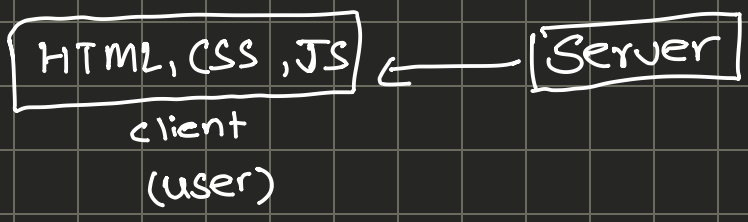
Project :- Multithreaded Web Server

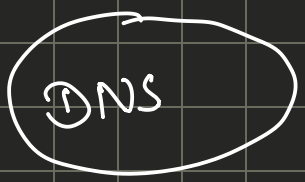


Request Cycle

client → www.google.com

192.215. . . . :8080 (IPv4 / v6) address

443 Secured connection.



Key → value mapping

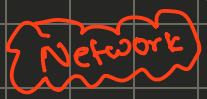
Data Flow

Client 101101 → Server

← 10 11101

medium used for the flow of data (Rules / Protocols).

Client → DNS



Server

client

TCP/UDP ✓ → Three handshake are needed for communication.

Server

TCP → Data matters and also its Sequence.

UDP → used in video streaming.

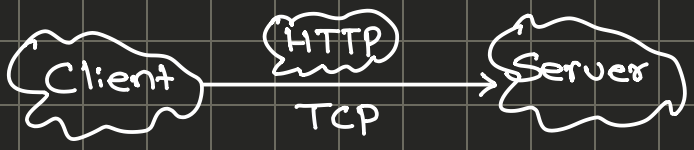
3 way handshake:

Client ——————→ Server

i) Client will send sync flag to the Server

ii) Socket in the server would send back Sync Accept to the client once server accept the sync.

iii) Finally, client will send acknowledgement to the server, and the communication channel is established between client and the Server.

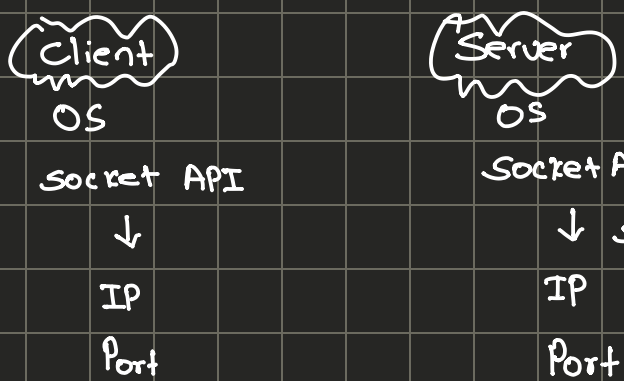


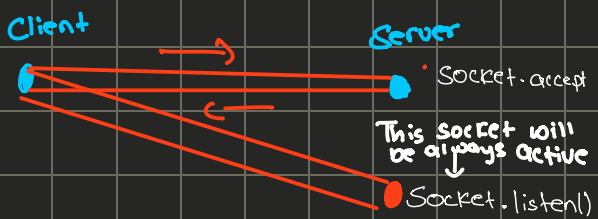
HTTP 1.0 → Non-Persistance. (Once the data is received by the client the socked is closed and the client disconnects with the server)

1.1 → Persistance.

↓

The socket would not be closed until a timeout until then a communication channel is established between the client and the Server. Note:- If there is no time out function then the client has to send Finish to the server manually, and the server would send back acknowledgement to the client and socked would be closed.

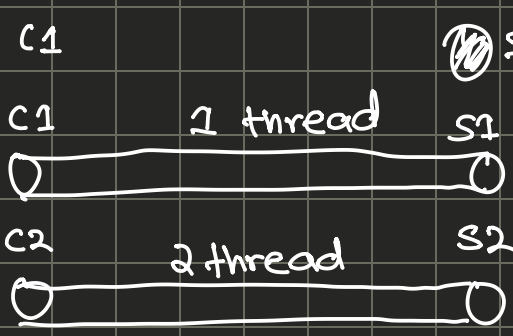




Note:- when the socket.accept() happens, it happens in other socket.

Thread :- Basically a sequence of instructions that can run independently within a program.

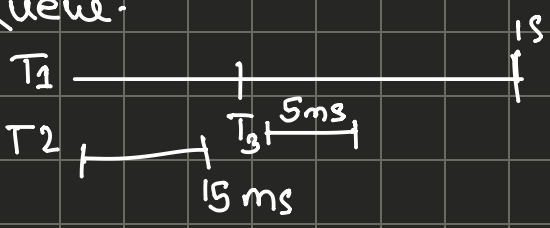
multi - threaded



In multithreading, each single thread would handle the communication between client and the server independently.

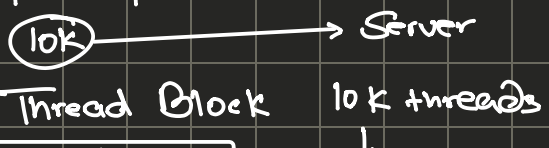
Context switching.

Basically works like a priority Queue.



* Algorithms used for context Switching
  i) Round Robins
  ii) First in First Out.

10K request per minute



A Time for which the thread has already worked

* In-memory lots of resources needed

Solution:-

Thread Pool