When a user enters an URL in the browser, how does the browser fetch the desired result?

Ans:

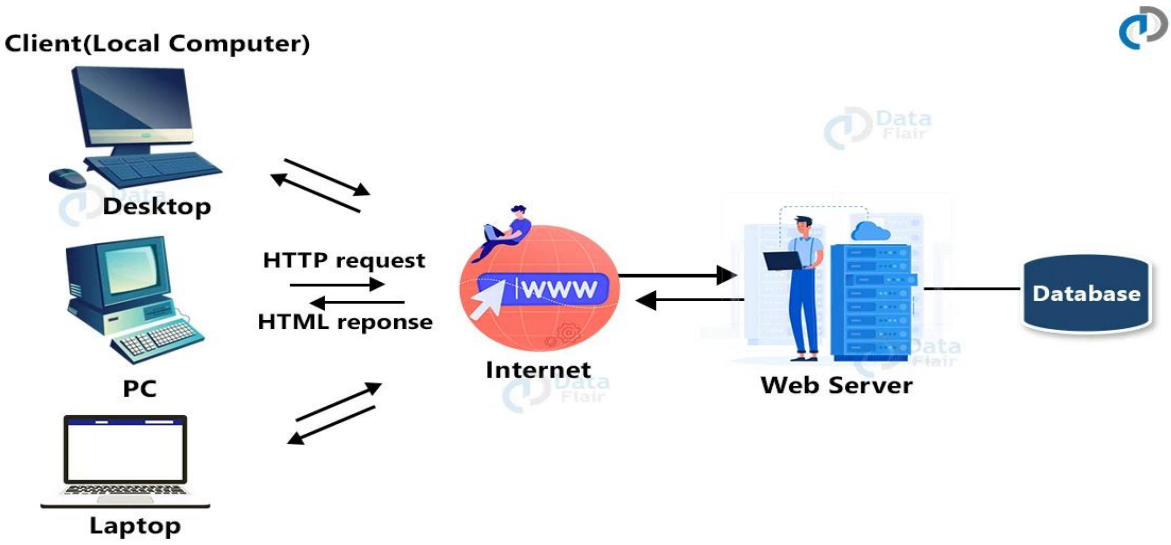
For Get to the Answer, First We need to understand the functionality of a Web Browser.

Web Browser:

- A web browser is an application software that is used to present or receive resources that is traversing through the **World Wide Web** (www)

- The main functions of web browser are to fetch or retrieve informative resources from World Wide Web to the client/ user on demand, translate those files received from web server and display those content to the user and allow the client /user to access all other relevant resources & information via hyperlinks.

- When the user inputs any URL (uniform resource locator) in the web browser, the user is navigated to that website by the browser quickly.



Components of a Web Browser:

1. User Interface

- It is an environment allowing users to use certain features like search bar, refresh button, menu, bookmarks, etc.

2. Browser Engine

- The bridge connects the interface and the engine. It monitors the rendition engine while manipulating the inputs coming from multiple user interfaces.

3. Networking

- The protocol provides an URL and manages all sorts of safety, privacy and communication.
- In addition, the store network traffic gets saved in retrieved documents.

4. Data Storage

- The cookies store information as the data store is an uniform layer that the browsers use. Storage processes like IndexedDB, WebSQL, localStorage, etc works well on browsers.

5. JavaScript Interpreter

- It allows conversion of JavaScript code in a document and the executes it. Then the engine shows the translation on the screen to the users.

Websites, Server and IP Address

- Websites are collections of files, often HTML, CSS And JavaScript and Images, that tells Browsers to how to display your websites. They need to be accessible to anyone from anywhere at any time, so hosting them on your computer at home isn't be scalable or reliable.

- A powerful external computer connected to the Internet, called a server, stores these files.

- When you point out to browser at www.https://example.com/anotherexapmle , browser has to figure out on which server on internet is hosting this site.

