HTTP / HTTPS

목차

1. HTTP
    - HTTP 요청 / 응답 구성
    - HTTP의 특성
    - 메시지
    - 상태 코드
    - 헤더
2. HTTPS
    - 사용 이유
    - 기타 특징
    - 사이트 인증 절차

HTTP

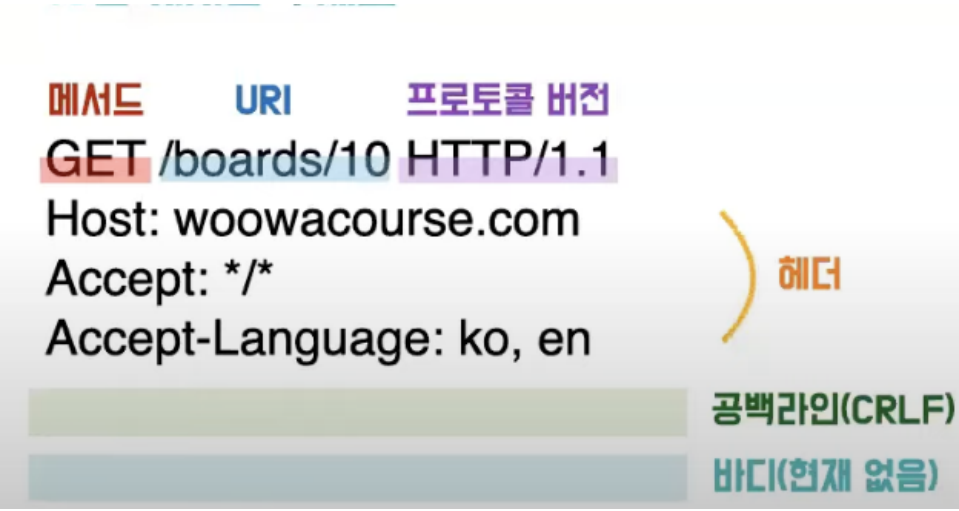
Hyper**t**ext **T**ransfer **P**rotocol

서로 다른 시스템 사이에서 메시지를 교환하기 위한 프로토콜로, 서버와 클라이언트에서 데이터를 주고받는 용도로 많이 사용

HTTP 프로토콜 요청 / 응답

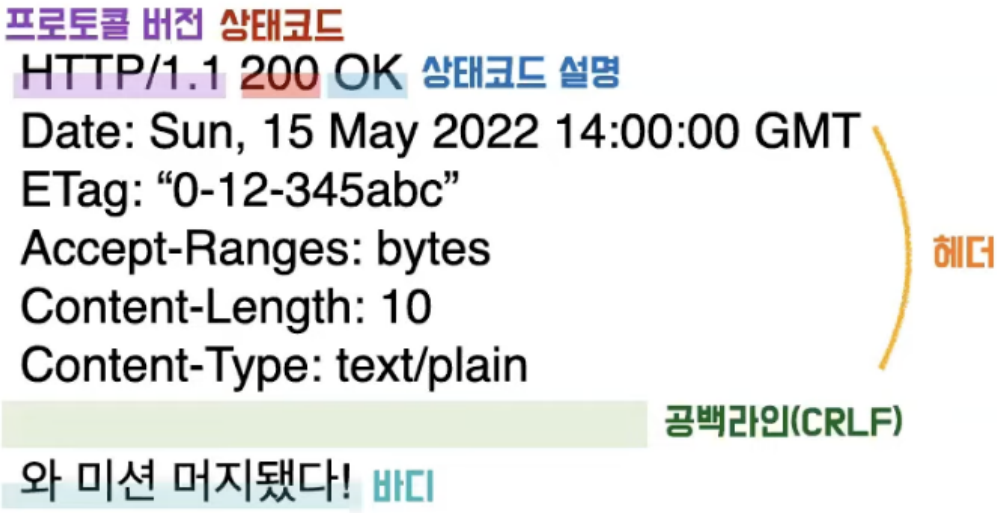
Request

- 메서드, URI, 프로토콜 버전, 헤더, 바디로 구성



Response

- 프로토콜 버전, 상태 코드와 설명, 헤더, 바디로 구성



HTTP 프로토콜 특성

1. Stateless

과거 정보를 남기지 않고 새로운 요청을 보낼 때마다 새로운 리스폰스를 보내준다.

