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

**INTRODUCTION TO HTTP**

**HTTP—>HyperText Transfer Protocol**

HTTP (Hypertext Transfer Protocol) is the set of rules for transferring files -- such as text, images, sound, video and other multimedia files -- over the web. As soon as a user opens their web browser, they are indirectly using HTTP.

HTTP is an application protocol that runs on top of the TCP/IP suite of protocols, which forms the foundation of the internet. The latest version of HTTP is HTTP/2, which was published in May 2015. It is an alternative to its predecessor, HTTP 1.1

| * **HTTP 1.1** | * **HTTP 2** |
| --- | --- |
| * It works on textual format | * It works on the binary protocol. |
| * 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. |
| * It uses requests resource Inlining for use getting multiple pages | * It uses PUSH frame by server that collects all multiple pages |
| * It compresses data by itself | * It uses HPACK for data compression |
| * There is no built-in prioritization mechanism. | * It allows clients to specify the priority of each request,helping to ensure the critical resources are loaded first. |
| * Server Push is not possible on HTTP 1.1 whereas client has to send the request to each resource individually | * It supports Server push,where the server can send additional resources(e.g CSS,Javascript)to the client before the client requests them. |
| * Stream dependencies are not available here | * It allows streams to depend on each other,ensuring the related resources are loaded in correct order. |
| * Error handling is less sophisticated. | * Error handling is improved,with more specific error code and the ability to continue * processing other requests on the same connection even if one request encounters an error |
| * It relies on TCP level flow control,which is less efficient | * It has built in flow control mechanisms,preventing congestion and optimizing resource delivery |
| * It opens and closes a new connection for each request/response pair | * It uses a single,long lived connection for multiple requests and responses,reducing connection setup and teardown overhead. |

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

**INTRODUCTION TO OBJECTS**

* An object is a collection of properties, and a property is an association between a name (or *key*) and a value. A property's value can be a function, in which case the property is known as a method.
* JavaScript object is a non-primitive data-type that allows you to store multiple collections of data.
* In JavaScript, an object is a standalone entity, with properties and type.
* Compare it with a car, for example. A car is an object, with properties. A car has a color, a design, weight, a material it is made of, etc.
* The same way, JavaScript objects can have properties, which define their characteristics.

**Declaration of Object**

**Syntax:**

**const object\_name = {**

**key1: value1,**

**key2: value2**

**}**

**Example for Object Creation**

**const car = {**

**brand: 'honda',**

**color: black,**

**model:’amaze’**

**};**

# **Objects and its properties**

* A JavaScript object has properties associated with it. A property of an object can be explained as a variable that is attached to the object.
* Object properties are basically the same as ordinary JavaScript variables, except for the attachment to objects.
* The properties of an object define the characteristics of the object. You access the properties of an object with a simple dot-notation:

**Syntax:**

**objectName.propertyName**

* + Like all JavaScript variables, both the object name (which could be a normal variable) and property name are case sensitive.
  + You can define a property by assigning it a value. For example, let’s create an object named pen and give it properties named brand,color and so on

**var myPen = new Object();**

**myPen.model = 'Hero';**

**myPen.color = ‘black’;**

Unassigned properties of an object are undefined (and not null).

**Example:**

**myPen.model;** **//undefined**

Properties of JavaScript objects can also be accessed or set using a bracket notation

**myPen['model'] = 'Hero';**

**myPen['color'] = ‘black’;**

## **Creating Objects in JavaScript:**

1. By object literal
2. By creating instance of Object directly (using new keyword)

**By object literal**

* This is the easiest way to create a JavaScript Object.
* Using an object literal, you both define and create an object in one statement.

**Example:**

**const person={**

**firstName:’Sofi’,**

**lastName:’John’**

**};**

**Using the ‘new’ keyword with’ in-built Object constructor function**

To create an object, use the new keyword with Object() constructor, like this:

**const person = new Object();**

Now, to add properties to this object, we have to do something like this:

**person.firstName:”Sofi”;**

**person.lastName:”John”;**