# REST API, REST, RESTful 정리

## REST란?

- REST(Representational State Transfer)의 약자.

- 자원을 이름(URI)으로 구분하고, HTTP Method를 통해 자원에 대한 CRUD Operation을 적용.

- HTTP URI: 자원(Resource)을 명시

- HTTP Method: POST, GET, PUT, DELETE, PATCH 등으로 행위(Verb)를 명시

- CRUD Operation: Create(POST), Read(GET), Update(PUT, PATCH), Delete(DELETE)

## REST 구성 요소

- 자원(Resource): HTTP URI

- 자원에 대한 행위(Verb): HTTP Method

- 자원에 대한 행위의 내용(Representation): HTTP Message Payload

## REST의 특징

- Server-Client 구조

- Stateless (무상태)

- Cacheable (캐시 처리 가능)

- Layered System (계층화 구조)

- Uniform Interface (일관된 인터페이스)

## REST의 장점

- 기존 HTTP 인프라를 그대로 사용 가능

- HTTP 표준 프로토콜을 활용한 범용성

- 다양한 플랫폼에서 사용 가능

- 명확한 메시지 구조로 이해 쉬움

- 서버와 클라이언트 역할 명확히 분리

## REST의 단점

- 표준 자체가 없어 정의 필요

- HTTP Method의 제한

- 헤더 정보 처리로 테스트 어려움

- 구형 브라우저와 호환성 문제

## REST API란?

- REST 원리를 따르는 API

- REST 설계 규칙을 따라야 진정한 RESTful API

## REST API 설계 규칙

- 1. URI는 동사보다 명사를 사용하며, 소문자를 사용.

- 2. 마지막에 슬래시(/) 포함하지 않음.

- 3. 언더바(\_) 대신 하이픈(-) 사용.

- 4. 파일 확장자 포함하지 않음.

- 5. URI에 행위 포함하지 않음.

## RESTful이란?

- REST API의 설계 규칙을 올바르게 지킨 시스템을 RESTful하다고 함.

- 단순히 REST API를 사용한다고 RESTful하지는 않음.