1. **Difference between HTTP1.1 vs HTTP2 ?**

|  |  |  |
| --- | --- | --- |
|  | **HTTP1.1** | **HTTP2** |
| * **Multiplexing** | HTTP1.1 loads resources one after the other,so if one resource cannot be loaded, it blocks all the other resources behind it. | HTTP2 is able to use a single TCP connection to send multiple streams of data at once so that no one resource blocks any other resource. |
| * **Header compression** | Headers are sent on every request leading to a lot of duplicate data being sent uncompressed across the wire. | HTTP2 uses a more advanced compression method called HPACK that eliminates redundant information in HTTP header packets. |
| * **Prioritization** | In HTTP1.1 prioritization is fixed since the page resources load one after the other | HTTP/2 offers a feature called weighted prioritization.  This allows developers to decide which page resources will load first, every time. |
| * **Connectivity** | 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. | Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. |
| * **Authentication** | It is relatively secure since it uses digest authentication | It is better equipped to deal with new TLS features like connection error of type - Inadequate Security. |
| * **Protocol Type** | Text based protocol that is in the readable form. | It is a binary protocol (HTTP requests are sent in the form of 0s and 1s).Needs to be converted back from binary in order to read it. |

**2. HTTP version history ?**

> **HTTP** (HyperText Transfer Protocol) is the underlying protocol of the World Wide Web. Developed by Tim Berners-Lee and his team between 1989-1991.

> Built over the existing TCP and IP protocols, it consists of 4 building blocks:

* A textual format to represent hypertext documents, the *[HyperText Markup Language](https://developer.mozilla.org/en-US/docs/Web/HTML)* (HTML).
* A simple protocol to exchange these documents, the *HypertText Transfer Protocol*(HTTP).
* A client to display (and accidentally edit) these documents, the first Web browser called *WorldWideWeb*.
* A server to give access to the document, an early version of *httpd*.

## > Basic Architecture

The following diagram shows a very basic architecture of a web application and depicts where HTTP sits:



The HTTP protocol is a request/response protocol based on the client/server based architecture where web browsers, robots and search engines, etc. act like HTTP clients, and the Web server acts as a server.

**3.List 5 difference between Browser js console vs Node js ?**

|  |  |  |
| --- | --- | --- |
|  | **Browser js** | **Node js** |

|  |  |  |
| --- | --- | --- |
| **WINDOW** | It is predefined global object which has functions and attributes, that have to deal with window that has been drawn. | Node doesn’t have a predefined “window” object cause it doesn’t have a window to draw anything. |
| **LOCATION** | It is predefined object in browsers, that has all the information about the url we have loaded. | “location” object is related to a particular url because it is for page specific , node does not require. |
| **DOCUMENT** | It is predefined global variable in browsers, has the html which is rendered. | Node doesn’t have “document” object because it never has to render anything in a page. |
| **GLOBAL** | Browsers may have an object named “global”, but it will be the exact one as “window”. | Node has “global”, which is a predefined global object. It contains several functions that are not available in browsers, cause they are needed for server side works only. |
| **REQUIRE** | Browsers don’t have “require” predefined. You may include it in your app for asynchronous file loading. | “require” object is predefined in Node which is used to include modules in the app |

**4.What happens when you type a URL in address bar in the browser ?**

**https://www.ilovemath.org/**

**Note :** DNS is short for Domain Name System. Like a phonebook, DNS maintains and maps the name of the website, i.e. URL, and particular IP address it links to. Every URL on the internet has a unique IP address which is of the computer which hosts the server of the website requested.

* Browser checks cache for DNS entry to find the corresponding [IP address](https://www.geeksforgeeks.org/introduction-of-classful-ip-addressing/) of website.

Example : Browser cache , router cache

* If not found in cache, ISP’s (Internet Service Provider) DNS server initiates a DNS query to find IP address of server that hosts the domain name.  
  The requests are sent using small data packets that contain information content of request and IP address it is destined for.
* Browser initiates a [TCP (Transfer Control Protocol)](https://www.geeksforgeeks.org/tcp-and-udp-in-transport-layer/) connection with the server using synchronize and acknowledge messages.
* Browser sends an [HTTP](https://www.geeksforgeeks.org/http-non-persistent-persistent-connection/) request to the web server. GET or POST request.
* Server on the host computer handles that request and sends back a response. It assembles a response in some format like JSON, [XML](https://www.geeksforgeeks.org/xml-basics/) and HTML.
* Server sends out an HTTP response along with the status of response.
* Browser displays [HTML](https://www.geeksforgeeks.org/html-tutorials/) content