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

|  |  |  |
| --- | --- | --- |
| **Aspect** | **HTTP 1.1** | **HTTP2** |
| **Connection handling** | Requires multiple connections for resources | Uses a single, multiplexed connection |
| **Header Compression** | Headers are not compressed | Employs header compression |
| **Multiplexing** | Limited parallelism due to head-of-line blocking | Allows multiple simultaneous requests/responses |
| **Server Push** | Not supported | Supports server push for proactive resource delivery |
| **Binary vs. Text** | Use plain text | Utilizes a binary protocol for more efficiency |
| **Header Overhead** | Headers can be lengthy, increasing overhead | Reduced header size due to compression |
| **Dependency** | Older and widely supported | A modern protocol gaining wider adoption |
| **Latency Reduction** | Prone to higher latency due to sequential requests | Reduces latency with multiplexing and prioritization |
| **complexity** | Simpler to implement | More complex due to multiplexing and push |
| **Bandwidth Usage** | inefficient use of bandwidth | Optimizes bandwidth usage through compression |
| **Resource Prioritization** | Lacks prioritization | Supports stream prioritization for critical resources |
| **Adoption** | Widespread but decreasing | Increasing adoption among websites and servers |
| **Backward Compatibility** | Compatible with older systems | Requires modern browser and server support |
| **Round Trips** | Requires more round trips for resource retrieval | Reduces round-trip delays with multiplexing and push |
| **Resource Loading** | slowing down page loads | Faster resource loading with multiplexing |

1. **Write a blog about objects and their internal representation in Javascript**

* **Objects are Key-Value Pairs:** JavaScript objects are collections of key-value pairs. Each key is a string (or Symbol), and each value can be of any data type.
* **Creating Objects:** Objects can be created using object literals {}, the Object() constructor, or custom constructor functions.
* **Object Properties:** The properties of an object can be accessed using dot notation or bracket notation.
* **Prototypes:** Objects in JavaScript can inherit properties and methods from their prototype, creating a prototype chain.
* **Deleting an object:** A Java script object can be deleted using the delete method.
* **Object creation with a new method:** A Java script can also be created with a new Object method. However, there is no difference between the above simple method and the new object method.
* **Duplicating Objects:** Objects can be duplicated using methods like Object assignors and the spread operator.
* **Accessing elements of Java script objects:** Object properties can either be accessed by using string indices or by object property.
* **Property Accessors:** JavaScript objects can have getter and setter methods to control access and modification of properties.
* **Enumerability:** Enumerability determines whether a property will be listed in for...in loops and Object keys().
* **Internal Representation:** Internally, objects in JavaScript are often implemented using hash tables or similar data structures, which provide efficient key-value lookup.