**Basics of HTTP**

HTTP is an extensible protocol that relies on concepts like resources and Uniform Resource Identifiers (URIs), simple message structure, and client-server communication flow. On top of these basic concepts, numerous extensions have been developed over the years that add updated functionality and semantics with new HTTP methods or headers.

[**Articles**](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP#articles)

[Overview of HTTP](https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview)

Describes what HTTP is and its role in web architecture, including its position in the protocol stack.

[Evolution of HTTP](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Evolution_of_HTTP)

HTTP was created in the early 1990s and has been extended several times. This article goes through its history and describes HTTP/0.9, HTTP/1.0, HTTP/1.1, and the modern HTTP/2, as well as novelties introduced over the years.

[Resources and URIs](https://developer.mozilla.org/en-US/docs/Web/HTTP/Resources_and_URIs)

A brief introduction to the concept of resources, identifiers, and locations on the web.

[Identifying resources on the Web](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Identifying_resources_on_the_Web)

Describes how web resources are referenced and how to locate them.

[Data URLs](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URLs)

A specific kind of URI that directly embeds the resource it represents. Data URLs are very convenient, but have some caveats.

[Resource URLs](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Resource_URLs) Non-standard

Resource URLs, those prefixed with the resource scheme are used by Firefox and Firefox browser extensions to load resources internally, but is also available to some sites the browser connects to as well.

[MIME types](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types)

Since HTTP/1.0, different types of content can be transmitted. This article explains how this is accomplished using the [Content-Type](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type) header and the MIME standard.

[Choosing between www and non-www URLs](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Choosing_between_www_and_non-www_URLs)

This article provides guidance on how to choose whether to use a www-prefixed domain or not, along with the consequences of that choice.

[Flow of an HTTP session](https://developer.mozilla.org/en-US/docs/Web/HTTP/Session)

This article describes a typical HTTP session; i.e. what happens when you follow a link or load an image into a web page.

[HTTP Messages](https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages)

HTTP Messages transmitted during requests or responses have a very clear structure. This introductory article describes this structure, its purpose, and its possibilities.

[Frame and message structure in HTTP/2](https://developer.mozilla.org/en-US/docs/Web/HTTP/Frame_and_message_structure_in_HTTP_2)

HTTP/2 encapsulates and represents HTTP/1.x messages in a binary frame. This article explains the frame structure, its purpose, and the way it's encoded.

[Connection management in HTTP/1.x](https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection_management_in_HTTP_1.x)

HTTP/1.1 was the first version of HTTP to support persistent connection and pipelining. This article explains both concepts.

[Connection management in HTTP/2](https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection_management_in_HTTP_2)

HTTP/2 completely revisited how connections are created and maintained. This article explains how HTTP frames allow multiplexing and solve the 'head-of-line' blocking problem of former HTTP versions.

[Content Negotiation](https://developer.mozilla.org/en-US/docs/Web/HTTP/Content_negotiation)

HTTP introduces a set of headers, starting with [Accept](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Accept) as a way for a browser to announce the format, language, or encoding it prefers. This article explains how this advertisement happens, how the server is expected to react, and how it chooses the most adequate response.