🤔

REST API와 HTTP 메서드, 상태 코드

REST 란?

- Representational State Transfer
- 웹에 존재하는 자원에 HTTP URI를 부여하여 구분하고, HTTP Method를 통해 자원에 대한 CURD 연산을 요청
- 웹의 장점을 최대한 활용할 수 있는 아키텍처 스타일

장점

- HTTP 프로토콜의 인프라 그대로 사용
  - REST API 별도 인프라 구축 필요 없음
  - HTTP로 통신하는 모든 플랫폼에서 사용 가능
  - 서버와 클라이언트 역할 분리

단점

- 사용할 수 있는 메서드 한정적
- 구형 브라우저라면 PUT, DELETE같은 HTTP 메서드 지원하지 못함

> 💡 추가) RESTful API란 REST 원리를 따르는 API이다.

REST API 구성 요소

1. **자원(Resource)**

- 특정 자원 나타냄
- HTTP URI로 표현

2. **행위(Verb), Method**

- 자원에 대한 행위 나타냄
- HTTP Method로 표현

3. **표현(Representations)**

- 자원 행위의 추가 내용
- HTTP Message payload로 표현 (= HTTP Message Body)
- GET, DELETE의 행위경우 작성하지 않음

REST API 중심 규칙

> 💡 REST의 기본적인 규칙
> 1. URI 자원 표현하는데 집중
> 2. 행위에 대한 정의는 HTTP Method

1. URI는 정보의 자원 표시

- 리소스 명은 동사보다 명사 사용
- URI에 리소스 행동표현 사용하지 않음

```
// URI에 동사 표현 -> bad
GET students/find/100

// 수정
GET students/100
```

2. 자원의 행위는 HTTP Method로 표현

- 주로 GET, POST, PUT, DELETE 로 표현

```
// 등록, 삭제 URI에 포함 -> bad
GET students/insert/100
GET students/delete/100

// 수정
POST students/100
DELETE students/100
```

URI 설계 시 주의점

1. 슬래시(/) - 계층 관계를 나타냄

URI 마지막에 슬래시 넣지 않기

URI는 리소스의 유일한 식별자를 나타내야 한다. URI가 다르면 리소스가 달라진다. 따라서 URI 경로에 마지막에 슬래시를 사용하지 않는다.

```
GET http://restapi.com/school/students/100
GET http://restapi.com/school/classroom/3

// 마지막 슬래시 -> bad
GET http://restapi.com/school/students/100/
```

2. 하이픈(-) - URI 가독성 높이기 위해 사용

아래 예시는 짧은 경우이지만 보통 URI가 길어질 때 가독성을 위해 사용한다.

```
GET http://restapi.com/school/classroom/3-3
```

3. 밑줄(_) - 사용하지 않음

밑줄은 보기 어렵거나 문자가 가려지는 문제가 발생할 수 있다. 따라서 밑줄 대신 하이픈을 사용한다.

```
// URI에 밑줄 사용 - bad
GET http://restapi.com/school/classroom/3_3

// 수정
GET http://restapi.com/school/classroom/3-3
```

4. URI는 소문자로 작성하기

URI 경로에 대문자는 가급적 피하는 것이 좋다. 그 이유는 대소문자에 따라 다른 리소스로 인식하기 때문이다.

```
// URI에 대문자 사용 - bad
GET http://restapi.com/School/Classroom/3-3

// 수정
GET http://restapi.com/school/classroom/3-3
```

5. 파일 확장자는 URI에 포함시키지 않음

URI에 파일 확장자를 표시하는 대신 Accept 헤더를 통해 작성한다.

```
// URI 경로에 확장자 표기 - bad
GET http://restapi.com/school/students/photo.jpg

// 수정
```

```
GET http://restapi.com/school/students/photo
HTTP/1.1 HOST: restapi.com Accept: image/jpg
```

HTTP 주요 메서드

- **안전(safe)**
  - 호출해도 리소스를 변경하지 않는다.
- 멱등(Idempotent)
  - 같은 요청 보내도 결과가 똑같다.
- 캐시 가능
  - 응답 결과 리소스를 캐시하여 사용 가능한지 판단한다.

	기능	payload 사용	안전	멱등	캐시 가능
GET	리소스 조회	예(보통 사용x)	예	예	예
POST	리소스 생성	예	아니오	아니오	예(보통 사용x)
PUT	리소스 수정(덮어쓰기)	예	아니오	예	아니오
PATCH	리소스 일부 수정	예	아니오	아니오	예(보통 사용x)
DELETE	리소스 삭제	예(보통 사용x)	아니오	예	아니오

HTTP 주요 상태코드

상태코드는 클라이언트가 보낸 요청의 처리 상태이다. 서버에서 클라이언트로 응답 메시지를 보낼 때 이 상태코드가 들어간다.

2xx - 요청 정상 처리

상태코드	
200	OK, 결과 정상적으로 처리하여 응답
201	Create, 요청 성공하여 새 리소스 생성됨

3xx - 리다이렉션

상태코드	
301	Moved Permanently, 리소스 URI가 영구적으로 이동
302	Found, 리소스의 URI가 일시적으로 변경, 리다이렉트시 요청 메서드가 GET으로 변함

4xx - 클라이언트 오류

오류의 원인이 클라이언트에 있다. 따라서 같은 요청을 보내도 실패한다.

상태코드	
400	Bad Request, 클라이언트의 잘못된 요청
401	Unauthorized, 클라이언트가 해당 리소스에 대한 인증이 필요
403	Forbidden, 서버가 요청을 이해했지만 승인 거부
404	Not Found, 요청 리소스를 찾을 수 없음

5xx - 서버 오류

오류의 원인이 서버에 있다.

상태코드	
500	Internal Server Error, 서버 내부 문제 오류
503	Service Unavailable, 서비스 이용 불가

참고자료

https://restfulapi.net/http-methods/

https://poiemaweb.com/js-rest-api

https://meetup.toast.com/posts/92