# 1.Difference between HTTP1.1 vs HTTP 2

HTTP stands for hypertext transfer protocol & it is used in client-server communication. By using HTTP user sends the request to the server & the server sends the response to the user. There are several stages of development of HTTP but we will focus mainly on HTTP/1.1 which was created in 1997 & the new one is HTTP/2 which was created in 2015

| **HTTP/1.1** | **HTTP/2** |
| --- | --- |
| It works on the textual format. | It works on the binary protocol. |
| There is head of line blocking that blocks all the requests behind it until it doesn’t get its all resources. | It allows multiplexing so one TCP connection is required for multiple requests. |
| It uses requests resource Inlining for use getting multiple pages | It uses PUSH frame by server that collects all multiple pages |
| It compresses data by itself. | It uses HPACK for data compression. |

# 2.OBJECTS AND ITS INTERNAL REPRESENTATION OF JAVASCRIPT

In JavaScript, objects play a fundamental role in organizing and manipulating data. They provide a powerful way to store collections of key-value pairs and represent complex entities. In this blog post, we will explore objects in JavaScript and delve into their internal representation, uncovering how they work behind the scenes.

## What are objects in javascript?

In JavaScript, objects are dynamic entities that serve as containers for properties. They allow you to group related data together and represent real-world entities or abstract concepts. Properties within an object can be of any data type, including other objects. Objects are created using two common methods: object literals and the Object constructor.

## Internal representation of javascript

Internally, JavaScript engines use a hash table or dictionary-like data structure to represent objects efficiently. Let's dive into how this internal representation works.

When an object is created, the JavaScript engine allocates memory to store the object and its properties. The properties are stored as key-value pairs, with the keys being strings (or symbols) and the values being any data type.

To access or modify a property of an object, the engine uses a hashing algorithm to compute a hash value for the property key. This hash value serves as an index in the underlying hash table, enabling fast property lookup and manipulation.

The use of a hash table allows JavaScript engines to achieve efficient object property access, insertion, and deletion. By generating unique hash values for keys, collisions are minimized, ensuring good performance even with a large number of properties.