장점: 상태를 저장하지 않으므로, 서버를 확장하기 편하다.

> 💡 상태 유지가 필요한 경우 (장바구니 결제 등) 쿠키, 세션, 토큰등을 이용한다.

2. 지속연결

TCP 커넥션을 유지합니다. 요청마다 새로운 커넥션을 만들고 종료하지 않습니다.

- 장점: 서버의부하를 줄이고 통신을 빠르게 할 수 있습니다.

추가) 따라서 응답을 기다리지 않고 바로 요청을 보내는 파이프라이닝도 가능하다. 하지만 성능 향상은 별로 없고 버그가 많아서 최신 브라우저에서는 활성화되지 않음

HTTP 메서드

멱등성(idempotent)

(수학, 전산학) 연산을 여러번 적용하더라도 결과가 달라지지 않는 성질
HTTP 에서는 여러번 요청(연산)한 결과 서버의 상태가 항상 동일한 경우를 뜻한다.

메서드별로 성립 여부가 다르다.

GET

특정 리소스의 표시를 요청한다. 오직 데이터를 받기만 함

본문을 담을 수 있긴 한데 안 담는게 바람직함 .

query를 통해 전달하고 싶은 정보를 전달할 수도 있음.

멱등성 보장

POST

메시지 바디를 통해 서버로 데이터를 전달할 때 사용

서버에 새로운 리소스를 생성하는 용도

서버의 상태가 변하니까 멱등성 없음

PUT / PATCH

PUT: 대상 리소스가 없다면 생성하고, 있다면 본문대로 교체하는 데 사용, 멱등성 보장

PATCH: 대상 리소스의 일부분을 수정. 변경 방식이 멱등해도 되고 아니어도 됨

{ name: "kim"} - 멱등

{ "operation": "add", "age": 10"} - 멱등 X

DELETE

리소스를 삭제하는 용도

멱등성 보장

> 💡 게시판의 마지막 글을 DELETE로 요청하는 경우
> - 여러번 요청할 때 제거되는 글이 다르므로, DELETE로 요청하면 안된다.

ETC

head,options, trace등 여러가지가 있음.

HTTP 상태 코드

응답의 상태를 나타내는 코드

앞자리는 5개의 클래스, 뒷자리는 구체적인 상태를 나타낸다.

1XX

처리중

- 처리중인데 해당 상태를 응답으로 보낼일이 많지 않음

2xx

처리 성공

- 200 OK: 서버가 요청 제대로 처리
- 201 CREATE: 서버가 잘 처리했고 새로운 리소스가 서버에 저장
- 204 No Content: 요청은 잘 처리했으나 요청에 따른 콘텐츠는 제공하지 않을 때 사용

3xx

요청에 추가적인 처리, 동작이 필요하다는 응답이 돌아올 때 사용

상태 코드	이름	Redirect	POST -> GET	URL 영구 이동
301	moved Permanently	O	X	O
302	Found	O	X	X
303	See Other	O	O	X
304	Not Modified	X	X	X
307	Temporary Redirect	O	X, 메소드 변경 금지	X
308	Permanent Redirect	O	X, 메소드 변경 금지	O

- 301: 요청한 리소스의 URI 변경
- 302: 요청한 리소스의 URI가 일시적 변경
- 303: 요청한 리소스를 다른 URI에서 GET으로 얻어야 할 경우
- 304: 리다이렉트되지 않는다. 마지막 요청이후 요청된 페이지가 변경되지 않았다는 뜻
    - 브라우저가 헤더에 if-modified-since를 사용한 경우. 로컬에 있는 리소스를 사용하면 된다.

