HTTP

The Hypertext Transfer Protocol (HTTP), which facilitates communication between clients and servers for online browsing, is essential. HTTP has changed over time to meet the increasing needs of contemporary online applications. In the field of web development, HTTP/1.1 and HTTP/2 are two key versions of the protocol. In this article, we'll examine the main distinctions between HTTP/1.1 and HTTP/2 to help you see how the later builds on the former.

The web has been powered by HTTP/1.1, the most popular version of HTTP up until recently. It is a text-based protocol that uses a request-response model of communication.

HTTP/1.1 provides a significant amount of redundant header information with each request, including cookies and user agents. The size of the requests as a whole grows due to the redundancy, which also adds needless overhead.Without a request, the server cannot push resources to the client in HTTP/1.1.

The Next Generation HTTP/2 standard was introduced in 2015 with the goal of overcoming the restrictions of HTTP/1.1 and improving online performance.

Multiplexing: Multiplexing is one of HTTP/2's key advancements. HTTP/2 enables the simultaneous sending of numerous requests and responses over a single connection as opposed to using multiple connections. As a result, head-of-line blocking is no longer necessary, and web page loading times are much enhanced.

HTTP/2 uses the HPACK compression technique, which is more effective at compressing header data. Performance is enhanced as a result of reducing redundancy and requests' size.

Server Push: Server Push is a feature of HTTP/2 that enables the server to transmit resources to the client in advance of those resources being requested. As a result, page rendering can happen more quickly since the server can push essential resources like stylesheets, scripts, and graphics without having to wait for individual requests.

Objects in JavaScript

JavaScript relies on objects to store and manipulate data. In order to design effective code, it is essential to understand how things are represented inside. We'll examine the internal representation of objects in JavaScript in this blog article.

Internal Object Representation: JavaScript engines employ a variety of methods to effectively represent objects. Here is a quick summary:

Objects are made up of key-value pairs called properties and methods, which are the building blocks of objects. They are kept in internal fields or slots.

Objects are given hidden classes or shapes that maintain the object's layout while optimising property access.

Each property has a descriptor that defines its attributes, including writability and enumerability.

Garbage collection: Algorithms for garbage collection are used by JavaScript engines to manage memory.