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

**HTTP1.1:**

\*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.

\*However, pipelining was hard to implement due to issues such as head-of-line blocking and was not a feasible solution.

\*Introduces a warning header field to carry additional information about the status of a message. Can define 24 status codes, error reporting is quicker and more efficient.

\*It is relatively secure since it uses digest authentication, NTLM authentication.

\*Expands on the caching support by using additional headers like cache-control, conditional headers like If-Match and by using entity tags.

**HTTP2:**

\*Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time.

\*It is done using streams which can be prioritized, can have dependencies and individual flow control.

\*It also provides a feature called server push that allows the server to send data that the client will need but has not yet requested.

\*Underlying semantics of HTTP such as headers, status codes remains the same.

\*Security concerns from previous versions will continue to be seen in HTTP/2. However, it is better equipped to deal with them due to new TLS features like connection error of type Inadequate Security.

**2. Objects and its internal representation in Javascript**

Objects are important data types in javascript. Objects are different than primitive datatypes (i.e. number, string, boolean, etc.). Primitive data types contain one value but Objects can hold many values in form of Key: value pair. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.

Every object has some property associated with some value. These values can be accessed using these properties associated with them.

var myCar = new Object();

myCar.make = 'Suzuki';

myCar.model = 'Altros';

myCar.year = 1978;

myCar.wheels = 2;

After creating myCar object, the value inside the object can be accessed using keys.

i.e.

myCar.year

Output: 1978

These values can be accessed using brackets notation also.

myCar.year

Output: 1978.