__/__/__

- What happens when we hit URL in browser ?

- What constitutes a URL ?

Human-readable way to know what we are interested in domain?

Optional KV pairs we can pass to additional optional info in the request

https://www.google.com/api/search?q=home

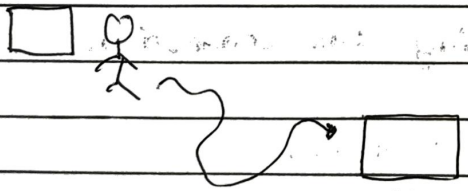
scheme

path

Tells browser which protocol to use while connecting other scheme: http, ws, etc

on the product we are accessing, I am interested in this path / resource.

- The DNS ~~to~~ resolution.



google.com

17.53.21.253

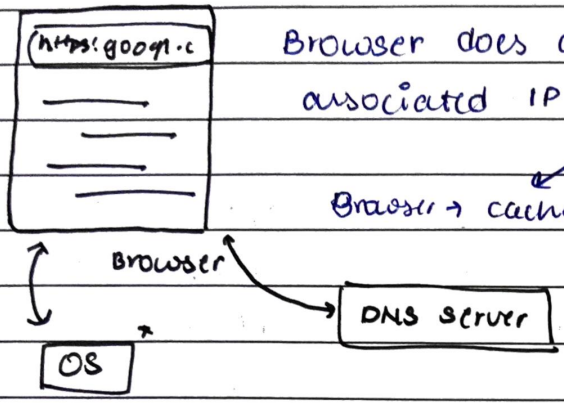
Every machine on the internet has an "address" enabling us to reach it over the network,

↳ this is IP address

* easier to remember "google.com" than the IP address. Hence, we need a way that converts

google.com ⟶ 17.53.21.253

__/__/____



Browser does a DNS lookup to get the associated IP address.

IP of google → not going to change

Browser → cached. DNS information is heavily cached

in the browser ~~in the~~

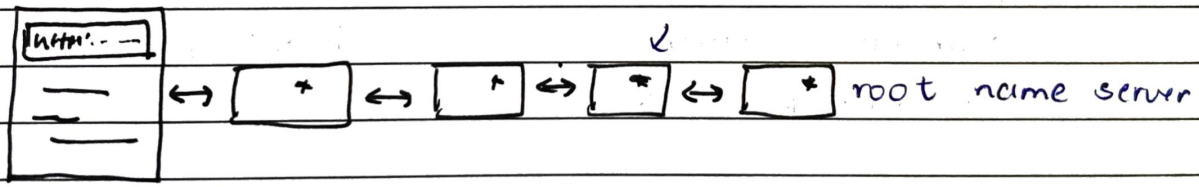
in the operating system

across all machines of DNS resolution.

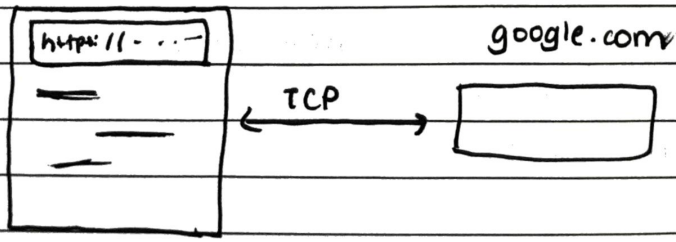
What looks like a simple call, actually involves lots of machine

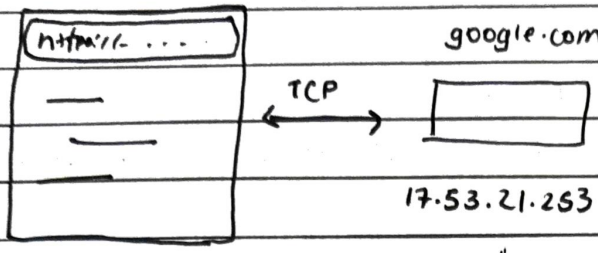
iterative / recursive

cached

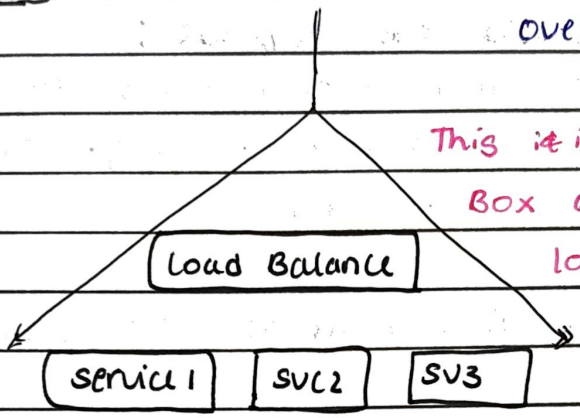


- After this resolution process, browser has the IP address to connect to establishing the connection.





Browser now establishes a TCP connection with the machine [server] and can now talk to it over the network.



This it in itself is a Pandora's Box and constitutes of 1000's of machines

Sending the request :-

other protocol possible

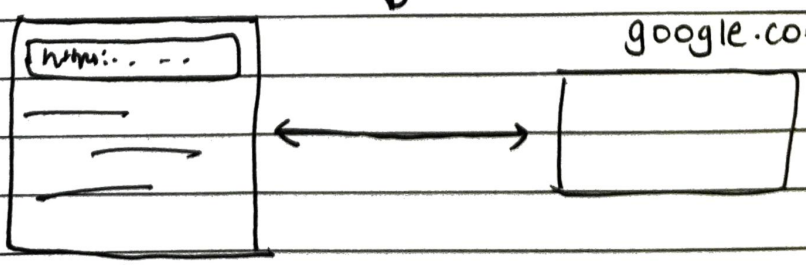
Browser now compliers the request into HTTP specification and sends it across to the server.

```
GET /api/search?q=home HTTP/1.1
Host : www.google.com
connection : Keep-alive
```

gives we are just hitting URL in browser it fires an HTTP GET to the server

→ HEADERS

↳ instructions to server + meta data



• why protocol exist "

__/__/____

+ HTTP is protocol that specifies :

1. how to pack the data
2. what to do before, during & after the request

• Server procuses the request :

Once the server ~~reci~~ receives the HTTP request, it parses the above message and understands what needs to be done.

* Server may just load the file from local disk & server
it may make call to database to get responses
it may throw error it malformed.

it complies a proper HTTP responses
and respond back over same of TCP connection
(html, text, JSON, etc)

```
HTTP/1.1  200  OK
Content-Type : text/html
Content-Length : 2042
```

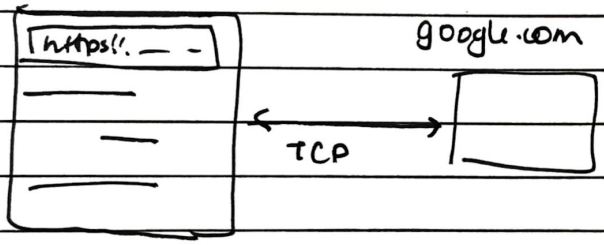
__/__/__

- Browser upon receiving the response :

HTTP/1.1 200K OK — status code	Browser upon recieving the response parses the message, extracts the info and "renders"
content-Type : text/html — type	
content-Length : 2042 — length	
<html> <head> ... — body	

↳ http message

↳ showing html as html
text as text, etc



+ if browser does not support the response type then it downloads the file locally.

- When HTML is rendered, browser may come across

1. linked CSS file
2. img tags to render an image
3. inline Javascript code

↳ it fetches the ~~additionally~~ additional files by going through the exact same process.

↳ starts executing it
[may involve making more HTTP requests]
↳ API calls