- Each server or device have unique number called IP Address

- An IP address contains four numbered parts: 93.184.216.34

- But numbers like this are hard to remember! That's where domain names come in. example.com is much easier to remember than 93.184.216.34.

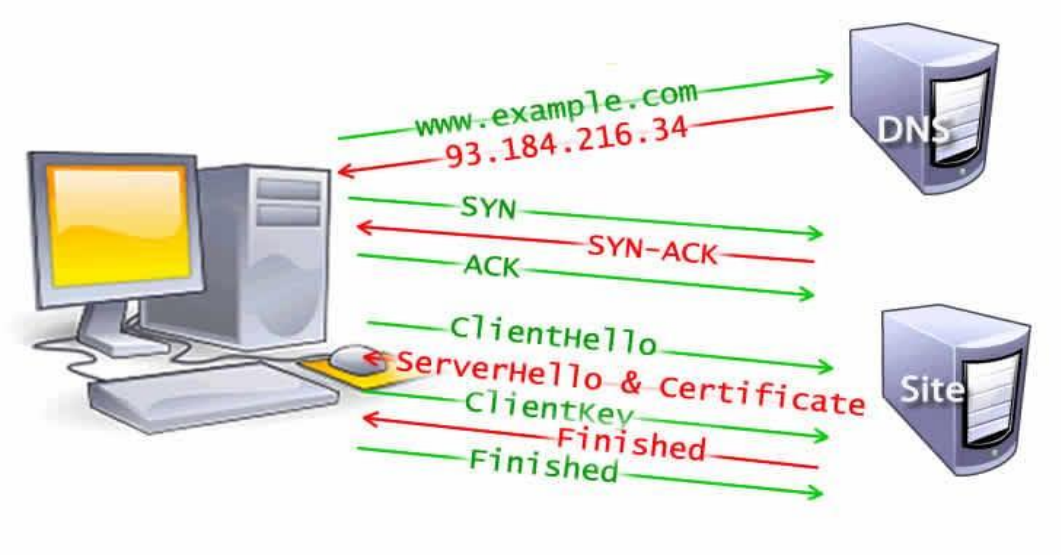
TCP Handshake:

- Once the IP address is known, the browser sets up a connection to the server via a TCP three-way handshake. This mechanism is designed so that two entities attempting to communicate—in this case the browser and web server—can negotiate the parameters of the network TCP socket connection before transmitting data, often over **HTTPS**.

TSL Negotiation:

- For secure connections established over HTTPS, another "**handshake**" is required. This handshake, or rather the **TLS negotiation**, determines which cipher will be used to encrypt the communication, verifies the server, and establishes that a secure connection is in place before beginning the actual transfer of data. This requires three more round trips to the server before the request for content is actually sent.

- While making the connection secure adds time to the page load, a secure connection is worth the latency expense, as the data transmitted between the browser and the web server cannot be decrypted by a third party.

