**TASK - 1**

**HTTP**

The Hyper Text Transfer Protocol - HTTP is a data communications protocol and acts as the foundation of the World Wide Web.

**Difference between HTTP/1.1 vs HTTP/2**

|  |  |  |
| --- | --- | --- |
| **S.No** | **HTTP/1.1** | **HTTP/2** |
| 1 | It published in “1997” | It published in “2015” |
| 2 | It works on the “Textual format” | It works on the “Binary protocol” |
| 3 | There is head of line blocking that blocks all the requests behind it until it doesn’t get its all resources. | It allows multiplexing so one TCP connection is required for multiple requests. |
| 4 | It uses requests resources inlining for use getting multiple pages. | It uses PUSH frame by server that collects all multiple pages. |
| 5 | It compresses data by itself. | It uses HPACK for data compression. |

**HTTP/ Motivation and Goals**

One of the motivating factors behind **HTTP/2** was the quest for greater performance. This was necessitated bye the fact that websites were becoming more media-rich, and offered significantly more interaction with the client. Server-side operations and client-side scripts were becoming larger and more complex and as such, were more demanding on resources including bandwidth.

Given that **HTTP/1.1** protocol is still widely deployed, including built in to middleboxes that are not likely to be upgraded in this respect, HTTP is backward compatible and is largely the same, the changes to **HTTP/2** can be primarily regarded as optimizations and bug fixes.

**TASK – 2**

**OBJECTS AND ITS INTERNAL REPRESENTATION IN JAVASCRIPT**

**OBJECTS IN JAVASCRIPT**

In JavaScript, objects are collections of key-value pairs, where keys are strings (or symbols) and values can be of any data type, including other objects. Objects are used to represent real-world entities, data structures, and more complex data types. In JavaScript, almost "everything" is an object.

* Booleans can be objects (if defined with the new keyword)
* Numbers can be objects (if defined with the new keyword)
* Strings can be objects (if defined with the new keyword)
* Dates are always objects
* Maths are always objects
* Regular expressions are always objects
* Arrays are always objects
* Functions are always objects
* Objects are always objects

(All JavaScript values, except primitives, are objects)

**OBJECTS AND ITS INTERNAL REPRESENTATION IN JAVASCRIPT**

**Introduction:**

JavaScript, the language that powers dynamic and interactive web applications, is built around the concept of objects. Objects are fundamental to the language, serving as a cornerstone for data manipulation and structuring. In this blog post, we’ll embark on a visual exploration of objects in JavaScript and unravel their internal representation.

**1.What is an Object in JavaScript?**

In JavaScript, an object is a complex data type that allows you to store and organize data in key-value pairs. Objects can represent real-world entities and are used to model and manipulate information efficiently.

**2. Creating Objects:**

Objects in JavaScript can be created using object literals or the `Object` constructor. Let’s take a look at a simple object literal:

Here, `person` is an object with three properties: `name`, `age`, and` is employed`

**3. Internal Representation of Objects:**

Internally, JavaScript engines represent objects using various data structures. One common representation is the hash table, where keys are hashed to optimize property access. The hash table allows for quick lookup of properties, making object access efficient.

**4. Object Properties and Methods:**

Objects can contain not only data properties but also methods, which are functions associated with the object. Let’s add a method to our `person` object:

javascript code :

person.greet = function() {

console.log(`Hello, I’m ${this.name}!`);

};

Now, our `person` object has a `greet` method that logs a greeting.

**5. Prototypes and Inheritance:**

JavaScript is a prototype-based language, and objects can inherit properties and methods from other objects through their prototypes. This mechanism enables the creation of hierarchical structures.In the JS code, you can see how objects and their prototypes are linked, forming a chain of inheritance.

**Conclusion:**

Objects lie at the heart of JavaScript, providing a powerful mechanism for organizing and manipulating data. The internal representation of objects, often implemented using hash tables, ensures efficient property access. As you dive deeper into JavaScript development, understanding how objects work and how they are internally represented will empower you to write more expressive and modular code. So, embrace the magic of objects, and let them be your companions in building dynamic and sophisticated web applications.