

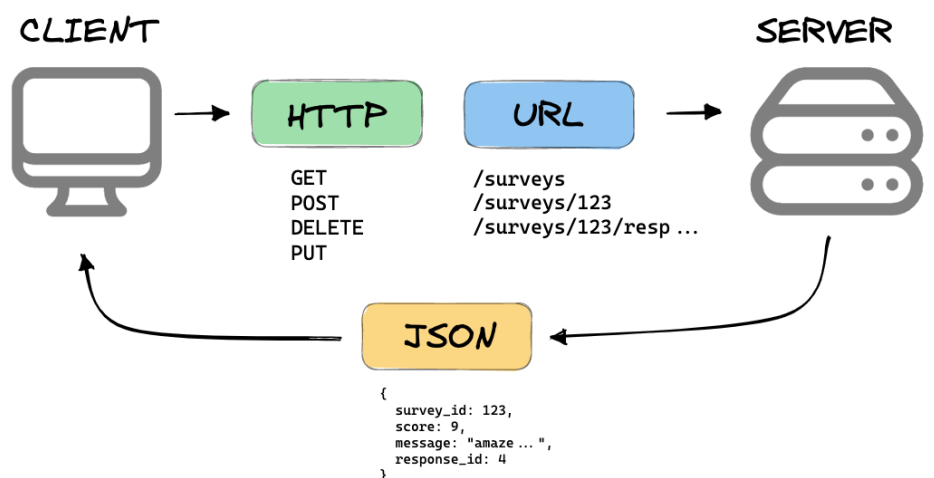


# RESTful(REST) API

≡ 작 성자	FE 박효빈
📅 날 짜(수, 토)	@2024년 6월 22일
≡ 주 제	RESTful API
≡ 참 고자 료	<a href="https://meetup.nhnccloud.com/posts/92">https://meetup.nhnccloud.com/posts/92</a> <a href="https://aws.amazon.com/ko/what-is/restful-api/">https://aws.amazon.com/ko/what-is/restful-api/</a> <a href="https://www.elancer.co.kr/blog/view?seq=74">https://www.elancer.co.kr/blog/view?seq=74</a> <a href="https://www.ibm.com/kr-ko/topics/rest-apis">https://www.ibm.com/kr-ko/topics/rest-apis</a>

## REST란?

### WHAT IS A REST API?



mannhowie.com

Representational State Transfer의 약자로 전반적인 웹 어플리케이션에서 상호작용하는데 사용되는 웹 아키텍처 모델이다. 자원을 주고받는 웹 상에서의 **통신 체계에 있어서 범용적인 스타일**을 규정한 아키텍처 라고 할 수 있습니다.

즉, 한마디로

1. HTTP URI(Uniform Resource Identifier)를 통해 자원(Resource)을 명시하고,
2. HTTP Method(POST, GET, PUT, DELETE, PATCH 등)를 통해 해당 자원을 조작한다.
3. JSON, XML을 통해 표현한다.

이러한 스타일을 규정한 아키텍처이다.

네트워크 상에서 클라이언트와 서버 사이의 통신하는 방식 중 하나의 개념이라고 생각하면 된다.

## REST 특징

### 1. Client-Server 구조

REST 서버는 API 제공, 클라이언트는 사용자 인증이나 컨텍스트(세션, 로그인 정보)등을 직접 관리하는 구조로 각각의 역할이 확실히 구분되기 때문에 클라이언트와 서버에서 개발해야 할 내용이 명확해지고 서로간 의존성이 줄어들게 된다.

### 2. 무상태 (Stateless)

REST는 무상태성 성격을 갖는다. 다시 말해 작업을 위한 상태 정보를 따로 저장하고 관리하지 않는다. 세션 정보나 쿠키정보를 별도로 저장하고 관리하지 않기 때문에 API 서버는 들어오는 요청만을 단순히 처리하면 된다. 때문에 서비스의 자유도가 높아지고 서버에서 불필요한 정보를 관리하지 않음으로써 구현이 단순해진다.

### 3. 캐시 가능(Cacheable)

REST의 가장 큰 특징 중 하나는 HTTP라는 기존 웹표준을 그대로 사용하기 때문에, 웹에서 사용하는 기존 인프라를 그대로 활용이 가능하다. 따라서 HTTP가 가진 캐싱 기능이 적용 가능하다. HTTP 프로토콜 표준에서 사용하는 Last-Modified태그나 E-Tag를 이용하면 캐싱 구현이 가능합니다.

#### ▼ HTTP 캐싱

**웹 리소스(HTML 페이지, 이미지, 자바스크립트 파일 등)를 사용자의 브라우저나 서버에 임시로 저장해두고, 동일한 요청이 있을 때 빠르게 제공하는 기술입니다.**

#### ▼ Last-Modified와 ETag

웹 개발에서,

**Last-Modified**와 **ETag**는 웹 브라우저와 서버 간의 효율적인 데이터 전송을 위해 사용되는 HTTP 헤더이다.

#### ▼ Last-Modified

Last-Modified response HTTP 헤더에는 오리진 서버가 리소스를 마지막으로 수정했다고 믿는 날짜와 시간이 포함되어 있습니다. 리소스가 이전에 저장된 리소스와 동일한지 여부를 확인하는 검증기로 사용됩니다.

