**HTTP**

The dream behind the Web is of a common information space in which we communicate by sharing information. Its universality is essential: the fact that a hypertext link can point to anything, be it personal, local or global, be it draft or highly polished.

Tim Berners-Lee, *The World Wide Web: A very short personal history*

The *Hypertext Transfer Protocol*, already mentioned in [Chapter 12](https://eloquentjavascript.net/2nd_edition/12_browser.html#web), is the mechanism through which data is requested and provided on the World Wide Web. This chapter describes the protocol in more detail and explains the way browser JavaScript has access to it.

**The protocol**

If you type *eloquentjavascript.net/17\_http.html* into your browser’s address bar, the browser first looks up the address of the server associated with *eloquentjavascript.net* and tries to open a TCP connection to it on port 80, the default port for HTTP traffic. If the server exists and accepts the connection, the browser sends something like this:

GET /17\_http.html HTTP/1.1

Host: eloquentjavascript.net

User-Agent: Your browser's name

Then the server responds, through that same connection.

HTTP/1.1 200 OK

Content-Length: 65585

Content-Type: text/html

Last-Modified: Wed, 09 Apr 2014 10:48:09 GMT

<!doctype html>

... the rest of the document

The browser then takes the part of the response after the blank line and displays it as an HTML document.

The information sent by the client is called the *request*. It starts with this line:

GET /17\_http.html HTTP/1.1

The first word is the *method* of the request. GET means that we want to *get* the specified resource. Other common methods are DELETE to delete a resource, PUT to replace it, and POST to send information to it. Note that the server is not obliged to carry out every request it gets. If you walk up to a random website and tell it to DELETE its main page, it’ll probably refuse.

The part after the method name is the path of the resource the request applies to. In the simplest case, a resource is simply a file on the server, but the protocol doesn’t require it to be. A resource may be anything that can be transferred *as if* it is a file. Many servers generate the responses they produce on the fly. For example, if you open [*twitter.com/marijnjh*](http://twitter.com/marijnjh), the server looks in its database for a user named *marijnjh*, and if it finds one, it will generate a profile page for that user.

After the resource path, the first line of the request mentions HTTP/1.1 to indicate the version of the HTTP protocol it is using.

The server’s response will start with a version as well, followed by the status of the response, first as a three-digit status code and then as a human-readable string.

HTTP/1.1 200 OK

Status codes starting with a 2 indicate that the request succeeded. Codes starting with 4 mean there was something wrong with the request. 404 is probably the most famous HTTP status code—it means that the resource that was requested could not be found. Codes that start with 5 mean an error happened on the server and the request is not to blame.

The first line of a request or response may be followed by any number of *headers*. These are lines in the form “name: value” that specify extra information about the request or response. These headers were part of the example response:

Content-Length: 65585

Content-Type: text/html

Last-Modified: Wed, 09 Apr 2014 10:48:09 GMT

This tells us the size and type of the response document. In this case, it is an HTML document of 65,585 bytes. It also tells us when that document was last modified.

For the most part, a client or server decides which headers to include in a request or response, though a few headers are required. For example, the Host header, which specifies the hostname, should be included in a request because a server might be serving multiple hostnames on a single IP address, and without that header, the server won’t know which host the client is trying to talk to.