Populating the page: how browsers work

Source : https://developer.mozilla.org/en-US/docs/Web/Performance/How_browsers_work

Overview

Users want web experiences with content that is:

- fast to load.
- smooth to interact with.

Two major issues in web performance are:

- latency
- for the most part, browsers are single-threaded.

Latency

Latency is the biggest threat against ensuring a fast-loading page. Network latency is the time it takes to transmit bytes over the air to computers. We have to enhance web performance i.e. we have to make the page load as quickly as possible.

Single-threaded

For the most part, browsers are considered single-threaded. That is, they execute(실행하다) a task from beginning to end before taking up another task.

Render time is key, ensuring the main thread can complete all the work we throw at it and still always be available to handle user interactions.

Web performance can be improved by understanding the single-threaded nature of the browser and minimizing the main thread's responsibilities, to ensure rendering is smooth and responses to interactions are immediate.

Navigation

Navigation : The first step in loading a web page. It occurs whenever a user requests a page by entering a URL into the address bar, clicking a link, submitting a form, as well as other actions.

One of the goals of web performance is to minimize the amount of time a navigation takes to complete. Latency and bandwidth are foes(적) which can cause delays.

DNS Lookup

The first step of navigating to a web page is finding where the assets for that page are located. If you navigate to https://example.com, the HTML page is located on the server with an IP address of 93.184.216.34. If you've never visited this site, a DNS lookup must happen.

Your browser requests a DNS lookup, which is eventually fielded by a name server, which in turn responds with an IP address. After the initial request, the IP will likely be cached for a time, which speeds up subsequent requests by retrieving(검색하다) the IP address from the cache instead of contacting a name server again.

So DNS lookups usually only need to be done once per hostname for a page load. However, DNS lookups must be done for each unique hostname the requested page references. If your fonts, images, scripts, ads, and metrics all have different hostnames, a DNS lookup will have to be made for each one.

호스트명(hostname) : 네트워크에 연결된 장치(컴퓨터, 파일 서버, 복사기, 케이블 모뎀 등)들에게 부여되는 고유한 이름이다. 특히 인터넷에서는 월드 와이드 웹, 전자 우편, 유즈넷 등에서 호스트명을 흔히 사용하며, 도메인 이름과 유사하지만 엄밀하게는 더 넓은 의미를 가지고 있다.

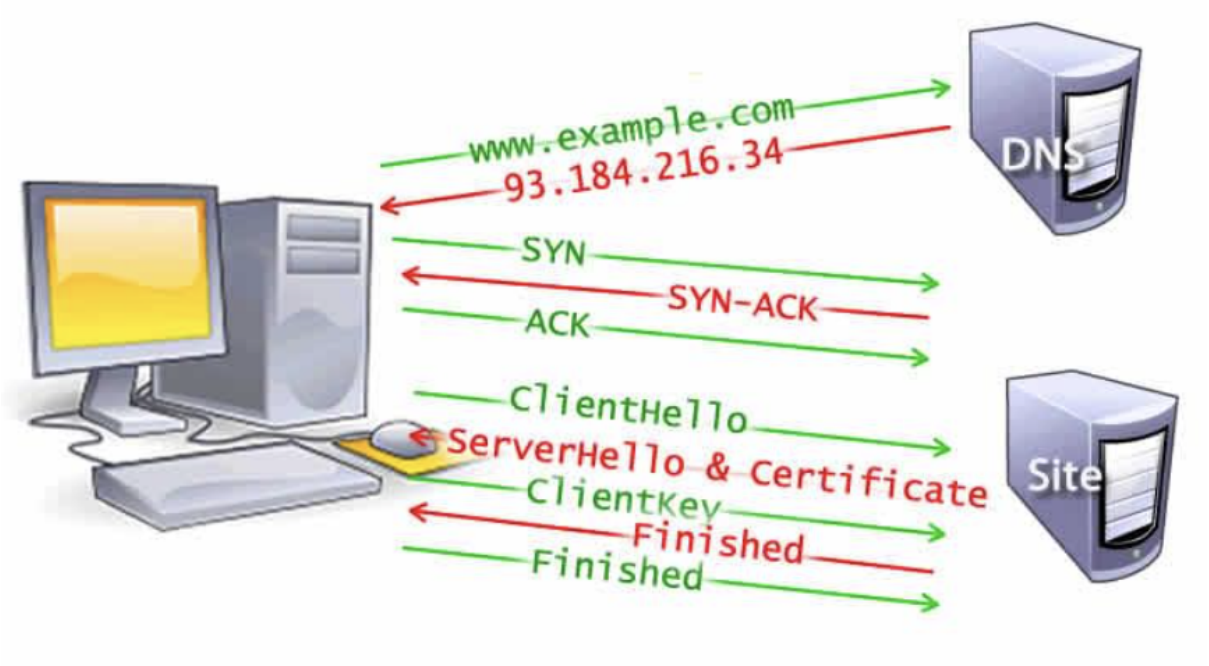
This can be problematic for performance, particularly on mobile networks.

