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

**ANS:**

|  |  |  |
| --- | --- | --- |
| **Feature** | **HTTP/1.1** | **HTTP/2** |
| **Connection Handling** | Separate connections for each resource. | Multiplexes multiple resources over a single connection. |
| **Header Compression** | Headers sent in plaintext, leading to larger overhead. | Utilizes header compression, reducing header size and improving efficiency. |
| **Server Push** | Not supported. Clients request resources individually. | Allows servers to push resources proactively to the client. |
| **Prioritization** | No built-in prioritization. All requests are treated equally. | Supports prioritization, fetching more critical resources first. |
| **Flow Control** | Lacks built-in flow control. | Implements advanced flow control, preventing congestion. |
| **Multiplexing** | Limited multiplexing capabilities, leading to slower page loading. | Efficiently multiplexes multiple streams over a sin |

* The first usable version of HTTP was created in 1997. Because it went through several stages of development, this first version of HTTP was called HTTP/1.1. This version is still use on the web.
* In 2015, a new version of HTTP called HTTP/2 was created. HTTP/2 solves several problems that creators of HTTP/1.1 did not anticipate.
* HTTP/2 is much faster and more efficient than HTTP/1.1. HTTP/2 is faster in how it prioritizes content during loading Process.
* HTTP/1.1 loads resources one after the other, so if one resource cannot be loaded, it blocks all the other resources behind it. HTTP/2 is able to use a single TCP connection to send multiple streams of data at once so that no one resource blocks any other resource.
* In HTTP/1.1 each request had to wait for a response before the next request Could be sent. HTTP/2 uses multiplexing to allow multiple concurrent requests and responses over a single TCP Connection.
* HTTP/1.1 relies on the transport layer to avoid buffer overflow, so each new TCP connection requires a separate flow control mechanism. HTTP/2 multiplexes streams within a single TCP connection.
* The major feature differentiates HTTP/2 from HTTP/1.1 is the binary framing layer. Unlike HTTP/1.1, HTTP/2 uses a binary framing layer.
* This layer encapsulates messages — converted to its binary equivalent — while making sure that its HTTP semantics(method, details, header Information, etc) remain Untamed.

HTTP/1.1 :

It works on the textual format.

There is head of line blocking that blocks all the requests behind it until it doesn’t get its all resources.

It uses requests resource Inlining for use getting multiple pages

It compresses data by itself.

HTTP/2:

It works on the binary protocol.

It allows multiplexing so one TCP connection is required for multiple requests.

It uses PUSH frame by server that collects all multiple pages

It uses HPACK for data compression.

**2.Write a blog about objects and its internal representation in Javascript?**

ANS:

* objects are collections of key-value pairs, where keys are strings (or symbols) and values can be of any data type,
* including other objects. Objects are used to represent real-world entities, data structures, and more complex data types.
* Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.
* 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.

For Eg. If your object is a student, it will have properties like name, age, address, id, etc and methods like updateAddress, updateNam, etc.

ex :

const person = {

name: "Sreeram K",

age: 22,

email: "sreeramuidesigner@gmail.com"

};

Internal Representation of Objects:

JavaScript engines use various data structures to represent objects efficiently.

One common approach is using a hash table or a similar structure to store the object’s properties and their corresponding values.

This allows for fast access and manipulation of properties.

Internal Representation:

{

name: "Sreeram K",

age: 22,

email: "sreeramuidesigner@gmail.com"

}