REST API 개념 정리

1. API (Application Programing Interface)

이미지 출처 > https://www.olyslager.com/api/

API는 애플리케이션 간의 상호작용을 가능하게 하는 규약이다. 클라이언트(사용자 인터페이스)가 API에 요청을 보낸 후 API는 서버와 상호 작용하여 요청을 처리하고 응답을 반환한다.

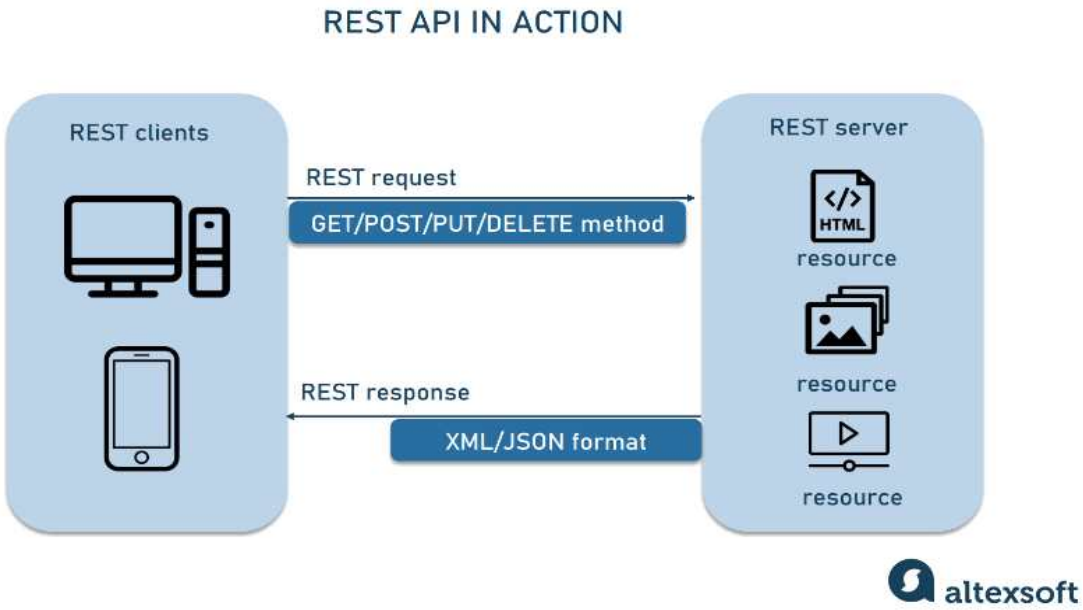
클라이언트: 사용자 인터페이스나 애플리케이션을 의미한다. 사용자의 요청을 API로 전달한다.

API: 클라이언트와 서버 사이의 중간자 역할을 한다. 클라이언트로부터 받은 요청을 서버가 이해할 수 있는 형태로 변환하고, 서버의 응답을 다시 클라이언트에게 전달한다.

서버: 요청을 처리하고 필요한 작업을 수행한 후, 그 결과를 API를 통해 클라이언트에게 전송한다.

API를 사용함으로써 클라이언트는 서버의 내부 구조나 복잡한 로직을 몰라도 서버와 상호작용할 수 있게 된다. 이를 통해 개발자들은 서비스의 다양한 기능을 더 쉽고 효율적으로 이용할 수 있다.

2. REST API (Representational State Transfer API)



이미지출처 > https://www.altexsoft.com/blog/rest-api-design/

- REST API는 웹 서비스를 구축하기 위한 일련의 지침과 원칙을 제공한다.
  클라이언트와 서버 간의 데이터 교환 및 상호작용을 가능하게 하는 인터페이스이다.

- REST는 인터넷 상에서 클라이언트와 서버 간의 통신 방식을 정의한 아키텍처 스타일이다. 웹의 기본 프로토콜인 HTTP를 사용하여 데이터와 자원을 주고받는다.

- REST API는 웹 개발에서 중요한 역할을 한다. 데이터와 자원의 효율적인 교환을 가능하게 하며 웹 서비스와 애플리케이션의 기능을 확장하는 데 필수적이다. REST API의 이해는 현대 웹 개발에서 매우 중요한 기술이다.

REST API의 핵심 원칙

1) Client-Server Architecture
- 클라이언트와 서버는 독립적으로 운영되며 서로의 요청과 응답으로 통신한다.

2) Stateless
- 각 요청은 독립적이며 서버는 클라이언트의 상태를 저장하지 않는다.

3) Cacheable
- 클라이언트는 서버의 응답을 캐시하여 성능을 향상시킬 수 있다.

4) Uniform Interface
 - 통일된 인터페이스를 통해 정보가 교환되며 이는 REST API를 쉽게 이해하고 사용할 수 있게 해준다.

REST API 사용 예시

- 자원 기반의 URL 사용하며 HTTP 메서드를 사용한 작업 표현

GET : 데이터 조회
POST : 새로운 데이터 생성
PUT : 데이터 전체 업데이트
PATCH : 데이터 부분 업데이트
DELETE : 데이터 삭제

예시)
http://example.com/product + GET , POST , PUT , DELETE
http://example.com/order + GET , POST , PUT , DELETE
http://example.com/notice + GET , POST , PUT , DELETE

REST API의 장점

1) 플랫폼 독립성
 - REST API는 HTTP를 기반으로 하므로 어떤 프로그래밍 언어나 플랫폼에서도 사용할 수 있다.

2) 확장성
 - REST 원칙에 따라 설계된 API는 유연하고 확장 가능하다.

3) 쉬운 통합
 - 다양한 서비스와 쉽게 통합될 수 있다.

REST API , RESTful API

- REST API는 REST의 원칙을 기반으로 하는 API이지만 반드시 모든 원칙을 엄격하게 따르는 것은 아니다. 반면 RESTful API는 REST의 원칙을 엄격하게 준수하는 API를 의미한다. 두 용어가 혼용되어 사용되기는 하지만 이러한 차이점을 이해하는 것이 중요하다.