What is HTTP? Why is HTTP/2 faster than HTTP/1.1?

HTTP stands for hypertext transfer protocol, and it is the basis for almost all web applications. More specifically, HTTP is the method computers and servers use to request and send information. For instance, when someone navigates to cloudflare.com on their laptop, their web browser sends an HTTP request to the Cloudflare servers for the content that appears on the page. Then, Cloudflare servers send HTTP responses with the text, images, and formatting that the browser displays to the user.

What is HTTP/1.1?

The first usable version of HTTP was created in 1997. Because it went through several stages of development, this first version of HTTP was called HTTP/1.1. This version is still in use on the web.

What is HTTP/2?

In 2015, a new version of HTTP called HTTP/2 was created. HTTP/2 solves several problems that the creators of HTTP/1.1 did not anticipate. In particular, HTTP/2 is much faster and more efficient than HTTP/1.1. One of the ways in which HTTP/2 is faster is in how it prioritizes content during the loading process.

What is prioritization?

In the context of web performance, prioritization refers to the order in which pieces of content are loaded. Suppose a user visits a news website and navigates to an article. Should the photo at the top of the article load first? Should the text of the article load first? Should the banner ads load first?

Prioritization affects a webpage's load time. For example, certain resources, like large JavaScript files, may block the rest of the page from loading if they have to load first. More of the page can load at once if these render-blocking resources load last.

In addition, the order in which these page resources load affects how the user perceives page load time. If only behind-the-scenes content (like a CSS file) or content the user can't see immediately (like banner ads at the bottom of the page) loads first, the user will think the page is not loading at all. If the content that's most

important to the user loads first, such as the image at the top of the page, then the user will perceive the page as loading faster.

How does prioritization in HTTP/2 affect performance?

In HTTP/2, developers have hands-on, detailed control over prioritization. This allows them to maximize perceived and actual page load speed to a degree that was not possible in HTTP/1.1.

HTTP/2 offers a feature called weighted prioritization. This allows developers to decide which page resources will load first, every time. In HTTP/2, when a client makes a request for a webpage, the server sends several streams of data to the client at once, instead of sending one thing after another. This method of data delivery is known as multiplexing. Developers can assign each of these data streams a different weighted value, and the value tells the client which data stream to render first.

Imagine that Alice wants to read a novel that her friend Bob wrote, but both Alice and Bob only communicate through the regular mail. Alice sends a letter to Bob and asks Bob to send her his novel. Bob decides to send the novel HTTP/1.1-style: He mails one chapter at a time, and he only mails the next chapter after receiving a reply letter from Alice confirming that she received the previous chapter. Using this method of content delivery, it takes Alice many weeks to read Bob's novel.

Now imagine that Bob decides to send Alice his novel HTTP/2-style: In this case, he sends each chapter of the novel separately (to stay within the postal service's size limits) but all at the same time. He also numbers each chapter: Chapter 1, Chapter 2, etc. Now, Alice receives the novel all at once and can assemble it in the correct order on her own time. If a chapter is missing, she may send a quick reply asking for that specific chapter, but otherwise the process is complete, and Alice can read the novel in just a few days.

In HTTP/2, data is sent all at once, much like Bob when he sends Alice multiple chapters at once. And just like Bob, developers get to number the chapters in HTTP/2. They can decide if the text of a webpage loads first, or the CSS files, or the JavaScript, or whatever they feel is most important for the user experience.

What are the other differences between HTTP/2 and HTTP/1.1 that impact performance?

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 one 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.

What is HTTP/3?

HTTP/3 is the next proposed version of the HTTP protocol. HTTP/3 does not have wide adoption on the web yet, but it is growing in usage. The key difference between HTTP/3 and previous versions of the protocol is that HTTP/3 runs over QUIC instead of TCP. QUIC is a faster and more secure transport layer protocol that is designed for the needs of the modern Internet.

Objects and its internal representation in JavaScript

JavaScript is designed on a simple object-based paradigm. An object is a collection of properties, and a property is an association between a name (or *key*) and a value. A property's value can be a function, in which case the property is known as a method.