- 1초 미만(0.x초) 단위로 캐시 조정이 불가능
- 날짜 기반의 로직 사용
- 데이터를 수정해서 날짜가 다르지만, 같은 데이터를 수정해서 데이터 결과가 똑같은 경우
- 서버에서 별도의 캐시 로직을 관리하고 싶은 경우 예)스페이스나 주식처럼 크게 영향이 없는 변경에서 캐시를 유지하고 싶은 경우

#### ▼ E-Tag

**ETag** HTTP 응답 헤더는 특정 버전의 리소스를 식별하는 식별자입니다. 웹 서버가 내용을 확인하고 변하지 않았으면, 웹 서버로 full 요청을 보내지 않기 때문에, 캐시가 더 효율적이게 되고, 대역폭도 아낄 수 있습니다.

- 캐시용 데이터에 임의의 고유한 버전 이름을 달아둠.
- 진짜 단순하게 ETag만 보내서 같으면 유지, 다르면 다시 받기.
- 클라이언트는 단순히 이값을 서버에 제공

#### 4. Self-descriptiveness (자체 표현 구조)

REST의 또 다른 큰 특징 중 하나는 REST API 메시지만 보고도 이를 쉽게 이해 할 수 있는 자체 표현 구조로 되어 있다는 것입니다.







#### 5. 계층형 구조

REST 서버는 다중 계층으로 구성될 수 있으며 보안, 로드 밸런싱, 암호화 계층을 추가해 구조상의 유연성을 둘 수 있고 PROXY, 게이트웨이 같은 네트워크 기반의 중간매체를 사용할 수 있게 합니다.

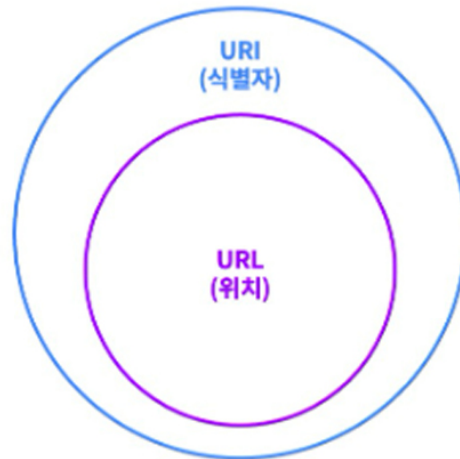
## REST API란?

REST(Representational State Transfer) 아키텍처 스타일의 설계 원칙을 준수하는 *API(애플리케이션프로그래밍 인터페이스)*입니다.

## REST API 디자인 규칙

- URI는 명사를 사용하자.
  -  /getAllUsers, /getUserById
  -  /users, /users/{id}
- URI 경로에는 소문자를 사용하자.
- 하이픈(-)은 URI 가독성을 높이는데 사용한다.
- 밑줄(\_)은 URI에 사용하지 않는다.
- HTTP 메소드를 활용하자. (자원에 대한 CRUD 작업은 HTTP 메소드로 표현해야한다.)
  - GET - 정보 요청
  - POST - 정보 생성
  - PUT - 정보 수정(전체)
  - PATCH - 정보 수정(일부)
  - DELETE - 정보 삭제
- HTTP 응답 상태 코드를 활용하자 (클라이언트는 해당 요청에 대한 처리 상태를 알 수 있어야 한다.)
  - 200 : OK
  - 201 : Created
  - 400 : Bad Request
  - 401 : Unauthorized
  - 404 : Not Found
  - 500 : Internal Server Error
- 계층 관계를 슬래시(/)로 표현하고, url 마지막에 슬래시(/)를 포함하지 않는다.
  -  /users/{id}/
  -  /users/{id}
- 파일 확장자는 URI에 포함하지 않는다.
  -  http://example.com/api/2/photo.png
  -  http://example.com/api/2/photo
- 조회 시 쿼리를 활용하자. (페이지나 필터링 정보는 쿼리 파라미터를 활용)
  - /users?page=2&count=20

## ▼ URI? URL?



## URI

Uniform Resource Identifier의 약자로 통합 자원 식별자이다.

## URL

Uniform Resource Locator의 약자로 네트워크상에서 통합자원의 “위치”를 나타내기 위한 규약이다.

정리하자면, URI는 식별자이고 URL은 식별자+위치이다.

- elancer.co.kr은 URI
- <https://elancer.co.kr>은 URL

따라서 URL은 URI안에 포함되는 개념이다. URL은 URI가 될 수 있지만 모든 URI가 URL이 될 수 없다.

# REST API와 RESTful API는 다른가?

## RESTful API

RESTful API는 REST 원칙을 따르는 API를 설명하는 데 사용되는 용어입니다. 따라서, RESTful API는 REST 아키텍처 스타일을 준수하는 API를 의미합니다. RESTful API는 다음과 같은 방식으로 설계됩니다:

1. **자원의 표현**: 자원은 JSON, XML 등의 포맷으로 표현됩니다.

2. **HTTP 메서드 사용:** 자원에 대한 CRUD(생성, 조회, 업데이트, 삭제) 작업은 HTTP 메서드(GET, POST, PUT, DELETE)를 사용하여 수행됩니다.
3. **명확한 URI 구조:** 각 자원은 명확하고 직관적인 URI로 식별됩니다.
4. **상태 전이:** 클라이언트는 상태 전이(예: 링크)를 통해 자원 간을 탐색할 수 있습니다.

## 차이점 요약

- **REST API:** REST 원칙을 따르는 모든 API를 의미합니다.
- **RESTful API:** REST 원칙을 준수하여 설계된 API를 의미합니다.

간단히 말해, 모든 RESTful API는 REST API의 한 종류이지만, 모든 REST API가 반드시 RESTful하지는 않을 수 있습니다. RESTful이라는 용어는 REST 원칙을 얼마나 잘 준수하고 있는지를 강조하기 위해 사용됩니다.