멱등성이 뭘까?

질문 - Put과 Patch의 차이를 따로 검색해보니 전체 수정/일부 수정 뿐만 아니라 멱등성에 관한 차이도 있던데, 그것에 관해서 좀 더 가르쳐주실 수 있을까요?

멱등성(Idempotence)이란?

> 첫 번째 수행을 한 뒤 여러 차례 적용해도 결과를 변경시키지 않는 작업 또는 기능의 속성

이해하기 너무 어렵다! 예를 들어보자

- 휴지통 비우기를 클릭해 휴지통을 비웠다.
    - 곧바로 다시 휴지통 비우기를 클릭해도 휴지통이 비어있다는 결과는 같다.
- 켜기, 끄기 버튼이 따로 있는 전등이 있다.
    - 전등을 켜고 싶어서 전등을 켰다.
        - 다시 여러번 켜기 버튼을 눌러도 전등은 켜짐 상태 그대로이다.
- DELETE/user/{userId} 를 통해 유저를 삭제했다.
    - 요청이 안 들어갔을까봐 여러번 날렸다. 그럼에도 결과는 그 유저가 삭제되었다는 것으로 같다.

HTTP에서의 멱등성?

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

PUT은 멱등하다! 하지만 Patch는...

PUT - 사용자의 비밀번호만 바꾸고 싶을 경우에도 사용자 정보 전체를 재 전송해야 함.

```
PUT /users/1

{
  "name": "홍길순",
  "email": "hong@test.com",
```

```
  "age": 20
}
```

하나만 바꾸고 싶을 때도 사용자 정보 전체를 재전송해야함.

다르게 말하면, **항상 사용자 정보 전체를 재전송함**

⇒ N번 PUT 요청을 보내면 N번 전체 내용을 덧씌우는 형태이므로, PUT은 멱등성을 보장할 수밖에 없음!

PATCH - 사용자의 비밀번호만 바꾸고 싶을 경우 비밀번호만 전송하면 됨.

```
PATCH /users/1

{
  "name": "홍길순"
}
```

위와 같이 무언가를 간단히 수정하는 형태라면 멱등성을 보장할 수 있음!

하지만...Patch를 통해 응답을 추가하는 경우라면?

```
PATCH /user/123/follow/456
{
  "follow": 456
}
```

유저가 다른 유저를 follow했다는 로그를 DB에 저장해둔다고 생각해보자!

여러번 응답을 보내면 그만큼 DB에 쌓이게 됨

N번 요청을 보내면 N개의 중복된 행이 DB에 저장되게됨!(DB의 상태가 계속 변함)

⇒ 따라서 멱등성을 보장하지 못함!

그러면 저런 경우엔 Patch쓰면 안됨?

멱등성을 보장하도록 코드로직을 수정하면 된다!

1. key를 통해 같은 요청이면 무시하기

```
PATCH /user/123/follow/456
{
    "key": 123-456
  "follow": 456
}
```

1. 팔로우를 하고난 뒤에만 로그를 추가하도록 하기
    1. 이미 팔로우가 되어있으면 중복된 요청이 무시될 것이기 때문에 멱등성을 보장함
2. (위의 예와 같이 DB에 기록하는경우) DB자체에서 중복된 행을 받지 않도록 하기
    - 설계상 위험할 수 있으니 주의

정리

HTTP 메소드	목적	멱등 여부	이유
GET	리소스를 조회함	O	같은 요청을 N번 호출해도 시스템으로부터 같은 결과가 조회됨
POST	리소스를 생성 또는 처리함	X	같은 요청을 N번 호출하면 새로운 리소스가 생성되거나 리소스의 상태가 달라지면서 호출 결과가 달라질 수 있음
PUT	리소스를 대체함	O	같은 요청을 N번 호출해도 항상 대상 리소스를 대체하여 동일한 상태로 만듦
PATCH	리소스를 수정함	X	기존 리소스에 응답을 추가(append)하는 경우에도 PATCH가 사용될 수 있으며, 이때 호출 결과가 달라질 수 있음
DELETE	리소스를 삭제함	O	N번 호출해도 항상 리소스가 없는 동일한 상태임

참고문헌

RFC 7231: Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content

멱등성이 뭔가요? | 토스페이먼츠 개발자센터

[HTTP] HTTP 메소드의 멱등성(Idempotence)과 Delete 메소드가 멱등한 이유