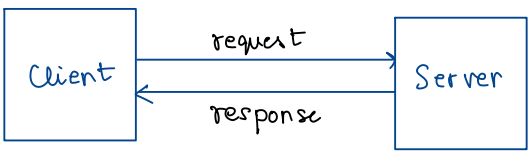
HTML 1 : Web Fundamentals & Intro to HTML

How Web Works



All the computers or systems which are connected to the internet are called clients and servers

→ **Clients** are typically web-users / internet-connected devices which are making request

I want this
page song movie ... etc.

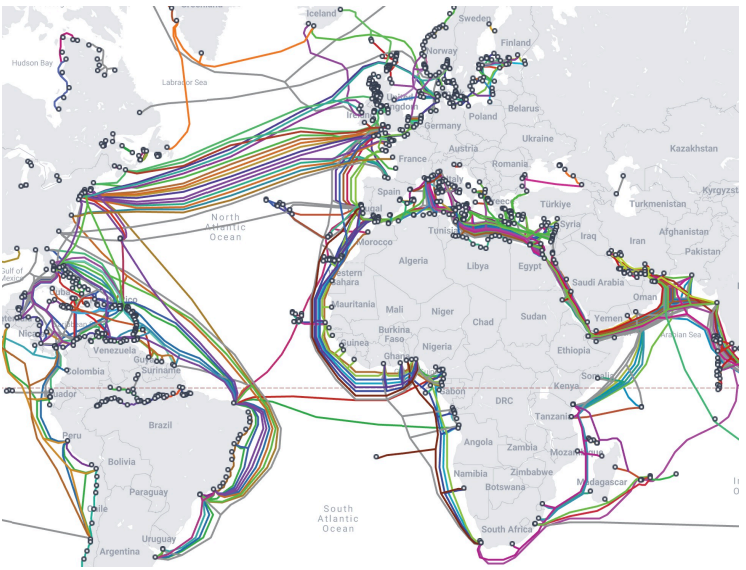
→ Simply put, anything from where the request originates is called the client.

→ **Servers** are systems which have all the data / information and issues responses

→ We can say that **internet connection** is **network of networks**

↳ We say so because, all the countries in the world, they are connected via fibre optic cables running at the sea bed

when the signals travel from one end of the cable to the other nearly at the speed of light



This is why we are able to connect to other parts of the world almost instantly!

Trivia

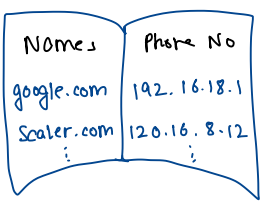
→ There are people like Elon Musk who have been trying to move towards satellites instead of these cables! (Starlink)

→ This is because some times ships / aquatic animals can damage these cables. Some countries in warfare as well have been reported damaging these cables to break / sabotage internet

TCP / IP are some communication protocols that define how data should travel across the internet

→ Just like when we are out on the road, there are some protocols that everyone should follow, for proper movement of traffic

DNS (Domain Name System) is like a directory, much like the



telephone book in the olden days, which consists of mapping between the website (name) and their respective IP Address

→ Every website is mapped to an IP address, which the browser needs to figure out whenever you name-drop that website.

→ It's much like when you call your 'friend' from contact list, your mobile needs to figure out what the actual contact number of your 'friend' is.

HTTP (HyperText Transfer Protocol) is a set of protocols / rules for transferring 'hyper text'

↓

The HTML documents that we're transferring from client to server

→ HTTP is the language of the browser for communication b/w client & server

Putting it all together

→ When we enter a ① URL (uniform resource locator)

↓

② https://scaler.com / domain name

↓ translated to

③ IP address (by DNS server)

→ But the browser will not repeat the same architecture every time.

→ If we open a URL very frequently, it won't hit DNS server each time to fetch IP.

