HTTP (HyperText Transfer Protocol) is the underlying protocol of the World Wide Web. Developed by Tim Berners-Lee and his team between 1989-1991, HTTP has seen many changes, keeping most of the simplicity and further shaping its flexibility. HTTP has evolved from an early protocol to exchange files in a semi-trusted laboratory environment, to the modern maze of the Internet, now carrying images, videos in high resolution and 3D.

**HTTP/0.9 – The one-line protocol**

The initial version of HTTP had no version number; it has been later called 0.9 to differentiate it from the later versions. HTTP/0.9 is extremely simple: requests consist of a single line and start with the only possible method GET followed by the path to the resource (not the URL as both the protocol, server, and port are unnecessary once connected to the server).

**HTTP/1.0 – Building extensibility**

HTTP/0.9 was very limited and both browsers and servers quickly extended it to be more versatile:

* Versioning information is now sent within each request (HTTP/1.0 is appended to the GET line)
* A status code line is also sent at the beginning of the response, allowing the browser itself to understand the success or failure of the request and to adapt its behavior in consequence (like in updating or using its local cache in a specific way)
* The notion of HTTP headers has been introduced, both for the requests and the responses, allowing metadata to be transmitted and making the protocol extremely flexible and extensible.
* With the help of the new HTTP headers, the ability to transmit other documents than plain HTML files has been added (thanks to the Content-Type header).

**HTTP/1.1 – The standardized protocol**

In parallel to the somewhat chaotic use of the diverse implementations of HTTP/1.0, and since 1995, well before the publication of HTTP/1.0 document the next year, proper standardization was in progress. The first standardized version of HTTP, HTTP/1.1 was published in early 1997, only a few months after HTTP/1.0.

HTTP/1.1 clarified ambiguities and introduced numerous improvements:

* A connection can be reused, saving the time to reopen it numerous times to display the resources embedded into the single original document retrieved.
* Pipelining has been added, allowing to send a second request before the answer for the first one is fully transmitted, lowering the latency of the communication.
* Chunked responses are now also supported.
* Additional cache control mechanisms have been introduced.
* Content negotiation, including language, encoding, or type, has been introduced, and allows a client and a server to agree on the most adequate content to exchange.
* Thanks to the Host header, the ability to host different domains at the same IP address now allows server colocation.

Thanks to its extensibility – creating new headers or methods is easy – and even if the HTTP/1.1 protocol was refined over two revisions, RFC 2616 published in June 1999 and the series of RFC 7230-RFC 7235 published in June 2014 in prevision of the release of HTTP/2, this protocol has been extremely stable over more than 15 years.

**HTTP/2 – A protocol for greater performance**

Over the years, Web pages have become much more complex, even becoming applications in their own right. The amount of visual media displayed, the volume and size of scripts adding interactivity, has also increased: much more data is transmitted over significantly more HTTP requests. HTTP/1.1 connections need requests sent in the correct order. Theoretically, several parallel connections could be used (typically between 5 and 8), bringing considerable overhead and complexity. For example, HTTP pipelining has emerged as a resource burden in Web development.

In the first half of the 2010s, Google demonstrated an alternative way of exchanging data between client and server, by implementing an experimental protocol SPDY. This amassed interest from developers working on both browsers and servers. Defining an increase in responsiveness, and solving the problem of duplication of data transmitted, SPDY served as the foundations of the HTTP/2 protocol.

The HTTP/2 protocol has several prime differences from the HTTP/1.1 version:

* It is a binary protocol rather than text. It can no longer be read and created manually. Despite this hurdle, improved optimization techniques can now be implemented.
* It is a multiplexed protocol. Parallel requests can be handled over the same connection, removing the order and blocking constraints of the HTTP/1.x protocol.
* It compresses headers. As these are often similar among a set of requests, this removes duplication and overhead of data transmitted.
* It allows a server to populate data in a client cache, in advance of it being required, through a mechanism called the server push.

Officially standardized, in May 2015, HTTP/2 has had much success. By July 2016, 8.7% of all Web sites[1] were already using it, representing more than 68% of all requests[2]. High-traffic Web sites showed the most rapid adoption, saving considerably on data transfer overheads and subsequent budgets.

This rapid adoption rate was likely as HTTP/2 does not require adaptation of Web sites and applications: using HTTP/1.1 or HTTP/2 is transparent for them. Having an up-to-date server communicating with a recent browser is enough to enable its use: only a limited set of groups were needed to trigger adoption, and as legacy browser and server versions are renewed, usage has naturally increased, without further Web developer efforts.