RESTful API

by 홍석주

REST란

- Representational State Transfer
- 자원의 이름(표현)으로 구분하여 해당 자원의 상태(정보)를 주고 받는 것
  - 자원: 소프트웨어가 관리하는 모든 것, 문서, 데이터 등
  - 표현: 위의 자원을 표현하기 위한 이름, 학생 데이터가 자원일 때 `students`로 표현
  - 상태: 데이터가 요청되어지는 시점에 자원의 상태를 전달. 주로 JSON으로 전달.
- 크게 자원(Resource), 행위(Verb), 표현(Representations)로 구성되어있다.

특징

- Stateless (무상태성): 서버가 상태 정보를 따로 저장하고 관리하지 않고, 요청만 처리
- Cacheable (캐시가능): HTTP 웹 표준을 사용하기에 HTTP 캐싱 기능 적용 가능
- 클라이언트-서버 구조: 서버는 API를 제공, 클라이언트는 사용자 인증이나 컨텍스트(세션, 로그인 정보) 등을 직접 관리하는 구조로 역할을 분리해서 서로의 의존성을 줄임

RESTful API

- 사실, REST는 아키텍쳐이지 뭐 표준이라던가 규약같은 것은 아니라 지킬 필요 X
- 다만, 개발자들의 경험들이 축적되면서 쌓인 컨벤션들이 REST하다는 의미로 RESTful로 불리며 아래처럼 어느정도의 가이드라인이 생겼다.

RESTful API 디자인 가이드 - 리소스

URI는 정보의 자원을 표현해야 한다.

- 도큐먼트(document): 객체 인스턴스나 데이터베이스 레코드와 유사한 개념
- 컬렉션(collection): 서버에서 관리하는 디렉터리 리소스
- 스토어(store): 클라이언트에서 관리하는 리소스 저장소

RESTful API 디자인 가이드 - HTTP Method

자원에 대한 행위는 HTTP Method(GET, POST, PUT, DELETE)로 표현한다.

Method	역할
POST	해당 URI를 요청하면 리소스를 생성
GET	리소스를 조회
PUT	리소스를 수정
DELETE	리소스를 삭제

> ex) 미네랄을 캐라. 미네랄이 자원(Resource), 캐라가 행위(Method)가 된다.

적용

- URI는 정보의 자원을 표현해야하기에 명사로 표현한다.
- 북수 단어 형태로 써준다.
- 행위에 대해서는 HTTP Method로 처리한다.

```
// 멤버 전체를 갖고 올 때
GET /members

// 특정 멤버를 갖고 올 때
GET /members/{id}

// 멤버를 추가 할 때
POST /members

// 멤버를 수정 할 때
PUT /members/{id}

// 멤버를 삭제 할 때
DELETE /members/{id}
```

적용

- 근데 리소스와 행위만으로 URI를 표현하는데 부족하기에 이럴 땐 컨트롤 URI를 추가 해준다.

```
POST /orders/{orderId}/start-delivery
```

convention

- URI 마지막 문자로 슬래쉬(/) 포함하지 않는다.
- 언더바(_) 대신에 대쉬(-)를 사용하자.
- 소문자를 사용하자.
- 파일 확장자는 URI에 포함하지 않는다.
- 리소스 간에 연관 관계(has 관계)가 있는 경우.
    - `GET /users/{id}/devices`