**HTTP**

1. Hypertext Transfer Protocol (HTTP) is an application protocol that is, currently, **the foundation** of data communication for the World Wide Web.
2. **HTTP is based on** the Client/Server model. Client/Server model can be explained as two computers, Client (receiver of service) and Server (provider of service) that are communicating via requests and responses.

**HTTP/1.1**

* Protocol negotiation mechanism — protocol electing, eg. HTTP/1.1, HTTP/2 or other.
* High-level compatibility with HTTP/1.1 — methods, status codes, URIs and header fields.
* HTTP1.1 used to process text commands to complete request-response cycles.
* 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.
* It is relatively secure since it uses digest authentication, NTLM authentication.
* 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.

**HTTP/2**

* HTTP/2 will use binary commands (in 1s and 0s) to execute the same tasks.
* Page load speed improvements
* Underlying semantics of HTTP such as headers, status codes remains the same.
* Compression of request headers
* Binary protocol
* HTTP/2 Server Push
* Request multiplexing over a single TCP connection
* Request pipelining
* HOL blocking (Head-of-line) — Package blocking
* 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.

**Objects and its internal representation in JavaScript**

* Objects, in JavaScript, is its 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.
* For example, let’s create an object named myCar and give it properties named make, model, and year as follows:

var myCar = new Object();  
 myCar.make = 'Ford';  
 myCar.model = 'Mustang';  
 myCar.year = 1969;