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

**Introduction:**

* The World Wide Web has evolved significantly since its inception, and one of the pivotal technologies that enable this evolution is the Hypertext Transfer Protocol (HTTP). In this blog, we will dive into the key differences between HTTP/1.1 and HTTP/2.

|  |  |
| --- | --- |
| **HTTP/1.1** | **HTTP/2** |
| * The Legacy Protocol has been the workhorse of the web for decades, serving as the foundation for web communication. | * A Modern Approach was developed to address the shortcomings of http/1.1 and to provide a more efficient way of transmitting data over the web. |
| **Connection Handling**:   * In HTTP/1.1, multiple resources on a webpage require separate connections, leading to a phenomenon called “head-of-line blocking.” This means that if a resource requested by a browser is delayed or blocked, it can slow down the loading of other resources, even if they're ready to be fetched. | **Multiplexing**:   * One of the most significant improvements in HTTP/2 is multiplexing. This feature allows multiple requests and responses to be sent and received simultaneously over a single connection. |
|  |
| **No Compression by Default:**   * HTTP/1.1 doesn't compress headers by default, which can lead to significant overhead in terms of data transfer, especially when dealing with large numbers of small requests. | **Server Push**:   * HTTP/2 supports server push, which enables the server to send additional resources to the client before they are explicitly requested. |
| **One Request per Connection:**   * HTTP/1.1 doesn't compress headers by default, which can lead to significant overhead in terms of data transfer, especially when dealing with large numbers of small requests. | **Binary Protocol**:   * HTTP/2 uses a binary protocol for data exchange, which is more compact and easier for computers to parse compared to the text-based protocol of HTTP/1.1. |

**Conclusion:**

* HTTP/1.1 will likely continue to be used for legacy systems embracing HTTP/2 can significantly enhance the speed, performance, and overall quality of modern web applications. As we move forward, it's important for web developers and administrators to understand the differences between these protocols and make informed decisions about which one to adopt based on their specific needs and goals.

1. **Write a blog about objects and its internal representation in Java Script.**

**Introduction:**

* JavaScript is a versatile programming language that plays a pivotal role in modern web development. One of its core concepts is the use of objects, which allow developers to organize and manipulate data in a structured manner.

**Understanding Objects in JavaScript:**

* At its heart, JavaScript is an object-oriented language. Objects are fundamental to how data is organized and manipulated in the language. An object in JavaScript is essentially a collection of key-value pairs, where each key is a property and each value can be a piece of data or a function (known as a method).

**Own Properties and Prototype Properties:**

* **Own Properties:** They can be added, modified, or deleted using dot notation or square brackets. Each own property is associated with a value or a function.
* **Prototype Properties:** Objects in JavaScript can have a prototype, which is essentially another object. This prototype object can contain properties and methods that are shared among multiple objects.

**Properties and Methods:**

* Properties in JavaScript objects can hold a wide range of values: Numbers, Strings, Booleans, Arrays, other objects, and even functions.
* Methods are essentially properties that hold function values.
* When a method is invoked on an object, the method has access to the object's properties and can manipulate them as needed.
* **Use Object Literal Syntax**: When creating simple objects, use the object literal syntax for cleaner code.
* **Prototype Chain**: Utilize the prototype chain to share properties and methods among objects, reducing memory consumption and promoting efficient code.
* **Constructor Functions and Classes**: For more complex object creation, constructor functions and ES6 classes provide a structured way to define objects with shared behaviors.

**Conclusion:**

Objects are the building blocks of JavaScript, allowing developers to model real-world entities and their behaviors in code. Understanding the internal representation of objects, their properties, methods, and the prototype chain is crucial for writing efficient, maintainable, and organized code.