Write a blog on Difference between HTTP1.1 vs HTTP2

|  |  |
| --- | --- |
| HTTP1.1 | HTTP2 |
| * What is HTTP1.1 : * The first usable version of HTTP was created in 1997. Because it went through several stages of development, this first version of HTTP was called HTTP/1.1. This version is still in use on the web. | * What is HTTP2 * In 2015, a new version of HTTP called HTTP/2 was created. HTTP/2 solves several problems that the creators of HTTP/1.1 did not anticipate. In particular, HTTP/2 is much faster and more efficient than HTTP/1.1. One of the ways in which HTTP/2 is faster is in how it prioritizes content during the loading process. |
| * **Multiplexing:** * HTTP/1.1 loads resources one after the other, so if one resource cannot be loaded, it blocks all the other resources behind it | * **Multiplexing:** * . In contrast, HTTP/2 is able to use a single [TCP](https://www.cloudflare.com/learning/ddos/glossary/tcp-ip/) connection to send multiple streams of data at once so that no one resource blocks any other resource. * HTTP/2 does this by splitting data into binary-code messages and numbering these messages so that the client knows which stream each binary message belongs to. |
| * **Server push:** * Small files load more quickly than large ones. To speed up web performance, both HTTP/1.1 | * **Server push:** * and HTTP/2 compress HTTP messages to make them smaller. However, HTTP/2 uses a more advanced compression method called HPACK that eliminates redundant information in HTTP header packets. |

|  |  |
| --- | --- |
| * It works on the textual format. | * It works on the binary protocol |
| * There is head of line blocking that blocks all the requests behind it until it doesn’t get its all resources. | * It allows multiplexing so one TCP connection is required for multiple requests. |
| * It uses requests resource Inlining for use getting multiple pages | * It uses PUSH frame by server that collects all multiple pages |
| * It compresses data by itself. | * It uses HPACK for data compression |