1. **HTTP/1.1 vs. HTTP/2 Protocol**

* **Binary protocols** – Binary protocols consume less bandwidth, are more efficiently parsed and are less error-prone than the textual protocols used by HTTP/1.1. Additionally, they can better handle elements such as whitespace, capitalization and line endings.
* **Multiplexing** – HTTP/2 is multiplexed, i.e., it can initiate multiple requests in parallel over a single TCP connection. As a result, web pages containing several elements are delivered over one TCP connection. These capabilities solve the head-of-line blocking problem in HTTP/1.1, in which a packet at the front of the line blocks others from being transmitted.
* **Header compression** – HTTP/2 uses header compression to reduce the overhead caused by TCP’s [slow-start](https://en.wikipedia.org/wiki/TCP_congestion_control#Slow_start) mechanism.
* **Server push** – HTTP/2 servers push likely-to-be-used resources into a browser’s cache, even before they’re requested. This allows browsers to display content without additional request cycles.
* **Increased security** – Web browsers only support HTTP/2 via encrypted connections, increasing user and application security.

1. **HTTP version history**

Invented by Tim Berners-Lee in 1989 -1991, **HTTP functions as a request–response protocol in the client–server computing model.**

HTTP has four versions —

1) HTTP/0.9

2) HTTP/1.0

3) HTTP/1.1

4) HTTP/2.0.

HTTP/0.9 — The One-line Protocol

* Initial version of HTTP — a simple client-server, request-response, telenet-friendly protocol
* Request nature: single-line (method + path for requested document)
* Methods supported: GET only
* Response type: hypertext only
* Connection nature: terminated immediately after the response
* No HTTP headers (cannot transfer other content type files), No status/error codes, No URLs, No versioning

HTTP/1.0 — Building extensibility

* Browser-friendly protocol
* Provided header fields including rich metadata about both request and response (HTTP version number, status code, content type)
* Response: not limited to hypertext (Content-Type header provided ability to transmit files other than plain HTML files — e.g. scripts, stylesheets, media)
* Methods supported: GET , HEAD , POST
* Connection nature: terminated immediately after the response

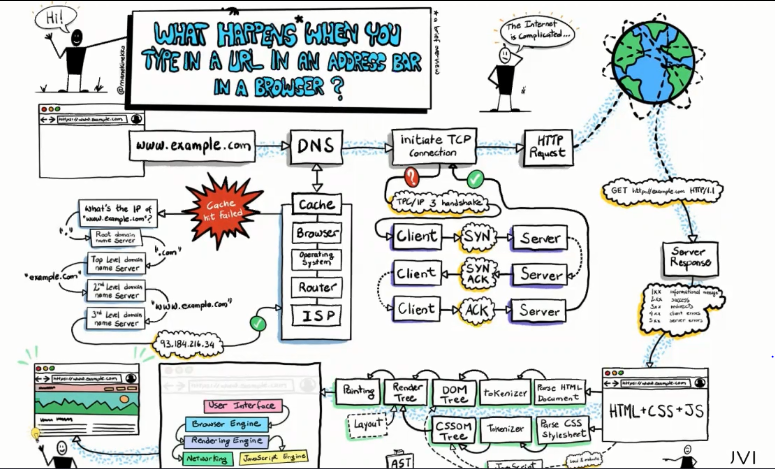
# HTTP/1.1 — The standardized protocol

* This is the HTTP version currently in common use.
* Introduced critical performance optimizations and feature enhancements — persistent and pipelined connections, chunked transfers, compression/decompression, content negotiations, virtual hosting (a server with a single IP Address hosting multiple domains), faster response and great bandwidth savings by adding cache support.
* Methods supported: GET , HEAD , POST , PUT , DELETE , TRACE , OPTIONS
* Connection nature: long-lived

**3) Difference between Javascript vs Node JS**

| **S.No** | **Javascript** | **NodeJS** |
| --- | --- | --- |
| 1. | Javascript is a programming language that is used for writing scripts on the website. | NodeJS is a Javascript runtime environment. |
| 2. | Javascript can only be run in the browsers. | NodeJS code can be run outside the browser. |
| 3. | It is basically used on the client-side. | It is mostly used on the server-side. |
| 4. | Javascript is capable enough to add HTML and play with the DOM. | Nodejs does not have capability to add HTML tags. |
| 5. | Javascript is used in frontend development. | Nodejs is used in server-side development. |

* 1. **) What happens when you type URL in browser?**

****

* When we type url ([www.example.com](http://www.example.com)) in browser , this hostname is converted to IP address with the help of DNS entity in a process called **FORWARD LOOKUP**.
* Based on the protocol HTTP issued , the request is sent to server to get the response back to the web browser . Based on the protocol used ,it looks for particular port in the web server (HTTP -80 & HTTPS -443) and looks for the index.html to get the resources .
* Once the responses reaches back to the browser , there with help of Rendering engine (will have parsed items ) & javascript (converts js file to 0’s & 1’s) will have the layout tree .Coloring of the components is taken care by UI backend and finally it is displayed on the browser screen .