HTTP

개요

> Hyper Text Transfer Protocol
>
> 브라우저와 서버가 통신할 수 있도록 만들어주는 여러 프로토콜 가운데 한 종류로 **웹 브라우저와 웹 서버 사이에 데이터를 주고 받는데 사용되는 통신**
>
> 초기에는 HTML 문서를 전송하기 위한 목적으로 만들어졌으나 지금에서는 Json 및 Image 파일 등 여러 형태의 데이터를 주고받을 수 있게 됐다

특징

- HTTP 는 애플리케이션 레벨의 프로토콜로 TCP/IP 위에서 동작한다
  - 브라우저에서 URL을 입력하고 조회하면 DNS 서버가 URL의 IP를 찾아 해당 IP의 서버와 TCP 연결을 맺고, 이후 HTTP 요청/응답이 발생한다

> TCP/IP 란?
>
> 일반적으로 전송 계층 프로토콜로 TCP 를 사용하고, 네트워크 계층 프로토콜로 IP 를 사용하는데 이 두 계층을 통틀어 TCP/IP 라고 칭한다.
>
> TCP/IP 에서는 IP 주소를 사용해 통신할 컴퓨터를 결정하고, 포트 번호를 사용해서 해당 컴퓨터의 어떤 프로그램과 통신할지 결정한다. 이때 HTTP에서는 기본적으로 80포트를 사용

- 비연결성 / 무상태성 프로토콜이다

> 비연결성 (Connectionless) / 무상태성 (Stateless) 란?
>
> 비연결성은 클라이언트와 서버가 한 번 연결을 맺은 후 , 클라이언트 요청에 대해 서버가 응답을 마치면 클라이언트의 상태를 저장하않고 맺었던 연결을 끊어 버리는 성질을 의미한다.
>
> 서버는 접속을 유지할 필요가 없고 상태를 저장할 필요 또한 없기 때문에 연결성을 가지는 프로토콜에 비해 최소한의 자원만을 사용하게 된다.
>
> 또한 이전 데이터 요청과 다음 데이터 요청이 서로 무관하다.

HTTP Message

HTTP Request

```
GET /index.html HTTP/1.1 -- Request Start Line

```

```
/* HTTP Header Start */
Host: example.com
User-Agent: Mozilla/5.0
Accept-Language: ko-KR
/* HTTP Header End */

/* Body Start */
/* Body End */
```

- Request Start Line
  - HTTP 메소드
  - 요청 타겟
    - URL
    - Protocol
    - Port
    - Domain
  - HTTP 버전
- HTTP Header
  - General Header
    - 요청과 응답에 모두 적용되지만, 데이터와는 관련이 없는 헤더
      - Date 등
  - Request Header
    - 요청하는 클라이언트에 대한 자세한 정보를 포함하는 헤더
      - Host, User-Agent, Cookie 등
  - Entity Header
    - 콘텐츠의 길이나 MIME 타입과 같이 Entity Body에 대한 자세한 정보를 포함하는 헤더
      - Content-Type, Content-Length
- Body
  - Single-Resource Bodies
  - Multiple-Resource Bodies

HTTP Response

```
HTTP/1.1 200 OK -- Response Status Line

/* HTTP Header Start */
Date: Sat, 09 Oct 2023 14:28:02 GMT
Server: Apache
Content-Type: text/html
/* HTTP Header End */

/* Body Start */
```

```
<html>
...
</html>
/* Body End */
```

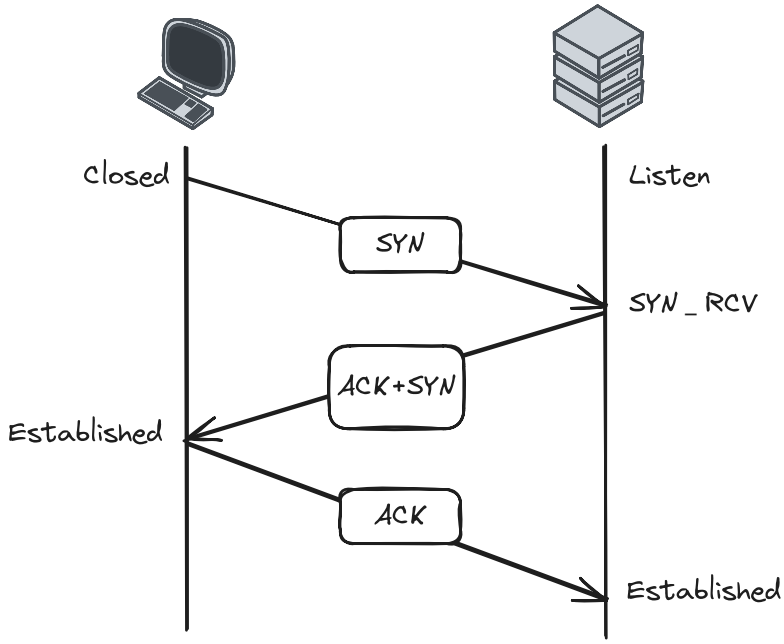
- Status Line
  - Protocol Version
    - 주로 HTTP/1.1 사용
  - 상태 코드
    - 200, 404 400 등
  - 상태 텍스트
    - 상태 코드에 대한 설명
- HTTP Header
  - General Header
    - 요청과 응답에 모두 적용되지만, 데이터와는 관련이 없는 헤더
      - Date 등
  - Request Header
    - 요청하는 클라이언트에 대한 자세한 정보를 포함하는 헤더
      - Host, User-Agent, Cookie 등
  - Entity Header
    - 콘텐츠의 길이나 MIME 타입과 같이 Entity Body에 대한 자세한 정보를 포함하는 헤더
      - Content-Type, Content-Length
- Body
  - Single-Resource Bodies
  - Multiple-Resource Bodies