TCP Handshake

Once the IP address is known, the browser sets up a connection to the server via(~을 통해) a TCP three-way handshake.

TLS Negotiation

For secure connections established over HTTPS (not HTTP), TLS negotiation is required. TLS negotiation determines which cipher(암호) 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.



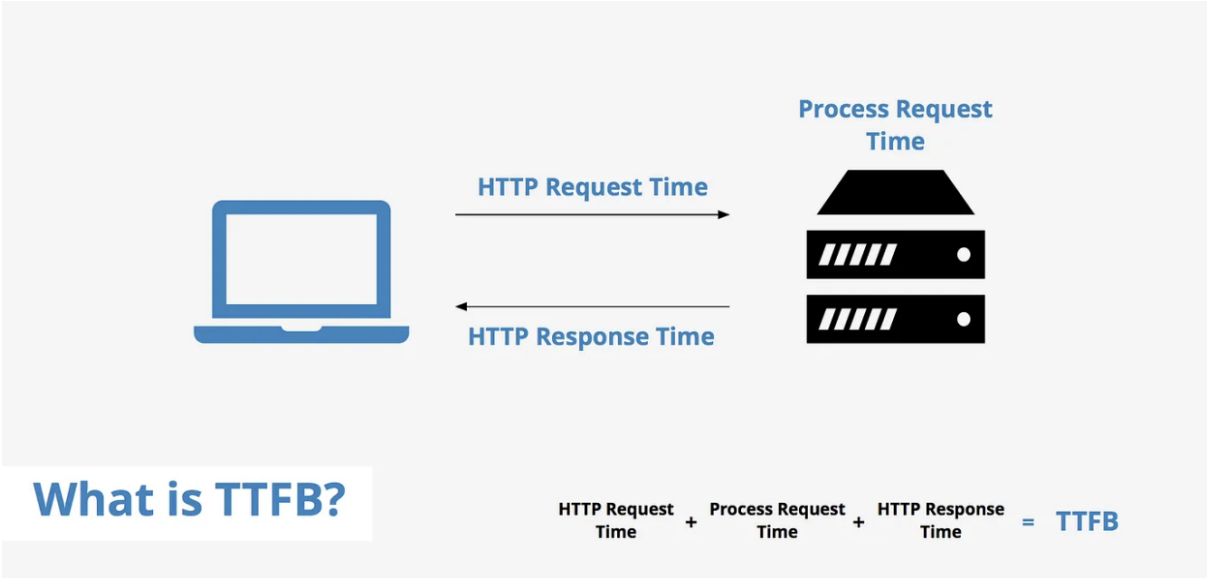
While making the connection secure adds time to the page load, a secure connection is worth the latency expense(비용), as the data transmitted between the browser and the web server cannot be decrypted by a third party.

SSL과 TLS의 차이점은 무엇인가요? :
https://aws.amazon.com/ko/compare/the-difference-between-ssl-and-tls/
“TLS는 SSL의 직접적인 후속이며 이제 모든 버전의 SSL이 더 이상 사용되지 않습니다. 하지만 TLS 연결을 설명하는 데 SSL이라는 용어가 많이 사용됩니다. 대부분의 경우 SSL 및 SSL/TLS라는 용어 모두 TLS 프로토콜과 TLS 인증서를 나타냅니다.”

Response

Once we have an established connection to a web server, the browser sends an initial HTTP GET request, 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.

TTFB (Time to First Byte) : The time between when the user made the request (clicking on a link) and the receipt(수신) of the first packet of HTML.



