M T W T F S S

COMPASS

Date :

Write a blog on Difference between HTTP1.1 vs HTTP2

HTTP/1.1 and HTTP/2 are different versions of the hypertext transfer protocol (HTTP), which is used for communication between web servers and web clients. Here are some key differences between HTTP/1.1 & HTTP/2:

1 Protocol Structure
HTTP/1.1 is a text-based protcol where requests and responses are sent in plain text.
HTTP/2, on the other is a binary protocol, meaning that the message are encoded in binary format for more efficient processing

2. Multiplexing
HTTP/1.1 only allows one request to be sent at a time over a single TCP connection.
HTTP/2 introduces multiplexing, which enables multiple requests and responses to be sent concurrently over a single connection.

3. Header Compression
HTTP/1.1, headers are sent in plain text with each request and response, resulting in redundant data being transmitted.
HTTP/2 utilizes header compression techniques, such as the HPACK algorithm, to significantly reduce the overhead of header data, resulting in improved network utilization

M T W T F S S

COMPASS
Date :

4 Server Push
HTTP/2 introduces server push, a feature that allows the server to proactively send additional resources to the client before they are explicitly requested. This can improve page load time by eliminating the need for additional round trips between the client & server.

5 Prioritization
HTTP/2 supports request prioritization, allowing clients to indicate the importance of different resources. This enables the server to allocate resources accordingly, ensuring that critical resources are delivered more quickly

6 Security
While both HTTP/1.1 & HTTP/2 can be used with & without encryption (HTTP vs HTTPS), the adoption of HTTPS is encouraged in HTTP/2. Many modern browsers only support HTTP/2 over secure connections

M T W T F S S
Date :

Write a blog about objects & its internal representation in Javascript

In Javascript, objects are collections of key-value pairs, where the keys are strings and the values can be of any type. They provide a way to represent entities or concepts in the code, allowing properties and methods to be associated with them. Here is an example of creating an object using object literal notation.

Internal Representation
Internally, Javascript objects are implemented as hash tables or dictionaries. This means that when an object is created, memory is allocated to store its properties & methods. Each property is associated with a unique key (property name) and its corresponding value. Javascript engines use various techniques to optimize object representation and access for efficient performance

Property Access and Lookup
When accessing a property of an object, Javascript engines perform property lookup using a process called property resolution. The engine starts by searching for the property within the object itself. If the

COMPASS

M T W T F S S

Date :

Property Access and Lookup
When accessing a property of an object, JS engines perform property lookup using a process called property resolution. The engine starts by searching for the property within the object itself

Adding and Modifying Properties:
Javascript allows properties to be added or modified dynamically on objects. This flexibility is possible due to the dynamic nature of the language. When a new property is added, the engine adjusts the internal representation of the object, allocating memory for the new property & updating the hash table accordingly.

Memory Management & Garbage Collection:
Javascript engines employ automatic memory management through a process called garbage collection. When an object is no longer reachable or referenced, it becomes eligible for garbage collection. The engine's garbage collector periodically frees up memory by identifying and removing unused objects, allowing efficient memory utilization