멱등성

주차	12주차
스터디 일자	@2024/01/03

1️⃣ 안정성과 멱등성

안정성(Safe)

HTTP 메서드의 **안정성**이란 보안 취약성을 말하는 것이 아닌, 호출해도 리소스가 변경되지 않는 성질을 말한다.

멱등성(Idempotent)

> **멱등(冪等)** : 수학이나 전산학에서 연산을 여러 번 적용하더라도 결과가 달라지지 않는 성질

HTTP 메서드의 **멱등성**이란 동일한 요청(Request)을 한 번 호출하든 여러번 호출하든 그 결과가 같음을 의미한다.

!! 이때 호출을 실행한 결과가 의미하는 것이 응답 상태 코드가 아닌, `서버의 상태` 라는 점을 유의하자.

HTTP 메서드별 속성

	안전성	멱등성
GET	✅	✅
POST	❌	❌
PUT	❌	✅
PATCH	❌	❌
DELETE	❌	✅

안정성과 멱등성은 다르다

- **HTTP 메서드의 안전**
  - 한 번을 호출하든 여러 번을 호출하든 **리소스에 수정이 발생하지 않는 속성**
- **HTTP 메서드의 멱등**
  - **리소스에 수정이 발생한다고 하더라도** 메서드를 **여러 번 실행한 결과**가 **한 번 실행한 결과**와 같다면 만족하는 속성이라는 차이점이 있다.
  - PUT과 DELETE는 멱등한 메서드지만, 리소스에 변화를 일으키기 때문에 안전한 메서드가 아님

멱등적이지 않은 설계

🧐 만약 개발자가 조회수 기능을 추가하면서, 게시글을 조회하면 동시에 조회수도 올리도록 구현했다.

> GET /post/1 요청
> 서버에서 id값이 1인 게시글을 조회
> 해당 게시글의
> **조회수 데이터를 1 증가시킴**
>
> 해당 게시글 데이터를 응답

이 경우, GET 요청을 여러번 보낼 경우 서버의 데이터 상태는 **매번 바뀌게 될 것이다.**

그러므로, 위 GET 요청 로직은 멱등성을 가지지 않는 것이며, 개발자는 HTTP 스펙에 부합하지 않게 API를 구현했다고 볼 수 있다.

따라서 GET의 멱등성에 맞게 API를 설계하기 위해서는, 조회수 컬럼의 값을 증가시키는 요청을 **PATCH 요청**으로 따로 분리하는 것이 올바르다.

2️⃣ API 관점에서 멱등성

- **멱등한 API**라면 두 번 이상 요청해도 결과는 처음 요청과 **똑같이 돌아온다.**
- 단순히 돌아온 값이 같을 뿐 아니라 **서버 상태(DB)**에도 영향을 미치지 않는다.

✨ 이렇게 **멱등한 API**는 시스템에 의도하지 않은 문제를 일으키지 않고 요청을 재시도할 수 있기 때문에, 멱등성은 **결함 없고 안전한 API를 만드는데 중요!**

3️⃣ 멱등성을 보장하는 법 (멱등키)

- **멱등키**를 API 요청에 포함하면 된다.
- **이전 요청과 동일한 멱등키**를 가진 요청을 받으면 서버에서 이 요청을 **중복**으로 판단한 뒤 실제로 처리하지 않고 첫 요청과 같은 응답을 반환하는 방식
- 요청 본문, URL 쿼리 매개변수, 헤더 중 하나에 멱등키를 포함해서 보내면 된다.
    - IETF(국제 인터넷 표준화 기구)에서는 요청 **헤더에 포함**하는 방법을 표준으로 제안하고 있다.

멱등키?

`Idempotency-Key: {IDEMPOTENCY_KEY}`

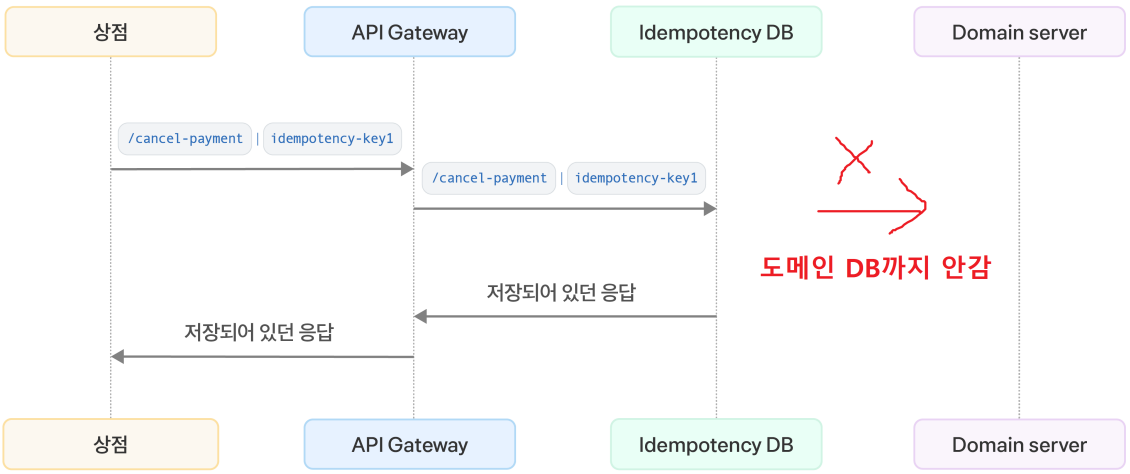
그냥 임의로 요청에 대한 무작위적인 고유 값을 생성해서 헤더에 넣어주는 것이다. (권장은 UUID)

```
# 토스 페이먼츠 결제 취소 API 예시
curl --request POST \
  --url https://api.tosspayments.com/v1/payments/5zJ4xY7m0kODnyR
  --header 'Authorization: Basic dGVzdF9za196WExrS0V5cE5BcldtbzU
  --header 'Content-Type: application/json' \
  --header 'Idempotency-Key: SAAABPQbcqjEXiDL' \
  --data '{"cancelReason":"고객 변심"}'
```

