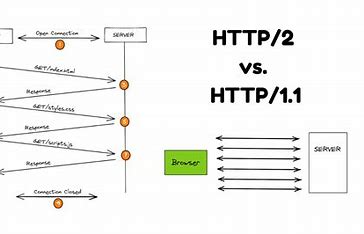
# Write a blog on the Difference between HTTP1.1 and HTTP2

**HTTP/1.1 vs. HTTP/2**



The protocol that supports our digital interactions is essential in the dynamic world of the internet. HTTP (Hypertext Transfer Protocol) has been the mainstay of online communication for many years. However, a new chapter in the history of the internet began with the introduction of HTTP/2. Let's examine the differences between HTTP/1.1 and HTTP/2 to understand why this change is important.

**The Legacy of HTTP/1.1:**

Since 1999, HTTP/1.1 has been the internet's standard, but its limitations have emerged in handling multiple concurrent connections and efficiently transmitting data.

1. HTTP/1.1 introduces latency issues as each request creates a separate connection, causing overhead due to the repeated establishment and removal of connections.

2. Head-of-line blocking occurs when a request is delayed or an error occurs, causing subsequent requests to be held up.

3. HTTP/1.1 sends redundant header fields with every request, increasing overhead and bandwidth consumption.

**Enter HTTP/2:**

HTTP/2 was introduced in 2015 to improve web performance and address the shortcomings of HTTP/1.1.

1. HTTP/2 introduces multiplexing, enabling simultaneous transmission of multiple requests and responses over a single connection, thereby reducing latency and addressing head-of-line blocking issues.

2. HTTP/2 employs header compression techniques like Huffman encoding to reduce redundant header fields, thereby minimizing overhead and conserving bandwidth.

3. HTTP/2 supports server push, allowing the server to proactively push resources to the client's cache, eliminating the need for subsequent requests.

4. HTTP/2 introduces stream prioritization, allowing clients to specify the priority of each request, ensuring timely delivery of critical resources and improving user experience.

Performance Benchmarks:

Studies show that HTTP/2 outperforms HTTP/1.1 in terms of page load times and throughput, particularly for websites with numerous assets.

Adoption Challenges:

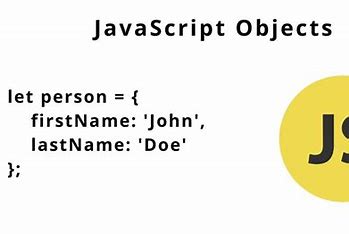
HTTP/2 adoption is gradual due to legacy infrastructure and browser support, but momentum is growing as more servers and browsers support it.

Conclusion:

HTTP/2 revolutionizes web communication, offering improved performance, reduced latency, and efficiency compared to HTTP/1.1. Despite challenges, it offers faster, more responsive experiences, demonstrating ongoing optimization and innovation in web technology.

# 2. Write a blog about objects and their internal representation in Javascript

**Unraveling Objects: Understanding Internal Representation in JavaScript**



Discover the internal representation of objects in JavaScript and understand their underlying mechanics, revealing their significance in modern web development from simple key-value pairs to complex data structures.

Anatomy of Objects:

JavaScript objects are key-value pairs, allowing developers to create dynamic and versatile data structures, characterized by their flexibility and ability to accept various data types.

Internal Representation:

JavaScript engines employ various mechanisms to efficiently represent objects, with fundamental concepts remaining consistent across different engines like V8, SpiderMonkey, or JavaScriptCore.

1. Object Properties:

JavaScript objects have a hidden internal property map, known as a "property map" or "dictionary," which allows dynamic addition, modification, or deletion of properties.

2. Prototypes and Inheritance:

JavaScript uses a prototype-based inheritance model, allowing objects to inherit properties and methods from prototypes. If a property isn't found, the JavaScript engine traverses the prototype chain.

3. Hidden Classes and Shapes\*:

JavaScript engines optimize object property access and storage by using hidden classes or shapes, which are internal representations used to track object structure.

The engine can create a new hidden class or adjust the existing one when an object's structure changes, such as adding or removing properties.

Performance Considerations:

Understanding the internal representation of objects is crucial for writing efficient JavaScript code, with performance considerations being a key aspect.

1. To prevent unnecessary overhead, The goal is to minimize property access in loops and performance-critical code sections.

2. Property caching is a technique that caches frequently accessed properties to reduce lookup time and enhance overall performance.

3. Object pools or object recycling techniques can help reduce memory allocation overhead, particularly for short-lived objects.

Conclusion:

Objects are the foundation of JavaScript programming, providing flexibility, versatility, and dynamic capabilities. Mastering their internal representation and JavaScript engines enhances code efficiency and performance.

# 3. Read about IP address, port, HTTP methods, MAC address

Certainly! Here's a brief overview of each:

1. IP Address (Internet Protocol Address)



IP addresses are unique numerical labels assigned to devices in computer networks, serving host identification and location addressing. Available in 32-bit and 128-bit versions, IPv6 is increasingly used.

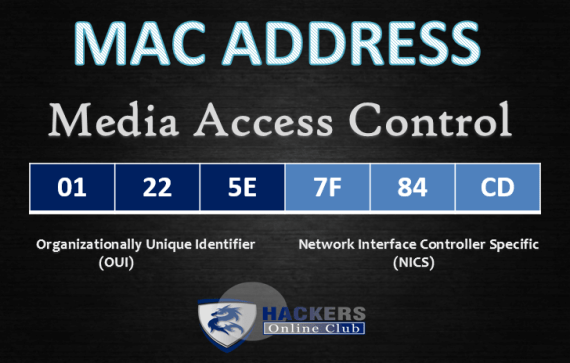
2. Port

A port is a numerically identified communication endpoint in computer networking, allowing multiple applications or services to use the same network connection simultaneously.

3. HTTP Methods (Hypertext Transfer Protocol Methods)

HTTP is a protocol for transmitting hypermedia documents like HTML, with common HTTP methods including GET, POST, PUT, DELETE, PATCH, HEAD, and OPTIONS. These methods define actions on web servers, allowing for data retrieval, processing, uploading, deletion, partial updates, and communication options.

4. MAC Address (Media Access Control Address)



MAC addresses are unique identifiers assigned to network interface controllers (NICs) for local communication within a network segment, typically assigned by hardware manufacturers, unlike IP addresses.

Understanding these concepts is essential for anyone working with computer networks or web development. Each plays a critical role in how data is transmitted and received over the internet.