**1. Difference between HTTP1.1 vs HTTP2**

**HTTP 1.1**

1. In HTTP 1.1 the loads are transferred one after the other. If any one data is not loaded, the other data is blocked.

2. Data is transferred in single queue.

3. In HTTP 1, the server is not allowed to push content to the client before client ask for it.

4. HTTP 1.1 does not have HPACK compression mechanism to eliminate redundant information in http header packets.

**HTTP/1.1:** For better understanding, let’s assume the situation when you make a request to the server for the geeksforgeeks.html page & server responds to you as a resource geeksforgeeks.html page. before sending the request and the response there is a TCP connection established between client & server. again, you make a request to the server for image img.jpg & the server gives a response as an image img.jpg. the connection was not lost here after the first request because we add a keep-alive header which is the part of the request so there is an open connection between the server & client. there is a persistent connection which means several requests & responses are merged in a single connection. These are the drawbacks that lead to the creation of HTTP/2: The first problem is HTTP/1.1 transfer all the requests & responses in the plain text message form. The second one is head of line blocking in which TCP connection is blocked all other requests until the response does not receive. all the information related to the header file is repeated in every request.

**HTTP 2**

1. HTTP 2 uses single TCP connection to send multiple streams of data at once. If anyone data is not loaded, it will not affect other data.

2. Data is splitted using binary code and numbering these data will help client to know which stream each data belongs to.

3. In HTTP 2, the server is allowed to push contents to the client before client ask for it.

4. HTTP 2 uses more advanced compression method called HPACK compression that eliminated redundant information in http header packets, which results in faster loading.

**HTTP/2:** HTTP/2 was developed over the SPDY protocol. HTTP/2 works on the binary framing layer instead of textual that converts all the messages in binary format. it works on fully multiplexed that is one TCP connection is used for multiple requests. HTTP/2 uses HPACK which is used to split data from header. it compresses the header. The server sends all the other files like CSS & JS without the request of the client using the PUSH frame.

| **HTTP/1.1** | **HTTP/2** |
| --- | --- |
| It works on 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 In lining 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. |

1. **IP address, port, HTTP methods, MAC address**

**IP address:**

IP stands for Internet Protocol. IP address is a numerical label that is given to a device to identify the device address in internet communication. There are IPV4 - 32 bits and IPv6 - 128 bits address.

An IP address works in helping your device, whatever you are accessing the internet on, to find whatever data or content is located to allow for retrieval. Common tasks for an IP address include both the identification of a host or a network, or identifying the location of a device. An IP address is not random.

IP addresses are generated through a hierarchical system involving the IANA, RIRs and ISPs. Common IP security threats include hijacking, blacklisting, and DDoS attacks. Users can protect their IP address by using firewalls, keeping software updated, using VPNs, and enabling two-factor authentication.

An IP address has two parts: the network ID, comprising the first three numbers of the address, and a host ID, the fourth number in the address. So, on your home network — 192.168. 1.1, for example – 192.168. 1 is the network ID, and the final number is the host ID.

**Types of IP Address**

IP Address is of two types:

**1. IPv4:**Internet Protocol version 4. It consists of 4 numbers separated by the dots. Each number can be from 0-255 in decimal numbers. But computers do not understand decimal numbers, they instead change them to binary numbers which are only 0 and 1. Therefore, in binary, this (0-255) range can be written as (00000000 – 11111111). Since each number N can be represented by a group of 8-digit binary digits. So, a whole IPv4 binary address can be represented by 32-bits of binary digits. In IPv4, a unique sequence of bits is assigned to a computer, so a total of (2^32) devices approximately = 4,294,967,296 can be assigned with IPv4.

IPv4 can be written as:

*189.123.123.90*

**2. IPv6:**But, there is a problem with the IPv4 address. With IPv4, we can connect only the above number of 4 billion devices uniquely, and apparently, there are much more devices in the world to be connected to the internet. So, gradually we are making our way to **IPv6 Address**which is a 128-bit IP address. In human-friendly form, IPv6 is written as a group of 8 hexadecimal numbers separated with colons (:). But in the computer-friendly form, it can be written as 128 bits of 0s and 1s. Since, a unique sequence of binary digits is given to computers, smartphones, and other devices to be connected to the internet. So, via IPv6 a total of (2^128) devices can be assigned with unique addresses which are actually more than enough for upcoming future generations.

IPv6 can be written as:

*2011:0bd9:75c5:0000:0000:6b3e: 0170:8394*

**Port**:

Port are virtual points that denotes the start and end of the connection. Ports are software based and managed by OS.

A port is a virtual or physical endpoint on a network device or computer that identifies a connection and directs data to a service. Ports are associated with a specific process or service and are assigned a number.

