**Difference between HTTP/1.1 and HTTP/2**

**Introduction:**

HTTP stands for **Hypertext Transfer Protocol** which is being used as a standard for client-server communication on the World Wide Web. There are several versions of HTTP, but here we are interested in analysing the difference between HTTP/1.1 which was released in 1997 and HTTP/2 which was released in 2015.

**Delivery Models:**

In both of these protocols, the requests and responses between the client and server are made. But HTTP/1.1 transfers these in plain-text messages, while HTTP/2 converts all the messages into binary format and does it in a binary framing layer.

**HTTP/1.1 - Pipelining and Head-of-Line Blocking:**

HTTP/1.1 uses persistent connections and pipelining mechanism. With persistent connections, HTTP/1.1 uses a single TCP connection and allows the client to send multiple requests along the same connection without waiting for a response to each request, which is known as pipe lining. But there is a possibility of request at the head of the queue being unable to retrieve its required resource, in such case, it will block all the requests behind it. This is known as head-of-line (HOL) blocking.

**HTTP/2:**

* **Binary Framing Layer:**

In HTTP/2, the binary framing layer converts requests/responses into binary-code messages. HTTP/2 uses a single TCP connection within which there are multiple streams consisting of multiple messages and each of these messages split into smaller units called frames. This process of cutting into smaller packets increases the flexibility of data transfer.

* **Multiplexing:**

In HTTP/2, when a [client](https://www.cloudflare.com/learning/serverless/glossary/client-side-vs-server-side/) makes a request for a webpage, the server sends several streams of data to the client at once, instead of sending one by one. This method of data delivery is known as multiplexing. Multiplexing resolves the head-of-line blocking issue in HTTP/1.1 by ensuring that no message has to wait for another to finish.

The multiple streams awaiting the same resource can still cause performance issues which is solved by using stream prioritization.

* **Stream Prioritization:**

In HTTP/2, the binary framing layer organizes messages into parallel streams of data and assigns a weight between 1 and 256 to each stream. The resource allocation for each stream will be based on the weight value. We can set a lower value of weight for loading a high resolution image.

**Predicting Resource Requests:**

**HTTP/1.1 - Resource Inlining:**

In HTTP/1.1, if a client needs additional resources to render the page, then including the required resource directly within the HTML document which the server sends in response to the initial GET request is called as resource inlining thus reducing the total number of requests that the client must send.

**HTTP/2 - Server Push:**

In HTTP/2, a server sends a resource to a client along with the requested HTML page, thus providing the resource before the client asks for it. This process is called server push. In this way, an HTTP/2 connection can achieve the same goal of resource inlining while maintaining the separation between the pushed resource (required resource) and the document (HTML page). In this way, the client can decide to cache or decline the pushed resource separate from the main HTML document, thus fixing the major drawback of resource inlining.

**Compression:**

**HTTP/2 - Header Compression:**

HTTP/2 can split headers from their data, resulting in a header frame and a data frame, and uses an advanced compression method called HPACK to compress the header frame and make them smaller. As small files load more quickly than large ones, this header compression helps resulting in faster loading.

**Conclusion:**

An estimates states that around one-third of all websites in the world supports HTTP/2. HTTP/2 differs from HTTP/1.1 in many ways, with some features which are used to better optimize web application performance.