## **Write a blog on Difference between HTTP1.1 vs HTTP2**

Hypertext Transfer Protocol(HTTP) is the protocol that allows communication between web servers and web browser/clients. The HTTP 1.2 has been developed to improve performance of loading webpages and this blog will state the key differences between the two versions.

Firstly, HTTP 1.2 outperforms its predecessor in loading webpages due to the method of delivery of the HTML content. In HTTP 1.1, the request and response of the HTML content is transmitted using a text-based protocol that transmits messages via Transmission Control Protocol (TCP) connections between the client and web server. The limitation of this is that it will lead to Head-of-line (HOL) blocking which may be mitigated by additional TCP connections. However, this leads to additional overhead and latency resulting in a longer load time of the webpage. Conversely, HTTP 1.2 encodes the messages to binary which enables data to be transmitted via stream using multiplexing with a single TCP connection. This leads to minimal resources consumed in establishing the TCP and prevents HOL blocking. Therefore, the load of websites is faster using HTTP 1.2 than 1.1.

Secondly, HTTP 1.2 introduces a feature called stream prioritization to counter the issues that can be caused by multiplexing mentioned earlier. Stream prioritization is the idea of assigning a specific order to the multiple data streams flowing between the client and web server. This helps developers to prioritize what type of content should be rendered first on the website and flexibility to change this priority by changing the weights which are used in prioritization of the streams. This helps to improve the user experience on the website and provides flexibility to developers.

Thirdly, HTTP 1.1 avoids buffer overflow by using the transport layer TCP whereby the size of the window/buffer is determined at both the client and server end. The space availability is conveyed between them using acknowledgement packets (ack), when the ack is zero it means the buffer is full. Each TCP connection used in HTTP 1.1 uses their very own control flow independent of each. On the other hand, HTTP allows the client and server to set their own flow control mechanisms at the application layer. This means that they can communicate the available buffer space and adjust the receive window for each stream. Hence, developers have more control over the data utilization on the streams and have more options when coming up with a strategy for the web page rendering.

Fourthly, HTTP 1.1 differs from HTTP 1.2 in terms of how subsequence requests are required to render a page. HTTP 1.1 adopts resource inlining whereby additional resources are embedded within the HTML document that will be needed in addition to the initial GET request. This works well if the additional resources are small in size but can slow down the loading of pages if they are larger. Furthermore, the client does not have control over the additional resource that is contained within the HTML in terms of rejecting them or caching them for faster retrieval. HTTP 1.2 has a method called server push which enables multiple responses to the initial GET request, thus the additional resources no longer need to be part of HTML. It works by the server sending a PUSH\_PROMISE frame containing headers of the message, to the client informing it about the future responses. This provides greater control over the response files such as caching them or rejecting the response using RST\_STREAM frame and reducing duplicate requests to server, thus reducing overheads and speeding up the rendering of the page. It is important to note that the server push needs to be handled with care, as some browsers may not be able to reject the requests and there will be duplicate requests made.

Lastly, the headers are handled differently in the two HTTP versions. In HTTP 1.1, headers cannot be compressed thus it might be still bulky at times leading to poor performance. Contrary to this, HTTP 1.2 uses HPACK compression to reduce the size of headers, resulting in improved efficiency and faster load time. Furthermore, HPACK is able to keep track of previously sent metadata fields and further compress them according to a dynamically altered index shared between the client and the server. Thereby, further reducing the size and improve the load time.

Reference: <https://www.digitalocean.com/community/tutorials/http-1-1-vs-http-2-what-s-the-difference>

## **Write a blog about objects and its internal representation in Javascript**

Objects are complex data types that are distinct from the primitive data types like strings and integers. It allows us to store and manage various types of data in a structured way that allows us to represent complex relationships between the data or entities.

In Javascript, objects can be created using two methods. The first way is using curly braces {} as hash-map that contain key-value pairs, where the key is a string and the value can be any data type. The second method is using the constructor with the *new* keyword with the object constructor. The objects are passed by reference in Javascript meaning they can be modified within a function as the pointer points to the original memory address of the instance of object.

The created objects inherit properties and methods from the Object.prototype template. This means that the features of the object are shared across all instances of the object.This leads to more efficient and maintainable code by promoting code reusability. Furthermore, the properties of the object can be easily retrieved and modified.

<https://copyprogramming.com/howto/objects-and-its-internal-representation-in-javascript>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object>