“TTFB measures the duration from the user or client making an HTTP request to the first byte of the page being received by the client's browser.”
Time to first byte (위키피디아) : https://en.wikipedia.org/wiki/Time_to_first_byte

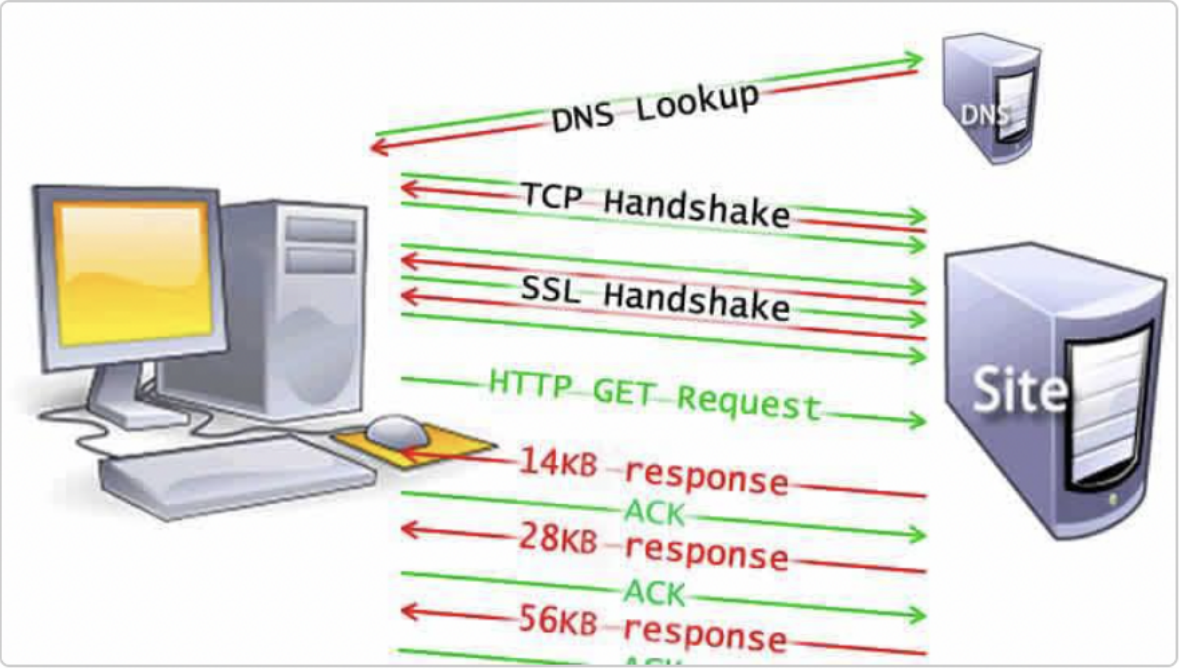
TTFB = HTTP Request Time + Process Request Time + HTTP Response Time
TTFB로 서비스 성능 측정하기 : https://equus3144.medium.com/ttfb%EB%A1%9C-%EC%84%9C%EB%B9%84%EC%8A%A4-%EC%84%B1%EB%8A%A5-%EC%B8%A1%EC%A0%95%ED%95%98%EA%B8%B0-21baef090c7d

TTFB = (요청한 페이지 첫 번째 packet의 첫 번째 바이트를 수신했을 때) - (첫 번째 request를 보내려고 할 때 (click on a link))
Time to First Byte (TTFB) : https://web.dev/ttfb/

TCP Slow Start / 14KB rule

The first response packet will be 14KB. This is part of the TCP slow start algorithm.

TCP slow start : An algorithm which balances the speed of a network connection. Slow start gradually(점차적으로) increases the amount of data transmitted until the network’s maximum bandwidth can be determined.



TCP slow start gradually builds up transmission speeds appropriate for the network's capabilities to avoid congestion.

Congestion control

As the server sends data in TCP packets, the client confirms delivery by returning acknowledgements(인정), or ACKs. The connection has a limited capacity depending on hardware and network conditions. If the server sends too many packets too quickly, they will be dropped. Meaning, there will be no acknowledgement. The server registers(등록하다) this as missing ACKs. Congestion control algorithms use this flow of sent packets and ACKs to determine a send rate.

TCP slow start is part of the congestion control algorithm.
https://en.wikipedia.org/wiki/TCP_congestion_control

Parsing

Once the browser receives the first chunk of data, it can begin parsing the information received. Parsing is the step the browser takes to turn the data it receives over the network into the DOM and CSSOM, which is used by the renderer to paint a page to the screen.

The DOM is the internal representation of the markup for the browser.

Even if the requested page's HTML is larger than the initial 14KB packet, the browser will begin parsing and attempting to render an experience based on the data it has. This is why it's important for web performance optimization to include everything the browser needs to start rendering a page, or at least a template of the page - the CSS and HTML needed for

the first render - in the first 14 kilobytes. But before anything is rendered to the screen, the HTML, CSS and JavaScript have to be parsed.

Render

Interactivity