4xx:

(에러) 잘못된 요청으로, 클라이언트가 잘못 보낸 경우

- 400 bad request 서버가 요청 구문을 인식하지 못한 경우
- 401 unauthorized : 인증이 필요한 경우. 로그인이 필요하거나 인증정보를 잘못 입력한 경우 받는다.
- 403 Forbidden: 인증은 됐더라도, 해당 사용자의 권한이 없는 경우
- 404 Not Found: 리소스를 찾을 수 없거나, 요청을 거부하는 이유를 비밀로 하는 경우

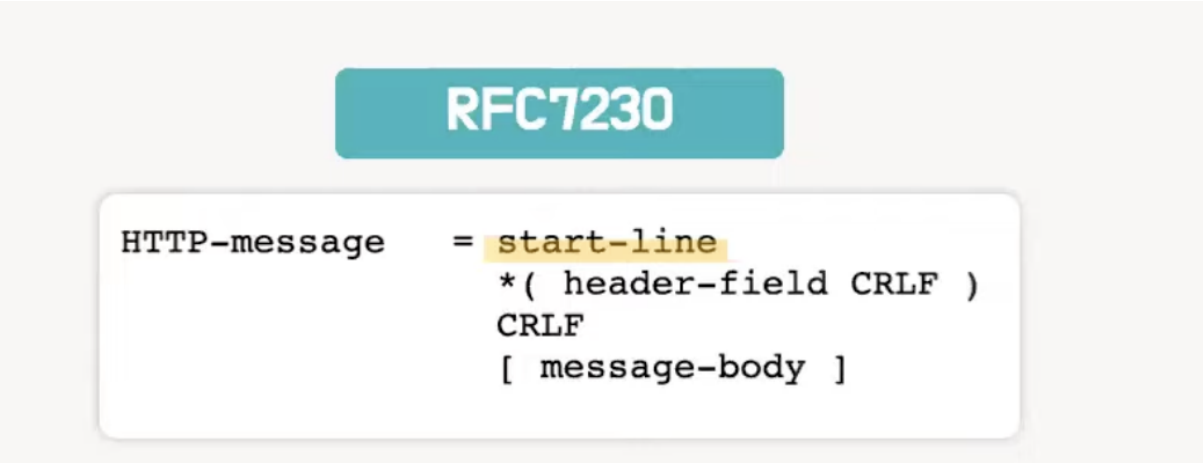
5xx

서버 내부의 오류

- 500 Internal Server Error : 서버에 오류가 발생하여 요청을 수행할 수 없을 때 사용
- 501 Not Implemented : 서버에서 해당 기능을 지원하지 않는 경우.
    - POST, GET은 지원하지만 DELETE를 지원하는