→ Instead, there are optimizations in place, for eg the browser / router will save the frequently visited URL's (caching)

↙

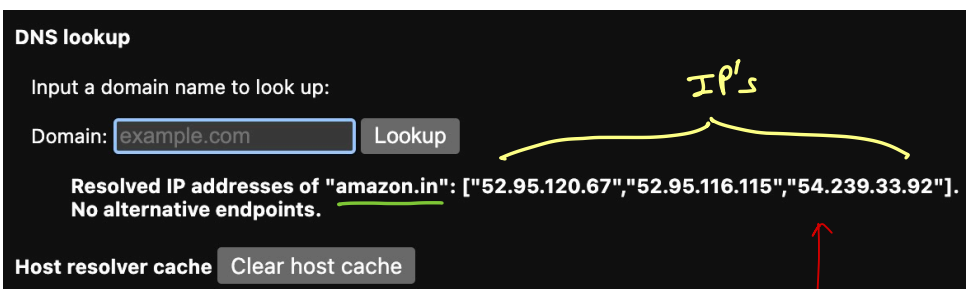
Our OS, ISP's like Airtel also does some caching

→ If the browser cache is not present, browser will talk to OS cache, if not even there, it goes to ISP's cache. Ultimately if not even there, then it hits the DNS server

→ This DNS server is configured as soon as we connect to the WiFi, such that the browser knows how to contact

Check this out!

Chrome has a DNS lookup feature, where you can see the IP's for the server's domain name provided



Any company might have multiple server's, hence IP's and depending upon which server is available to respond faster, we get the returned IP's (like a building with multiple entry gates)

→ Over the period of time, these listed IP's were received by the browser, so it cached those

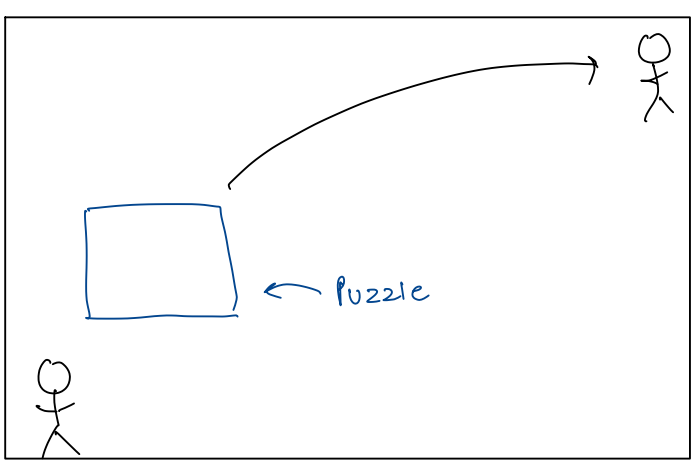
④ HTTP messages will now be sent from client to server, basically an exchange where the client sends a request to the server asking it send a copy of the website (HTML/CSS/JS)

⑤ All this exchange/communication between two or multiple systems, we need to have a protocol → TCP/IP

↳ The interacting computers/systems need to know ahead-of-time how they are expected to communicate

- how to start the communication
- how will system know that message sent was transferred correctly or not
- how will the communication end?

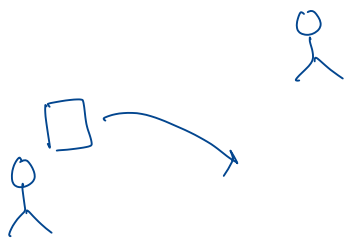
Practical Example



Description → You and your friend are standing very far apart from each other in a large room.

→ You have to pass a puzzle which you have to your friend

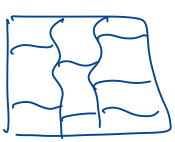
→ Since it's a very big room, there are high chances that it might fall somewhere in between



Approach ?

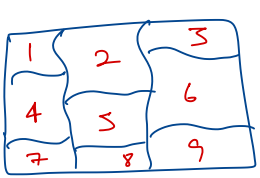
→ You do an interesting thing. Since it's a puzzle, it has smaller pieces in it. You break it into smaller pieces

