

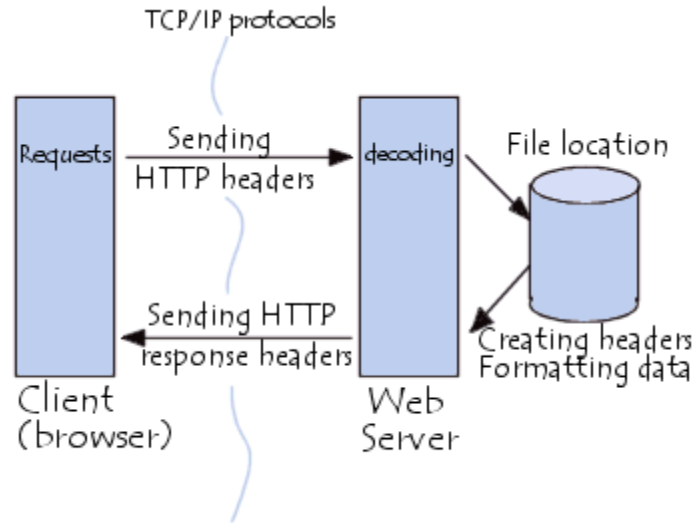
Introduction to the HTTP protocol

Since 1990 HTTP [protocol](#) (HyperText Transfer Protocol) has been the most widely used protocol on the Internet. Version 0.9 was only intended to transfer data over the Internet (in particular Web pages written in [HTML](#). Version 1.0 of the protocol (the most used) now allows the transfer of messages with headers describing the content of the message by using [MIME](#) type coding.

The aim of HTTP protocol is to allow transfer of files (essentially in HTML format) between a browser (the client) and a Web server (called among other things *httpd* on [UNIX](#) machines) located using a character string called a [URL](#).

Communication between browser and server

Communication between the browser and server takes place in two stages:



- The navigator makes a **HTTP request**
- The server processes the request then sends a **HTTP response**

In reality, the communication is conducted in more stages if you consider the processing of the request by the server. Given that we are only concerned with HTTP protocol, server side processing will not be explained as part of this article.

HTTP Request

A HTTP request is a collection of lines sent to the server by the browser. It includes:

- **A request line:** This is a line specifying the type of document requested, the method which must be applied, and the version of the protocol used. The line is made up of three elements which must be separated by a space:
 - The method
 - The URL
 - The version of the protocol used by the client (generally *HTTP/1.0*)
- **The request header fields:** This is a collection of optional lines allowing additional information about the request and/or the client to be given (browser, operating system, etc.). Each of these lines is composed of a name describing the header type, followed by a colon (:) and the value of the header

- **The body of the request:** This is a collection of optional lines which must be separated from preceding lines by an empty line and for example allowing data to be sent by a POST command during the sending of data to the server using a form

A HTTP request therefore has the following syntax (<crLf> meaning carriage return and line feed):

```
METHOD URL VERSION<crLf>
HEADER: Value<crLf>
.
.
.
HEADER: Value<crLf>
Empty line <crLf>
BODY OF THE REQUEST
```

Here is an example of a HTTP request:

```
GET http://en.kioskea.net/ HTTP/1.0
Accept: text/html
If-Modified-Since: Saturday, 15-January-2000 14:37:11 GMT
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 95)
```

Commands

Command	Description
GET	Request for the resource located at the specified URL
HEAD	Request for the header of the resource located at the specified URL
POST	Sends data to the program located at the specified URL
PUT	Sends data to the specified URL
DELETE	Deletes the resource located at the specified URL

Headers

Header name	Description
Accept	Type of content accepted by the browser (for example <i>text/html</i>). See MIME types
Accept-Charset	Character set expected by the browser
Accept-Encoding	Data coding accepted by the browser
Accept-Language	Language expected by the browser (English by default)
Authorization	Identification of the browser to the server
Content-Encoding	Type of coding for the body of the request

Content-Language	Type of language in the body of the request
Content-Length	Length of the body of the request:
Content-Type	Type of content of the body of the request (for example <i>text/html</i>). See MIME types
Date	Date data transfer starts.
Forwarded	Used by intermediary machines between the browser and server
From	Allows the client email address to be specified
From	Makes it possible to specify that the document must be sent if it has been modified since a certain date.
Link	Link between two URLs
Orig-URL	URL from which the request originated
Referer	Link URL from which the request has been made
User-Agent	String giving information about the client, such as the name and version of the browser and the operating system