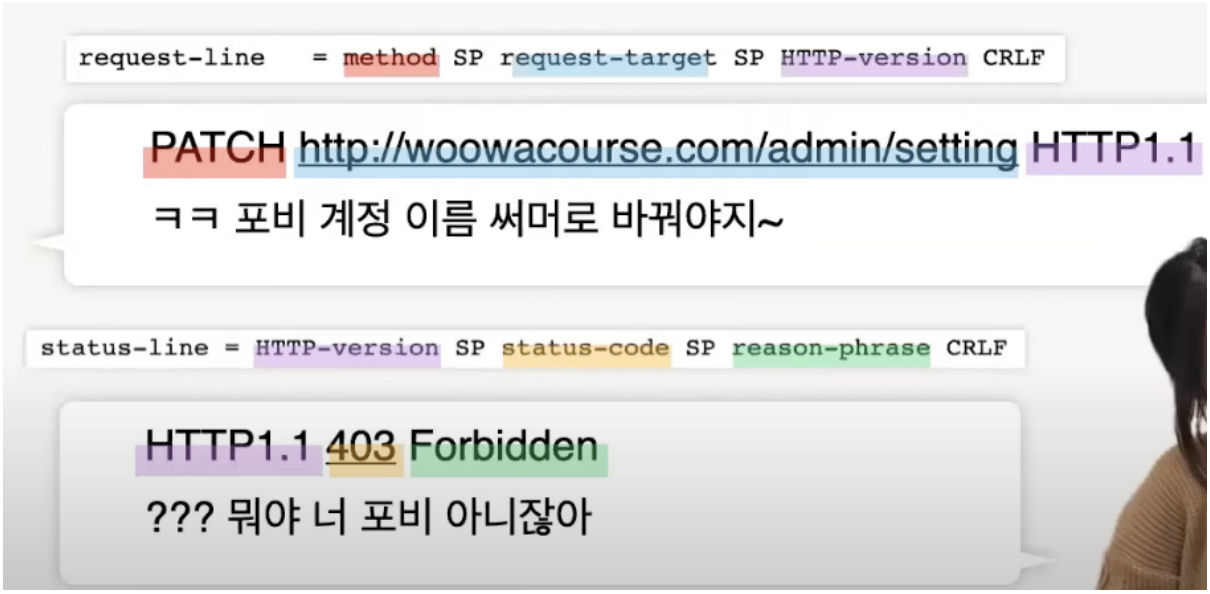
- 503 Service Unavailable: 서버 다운

HTTP 메시지



start-line

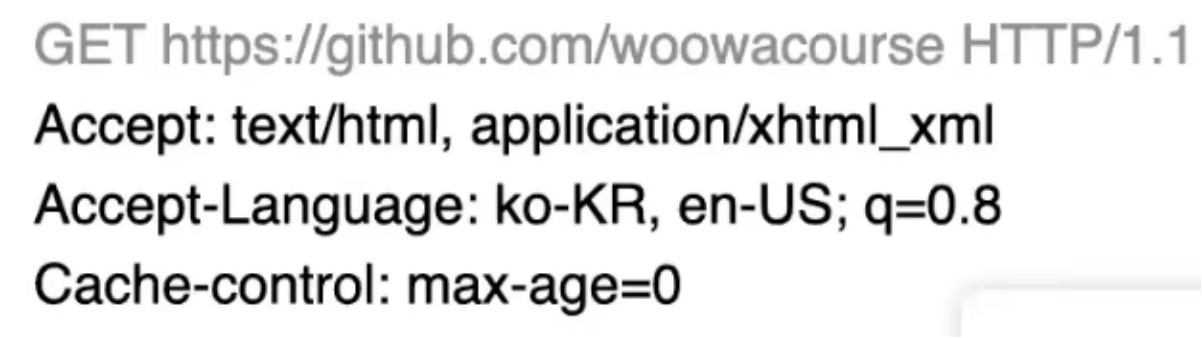
요청은 request line, 응답은 status line 이라고 부른다.



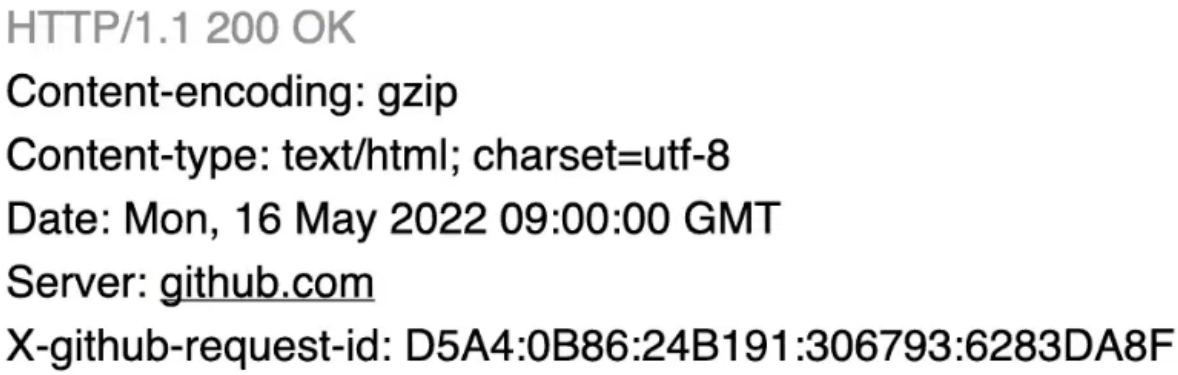
header

필드 - 값으로 이루어짐. 값은

[요청 헤더]



[응답 헤더]



Accept: application/json, text/plain, */*

- > json > text > all type 순으로 받는다는 표현이다.