→ First optimization



→ Next, you number the individual pieces

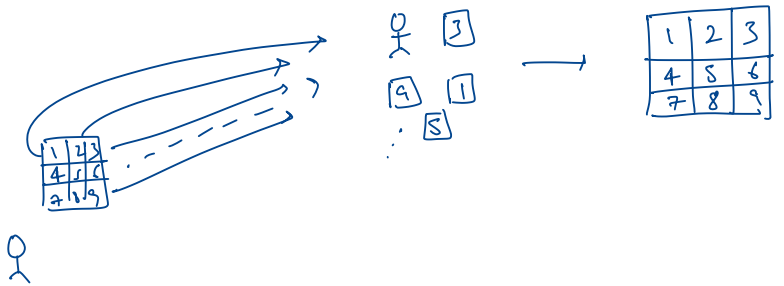
→ second optimization



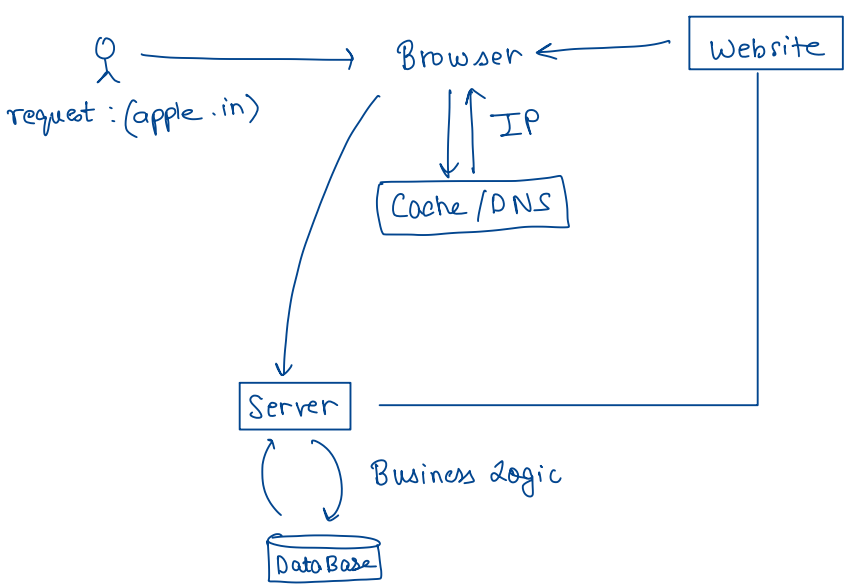
→ Now, you start throwing these pieces individually, not in any necessary order.

→ Since these are all smaller pieces, you can throw easily and will be able to transmit easily

→ Your friend on the other hand, receives them and based on the number, picks them up and recreates what you wanted to send in the first place



→ This is what takes place by the virtue of TCP/IP



Additional Read (Optional)

When you access a website, your browser communicates with a web server to retrieve the necessary files (like HTML, CSS, and JavaScript) that make up the webpage. These files are transmitted over the internet using the TCP/IP protocol suite, which breaks down data into packets for efficient and reliable delivery. Here's how the process works:

