**DAY 1**

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

|  |  |
| --- | --- |
| **HTTP 1.1** | **HTTP 2** |
| It supports connection reuse i.e. for every TCP connection there could be multiple requests and responses, and pipe lining where the client can request several resources from the server at once. However, pipe lining was hard to implement due to issues such as head-of-line blocking and was not a feasible solution. | Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. It is done using streams which can be prioritized, can have dependencies and individual flow control. It also provides a feature called server push that allows the server to send data that the client will need but has not yet requested. |
| Introduces a warning header field to carry additional information about the status of a message. Can define 24 status codes, error reporting is quicker and more efficient. | Underlying semantics of HTTP such as headers, status codes remains the same. |
| It is relatively secure since it uses digest authentication, NTLM authentication. | Security concerns from previous versions will continue to be seen in HTTP/2. However, it is better equipped to deal with them due to new TLS features like connection error of type Inadequate\_Security. |
| Expands on the caching support by using additional headers like cache-control, conditional headers like If-Match and by using entity tags. | HTTP/2 does not change much in terms of caching. With the server push feature if the client finds the resources are already present in the cache, it can cancel the pushed stream. |
| HTTP/1.1 provides faster delivery of web pages and reduces web traffic as compared to HTTP/1.0. However, TCP starts slowly and with domain sharing (resources can be downloaded simultaneously by using multiple domains), connection reuse and pipe lining, there is an increased risk of network congestion. | HTTP/2 utilizes multiplexing and server push to effectively reduce the page load time by a greater margin along with being less sensitive to network delays. |

1. **Write a blog about objects and its internal representation in JavaScript?**

Objects – Objects in JavaScript can be compared with real life objects(for example – A car is an object). Objects are collection of properties which has association with the key and the value. Based upon this association objects are classified into three types namely- simple objects, array of objects, objects of objects(nested objects). JavaScript objects have properties which describes their properties. Objects are defined as an unordered collection of related data in the form of “key: value” pairs. These values can be updated using methods. Objects are variables too but can contain many values. Objects can be created in several methods in JavaScript namely,

* Object literals - In this method objects are created in between {…} which is the most simplest way of creating it. Object literals are expressions and each of them results in a new object whenever it is executed. The inputs can be anything. For example)

Var obj ={

Input1: value1,

Input2: value2,

};

* Object constructor - With the help of the constructor function an object is created in this method which defines the characteristics of the object. Once we create a constructor function we can create numerous objects. For example)

**Creating a constructor:**

function info(name, age, d.o.b){

studentname=name,

studentage=age,

studentd.o.b=d.o.b,

}

(now we can create as much object as we can since we have created the object constructor)

1. Varstudent1= new info (‘ram’,’18’,’12/3/2002’)
2. Varstudent2= new info (‘ashok’,’18’,’2/6/2002’)

* Prototypes - With the help of a fixed object prototype created several new objects can be created by calling the prototype. These objects are created by calling object.create(). For example)

Let students={

Assemblyrole:”singing”

}

Let student1= object.create(students);

Console.log(student1:assemblyrole);

Thus objects are represented in JavaScript and these object members are accessed using dot notations, bracket notations. There is delete operator in order to remove the unwanted property of an object.