Accept-Language: en-US,en;q=0.5

- ;는 en과 그 가중치를 표현하기 위해 사용했음

Accept-Encoding: gzip, deflate, br

- > gzip, deflate, br(Brotli) 등등의 압축 포맷을 받는다는 표현이다

comma: field를 구분하는 데 사용

semicolon: 한 필드의 value들을 구분하는데 사용. 값을 extend 하는 느낌

==========

HTTPS

Hypertext Transfer Protocol Secure

TLS를 이용해 암호화된 연결을 하는 HTTP

HTTP는 전송되는 데이터 암호화 X → 보안 이슈 해결을 위해 HTTPS많이 사용

> 💡 SSL: Socket Layer Security
> TLS : Transport Layer Security
> - 암호화된 연결을 만들 수 있게 해준다.
> - SSL의 개선판이 TLS라고 볼 수 있는데 혼용되고 있음

사용 이유

1. 보안성

   메시지 바디와 헤더를 암호화해서 전송하므로, 데이터를 가로채도 내용을 알 수 없다

2. SEO (검색엔진최적화)

   HTTPS를 사용하는 웹사이트에게 검색엔진이 가산점을 준다.

기타 특징

SSL/TLS는 공개키/개인키 방식 + 대칭키 방식 암호화를 둘다 사용함

- 공개키 / 개인키 방식은 탈취 문제가 없고, 대칭키 방식은 복호화 연산이 간단함

인증기관(CA), 유저, 서버가 모두 개입되어 있음

공개/개인키 + 대칭키 이용 통신

대칭키를 공개키로 암호화하여 전달하고, 개인 키로 복호화하는 방식이다. 대칭키가 암호화되어 있어서 탈취되어도 안전하다. 이후 대칭키를 이용해 암호화 통신한다.

1. A → B 연결 요청
2. B → A : 공개 키 전송
3. A → B : 공개키로 대칭키를 암호화해 전송
4. B: 암호화된 대칭키를 개인키로 복호화
5. 이후 통신

사이트 유효성 검증

인증기관

사용자

사이트

사용자 접속 전

1. 사이트 정보 + 사이트 공개키를 인증기관에 전달
2. 사이트 정보 + 사이트 공개키를 검증하고 인증기관 개인키로 서명해 사이트 인증서 생성
3. 인증기관 → 사이트로 사이트 인증서 전달
4. 인증기관 공개키를 사용자에 전달 (브라우저 내장)

사용자가 가진 것: 인증기관 공개키

사이트가 가진 것: 사이트인증서 (사이트 정보와 사이트 공개키가 인증기관 개인키로 서명됨)

사용자 접속요청

1. 사용자가 사이트에 접속 요청
2. 사이트 → 사용자에게 인증서 전달

3. 사용자: 인증기관 공개키로 사이트 인증서를 복호화해서 사이트 정보와 사이트 공개키 획득
4. 사용자: 사이트 정보 및 공개 키를 획 사이트 공개 키로 사용자 대칭키 암호화 → 암호화된 대칭키를 사이트에 전송
5. 사이트 개인키로 대칭키를 복호화해서 저장
6. 대칭키를 이용해서 암호문 통신

핵심: 인증기관 개인키가 탈취되지 않으므로, 브라우저에 내장된 인증기관 공개키로 복호화되는 SSL 인증서라면 인증된 것임.

사이트, 사용자, 인증기관이 협력하기 때문에 안전한 접속방법이 된다.

Reference

https://www.youtube.com/watch?v=IjxkKQvn8Bc

https://www.youtube.com/watch?v=wPdH7lJ8jf0

https://www.inflearn.com/questions/110644/patch-메서드가-멱등이-아닌-이유