Write a blog on Difference between HTTP1.1 vs HTTP2?

HTTP1.1:

\* In HTTP/1.1, flow control relies on the underlying TCP connection.

\* When this connection initiates, both client and server establish their buffer sizes using their system default settings.

\* If the receiver’s buffer is partially filled with data, it will tell the sender its receive window, i.e., the amount of available space that remains in its buffer.

\* This receive window is advertised in a signal known as an ACK packet, which is the data packet that the receiver sends to acknowledge that it received the opening signal.

\* If this advertised receive window size is zero, the sender will send no more data until the client clears its internal buffer and then requests to resume data transmission.

\* It is important to note here that using receive windows based on the underlying TCP connection can only implement flow control on either end of the connection.

HTTP.2:

\* HTTP/2 multiplexes streams of data within a single TCP connection.

\* As a result, receive windows on the level of the TCP connection are not sufficient to regulate the delivery of individual streams.

\* HTTP/2 solves this problem by allowing the client and server to implement their own flow controls, rather than relying on the transport layer.

\* The application layer communicates the available buffer space, allowing the client and server to set the receive window on the level of the multiplexed streams.

\* This fine-scale flow control can be modified or maintained after the initial connection via a i.e WINDOW\_UPDATE frame.

Write a blog about objects and its internal representation in Javascript

* Objects, in JavaScript, is it’s most important data-type and forms the building blocks for modern JavaScript.
* These objects are quite different from JavaScript’s primitive data-types(Number, String, Boolean, null, undefined and symbol) in the sense that while these primitive data-types all store a single value each (depending on their types).
* Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.
* An object, is a reference data type. Variables that are assigned a reference value are given a reference or a pointer to that value.
* That reference or pointer points to the location in memory where the object is stored. The variables don’t actually store the value.