## **Types of Ports**

Ports are further divided into three categories:

* Well Known Port
* Registered port
* Dynamic Port

Here are some examples of ports and their associated services:

* **Port 20 and 21**: File Transfer Protocol (FTP)
* **Port 22**: Secure Shell (SSH)
* **Port 25**: Simple Mail Transfer Protocol (SMTP)
* **Port 53**: Domain Name System (DNS)
* **Port 80**: Hypertext Transfer Protocol (HTTP)
* **Port 443**: Hypertext Transfer Protocol Secure (HTTPS)

Ports are part of the transport layer of the OSI model of the internet. They allow multiple applications to use the network simultaneously without any overlapping. The device uses the port number to route the incoming data to the intended application.

## **Importance of Ports:**

Ports have many significances. Some of them are-

* **Identification of service-** Different application/services that work on the same device can be differentiated by their port numbers. For example, [HTTP](https://www.geeksforgeeks.org/difference-between-http-and-https-2/)(Port number 80) and SMTP(port number 25) in the same computer uses different port number to ensure their data goes to the correct service
* **Efficient Data Routing-** When a network device receives data from different places it uses port numbers to efficiently route those data packets to the respective application
* **Block traffic from specific applications/services-** When we have to block incoming or outgoing traffic from a specific application/service then we need to install a firewall and specify the port number of that application/service. We block traffic from/to some specific applications/services when we find any potential threats from those applications/services
* **Scalability of services-** Many services can run simultaneously on the same device and can be differentiated using their port number. This helps the device to scale and support many services at the same time.

**HTTP:**

Hyper Text Transfer Protocol is used to load webpages in hypertext links. It is an application layer protocol designed to share information in networked devices.

* **Basic Structure**: HTTP forms the foundation of the web, enabling data communication and file sharing.
* **Web Browsing**: Most websites use HTTP, so when you click on a link or download a file, HTTP is at work.
* **Client-Server Model**: HTTP works on a request-response system. Your browser (client) asks for information, and the website’s server responds with the data.
* **Application Layer Protocol**: HTTP operates within the Internet Protocol Suite, managing how data is transmitted and received.

## **HTTP Request / Response**

Communication between clients and servers is done by **requests** and **responses.**

1. A client (a browser) sends an **HTTP request** to the web
2. A web server receives the request
3. The server runs an application to process the request
4. The server returns an **HTTP response** (output) to the browser
5. The client (the browser) receives the response

## **HTTP Request**

HTTP request is simply termed as the information or data that is needed by Internet browsers for loading a website. This is simply known as HTTP Request.

There is some common information that is generally present in all HTTP requests. These are mentioned below.

* HTTP Version
* URL
* HTTP Method
* HTTP Request Headers
* HTTP Body

## **HTTP Response**

HTTP Response is simply the answer to what a Server gets when the request is raised. There are various things contained in HTTP Response, some of them are listed below.

* HTTP Status Code
* HTTP Headers
* HTTP Body

## **Characteristics of HTTP**

HTTP is IP based communication protocol that is used to deliver data from server to client or vice-versa.

* The server processes a request, which is raised by the client, and also server and client know each other only during the current bid and response period.
* Any type of content can be exchanged as long as the server and client are compatible with it.
* Once data is exchanged, servers and clients are no longer connected.
* It is a request and response protocol based on client and server requirements.
* It is a connection-less protocol because after the connection is closed, the server does not remember anything about the client and the client does not remember anything about the server.
* It is a stateless protocol because both client and server do not expect anything from each other but they are still able to communicate.

**MAC:**

Media Access Control is a sometimes known as physical address. It is a 12-digit alphanumeric address used to identify the electronic devices connected to the network.

* MAC address is the physical address, which uniquely identifies each device on a given network. To make communication between two networked devices, we need two addresses: **IP address and MAC address.** It is assigned to the NIC (Network Interface card) of each device that can be connected to the internet.
* It stands for **Media Access Control**, and also known as **Physical address, hardware address, or BIA (Burned in Address).**
* It is globally unique; it means two devices cannot have the same MAC address. It is represented in a hexadecimal format on each device, such as **00:0a:95:9d:67:16.**
* It is 12-digit, and 48 bits long, out of which the first *24 bits are used for****OUI*** *(Organization Unique Identifier),* and *24 bits are for NIC/vendor-specific.*
* It works on the data link layer of the OSI model.
* It is provided by the device's vendor at the time of manufacturing and embedded in its NIC, which is ideally cannot be changed.
* The **ARP protocol** is used to associate a logical address with a physical or MAC address.

**3.Objects and its internal representation in Javascript**

**Objects**:

In JavaScript Objects are values which are written as key : value pairs. Almost everything in JavaScript are objects except for primitive types. The primitive types are hard loaded data.

**representation**:

let person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

here firstName is key and John is the value.

Objects can also be return in the below format.

const person = new Object ();

person.firstName = "John";

person.lastName = "Doe";

person.age = 50;

person.eyeColor = "blue";