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. We can 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:

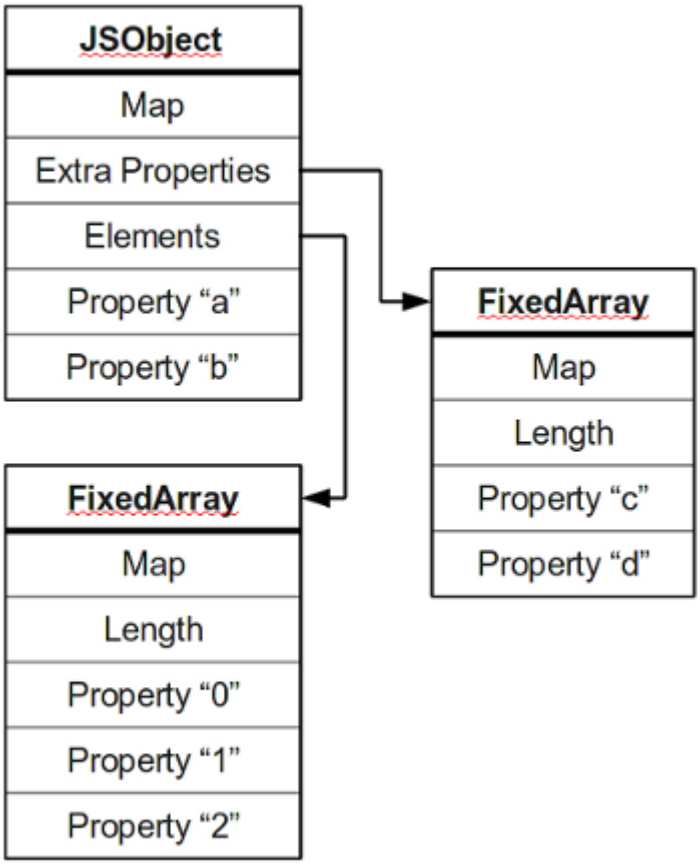
```
var myCar = new Object();
myCar.make = 'Ford';
myCar.model = 'Mustang';
myCar.year = 1969;
```

The above example could also be written using an **object initializer**, which is a comma-delimited list of zero or more pairs of property names and associated values of an object, enclosed in curly braces ({}):

```
var myCar = {
make: 'Ford',
model: 'Mustang',
year: 1969
};
```

JavaScript's internal representation of Objects:

A simple diagram is probably the best way to give a quick overview of the object representation in Javascript.



Most objects contain all their properties in a single block of memory *(‘a’ and ‘b’)*. All blocks of memory have a pointer to a map, which describes their structure.

Named properties that don’t fit in an object are usually stored in an overflow array *(‘c’ and ‘d’)*.

Numbered properties are stored separately, usually in a contiguous array.

The JavaScript standard allows developers to define objects in a very flexible way, and it is hard to come up with an efficient representation that works for everything. An object is essentially a collection of *properties*: basically key-value pairs. We can access properties using two different kinds of expressions:

- obj.prop
- obj[“prop”]

According to the spec, property names are always strings. If we use a name that is not a string, it is implicitly converted to a string. This may be a little surprising: if we use a number as a property name, it gets converted to a string as well. So a JavaScript object is basically a map from strings to values.

What Is a MAC Address?

Multiple hardware and software elements work together every day to connect us to the internet and get data to our devices. Hardware devices like routers and cables transmit the data we need, while software like border gateway protocol (BGP) and internet protocol (IP) addresses direct those data packets to and from those devices. Without both working together, we couldn't get online.

One of those critical elements is the media access control (MAC) address. MAC addresses are associated with specific devices and assigned to them by the manufacturer.

Other names used for MAC addresses include:

- Networking hardware address
- Burned-in address (BIA)
- Physical address
- Ethernet hardware address (EHA)

Wi-Fi, Bluetooth, and Ethernet connections all use MAC addresses.

