Http

Http는 www 상의 클라이언트와 서버 사이에 이루어지는 요청/응답 프로토콜이다.

- **URI, URL, URN**
  - 특정 자원에 대한 식별자라는 공통점. URI에 URL과 URN이 속함.
  - URL은 상대 경로, URN은 절대 경로
  - URN 만으로 실제 리소스 찾기 어려움.

- **역사**
  - 0.9 - 1991, GET 지원. 헤더 X
  - 1.0 - 1996, 메서드, 헤더 추가
  - 1.1 - 1997, 현대 기능
  - 2,3 → 성능 개선

- **특징**
  - 클라이언트-서버 구조 ⇒ 각각 독립적인 발전이 가능해졌음.
    - 클라이언트 - UI, 사용성
    - 서버 - 데이터, 비지니스 로직
  - 무상태 프로토콜(Stateless) → 스케일 아웃에 유리
    - ↔ stateful → 항상 같은 서버 유지 필요, 장애 발생 시 데이터 훼손
    - 필요한 데이터는 쿠키와 세션으로 최소화
  - 비연결성(Connectionless) → 서버 자원 관리에 유리
    - but 많은 데이터 전송에 어려움

      → http 지속 연결(Persistent Connection)으로 문제 해결 == keep alive

- **HTTP 메서드 속성**

- 안전(Safety)
  - 메서드 호출 시에도 리소스를 변경하지 않음 - GET, HEAD
  - 계속 호출해서 로그 쌓이면? - 안전은 해당 리소스에 대한 고려, 그 외는 X
- 멱등(Idempotent)
  - 동일한 요청에 대해서는 호출 회수에 관계없이 결과 동일
  - GET, PUT(최종 결과는 동일), DELETE
  - 재요청 중 다른 곳에서 리소스 변경되면? - 멱등은 외부 요인으로 중간에 리소스가 변경되는 것까지는 고려하지 않음.
- 캐시 가능(Cacheable)
  - 응답 결과 리소스에 대한 캐시 사용 여부
  - GET, HEAD, POST, PATCH 가능
  - but 보통은 GET/HEAD 정도만 캐시, POST, PATCH는 본문 내용까지 캐시하려면 구현 어려움.

HTTP 메소드	RFC	요청에 Body가 있음	응답에 Body가 있음	안전	멱등(Idempotent)	캐시 가능
GET	RFC 7231	아니요	예	예	예	예
HEAD	RFC 7231	아니요	아니요	예	예	예
POST	RFC 7231	예	예	아니요	아니요	예
PUT	RFC 7231	예	예	아니요	예	아니요
DELETE	RFC 7231	아니요	예	아니요	예	아니요
CONNECT	RFC 7231	예	예	아니요	아니요	아니요
OPTIONS	RFC 7231	선택 사항	예	예	예	아니요
TRACE	RFC 7231	아니요	예	예	예	아니요
PATCH	RFC 5789	예	예	아니요	아니요	예

- **API URI 설계 시**
  - URI는 리소스만을 식별, Method를 통해 행위를 분리
  - GET → 요청 리소스 조회, query parameter
  - POST → 새 리소스 생성(Collection, 서버가 관리하는 리소스 디렉토리), 리소스의 의미 관련한 특정 프로세스 처리(리소스 + 동사URI → 컨트롤 URI- 프로세스 상태 변경), 애매한 처리, 메시지 바디로 데이터 전송 및 처리
  - PUT → 리소스의 완전한 대체, (Store, 클라이언트가 관리하는 리소스 저장소)

- PATCH → 부분 변경
- DELETE → 삭제