멱등키 식별 플로우

1. API 서버는 취소 요청마다 헤더에 멱등키가 있는지 확인
2. 멱등키를 저장하기 위해 **DB를 만들어둔 뒤**, 멱등키가 포함된 취소 요청이 들어왔을 때 이 **DB를 쿼리해서** 요청이 들어온 멱등키와 매칭되는 요청 기록이 있는지 확인
    - 멱등한 요청 기록을 DB에 저장하는 기간을 정해두어, 그 기간이 지나면 DB에 저장된 멱등키와 기록이 없기 때문에 같은 멱등키를 사용해서 새로운 요청을 보낼 수 있게 한다.
      ⇒ 멱등키의 유효기간을 설정할 수 있다.

3. 만약 이전에 **같은 멱등키로 들어온 요청이 있었다면**, 서버에서 실제 요청을 실행하지 않고 저장되어 있던 응답 데이터를 돌려준다.



→ 만약 멱등키와 매칭되는 **이전 기록이 없다면**, 새로 생성된 응답을 저장하는 새로운 기록을 만들고 응답을 클라이언트에 돌려준다.

멱등성 제공 이유 (장점)

- 실수로 **중복 요청**이 되더라도(일명 '따닥') 실제로는 요청이 되지 않아서 안심하고 여러 번 요청할 수 있다. (안전하다)
  - 결제 같은 경우 완전 중요하겠죠…
- 멱등키를 가진 요청은 도메인 서버로 바로 처리되지 않기 때문에 도메인 서버 로직의 복잡도가 높을 때 **API 성능 개선**에 도움이 됨
  - 이미 기록이 있으면 도메인 서버를 거치지 않고도 결과 반환이 가능하니

4️⃣ 코드로 감 잡아보기

> 토스 페이먼츠의 **멱등성이 보장된 결제 취소 API의 처리 프로세스** (JavaScript)

JavaScript 코드를 java 코드로 변환하여 살펴보자~

```
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class CancelServlet extends HttpServlet {

    private Map<String, String> idempotencyResponses = new HashM

    protected void doPost(HttpServletRequest request, HttpServle
        String orderId = request.getParameter("orderId");
        String amount = request.getParameter("amount");
                // 멱등키 받아옴
        String idempotencyKey = request.getHeader("idempotencyKe

        // 멱등키가 있고 멱등 응답도 저장되어 있다면 실제 처리하지 않고 저장
        if (idempotencyKey != null && idempotencyResponses.conta
            String savedResponse = idempotencyResponses.get(iden
            response.setStatus(HttpServletResponse.SC_OK);
            response.getWriter().write(savedResponse);
            return;
        }

        // 실제 결제 취소 처리
        String result = cancelOrder(orderId, amount);

        // 멱등키가 있으면 멱등응답을 저장한다
        if (idempotencyKey != null) {
            idempotencyResponses.put(idempotencyKey, result);
        }
```

```
            // 결제 취소 성공 메시지 반환
            String responseBody = "{\"message\": \"결제 취소 성공\"}";
            response.setStatus(HttpServletResponse.SC_OK);
            response.getWriter().write(responseBody);
        }
    }
```

- 만약 여러 API에서 모두 멱등성을 보장하고 싶다면 하나하나 일일이 구현하지 말고 멱등성 컴포넌트를 만들어서 재사용하자!

5️⃣ 에러 처리

IETF 명세에서 제공하는 에러 시나리오는 다음과 같다.

에러 코드	시나리오
400 Bad Request	멱등해야 하는 API 요청에 **멱등키가 누락**됐거나 **형식에 맞지 않는 키 값**이 들어왔을 때
409 Conflict	이전 요청 처리가 **아직 진행 중인데 같은 멱등키로 새로운 요청이 올 때**
422 Unprocessable Entity	재시도 된 요청 본문(payload)이 **처음 요청과 다른데 같은 멱등키를 또 사용했을 때**

▼ 에러 코드 의미

400 : 서버가 **클라이언트 오류**(예: 잘못된 요청 구문, 유효하지 않은 요청 메시지 프레이밍, 또는 변조된 요청 라우팅)를 감지해 요청을 처리할 수 없거나, 하지 않는다는 것을 의미

409 : 서버의 **현재 상태**와 **요청**이 **충돌**했음을 의미

422 : 서버가 요청 엔티티의 콘텐츠 유형을 이해하고 요청 엔티티의 구문이 정확하지만 **포함된 명령을 처리할 수 없음**을 의미

⚠️ 끝.

참고한 곳

https://inpa.tistory.com/entry/WEB-🌐-HTTP의-멱등성---안정성---캐시성-💯-완벽-이해하기#멱등성idempotent

https://thisdev.tistory.com/5

https://velog.io/@tosspayments/멱등성이-뭔가요#멱등idempotent하다는-것