**JAVASCRIPT:INTRODUCTION TO BROWSER AND WEB**

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

The Hypertext Transfer Protocol(HTTP) is the foundation of data communication on theWorld Wide Web.It enables the transfer of data between a client and a server.Two of the most notable versions of HTTP are HTTP/1.1 and HTTP/2.

HTTP protocol was developed in 1989 as the common language that enables client and server machines’ interaction. Process steps are as enlisted:

1. The client (browser) has to send a request to the server using the method (GET/POST).
2. Server responds with the requested resource, for example – image, alongside the status of what it did to the client’s request.

|  |  |
| --- | --- |
| HTTP/1.1 | HTTP/2 |
| It works in the 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. |

Pipelining is a technique used in HTTP/1.1 to send multiple requests over a single connection without waiting for each response. However, it is limited by head-of-line blocking, which can still cause performance issues. Multiplexing, introduced in HTTP/2, allows multiple requests and responses to be sent concurrently over a single connection, eliminating head-of-line blocking and significantly improving performance.

In conclusion, the transition from HTTP/1.1 to HTTP/2 has brought substantial improvements in web performance.HTTP/2’s multiplexing,binary framing ,serverpush, and header compression have modernized web communication ,making it more efficient and responsive.While HTTP/1.1 is still in use,HTTP/2 is now the preferred choice for serving web content,ensuring a better user experience and improved website performance.

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

Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.An object, is a reference data type. Variables that are assigned a reference value are given a reference or a pointer to that value. That reference or pointer points to the location in memory where the object is stored. The variables don’t actually store the value.

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:

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 myCar and give it properties named make, model, and year as follows:

var myCar = new Object();  
myCar.make = 'Ford';  
myCar.model = 'Mustang';  
myCar.year = 1969;

Unassigned properties of an object are [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined" \t "https://medium.com/analytics-vidhya/_blank) (and not [null](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/null" \t "https://medium.com/analytics-vidhya/_blank)).

myCar.color; // undefined

Properties of JavaScript objects can also be accessed or set using a bracket notation (for more details see [property accessors](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Property_Accessors" \t "https://medium.com/analytics-vidhya/_blank)). Objects are sometimes called *associative arrays*, since each property is associated with a string value that can be used to access it. So, for example, you could access the properties of the myCar object as follows:

myCar['make'] = 'Ford';  
myCar['model'] = 'Mustang';  
myCar['year'] = 1969;

An object property name can be any valid JavaScript string, or anything that can be converted to a string, including the empty string. However, any property name that is not a valid JavaScript identifier (for example, a property name that has a space or a hyphen, or that starts with a number) can only be accessed using the square bracket notation. This notation is also very useful when property names are to be dynamically determined (when the property name is not determined until runtime). Examples are as follows:

// four variables are created and assigned in a single go,   
// separated by commas  
var myObj = new Object(),  
 str = 'myString',  
 rand = Math.random(),  
 obj = new Object();  
myObj.type = 'Dot syntax';  
myObj['date created'] = 'String with space';  
myObj[str] = 'String value';  
myObj[rand] = 'Random Number';  
myObj[obj] = 'Object';  
myObj[''] = 'Even an empty string';console.log(myObj);

You can also access properties by using a string value that is stored in a variable:

var propertyName = 'make';  
myCar[propertyName] = 'Ford';propertyName = 'model';  
myCar[propertyName] = 'Mustang';

# **Using the JavaScript Keyword new**

The following example also creates a new JavaScript object with four properties:

Example

var person = new Object();  
person.firstName = “John”;  
person.lastName = “Doe”;  
person.age = 50;  
person.eyeColor = “blue”;