JavaScript - Day -1: Introduction to Browser & web

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

HTTP1.1

HTTP/1.1, or Hypertext Transfer Protocol 1.1, is an older version of the HTTP protocol used for transferring data over the World Wide Web but comes with limitations that can impact web performance.

Advantages of HTTP/1.1:

- **Compatibility:** HTTP/1.1 is widely supported and compatible with most web servers, browsers, and web applications. This ensures that it can be used effectively on a wide range of systems.
- **Simple Implementation:** HTTP/1.1 is relatively simple to implement, making it a practical choice for many websites and web services.
- **Caching:** HTTP/1.1 includes caching mechanisms, allowing browsers to cache static assets like images, stylesheets, and scripts, which can improve page load times for returning visitors.
- **Backward Compatibility:** HTTP/1.1 is backward compatible with HTTP/1.0, which means it can work with older web servers and clients, ensuring that older technology isn't left behind.

Disadvantages of HTTP/1.1:

- **Serialization:** One of the most significant drawbacks of HTTP/1.1 is that it processes requests and responses serially. This means that each request has to wait for the previous one to complete, which can lead to performance bottlenecks and slower page load times.

- **Header Overhead:** Every HTTP/1.1 request and response includes headers that can be substantial in size. These headers need to be sent with each request, consuming bandwidth and slowing down page loading, particularly when making multiple requests.
- **Resource Bundling:** To mitigate the impact of multiple requests, developers often bundle resources (e.g., JavaScript and CSS files) into a single, larger file. While this reduces the number of requests, it makes caching and updates more challenging.
- **Limited Security:** HTTP/1.1 does not inherently provide strong security features. While HTTPS can be implemented, it's not a default feature of HTTP/1.1, leaving connections vulnerable to security threats.
- **Reduced Performance with Latency:** The serial processing and high overhead can be especially problematic for high-latency connections, causing noticeable delays in web page loading.

HTTP 2

HTTP/2 is a significant improvement over HTTP/1.1, offering faster page loading times, reduced overhead, and enhanced efficiency in web communication. Its advantages, such as multiplexing, header compression, server push, and prioritization, have made it a preferred choice for modern web applications. HTTP/2 has become the standard for many websites and is widely adopted to provide a better user experience on the web.

Advantages of HTTP/2:

- **Multiplexing:** HTTP/2 allows for the concurrent processing of multiple requests and responses over a single connection. This means that multiple assets, such as images, scripts, and stylesheets, can be transferred in parallel, reducing page load times significantly.
- **Header Compression:** HTTP/2 uses header compression techniques, which reduce the size of headers, minimizing overhead. This results in more efficient data transmission and faster loading times.
- **Server Push:** HTTP/2 enables servers to proactively push resources to clients before they request them. This can eliminate the need for additional requests and further improve page loading speed.

- **Binary Protocol:** Unlike the plain text of HTTP/1.1, HTTP/2 uses a binary protocol, which simplifies parsing and processing, making it more efficient.
- **Prioritization:** HTTP/2 allows clients to prioritize requests. This means that critical assets can be fetched before less important ones, enhancing the perceived speed of a web page.

Disadvantages of HTTP/2:

- **Compatibility:** While HTTP/2 is widely supported by modern web browsers and servers, there are still some older systems and devices that may not fully support it. This can create challenges in maintaining compatibility across all platforms.
- **Complexity:** Implementing and debugging HTTP/2 can be more complex than HTTP/1.1 due to the additional features and requirements. This complexity may pose challenges for some web developers and administrators.
- **Resource Consumption:** While HTTP/2's multiplexing feature improves overall performance, it can lead to increased resource consumption on servers. Proper server configuration and resource management are essential to make the most of HTTP/2's benefits.

2. Write a blog about objects and its internal representation in JavaScript

In JavaScript, objects are collections of key-value pairs. The keys are typically strings (or Symbols), and the values can be of any data type, including other objects. Objects are used to organize and store data in a structured manner.

```
const person = {
  name: "John",
  age: 30
};
```

Property Access:

In JavaScript, you can access properties of an object using dot notation (.) or square bracket notation ([]). When you access a property, the JavaScript engine internally looks up the property in the object's property table using the hidden class associated with the object, allowing for fast and efficient property access.

```
const person = {
  name: "John",
  age: 30,
};

// Access properties using dot notation
console.log(person.name); // "John"
console.log(person.age);  // 30

// Access properties using square bracket notation
console.log(person['name']); // "John"
console.log(person['age']);  // 30
```

Property Deletion:

In JavaScript, you can delete properties from objects using the **delete** operator. When you delete a property, it is marked as deleted, and the property is removed from the object.

```
const person = {
  name: "John",
  age: 30,
};
```

```
// Delete the 'age' property
delete person.age;

console.log(person); // { name: "John" }
```