RestFull API 정리

작성: 이찬영

REST API?

1. 해당 프로젝트에서 SPring 서버와 클라이언트의 통신을 REST API로 진행한다.
2. HTTP를 지원하는 모든 환경에서 사용이가능하다.
3. 시스템을 분산해 확장성과 재사용성을 높여 유지보수 및 운용을 편리하게 할 수 있다.

REST API 설계 기본 구칙

1. URI는 정보의 자원을 표현해야 한다.
    - resource는 동사보다는 명사를, 대문자보다는 소문자를 사용한다.
    - resource의 도큐먼트 이름으로는 단수 명사를 사용해야 한다.
    - resource의 컬렉션 이름으로는 복수 명사를 사용해야 한다.
    - resource의 스토어 이름으로는 복수 명사를 사용해야 한다.
        - Ex) GET /Member/1 -> GET /members/1

2. 자원에 대한 행위는 HTTP Method(GET, PUT, POST, DELETE 등)로 표현한다.
   - URI에 HTTP Method가 들어가면 안된다.
     - Ex) GET /members/delete/1 -> DELETE /members/1
   - URI에 행위에 대한 동사 표현이 들어가면 안된다.(즉, CRUD 기능을 나타내는 것은 URI에 사용하지 않는다.)
     - Ex) GET /members/show/1 -> GET /members/1
     - Ex) GET /members/insert/2 -> POST /members/2
   - 경로 부분 중 변하는 부분은 유일한 값으로 대체한다.(즉, :id는 하나의 특정 resource를 나타내는 고유값이다.)
     - Ex) student를 생성하는 route: POST /students
     - Ex) id=12인 student를 삭제하는 route: DELETE /students/12

REST API 설계 규칙

1. 슬래시 구분자(/)는 계층 관계를 나타내는데 사용한다.
   - Ex) http://restapi.example.com/houses/apartments
2. URI 마지막 문자로 슬래시(/)를 포함하지 않는다.
   - URI에 포함되는 모든 글자는 리소스의 유일한 식별자로 사용되어야 하며 URI가 다르다는 것은 리소스가 다르다는 것이고, 역으로 리소스가 다르면 URI도 달라져야 한다.
   - REST API는 분명한 URI를 만들어 통신을 해야 하기 때문에 혼동을 주지 않도록 URI 경로의 마지막에는 슬래시(/)를 사용하지 않는다.
   - Ex) http://restapi.example.com/houses/apartments/ (X)

3. 하이픈(-)은 URI 가독성을 높이는데 사용
   - 불가피하게 긴 URI경로를 사용하게 된다면 하이픈을 사용해 가독성을 높인다.
4. 밑줄(_)은 URI에 사용하지 않는다.
   - 밑줄은 보기 어렵거나 밑줄 때문에 문자가 가려지기도 하므로 가독성을 위해 밑줄은 사용하지 않는다.
5. URI 경로에는 소문자가 적합하다.
   - URI 경로에 대문자 사용은 피하도록 한다.
   - RFC 3986(URI 문법 형식)은 URI 스키마와 호스트를 제외하고는 대소문자를 구별하도록 규정하기 때문

6. 파일확장자는 URI에 포함하지 않는다.
   - REST API에서는 메시지 바디 내용의 포맷을 나타내기 위한 파일 확장자를 URI 안에 포함시키지 않는다.
   - Accept header를 사용한다.
   - Ex) http://restapi.example.com/members/soccer/345/photo.jpg (X)
   - Ex) GET / members/soccer/345/photo HTTP/1.1 Host: restapi.example.com Accept: image/jpg (O)
7. 리소스 간에는 연관 관계가 있는 경우
   - /리소스명/리소스 ID/관계가 있는 다른 리소스명
   - Ex) GET : /users/{userid}/devices (일반적으로 소유 'has'의 관계를 표현할 때)

REST API 설계 예시

CRUD	HTTP verbs	Route
목록 표시	GET	/resource
1개 표시	GET	/resource/:id
생성	POST	/resource
수정	PUT/PATCH	/resource/:id
삭제	DELETE	/resource/:id

RESTful

- RESTful은 일반적으로 REST라는 아키텍처를 구현하는 웹 서비스를 나타내기 위해 사용되는 용어이다.
  - ‘REST API’를 제공하는 웹 서비스를 ‘RESTful’하다고 할 수 있다.
- RESTful은 REST를 REST답게 쓰기 위한 방법으로, 누군가가 공식적으로 발표한 것이 아니다.
  -즉, REST 원리를 따르는 시스템은 RESTful이란 용어로 지칭된다.

RESTful의 목적

- 이해하기 쉽고 사용하기 쉬운 REST API를 만드는 것
- RESTful한 API를 구현하는 근본적인 목적이 성능 향상에 있는 것이 아니라 일관적인 컨벤션을 통한 API의 이해도 및 호환성을 높이는 것이 주 동기이니, 성능이 중요한 상황에서는 굳이 RESTful한 API를 구현할 필요는 없다.