1. Difference between HTTP1.1 vs HTTP2

HTTP1.1 and HTTP2 are different versions of the Hypertext Transfer Protocol (HTTP), the protocol used for transmitting and receiving web data between a client (like a web browser) and a server.

* HTTP1.1: It's a request-response protocol where only one request can be outstanding on a connection at a time. Multiple requests need to wait in line, leading to potential delays due to head-of-line blocking.
* HTTP2: Supports multiplexing, allowing multiple requests and responses to be sent and received in parallel over a single connection. This significantly improves efficiency and reduces latency, as it eliminates head-of-line blocking.
* HTTP1.1: Text-based protocol where data is sent in plain text. This can lead to inefficiencies due to the verbosity of headers and text-based nature.
* HTTP2: Uses a binary protocol for framing, which reduces overhead and improves parsing efficiency. Headers are compressed, reducing the amount of data transmitted.
* HTTP1.1: Headers are not compressed, leading to redundant data being sent with each request.
* HTTP2: Implements header compression, reducing overhead by compressing headers, thus reducing the amount of data transmitted and improving performance.

Both HTTP1.1 and HTTP2 can use the secure version, HTTPS, providing encryption and authentication.

HTTP2 improves upon several limitations of HTTP1.1

resulting in better performance, reduced latency, and improved efficiency in handling web traffic.

1. objects and its internal representation in Javascript:

In JavaScript, objects are fundamental data structures that represent a collection of key-value pairs. They are used to store various data and functionalities. Internally, how objects are represented and handled can vary based on the JavaScript engine

* + Objects consist of properties (data associated with the object) and methods (functions associated with the object). These properties and methods are stored within the object's internal structure.
  + Each object in JavaScript is linked to a prototype object from which it inherits properties and methods. This forms a prototype chain, allowing objects to inherit and share functionalities.
  + In JavaScript, objects are reference types, meaning they are stored and passed by reference. When assigning an object to a variable, you're storing a reference to the object in memory rather than the object itself.

JavaScript objects are dynamic, allowing properties to be added, modified, or deleted at runtime. This flexibility is a fundamental aspect of JavaScript's object model.