Appendix

TCP 3-way Handshake 란?

TCP 는 장치들 사이에 논리적인 접속을 성립하기 위하여 3-way handshake 를 사용한다.

3-way handshake 는 TCP/IP 프로토콜을 이용해서 통신을 하는 응용 프로그램이 **데이터를 전송하기 전에 먼저 정확한 전송을 보장하기 위해 상대방 컴퓨터와 사전에 세션을 수립하는 과정을 의미한다.**



[STEP 01]
클라이언트는 서버에 접속을 요청하는 SYN 패킷을 보낸다. 이 때 A클라이언트는 SYN을 보내고 SYN/ACK 응답을 기다리는 **SYN_SENT** 상태가 된다

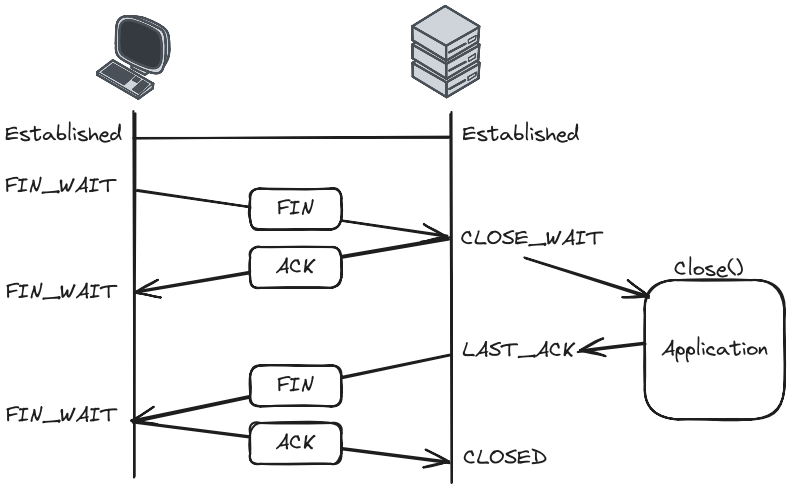
[STEP 02]
서버는 SYN 요청을 받고 클라이언트에게 요청을 수락한다는 ACK와 SYN Flag 가 설정된 패킷을 발송한다. 이후 클라이언트가 다시 ACK로 응답하기를 기다린다.

이때 서버는 **SYN_RECEIVED** 상태가 된다

[STEP 03]
클라이언트는 서버에게 ACK를 보내고 서버가 이를 받으면, 연결이 이루어지고 데이터가 오가게 된다. 이때 서버 상태는 **ESTABLISHED** 상태가 된다

TCP 4-way Handshake 란?

4-way Handshake 는 세션을 종료하기 위해 수행되는 절차다.



[STEP 01]
클라이언트가 연결을 종료하겠다는 FIN Flag 를 서버로 전송한다

[STEP 02]
서버는 확인메시지를 보내고 자신의 통신이 끝날때까지 기다리는데 이 상태가 **TIME_WAIT** 상태다

> TIME_WAIT
>
> 만약 서버에서 FIN을 전송하기 전에 전송한 패킷이 FIN보다 늦게 도착하는 상황이 발생할 경우를 대비하기 위해 클라이언트는 서버로부터 FIN을 수신하더라도 일정시간 동안 세션을 남겨놓고 잉여 패킷을 기다리는 과정을 거치게 된다.

[STEP 03]
서버가 통신이 끝났으면 연결이 종료되었다고 클라이언트에게 FIN Flag 를 전송한다

[STEP 04]
클라이언트는 확인했다는 메시지를 보낸다