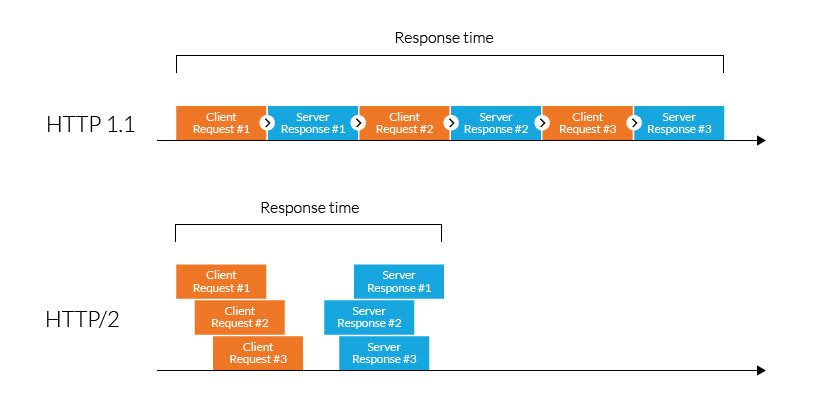
# **1)HTTP 1.1 VS HTTP 2**

****

HTTP/2 improved on HTTP/1.1 in a number of ways that allowed for speedier content delivery and improved user experience.

***Binary protocols:***

Binary protocols consume less bandwidth, are more efficiently parsed and are less error-prone than the textual protocols used by HTTP/1.1. Additionally, they can better handle elements such as whitespace, capitalization and line endings.

***Multiplexing:***

HTTP/1.1 loads resources one after the other, so if one resource cannot be loaded, it blocks all the other resources behind it. In contrast, HTTP/2 is able to use a single TCP connection to send multiple streams of data at once so that no resource blocks any other resource. HTTP/2 does this by splitting data into binary-code messages and numbering these messages so that the client knows which stream each binary message belongs to.

***Server push:***

Typically, a server only serves content to a client device if the client asks for it. However, this approach is not always practical for modern webpages, which often involve several dozen separate resources that the client must request. HTTP/2 solves this problem by allowing a server to "push" content to a client before the client asks for it. The server also sends a message letting the client know what pushed content to expect – like if Bob had sent Alice a Table of Contents of his novel before sending the whole thing.

***Header compression:***

Small files load more quickly than large ones. To speed up web performance, both HTTP/1.1 and HTTP/2 compress HTTP messages to make them smaller. However, HTTP/2 uses a more advanced compression method called HPACK that eliminates redundant information in HTTP header packets. This eliminates a few bytes from every HTTP packet. Given the volume of HTTP packets involved in loading even a single webpage, those bytes add up quickly, resulting in faster loading.

***Increased security:***

Web browsers only support HTTP/2 via encrypted connections, increasing user and application security.

# **2)Objects and its internal representation in javascript**

Objects are important data types in javascript. Objects are different than primitive datatypes (i.e. number, string, boolean, etc.). Primitive data types contain one value but Objects can hold many values in form of Key: value pair. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.

Every object has some property associated with some value. These values can be accessed using these properties associated with them.

var myCar = new Object();

myCar.make = 'Suzuki';

myCar.model = 'Altros';

myCar.year = 1978;

myCar.wheels = 2;

After creating myCar object, the value inside the object can be accessed using keys.

i.e.*myCar.year—>*Output: 1978

These values can be accessed using brackets notation also.

*myCar[year] —>*Output: 1978

***Syntax for adding a property*** to an object is :

ObjectName.ObjectProperty = propertyValue;

***Syntax for deleting a property*** from an object is:

delete ObjectName.ObjectProperty;

***Syntax to access a property*** from an object is:

objectName.property //or

objectName["property”] //or

objectName[expression]

So, conclusion and simple definition for Java Script properties is “Properties are the values associated with a JavaScript object”.