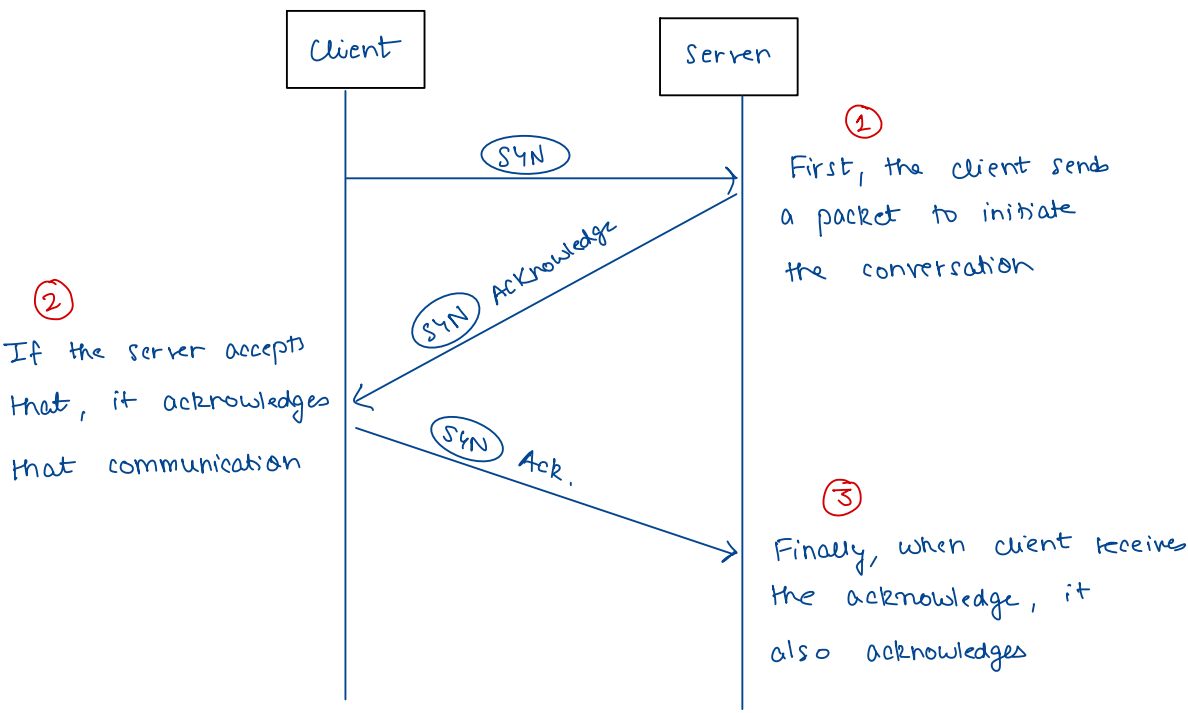
1. **Breaking Down Files into Packets**:
   - **Application Layer**: Your browser sends an HTTP request to the server. The server prepares an HTTP response containing the requested files.
   - **Transport Layer (TCP)**: The server's operating system receives the data from the application layer and segments it into smaller chunks. This segmentation is necessary because networks have a maximum transmission unit (MTU) size, typically around 1500 bytes for Ethernet networks.
   - **TCP Segmentation**: The data is divided into TCP segments. Each segment includes a portion of the file's data and a TCP header containing essential control information like sequence numbers.
   - **Network Layer (IP)**: Each TCP segment is encapsulated within an IP packet. The IP layer adds its own header, including source and destination IP addresses.
2. **Contents of the Packets**:
   - **TCP Header**: Contains information such as source and destination ports, sequence numbers, acknowledgment numbers, flags (like SYN, ACK, FIN), window size for flow control, and a checksum for error checking.
   - **IP Header**: Includes source and destination IP addresses, version (IPv4 or IPv6), header length, total packet length, identification for reassembly, time-to-live (TTL), and protocol (indicating TCP).
   - **Data Payload**: The actual chunk of the file's data being transmitted.
3. **Reassembly of Packets**:
   - **Transport Layer at the Client**: When the packets reach your device, the IP layer checks the IP headers to ensure they are meant for your device and passes the data up to the TCP layer.
   - **Sequence Numbers**: The TCP layer uses the sequence numbers in the TCP headers to reassemble the segments in the correct order, even if they arrive out of sequence.
   - **Error Checking and Retransmission**: TCP checks for errors using checksums. If a packet is missing or corrupted, the client requests retransmission.
   - **Data Delivery**: Once all segments are correctly reassembled, the TCP layer passes the complete data stream up to the application layer (your browser).
4. **Time Taken for the Process**:
   - **Latency Factors**: The time depends on factors like network latency (distance between client and server), bandwidth, network congestion, and the size of the files.
   - **Typical Timeframes**: For a well-optimized website and a decent internet connection, the initial page load might take between 0.5 to 2 seconds. Subsequent resource loading (like images or scripts) can occur concurrently, improving overall load times.
   - **Optimizations**: Techniques like HTTP persistent connections, content compression (gzip), content delivery networks (CDNs), and caching mechanisms are used to reduce load times.

Server Response

→ Generally, the server sends the HTML/CSS/JS as response

How does the communication start b/w the server and the client?

→ Before any real communication takes place between the client and server, a three way handshake is done, which is also known as the TCP/IP handshake.



① First, the client sends a packet to initiate the conversation

② If the server accepts that, it acknowledges that communication

③ Finally, when client receives the acknowledge, it also acknowledges

HTML / CSS

→ Suppose, you had to build a shopping mall

1) **Blueprint** (Wireframes)

Figma → these tools are used to create wireframes, for how the website will eventually look

↓

design is done pre-coding by UX engineers

→ planning of interface

→ what will be the flow?

→ how will login flow look like?

→ what are the different options & screens?

→ to buy a product, from selecting till checkout, what are the different screens that user encounters?

2) **Structure** (Brick-Mortar / cementing)

→ building bit by bit / brick by brick (skeleton that will support the structure)

→ this is where HTML creation happens

→ when the structure is ready, we start painting it, make it beautiful → CSS comes into picture

3) **Interactivity**

↳ to make shopping mall come to life.

↳ JavaScript

Boilerplate

→ <!DOCTYPE html> to inform browser we're using HTML 5

→ meta charset UTF-8, to let browser support many languages etc.

→ meta viewport, to make webpage mobile compatible

What is SEO?

→ It's a technique, which affects the ranking for a website in the website results displayed when a query is given

→ Semantic tags play a role in the SEO for that website

→ All this is done to make the search engine understand the website better, at the same time making the markup more readable and intuitive.