**1)Difference between HTTP1.1 vs HTTP2**

A)HTTP stands for hypertext transfer protocol & it is used in client-server communication, HTTP/1.1 was its 1st standardised version that was available for use in the year 1997 for the end-users and now recently the IETF, Google, Microsoft, and Facebook released the fully-comprehensive and well-tested newer version HTTP2 in 2015. HTTP2 helped in several methods to decrease latency, especially when dealing with mobile platforms and server-intensive graphics and videos.

The HTTP2 is created based on SPDY protocol and developed to address the inherent limitations of HTTP1.1 and further progress the Internet.

The difference between HTTP1.1 and HTTP2 is also the reason for the creation of HTTP2

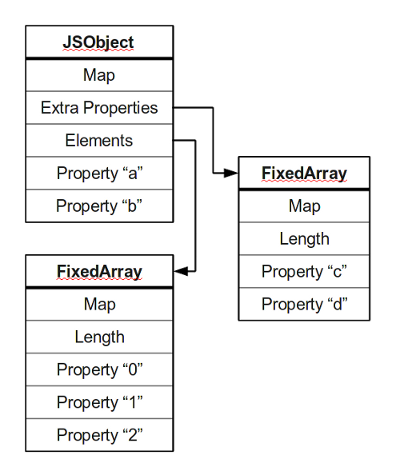
| Feature | HTTP1.1 | HTTP2 |
| --- | --- | --- |
| Multiplexing | Single request at a time per connection | Multiple requests in parallel over a single connection |
| Header Compression | Headers not compressed | Headers are compressed for reduced overhead |
| Binary Protocol | Text-based protocol | Binary protocol for more efficient parsing |
| Server Push | Not supported | Supports server push for proactive resource delivery |
| Connection Management | Multiple connections are often required | Single, multiplexed connection |
| Prioritization | Requests processed in order | Supports request prioritisation |

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

A)“A JavaScript object is a collection of named values having state and behavior (properties and method)”.

For example: Person, car, pen, bike, Personal Computer , Washing Machine etc.

unlike Primitive data types which can only contain one value, objects can contain multiple values in the form of key-value pairs. An object can be created with figure brackets {} with an optional list of properties. A property is a “key: value” pair; a key is the property name value can be anything.



**Properties:**

Objects in JavaScript have properties, which are key-value pairs. The key (also known as a property name) is always a string or a symbol, and the value can be any valid JavaScript data type, including other objects.

**Methods:**

Properties in JavaScript objects can also hold functions, which are then called methods. These methods are functions that are associated with the object and can perform actions or provide functionality related to the object.

**Prototype Chain:**

JavaScript objects can inherit properties and methods from other objects through a mechanism called the prototype chain. Each object in JavaScript has an associated prototype object, and if a property or method is not found on the object itself, JavaScript looks for it in the object's prototype and continues up the chain until it reaches the Object prototype.

**[[Prototype]]:**

In JavaScript, the [[Prototype]] internal property is used to establish the prototype chain. It's an internal link that connects an object to its prototype. In modern JavaScript, you can access an object's prototype using the Object.getPrototypeOf() method or the \_\_proto\_\_ property.

A simple example to illustrate the internal representation of an object in JavaScript

let person = {

firstname: 'John',

lastName: 'Doe',

sayHello: function() {

console.log(`Hello, ${this.firstName} ${this.lastName}!`);

}

};

// Accessing properties

console.log(person.firstName); // Output: John

// Calling a method

person.sayHello(); // Output: Hello, John Doe!

// Accessing the object's prototype

let prototypeOfPerson = Object.getPrototypeOf(person);

console.log(prototypeOfPerson); // Output: Object { ... }

In this example, a person is an object with properties (firstName, lastName) and a method (sayHello). The Object.getPrototypeOf() method is used to access the prototype of the person object, revealing the prototype chain.