**Rest / Rest API / Restful API란?**

**{ REST }**

REST (Representational State Transfer) (=이동 상태 묘사)

1. HTTP URI(Uniform Resource Identifier)를 통해 자원(Resource)을 명시하고,
2. HTTP Method(POST, GET, PUT, DELETE, PATCH 등)를 통해

해당 자원(URI)에 대한 CRUD Operation을 적용하는 것을 의미합니다.

여기서 잠깐! (URI 와 URL 을 혼동하지 말자!)

텍스트이(가) 표시된 사진

자동 생성된 설명

-즉 , URI 안에 URL 이라는 개념이 들어간다! ( URI > URL/URN )

**CRUD Operation이란?**

-**CRUD**는 대부분의 컴퓨터 소프트웨어가 가지는 **기본적인** **데이터 처리 기능**인 Create(생성), Read(읽기), Update(갱신), Delete(삭제)를 묶어서 일컫는 말

> Create : 데이터 생성(POST)

> Read : 데이터 조회(GET)

> Update : 데이터 수정(PUT)

> Delete : 데이터 삭제(DELETE)

**REST 구성 요소**

REST는 다음과 같은 3가지로 구성이 되어있다.

1. **자원(Resource) : HTTP URI**
2. **자원에 대한 행위(Verb) : HTTP Method**

-CRUD

1. **자원에 대한 행위의 내용 (Representations) : HTTP Message Pay Load**

-Client가 자원의 상태(정보)에 대한 조작을 요청하면 Server는 이에 적절한 응답(Representation)을 보낸다.

-JSON 혹은 XML를 통해 데이터를 주고 받는 것이 일반적이다.

### 테이블이(가) 표시된 사진 자동 생성된 설명

### REST의 장단점

장점

* HTTP 프로토콜의 인프라를 그대로 사용하므로 REST API 사용을 위한 별도의 인프라를 구출할 필요가 없다.
* REST API 메시지가 의도하는 바를 명확하게 나타내므로 의도하는 바를 쉽게 파악할 수 있다.
* 서버와 클라이언트의 역할을 명확하게 분리한다.

단점

* 표준이 자체가 존재하지 않아 정의가 필요하다.
* HTTP Method 형태가 제한적이다.

**{ REST API }**

RESPT API란 **REST의 원리를 따르는 API**를 의미합니다**.**

하지만 **REST API**를 올바르게 **설계**하기 위해서는 **지켜야 하는 몇가지 규칙**이 있으며 해당 규칙을 알아 보자.

**REST API 설계 예시**

**1. URI는 동사보다는 명사를, 대문자보다는 소문자를 사용하여야 한다.**

**Bad Example**[http://khj93.com/Running/](http://khj93.com/test/) **Good Example**[http://khj93.com/run/](http://khj93.com/test/)

**2. 마지막에 슬래시 (/)를 포함하지 않는다.**

**Bad Example**<http://khj93.com/test/>**Good Example**[http://khj93.com/test](http://khj93.com/test/)

**3. 언더바 대신 하이폰을 사용한다.**

**Bad Example**[http://khj93.com/test\_blog](http://khj93.com/test/) **Good Example**[http://khj93.com/test-blog](http://khj93.com/test/)

**4. 파일확장자는 URI에 포함하지 않는다.**

**Bad Example**[http://khj93.com/photo.jpg](http://khj93.com/test/)**Good Example**[http://khj93.com/photo](http://khj93.com/test/)

**5. 행위를 포함하지 않는다.**

**Bad Example**[http://khj93.com/delete-post/1](http://khj93.com/test/)**Good Example**[http://khj93.com/post/1](http://khj93.com/test/)

**{ RESTFUL API }**

**RESTFUL이란 REST의 원리를 따르는 시스템을 의미**한다. 하지만 REST를 사용했다 하여 모두가 RESTful 한 것은 아니다.  **REST API의 설계 규칙을 올바르게 지킨 시스템을 RESTful하다 말할 수 있으며**

모든 CRUD 기능을 POST로 처리 하는 API 혹은 URI 규칙을 올바르게 지키지 않은 API는 **REST API의 설계 규칙을 올바르게 지키지 못한 시스템은 REST API를 사용하였지만 RESTful 하지 못한 시스템**이라고 할 수 있다!.

Reference

-[**https://khj93.tistory.com/entry/%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC-REST-API%EB%9E%80-REST-RESTful%EC%9D%B4%EB%9E%80**](https://khj93.tistory.com/entry/%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC-REST-API%EB%9E%80-REST-RESTful%EC%9D%B4%EB%9E%80)

-<https://velog.io/@syoung125/%EA%B0%9C%EB%85%90%EA%B3%B5%EB%B6%80-URI%EB%8A%94-%EB%AC%B4%EC%97%87%EC%9D%B4%EA%B3%A0-URL%EC%9D%80-%EB%AC%B4%EC%97%87%EC%9D%BC%EA%B9%8C>