What Happens When you Type "google.com" and Press Enter?

When I type "google.com" into my web browser and press Enter, a complex series of events occurs to display the Google homepage. This process can be broken down into several key stages:

1. DNS Lookup

Step 1: Browser Cache

- The browser first checks its cache to see if it has a recent record of the IP address for "google.com".

Step 2: Operating System Cache

- If the browser cache does not contain the necessary information, the request is forwarded to the operating system's DNS cache.

Step 3: Router Cache

- If the operating system's cache is also empty, the request is sent to the local router, which may have its own DNS cache.

Step 4: ISP DNS Cache

- If the router does not have the information, the request is sent to the ISP's DNS server.

Step 5: Recursive DNS Lookup

- If the ISP's DNS server does not have the information, it performs a recursive DNS lookup, querying other DNS servers on the internet. This involves querying:
    - The root DNS servers
    - The TLD (Top-Level Domain) servers (for ".com" in this case)
    - The authoritative DNS servers for "google.com"

Step 6: DNS Response

- The IP address for "google.com" is returned through the chain: authoritative DNS server → ISP DNS server → router → operating system → browser.

2. TCP Connection

Step 1: Three-Way Handshake

- The browser establishes a TCP connection with the server at the IP address received from the DNS lookup. This involves:

- **SYN**: The browser sends a SYN (synchronize) packet to the server to initiate a connection.
- **SYN-ACK**: The server responds with a SYN-ACK (synchronize-acknowledge) packet.
- **ACK**: The browser sends an ACK (acknowledge) packet back to the server.

3. TLS Handshake (if HTTPS)

Step 1: ClientHello

- The browser sends a "ClientHello" message to the server, specifying supported encryption methods.

Step 2: ServerHello

- The server responds with a "ServerHello" message, choosing the encryption method from the list provided by the client.

Step 3: Server Certificate

- The server sends its SSL/TLS certificate to the browser to verify its identity.

Step 4: Key Exchange

- The browser and server exchange keys to establish a secure connection.

Step 5: Secure Connection Established

- Once the keys are exchanged and verified, a secure encrypted connection is established.

4. HTTP Request

Step 1: Request Line

- The browser sends an HTTP GET request to the server, specifying the desired resource ("/" for the homepage).

Step 2: Headers

- The browser includes additional information in the headers, such as the type of content it can accept, the browser type, and other relevant details.

5. Server Processing

Step 1: Request Handling

- The server processes the incoming HTTP request, determines the requested resource, and prepares a response.

Step 2: Fetch Resource

- The server fetches the requested resource (e.g., HTML file) from its storage.

6. HTTP Response

Step 1: Status Line

- The server sends an HTTP response back to the browser, starting with a status line (e.g., "HTTP/1.1 200 OK").

Step 2: Headers

- The server includes response headers, specifying the type of content, its length, caching policies, and other relevant details.

Step 3: Body

- The server includes the requested resource (e.g., HTML of the Google homepage) in the body of the response.

7. Browser Rendering

Step 1: Parsing HTML

- The browser parses the HTML content received from the server, building a Document Object Model (DOM) tree.

Step 2: Fetching Resources

- The browser identifies additional resources required (e.g., CSS, JavaScript, images) and sends additional HTTP requests to fetch them.

Step 3: Applying Styles

- The browser parses the CSS and applies styles to the HTML elements, calculating the layout.

Step 4: Executing JavaScript

- The browser executes any JavaScript code, which may manipulate the DOM, fetch additional resources, or perform other tasks.

Step 5: Rendering

- The browser renders the final webpage on the screen, combining the layout, styles, and executed scripts.

8. User Interaction

Step 1: User Inputs

- I can now interact with the webpage (e.g., clicking links, and entering search queries).

Step 2: Event Handling

- The browser handles my inputs by executing corresponding JavaScript event handlers, which may result in further HTTP requests, DOM updates, and re-rendering.

This entire process happens within milliseconds, providing a smooth and seamless experience when I type "google.com" and press Enter.