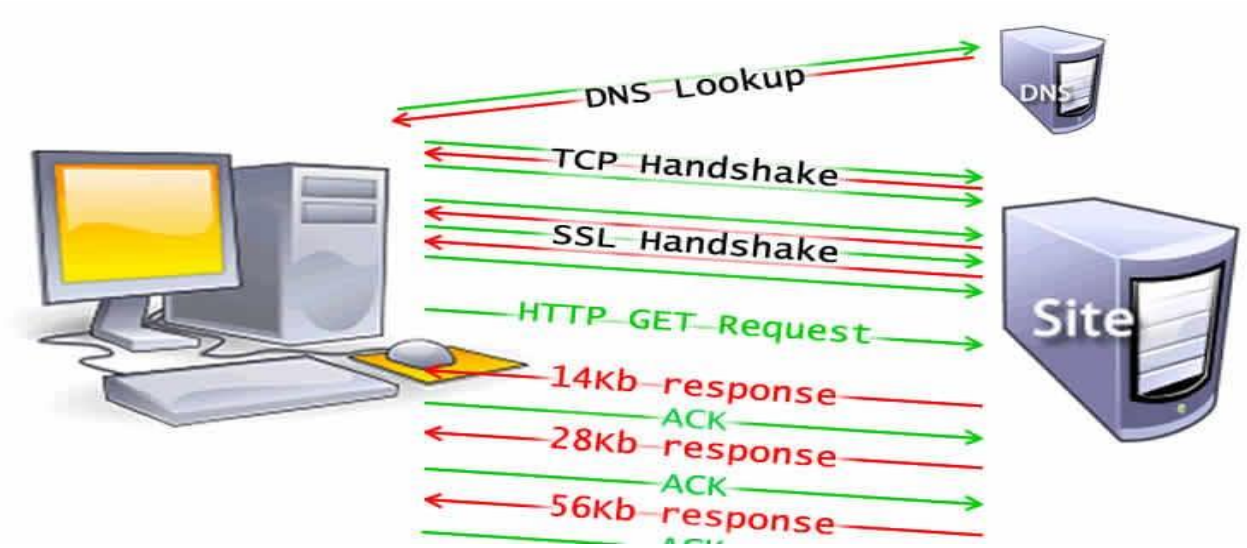


- Once we have an established connection to a web server, the browser sends an initial HTTP GET request on behalf of the user, which for websites is most often an HTML file. Once the server receives the request, it will reply with relevant response headers and the contents of the HTML.

TCP Slow Start / 14kb rule:

- The first response packet will be 14Kb. This is part of TCP slow start, an algorithm which balances the speed of a network connection. Slow start gradually increases the amount of data transmitted until the network's maximum bandwidth can be determined.

- In TCP slow start, after receipt of the initial packet, the server doubles the size of the next packet to around 28Kb. Subsequent packets increase in size until a predetermined threshold is reached, or congestion is experienced.



Critical rendering path:

The Critical Rendering Path is the sequence of steps the browser goes through to convert the HTML, CSS, and JavaScript into pixels on the screen.

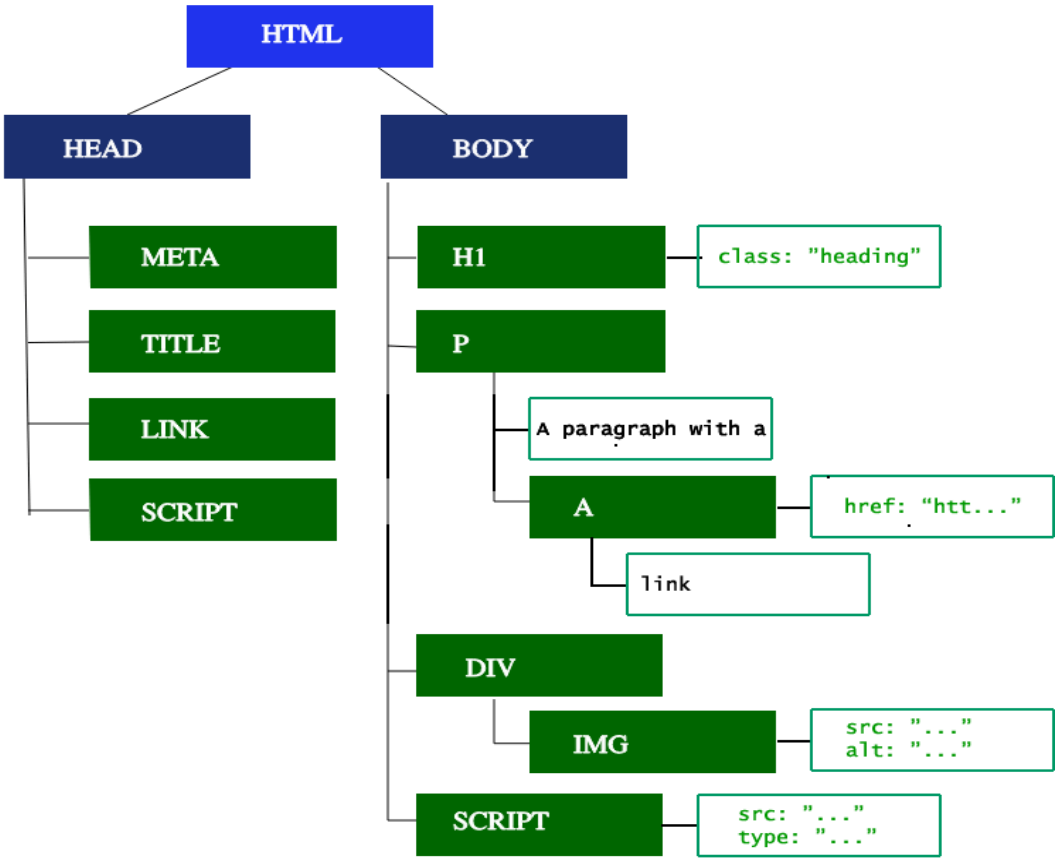
There are five steps in critical rendering path:

