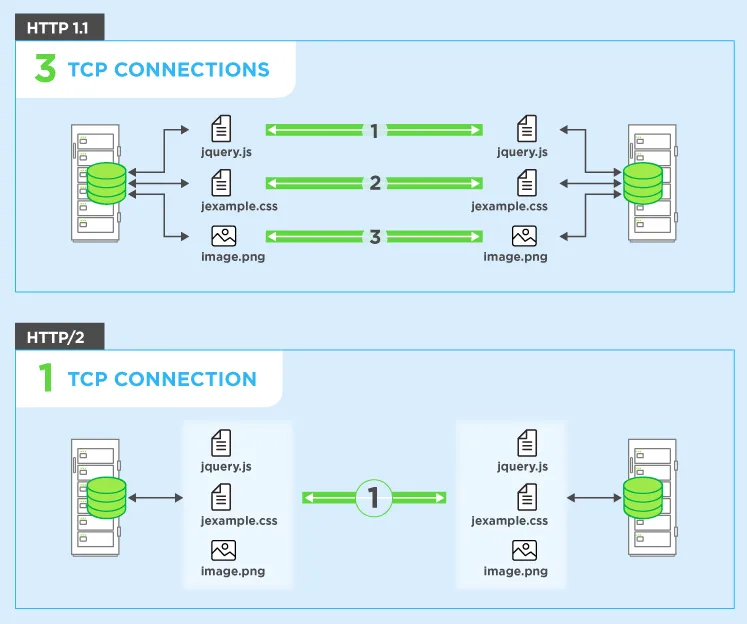
**1.Write a blog on Difference between HTTP1.1 vs HTTP2**

**The Hypertext Transfer Protoco**l (HTTP) is the backbone of the World Wide Web, allowing web servers and clients to communicate and exchange information. **HTTP 1.1** has been the dominant version of **HTTP** since its release in 1999, but in recent years, a new version, **HTTP 2**, has been developed to address some of the shortcomings of its predecessor. In this blog, I will explore the differences between **HTTP 1.1** and **HTTP 2**.

1. **Multiplexing**

One of the most significant differences between **HTTP 1.1** and **HTTP 2** is the way they handle multiple requests.

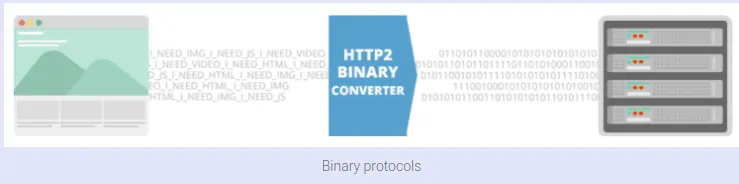
* In **HTTP 1.1**, each request and response requires a separate connection, which can be **slow and inefficient**, especially when dealing with large amounts of data.
* In contrast, **HTTP 2** uses a technique called **multiplexing**, which allows multiple requests to be sent and received over a single connection simultaneously.
* This results in faster and more efficient communication between the server and client.



1. **Binary vs. Text Protocol**

* **HTTP 1.1** uses a **text-based protocol**, which means that all data sent over the connection is in plain text.
* While this makes it easy to read and debug, it also means that the amount of data that can be sent in a single packet is limited.

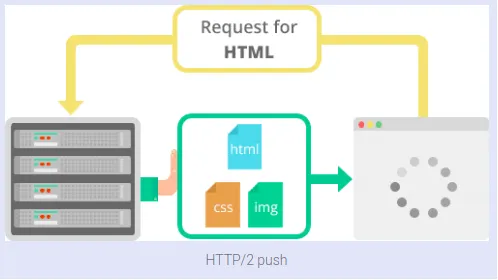
* **HTTP 2**, on the other hand, uses a **binary protocol** that allows for more efficient compression of data and reduces the overhead of the protocol itself.



