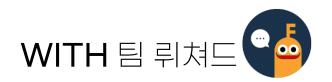


K-Digital Training 웹 풀스택 과정

# Spring API





### Rest API



#### API란?

- Application Programming Interface
- Interface란, 어떤 장치 간 정보를 교환하기 위한 수단이나 방법
- 대표적인 interface는 마우스, 키보드, 터치패드
- 개발에서의 **API**는 요청과 응답을 구성하는 방법에 대한 정보가 들어있다.



#### Rest란?

- Representational State Transfer
- 서버와 클라이언트 통신 방법 중 하나로, http URI 를 통해 자원을 명시하고 http method를 이용해 자원을 교환하는 통신 방법





#### Rest의 특징

- Server-Client ¬조
- Stateless (무상태)
- Cacheable (캐시 처리 가능)
- Layered System (계층화)
- Uniform Interface (인터페이스 일관성)



#### Rest의 장단점

- 장점
  - http 프로토콜의 인프라를 그대로 사용하므로 REST API 사용을 위한 별도의 인프라를 구축할 필요가 없다.
  - http 표준 프로토콜을 따르는 모든 플랫폼에서 사용이 가능하다.
  - 서버와 클라이언트의 역할을 명확하게 분리한다.
- 단점
  - http method 형태가 제한적이다.
  - 구형 브라우저에서는 호환이 되지 않아 지원해주지 못하는 동작이 많다. (IE)



#### REST API란?

- REST 아키텍처의 조건을 준수하는 API
- REST의 규칙을 모두 지켜 구현된 웹 서비스를 RESTful 하다고

마하다

규칙은 직접 알아보기!



| GET    | /movies     | Get list of movies       |
|--------|-------------|--------------------------|
| GET    | /movies/:id | Find a movie by its ID   |
| POST   | /movies     | Create a new movie       |
| PUT    | /movies     | Update an existing movie |
| DELETE | /movies     | Delete an existing movie |



#### RESTful이란?

- REST의 원리를 따르는 시스템
- 원리를 따르기만 하면 다 RESTful? NO!
- 원리를 따르면서 REST API 설계 규칙을 올바르게 지킨 시스템이 RESTful 한 시스템이다.



## DTO

#### DTO 란?

- Data Transfer Object
- 계층 간 데이터 교환을 위해 사용하는 객체
- 다른 로직을 가지지 않는 순수한 데이터 객체 ( Java Beans )

```
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class DTOExample{
    private String title;
    private String content;
    private String writer;
}
```



#### DTO 란?

```
⇒import lombok.Getter;
@Getter
@Setter
 public class DTOExample{
    private String title;
    private String content;
    private String writer;
```

```
public String getTitle() {
public void setTitle(String title) {
   this.title = title;
public String getContent() {
   return content;
public void setContent(String content) {
    this.content = content;
public String getWriter() {
public void setWriter(String writer) {
    this.writer = writer;
```







#### VO란?

- Value Object
- DTO와 비슷하지만, VO는 read-Only 속성을 갖고 있는 객체
- Getter 만 가지고 있어 값에 대한 수정이 불가능하다.



#### DTO vs VOS

|     | DTO                                 | vo                             |
|-----|-------------------------------------|--------------------------------|
| 목적  | 계층간 데이터 전달                          | 값 자체 표현                        |
| 동등성 | 필드값이 같아도<br>같은 객체 x                 | 필드값이 같으면 같은 객체                 |
| 가변성 | setter 존재 시 가변<br>setter 비 존재 시 불가변 | 불변                             |
| 로직  | getter/setter외의<br>로직이 필요하지 않음      | getter/setter외의<br>로직이 있어도 무방함 |