MAC addresses work with the card in your device that lets it connect wirelessly to the internet, called a Network Interface Controller (NIC). MAC addresses are used to identify which device is which on your local network so that data gets sent to your computer and not your roommate's smartphone.

How Does a MAC Address Work?

When data packets from the internet hit your router, that router needs to be able to send them to the right device on its network. It does this using MAC addresses, assigning a private IP address to each network-connected device based on that device's MAC address. This is different from the IP address your internet service provider (ISP) assigns you---that's your public IP address.

Your router tracks outbound data requests so that when the data comes back, it can attach the correct private IP to the data packets, then send them along to whichever device's MAC address matches that private IP.

Devices can have more than one MAC address because they get one for every place they can connect to the internet. If your laptop has an ethernet port and Wi-Fi, for example, it would have different MAC addresses for the Wi-Fi connection and the Ethernet connection. Bluetooth also uses its own MAC address.

What is an IP Address?

All the computers of the world on the Internet network communicate with each other with underground or underwater cables or wirelessly. If I want to download a file from the internet or load a web page or literally do anything related to the internet, my computer must have an address so that other computers can find and locate mine in order to deliver that particular file or webpage that I am requesting. In technical terms, that address is called **IP Address or Internet Protocol Address**.

Let us understand it with another example, like if someone wants to send you a mail then he/she must have your home address. Similarly, your computer too needs an address so that other computers on the internet can communicate with each other without the confusion of delivering information to someone else's computer. And that is why each computer in this world has a unique IP Address. Or in other words, an IP address is a unique address that is used to identify computers or nodes on the internet. This address is just a string of numbers written in a certain format. It is generally expressed in a set of numbers for example 192.155.12.1. Here each number in the set is from 0 to 255 range. Or we can say that a full IP address ranges from 0.0.0.0 to 255.255.255.255. And these IP addresses are assigned by IANA(known as Internet Corporation For Internet Assigned Numbers Authority).

But what is Internet protocol? This is just a set of rules that makes the internet work. You are able to read this article because your computer or phone has a unique address where the page that you requested (to read this article from GeeksforGeeks) has been delivered successfully.

The working of IP addresses is similar to other languages. It can also use some set of rules to send information. Using these protocols we can easily send, and receive data or files to the connected devices. There are several steps behind the scenes. Let us look at them

Working of IP addresses

- Your device directly requests your Internet Service Provider which then grants your device access to the web.
- And an IP Address is assigned to your device from the given range available.

- Your internet activity goes through your service provider, and they route it back to you, using your IP address.
- Your IP address can change. For example, turning your router on or off can change your IP Address.
- When you are out from your home location your home IP address doesn't accompany you. It changes as you change the network of your device.

Classification of IP Address

An IP address is classified into the following types:

1. Public IP Address: This address is available publicly and it is assigned by your network provider to your router, which further divides it to your devices. Public IP Addresses are of two types,

- **Dynamic IP Address:** When you connect a smartphone or computer to the internet, your Internet Service Provider provides you an IP Address from the range of available IP Addresses. Now, your device has an IP Address and you can simply connect your device to the Internet and send and receive data to and from your device. The very next time when you try to connect to the internet with the same device, your provider provides you with different IP Addresses to the same device and also from the same available range. Since IP Address keeps on changing every time when you connect to the internet, it is called a Dynamic IP Address.

- **Static IP Address:** Static address never changes. They serve as a permanent internet address. These are used by DNS servers. What are DNS servers? Actually, these are computers that help you to open a website on your computer. Static IP Address provides information such as device is located on which continent, which country, which city, and which Internet Service Provider provides internet connection to that particular device. Once, we know who is the ISP, we can trace the location of the device connected to the internet. Static IP Addresses provide less security than Dynamic IP Addresses because they are easier to track.

2. Private IP Address: This is an internal address of your device which are not routed to the internet and no exchange of data can take place between a private address and the internet.

3. Shared IP addresses: Many websites use shared IP addresses where the traffic is not huge and very much controllable, they decide to rent it to other similar websites so to make it cost-friendly. Several companies and email sending servers use the same IP address (within a single mail server) to cut down the cost so that they could save for the time the server is idle.

