Difference between HTTP1.1 vs HTTP2.

Http is an application level protocol which is used to communicate between the client and the server. Http2 is the newest version of Http. Http2 aims to improve performance and reducing page loading latency.

Disadvantages of http1.1 – Modern day applications requires huge amounts of data to be transferred between the client and the server which showcased the shortcomings of Http1.1. Here's some of them.

One open request per connection –

Http allows only one request per connection. That means whenever a request for a resource is made a whole new connection is to be made. Although features such as keep-alive fixes this, it causes problems such as Head-of-line Blocking which is a bottleneck for faster loading application.

Header data Duplication –

We know that every time we request something or get a response back from the server, lots of redundant data such as Cookies and other headers. While the data received from the server can be compressed using gzip, the same is not possible for headers in http1.1.

Buffer Overflow –

In http1.1, flow control relies on the underlying TCP connection. In the both the client and the server establish their buffer sizes. If the client has its buffer full at the time of transmission, then the server must wait for the client to send a signal. We should also remember since we use TCP connection, we need separate flow control mechanism for each new TCP connection.

Predicting Resource Requests –

A typical webpage consists pf various resources such as Javascript , CSS, images, etc. which have to be downloaded only after the first GET request. These additional requests ultimately increase connection load time.

How Http2 solves above issues – Instead of using plain text to send and receive data http2 stores and sends its requests in binary format. This allows for different techniques to overcome the draw backs of http1.1.

Multiplexing –

Http2 establishes a single connection object between the two systems. The Binary Framing Layer chops the request into multiple streams of similar request/response format. Theses streams are further chopped into frames. These frames are tagged to a stream which allows for out of order transfer thus no packet of data must wait for another packet. Multiplexing allows for multiple streams at a time with a single TCP connection.

Compression –

Unlike Http1.1, Http2 can compress headers leveraging the fact that it stores the requests in binary encoded format. Http2 can split header from their data resulting into header frame and data frame. Http2 uses specific compression program called HPACK for header compression. It uses Huffman Coding to drastically reduce the size of the header requests.

Server Push –

Since HTTP/2 enables multiple concurrent responses to a client's initial `GET` request, a server can send a resource to a client along with the requested HTML page, providing the resource before the client asks for it. This process is called *server push*. In this way, an HTTP/2 connection can accomplish the same goal of resource in-lining while maintaining the separation between the pushed resource and the document. This means that the client can decide to cache or decline the pushed resource separate from the main HTML document, fixing the major drawback of resource in-lining.

Http version history

In 1989, while he was working at CERN, Tim Berners-Lee wrote a proposal to build a hypertext system over the Internet. Initially calling it the *Mesh*, it was later renamed to *World Wide Web* during its implementation in 1990.

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

List 5 difference between Browser JS(console) vs Nodejs

S.NO	BROWSER JAVASCRIPT	NODEJS
1.	Programming language that is used for writing scripts on the website.	NodeJS is a JavaScript runtime environment.
2.	Cannot manipulate File Systems	NodeJS code can manipulate file systems.
3.	It is basically used on the client-side scripting	It is used on the server-side scripting
4.	Consists of objects to interact with the browser.(Example- Window, Document)	Nodejs does not have capability to manipulate contents of the browser.
5.	JavaScript can run in any browser engine as like Spider-Monkey in Firefox, Chakra on IE.	Nodejs can only run in V8 engine of google chrome.

what happens when you type a URL in the address bar in the browser?

When we type a URL in the address bar of a browser the following happens: -

- The browser looks up the IP address of the URL in the DNS.
- Then sends an HTTP request to the corresponding server.
- The server sends back an http response back to the browser(client).
- The browser begins rendering HTML.
- The browser requests for additional files such as images, css, javascript.