**HTTP/1.1 and HTTP/2**

Introduction ;

HTTP stands for **Hypertext Transfer Protocol**, and it is the basis for almost all web applications. HTTP is the method computers and servers use to request and send information. For instance, when someone navigates to Guvi.com on their laptop, their web browser sends an HTTP request to the Guvi servers for the content that appears on the page. Then, Guvi servers send HTTP responses with the text, images, and formatting that the browser displays to the user.

**HTTP / 1.1 ;**

Developed by Timothy Berners-Lee in 1989 as a communication standard for the (WWW) World Wide Web, HTTP is a top-level application protocol that exchanges information between a client computer and a local or remote web server. In this process, a client sends a text-based request to a server by calling a method like GET or POST. In response, the server sends a resource like an HTML page back to the client.

# 

## Fig.1

**HTTP / 2 ;**

 HTTP/2 was developed over the SPDY protocol. HTTP/2 works on the **binary framing** layer instead of textual that converts all the messages in binary format. it works on fully multiplexed that is one TCP connection is used for multiple requests. HTTP/2 uses **HPACK** which is used to split data from header. it compresses the header. The server sends all the other files like CSS & JS without the request of the client using the **PUSH** frame.

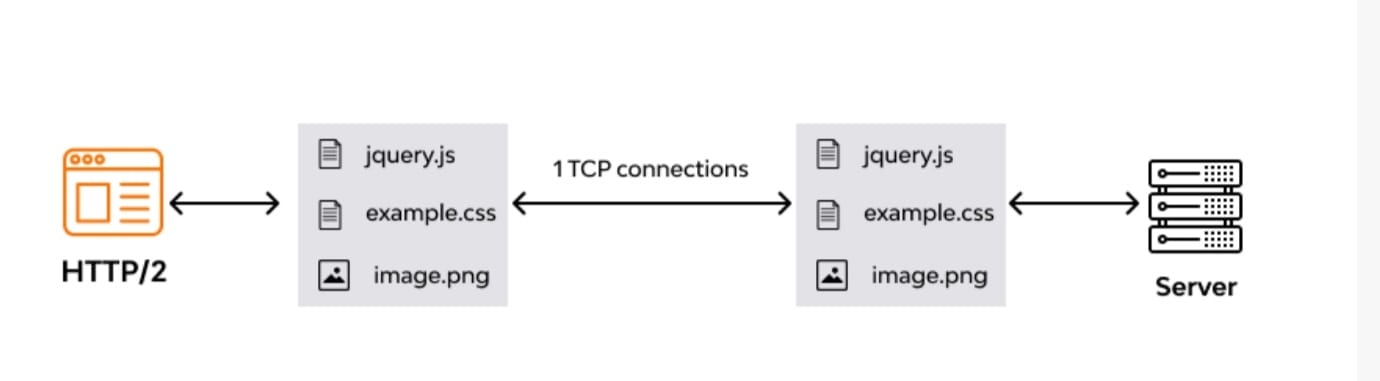


Fig.2

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

|  |  |  |
| --- | --- | --- |
| Difference | HTTP/1.1 | HTTP/2 |
| Key Features | * It supports connection reuse i.e. for every TCP connection there could be multiple requests and responses, and pipelining where the client can request several resources from the server at once. However, pipelining was hard to implement due to issues such as head-of-line blocking and was not a feasible solution | * Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. It is done using streams which can be prioritized, can have dependencies and individual flow control. It also provides a feature called server push that allows the server to send data that the client will need but has not yet requested. |
| Status Code | * Introduces a warning header field to carry additional information about the status of a message. Can define 24 status codes, error reporting is quicker and more efficient. | * Underlying semantics of HTTP such as headers, status codes remains the same. |
| **Authentication Mechanism** | * It is relatively secure since it uses digest authentication, NTLM authentication | * Security concerns from previous versions will continue to be seen in HTTP/2. However, it is better equipped to deal with them due to new TLS features like connection error of type Inadequate\_Security. |
| Caching | * Expands on the caching support by using additional headers like cache-control, conditional headers like If-Match and by using entity tags | * HTTP/2 does not change much in terms of caching. With the server push feature if the client finds the resources are already present in the cache, it can cancel the pushed stream |
| Web traffic | * HTTP/1.1 provides faster delivery of web pages and reduces web traffic as compared to HTTP/1.0. However, TCP starts slowly and with domain sharding (resources can be downloaded simultaneously by using multiple domains), connection reuse and pipelining, there is an increased risk of network congestion. | * HTTP/2 utilizes multiplexing and server push to effectively reduce the page load time by a greater margin along with being less sensitive to network delays |

Thank You