HTTP Response

A HTTP response is a collection of lines sent to the server by the browser. It includes:

- **A status line:** this is a line specifying the protocol version used and the status of the request being processed using a code and explanatory text. The line is made up of three elements which must be separated by a space:
 - The version of the protocol used
 - The status code:
 - The meaning of the code
- **The response header fields:** This is a collection of optional lines allowing additional information about the response and/or the client to be given (browser, operating system, etc.). Each of these lines is composed of a name describing the header type, followed by a colon (:) and the value of the header
- **The body of the response:** contains the requested document

A HTTP response therefore has the following syntax (<crLf> meaning carriage return and line feed):

```
VERSION-HTTP CODE EXPLANATION<crLf>
HEADER: Value<crLf>
.
.
.
HEADER: Value<crLf>
Empty line <crLf>
BODY OF THE RESPONSE
```

Here is an example of a HTTP response:

```
HTTP/1.0 200 OK
Date: Sat, 15 Jan 2000 14:37:12 GMT
Server: Microsoft-IIS/2.0
Content-Type: text/HTML
Content-Length: 1245
Last-Modified: Fri, 14 Jan 2000 08:25:13 GMT
```

Response headers

Header name	Description
Content-Encoding	Type of coding for the body of the response
Content-Language	Type of language in the body of the response
Content-Length	Length of the body of the response
Content-Type	Type of content of the body of the response (for example <i>text/html</i>). See MIME types
Date	Date data transfer starts.
Expires	Data use by date
Forwarded	Used by intermediary machines between the browser and server
Location	Redirection to a new URL associated with the document
Server	Features of the server having sent the response

The response codes

These are the codes that you see when the browser cannot display the requested page. The response code is made up of three digits: the first indicates the status and the following two digits explain the exact nature of the error.

Code	Message	Description
10x	Information message	These codes are not used in version 1.0 of the protocol
20x	Success	These codes indicate the smooth running of the transaction
200	OK	The request has been accomplished correctly
201	CREATED	This follows a POST command and indicates success, the remaining body of the document indicates the URL where the newly created document must be located.
202	ACCEPTED	The request has been accepted, the procedure which follows has not been accomplished
203	PARTIAL	When this code is received in response to a GET command it indicates that the

	INFORMATION	response is not complete.
204	NO RESPONSE	The server has received the request by there is no information to send back
205	RESET CONTENT	The server tells the browser to delete the content in the fields of a form
206	PARTIAL CONTENT	This is a response to a request consisting of the header <i>range</i> . The server must indicate the header <i>content-Range</i>
30x	Redirection	These codes indicate that resource is no longer in the location specified
301	MOVED	The requested data has been transferred to a new address
302	FOUND	The requested data is at a new URL, but has however maybe been moved since...
303	METHOD	This means that the client must try a new address, preferably by trying another method to GET
304	NOT MODIFIED	If the client has carried out a conditional GET command (by requesting if the document has been modified since the last time) and the document has not been modified it sends back this code.
40x	Error due to the client	These codes indicate that the request is incorrect
400	BAD REQUEST	The syntax of the request is badly formulated or is impossible to satisfy
401	UNAUTHORIZED	The parameters of the message give specifications of unacceptable forms of authorisation. The client must reformulate its request with the correct authorisation data
402	PAYMENT REQUIRED	The client must reformulate its request with the correct payment data
403	FORBIDDEN	Access to the resource is quite simply denied
404	NOT FOUND	Classic! The server has not found anything at the specified address. Left without leaving a forwarding address.....)
50x	Error due to the server	These codes indicate that there is an internal error in the server
500	INTERNAL ERROR	The server has encountered an unexpected condition which prevented it from following up the request (just one of those things that happen to servers...)
501	NOT IMPLEMENTED	The server does not support the service requested (it cannot know everything...)
502	BAD GATEWAY	The server has received an invalid response from the server which it is trying to access by acting as a gateway or proxy
503	SERVICE	The server cannot respond to you at the present time since it is too busy (all

	UNAVAILABLE	your communication lines are busy, please try again later)
504	GATEWAY TIMEOUT	The response from the server has taken too long in relation to the time for which the gateway had been prepared to way (the time that was assigned to you has now passed...)