1. **Server Push**

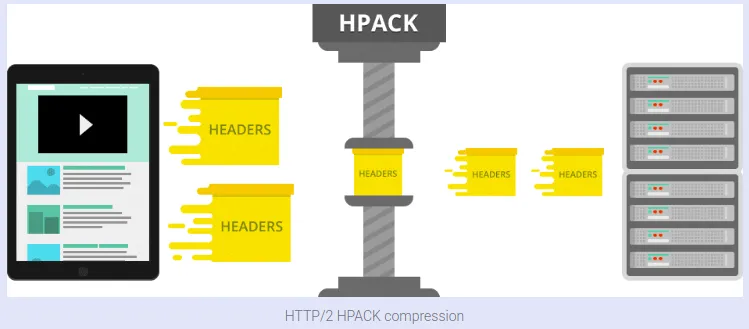
Another significant difference between HTTP 1.1 and HTTP 2 is the way they handle server push.

* In **HTTP 1.1**, a client must explicitly request each resource it needs from the server, even if some of those resources are required to display the page.
* In contrast, **HTTP 2** allows the server to push resources to the client before they are requested.
* This can significantly improve the performance of web pages, especially those that require multiple resources to be displayed.



1. **Header Compression**

* **HTTP 1.1** sends headers with every request and response, which can add significant overhead to the communication.
* **HTTP 2** addresses this issue by compressing headers, reducing the amount of data that needs to be sent over the connection.
* This can result in faster communication between the server and client and reduce the bandwidth requirements for the website.



1. **Security**

* Finally, while both **HTTP 1.1** and **HTTP 2** support the same security protocols, **HTTP 2** requires that all connections be secured using **Transport Layer Security (TLS) or Secure Sockets Layer (SSL).**
* This means that all communication between the server and client is encrypted, providing an additional layer of security.

In conclusion, HTTP/2 offers several significant improvements over HTTP/1.1, including multiplexing, a binary protocol, server push, header compression, and improved security.

While HTTP/1.1 is still widely used, and will likely remain in use for some time, HTTP/2 is the future of the web, offering faster and more efficient communication between servers and clients.

**Difference between HTTP 1.1 and HTTP 2 are:**

|  |  |  |
| --- | --- | --- |
| **S.No** | **HTTP 1.1** | **HTTP 2** |
| 1. | The uses works on the **textual format.** | It works on the **binary protocol**. |
| 2. | 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. |
| 3. | It uses requests resource Inlining for use getting multiple pages | It uses **PUSH frame** by server that collects all multiple pages |
| 4. | It compresses data by itself. | It uses **HPACK** for data compression. |

**2.Write a blog about 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.
* Objects are a structure of K:V pair
* Syntax : var objectname={keyname:value};

**var** studentname={

**name** : "john doe",

**age** : 32,

**year** : 2018,

**address** : "san fransico"

};

* Object name is also called as the reference name
* Object don't have indexes
* Accessing the elements inside the Object
* we can access the elements of the objects using keyname

**Two types of accessing;**

Method-1 : DOT method (.)

* Syntax: objectname.keyname
* **console.log**(studentname.age);

Method-2: Box Method[]

* Syntax: objectname["keyname"]
  + **console.log**(studentname[age]);
* Without Key We don't have values

**Object looping concept:**

* Applicable in looping
* Printing the Objects
* **for-in loop**- It is applicable only for objects (exception: even for arrays too)
* Syntax: for ( **var** key in **objectname**){//code };
* It is a Uni-direction (Only forward)
* It uses box method for the printing of elements
  + **console.log**(studentname[age]);

**for** (**var** **a** in **studentname**){

**console.log(a**,**studentname**[**a**]);

}

1. **Insertion:**

* Here a new **Key : Value** pair will be added at the end of the line.
* Syntax**: objectname .** keyname=value;

Example;

* I need to add gender:"male"
* studentname.gender="male";
* console.log(studentname);

**var** studentname={

**name** : "john doe",

**age** : 32,

**year** : 2018,

**address** : "san fransico"

**genter** :”male” };

1. **Updation:**

* it will be applicable for the key which is already present
* Syntax**: objectname .** keyname=value;

Example;

* + - studentname.address="california";
    - console.log(studentname);

**var** studentname={

**name** : "john doe",

**age** : 32,

**year** : 2018,

* + **(address** : "san fransico") to **address** : "califonia"

};

1. **Deletion:**

* Syntax: delete **objectname . keyname**

Example;

* delete studentname.age;
* console.log(studentname);

**var** studentname={

**name** : "john doe",

* **age** : 32, age will be totally delete.

**year** : 2018,

**address** : "califonia" };