4. Dedicated IP addresses: A dedicated IP Address is an address used by a single company or an individual which gives them certain benefits using a private Secure Sockets Layer (SSL) certificate which is not in the case of a shared IP address. It allows to access the website or log in via File Transfer Protocol (FTP) by IP address instead

of its domain name. It increases the performance of the website when the traffic is high. It also protects from a shared IP address that is black-listed due to spam.

Lookup IP addresses

To know your public IP, you can simply search “What is my IP?” on google. Other websites will show you equivalent information: they will see your public IP address because, by visiting the location, your router has made an invitation/request and thus revealed the information. the location IP location goes further by showing the name of your Internet Service Provider and your current city.

Finding your device’s private IP Address depends on the OS or platform you are using.

- **On Windows:** Click Start and type “cmd” in the search box and run the command prompt. In the black command prompt dialog box type “ipconfig” and press enter. You will be able to see your IP Address there.
- **On Mac:** Go to system preferences and select Network, you will be able to see the information regarding your network which includes your IP Address.

What is Ports in Networking?

Whenever any application in one computer sends data to another application of a different computer then it sends using IP Address and MAC Address but how does our computer know that this data is for a specific application and this data is sent by any specific application? There comes the concept of Port.
For instance, imagine your MAC Address or IP Address as the PIN code of the nearest Post Office and your house address as a Port. Whenever any parcel is sent to you it gets received by the nearest post office and then it is identified by your address where to deliver that parcel. Similarly in a computer data is first received using their IP or MAC address then it is delivered to the application whose port number is with the data packets.
Port is a logical address of a 16-bit unsigned integer that is allotted to every application on the computer that uses the internet to send or receive data.
Now every time any application sends any data, it is identified by the port that which the application sent that data and the data is to be transferred to the receiver application according to its port. We often call port as port number.
In the OSI Model ports are used in the Transport layer. In the headers of Transport layer protocols like TCP and UDP, we have a section to define port(port number). The network layer has to do nothing with ports, their protocols only care about IP Addresses.
Ports are assigned by computer i.e. operating system to different applications. Ports help computer to differentiate between incoming and outgoing traffic. Since the port is a 16-bit unsigned number it ranges from 0 to 65535.

Types of Ports

Ports are further divided into three categories:

- Well Known Port

- Registered port
- Dynamic Port

Well Known Port

- It is from the range 0 to 1023
- It is reserved for common and specifically used service
- It is used by some widely adopted protocols and services like HTTP(port 80), FTP(port 21), DNS(Port 53), SSH(port 22), etc…..

Registered Port

- It is from range 1024 to 49151
- These are used by applications or services that are not as common
- But it is used by those applications or services which require its specific port
- Organizations can ask IANA(Internet Assigned Number Authority) for any specific port number within this range

Dynamic Port

- It is from range 49152 to 65535
- It is also known as Ephemeral or Private Port
- It is used for those connections that are temporary or short-lived
- It is not registered or assigned and can be used by any process

Importance of Ports

Ports have many significance. 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(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 request methods

HTTP defines a set of **request methods** to indicate the desired action to be performed for a given resource. Although they can also be nouns, these request methods are sometimes referred to as *HTTP verbs*. Each of them implements a different semantic, but some common features are shared by a group of them: e.g. a request method can be safe, idempotent, or cacheable.

GET

> The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.

HEAD

> The HEAD method asks for a response identical to a GET request, but without the response body.

POST

> The POST method submits an entity to the specified resource, often causing a change in state or side effects on the server.

PUT

> The PUT method replaces all current representations of the target resource with the request payload.

DELETE

> The DELETE method deletes the specified resource.

CONNECT

> The CONNECT method establishes a tunnel to the server identified by the target resource.

OPTIONS

> The OPTIONS method describes the communication options for the target resource.

TRACE

> The TRACE method performs a message loop-back test along the path to the target resource.

PATCH

The PATCH method applies partial modifications to a resource.