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

**HTTP1**

Developed by Timothy Berners-Lee in 1989 as a communication standard for the World Wide Web, HTTP is a top-level application protocol that exchanges information between a client computer and a local or remote web server.

In this process, a client sends a text-based request to a server by calling a method like GET or POST.

In response, the server sends a resource like an HTML page back to the client.

For example, let’s say you are visiting a website at the domain [www.example.com](http://www.example.com).

When you navigate to this URL, the web browser on your computer sends an HTTP request in the form of a text-based message, similar to the one shown here:

This request uses the GET method, which asks for data from the host server listed after Host:.

In response to this request, the example.com web server returns an HTML page to the requesting client, in addition to any images, stylesheets, or other resources called for in the HTML. Note that not all of the resources are returned to the client in the first call for data.

The requests and responses will go back and forth between the server and client until the web browser has received all the resources necessary to render the contents of the HTML page on your screen.

You can think of this exchange of requests and responses as a single application layer of the internet protocol stack, sitting on top of the transfer layer (usually using the Transmission Control Protocol, or TCP) and networking layers (using the Internet Protocol, or IP):

**HTTP2**

HTTP/2 began as the SPDY protocol, developed primarily at Google with the intention of reducing web page load latency by using techniques such as compression, multiplexing, and prioritization.

This protocol served as a template for HTTP/2 when the Hypertext Transfer Protocol working group httpbis of the [IETF (Internet Engineering Task Force)](https://www.ietf.org/) put the standard together, culminating in the publication of HTTP/2 in May 2015.

From the beginning, many browsers supported this standardization effort, including Chrome, Opera, Internet Explorer, and Safari.

Due in part to this browser support, there has been a significant adoption rate of the protocol since 2015, with especially high rates among new sites.

From a technical point of view, one of the most significant features that distinguishes HTTP/1.1 and HTTP/2 is the binary framing layer, which can be thought of as a part of the application layer in the internet protocol stack.

As opposed to HTTP/1.1, which keeps all requests and responses in plain text format, HTTP/2 uses the binary framing layer to encapsulate all messages in binary format, while still maintaining HTTP semantics, such as verbs, methods, and headers.

An application level API would still create messages in the conventional HTTP formats, but the underlying layer would then convert these messages into binary.

This ensures that web applications created before HTTP/2 can continue functioning as normal when interacting with the new protocol.

**2.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)

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;

Unassigned properties of an object are [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined)

myCar.color; // undefined