modified: Sat, 08 Feb 1997 15:55:03 GMT
ETag: "afc0-8-32fca1d7"
Accept-Ranges: bytes
Content-Length: 8
Connection: close
Content-Type: text/plain

Connection closed by foreign host.
```

We can notice that this time the reply is much more complicated than before, it contains more data than before. To send this extra data, the client as well the server use what we call some **directives**. For instance, "Content-Type:" is a directive that specifies the type of the returned resource (in the example, it is plain text).

Making a request

A request is what we ask the server. In the first example, it was a GET request used to get (!!) a document. In the second example, we made a HEAD request in order to get only the header part of a complete reply we should get with a GET request (we will see later what this HEAD request is used for).

the different requests

HTTP request are performed by what we call **methods**. Here is the list of all available methods, and the version number with which they appeared:

request	HTTP version	description
GET	HTTP/0.9	to get a document
HEAD	HTTP/1.0	to get the header part of a complete reply
POST	HTTP/1.0	to send data to the server
PUT	HTTP/1.1	to ask the server to save the resource we are sending
DELETE	HTTP/1.1	to delete a file on the server
TRACE	HTTP/1.1	used to check what request the server has recieved
CONNECT	HTTP/1.1	word reserved for proxys to create tunnels
OPTIONS	HTTP/1.1	list of options available for a given resource

The method used for the request is the very first thing that the client provides (first line).

Formatting a request

Requests have a specific format only since HTTP/1.0.

For HTTP/0.9, the only possible request is the one seen in the first example:

```
GET http://www2.themanualpage.org/http/hello.txt
```

For HTTP/1.0 and later, requests are composed by 2 different parts (if we don't take the first line into account, since it only contains the method used for the request): the **headers** and the **body** of the request (in a given request, the body of the request is also called the **entity's body**). The headers are used to complete the request with directives (to say who is requesting the resource, what kind of web browser you are using...). Strictly speaking, headers are indispensable but as there is no header in HTTP/0.9, they are not mandatory in HTTP/1.0. Conversely, there is rarely a body in a request; there is one only when we use the POST or PUT methods, that is to say when the web browser wants to send data.

To learn more about this, see the **HTTP/1.0 headers**.

Server's replies

In HTTP/0.9, the reply is really simple, since the server directly sends the answer (HTML page, image...).

On the contrary, since HTTP/1.0, the server's reply is similar to a request, as it is made of 2 parts (except the first line): the **headers** and the **body** of the response (this body may be called the **entity's body**, too). Headers contain information about the response and the entity, once again defined with directives. For instance, the server says if the request was perfectly understood and treated, specifies the format of the sent resource... The entity's body is actually the result of the request.

To read more about this, see **server's replies in HTTP/1.0**.

 **printable format**