1. **Building DOM**
2. **Building CSSOM**
3. **Style**
4. **Layout**
5. **Paint**

1. Building DOM:

- The first step is processing the HTML markup and building the DOM tree

- The DOM tree describes the content of the document. The **<html>** element is the first tag and root node of the document tree. The tree reflects the relationships and hierarchies between different tags. Tags nested within other tags are child nodes. The greater the number of DOM nodes, the longer it takes to construct the DOM tree.



Preload scanner

While the browser builds the DOM tree, this process occupies the main thread. As this happens, the preload scanner will parse through the content available and request high priority resources like CSS, JavaScript, and web fonts. Thanks to the preload scanner, we don't have to wait until the parser

finds a reference to an external resource to request it. It will retrieve resources in the background so that by the time the main HTML parser reaches requested assets, they may possibly already be in flight, or have been downloaded. The optimizations the preload scanner provides reduce blockages.

2. Building the CSSOM:

- The second step in the critical rendering path is processing CSS and building the CSSOM tree. The CSS object model is similar to the DOM. The DOM and CSSOM are both trees. They are independent data structures. The browser converts the CSS rules into a map of styles it can understand and work with. The browser goes through each rule set in the CSS, creating a tree of nodes with parent, child, and sibling relationships based on the CSS selectors.

- As with HTML, the browser needs to convert the received CSS rules into something it can work with. Hence, it repeats the HTML-to-object process, but for the CSS.

- While the CSS is being parsed and the CSSOM created, other assets, including JavaScript files, are downloading (thanks to the preload scanner). JavaScript is interpreted, compiled, parsed and executed.

Render

Rendering steps include style, layout, paint and, in some cases, compositing. The CSSOM and DOM trees created in the parsing step are

combined into a render tree which is then used to compute the layout of every visible element

3. Style

- The third step in the critical rendering path is combining the DOM and CSSOM into a render tree. The computed style tree, or render tree, construction starts with the root of the DOM tree, traversing each visible node.

4.Layout

- The fourth step in the critical rendering path is running layout on the render tree to compute the geometry of each node. Layout is the process by which the width, height, and location of all the nodes in the render tree are determined, plus the determination of the size and position of each object on the page. Reflow is any subsequent size and position determination of any part of the page or the entire document.

- Once the render tree is built, layout commences. The render tree identified which nodes are displayed (even if invisible) along with their computed styles, but not the dimensions or location of each node. To determine the exact size and location of each object, the browser starts at the root of the render tree and traverses it.

5. Paint

- The last step in the critical rendering path is painting the individual nodes to the screen, the first occurrence of which is called the first meaningful paint. In the painting or rasterization phase, the browser converts each box calculated in the layout phase to actual pixels on the screen. Painting involves drawing every visual part of an element to the screen, including text, colors, borders, shadows, and replaced elements like buttons and images. The browser needs to do this super quickly.

- To ensure smooth scrolling and animation, everything occupying the main thread, including calculating styles, along with reflow and paint, must take the browser less than 16.67ms to accomplish