ASSIGNMENT

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

The World Wide Web is constantly evolving, and at the heart of its functioning lies the Hypertext Transfer Protocol (HTTP). Over the years, various versions of HTTP have been developed to enhance the efficiency and performance of web communications. In this blog post, we'll delve into the key differences between two prominent versions: HTTP/1.1 and HTTP/2.

HTTP/1.1: The Traditional Protocol:

HTTP/1.1 has been the backbone of web communication for many years. However, as the complexity of websites increased, HTTP/1.1 faced challenges in terms of performance, especially with multiple requests and larger file sizes.

**Sequential Loading:** One of the major limitations of HTTP/1.1 is its tendency to load resources sequentially. Each request must be completed before the next one begins, leading to latency issues, especially on bandwidth-intensive websites.

**Header Overhead:** Every HTTP request and response comes with a set of headers containing metadata. In HTTP/1.1, these headers are repeated for each request, leading to redundant data transfer and increased latency.

**Connection Handling:** In HTTP/1.1, a new connection is established for each request. While this is suitable for small websites, it results in increased latency for larger, more complex pages with numerous resources.

HTTP/2: The Modern Upgrade:

Recognizing the limitations of HTTP/1.1, HTTP/2 was introduced to address these issues and optimize web performance. Let's explore the key features that set HTTP/2 apart:

**Multiplexing:** Perhaps the most significant improvement, HTTP/2 introduces multiplexing, allowing multiple requests and responses to be sent concurrently over a single connection. This reduces latency and speeds up the loading of web pages with numerous resources.

**Header Compression:** To address the issue of header overhead, HTTP/2 utilizes header compression techniques. Headers are compressed before transmission, reducing redundant data and further improving performance.

**Binary Protocol:** While HTTP/1.1 relies on text-based protocols, HTTP/2 uses a binary protocol. This not only makes it more efficient but also allows for easier parsing and improved compression.

**Server Push:** Another notable feature of HTTP/2 is server push, where the server can proactively send resources to the client before they are explicitly requested. This optimizes the loading process by anticipating the client's needs.

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

JavaScript, the language that powers the dynamic and interactive aspects of the web, relies heavily on objects. Objects serve as the fundamental building blocks, allowing developers to structure, organize, and manipulate data in their applications. In this blog post, we'll dive into the fascinating world of objects in JavaScript, exploring their internal representation and the magic that happens under the hood.

Understanding Objects in JavaScript:objects are versatile entities that can represent real-world entities, encapsulate data, and provide functionality. Objects can be created using literal notation or through constructor functions, offering flexibility in design and implementation.

// Object literal notation

const person = {

name: 'John Doe',

age: 25,

profession: 'Developer'

};

The Internal Structure of Objects:

To comprehend the internal representation of objects in JavaScript, it's essential to grasp the underlying concepts of properties and prototypes.

1. **Properties:**

- Each object in JavaScript is a collection of key-value pairs, where keys are called properties and values can be any data type.

- Properties can be accessed using dot notation (`object.property`) or bracket notation (`object['property']`).

2. **Prototypes:**

- JavaScript is a prototype-based language, and objects can inherit properties and methods from other objects through prototypes.

- When a property is not found on an object, JavaScript looks up the prototype chain until it finds the property or reaches the end of the chain.

// Object with a prototype

const person = {

name: 'John Doe',

age: 25,

sayHello: function() {

console.log('Hello!');

}

};

// Creating an object that inherits from the 'person' object

const developer = Object.create(person);

developer.profession = 'Developer';

Memory Representation:

Under the hood, JavaScript engines manage the memory for objects using various data structures. Objects are stored in memory as key-value maps, and the engine employs optimization techniques to ensure efficient access and manipulation.

1. **Property Descriptors:**

- Each property in an object is associated with a property descriptor that defines its attributes, such as whether it is writable, enumerable, or configurable.

2. **Hidden Classes:**

- JavaScript engines use hidden classes to optimize object property access and improve performance. Objects with similar structures share the same hidden class, making property access faster.

// Example of hidden classes

function Person(name, age) {

this.name = name;

this.age = age;

}

const john = new Person('John Doe', 25);

const jane = new Person('Jane Doe', 30);

Certainly! Let's delve into each of these networking concepts:

**Read about IP address, port, HTTP methods, MAC address**

1. **IP Address (Internet Protocol Address):**

- An IP address is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication.

- It serves two main purposes: host or network interface identification and location addressing.

- There are two types of IP addresses: IPv4 (32-bit) and IPv6 (128-bit). IPv4 addresses are more common and look like this: 192.168.1.1, while IPv6 addresses are longer and look like this: 2001:0db8:85a3:0000:0000:8a2e:0370:7334.

2. **Port**:

- A port is a logical endpoint for communication in a computer network. It is identified by a 16-bit unsigned number, thus allowing for a total of 65,536 different ports.

- Ports are used to distinguish different services or processes on the same device. For example, HTTP typically uses port 80, and HTTPS uses port 443.

- Ports are categorized into three ranges: well-known ports (0-1023), registered ports (1024-49151), and dynamic or private ports (49152-65535).

3. **Methods (Hypertext Transfer Protocol Methods):**

- HTTP methods, also known as HTTP verbs, indicate the desired action to be performed on a resource. They are an essential part of the HTTP protocol, used for communication between web browsers and web servers.

- Some common HTTP methods include:

- \*\*GET:\*\* Retrieve data from the server.

- \*\*POST:\*\* Submit data to the server to be processed.

- \*\*PUT:\*\* Update a resource on the server.

- \*\*DELETE:\*\* Remove a resource from the server.

- \*\*HEAD:\*\* Retrieve headers of a resource without the body.

- \*\*OPTIONS:\*\* Fetch information about the communication options for the target resource.

4. **MAC Address (Media Access Control Address):**

- A MAC address is a unique identifier assigned to a network interface controller (NIC) for use as a network address in communications within a network segment.

- It is a hardware address that is assigned by the manufacturer and usually represented as six groups of two hexadecimal digits separated by colons or hyphens. For example, 00:1A:2B:3C:4D:5E.

- MAC addresses are used at the data link layer (Layer 2) of the OSI model and are essential for the functioning of local area networks (LANs).