


SeSAC 4기, 

Spring API 수업

WITH 팀 리처드



# Rest API

# API란?

- Application Programming Interface
- Interface란, 어떤 장치 간 정보를 교환하기 위한 수단이나 방법
- 대표적인 interface는 마우스, 키보드, 터치패드
- 개발에서의 API는 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

{REST}

# Rest의 특징

- Server-Client 구조
- 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 말한다.

규칙은 직접 알아보기!

“REST”

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.



CRUD

# GET 방식

- Get method 의 URL을 받을 때는 Controller에서 `@GetMapping(url)`을 이용해야 한다.

```
@GetMapping(url주소)
public String 함수이름() {
    return 템플릿 파일 명;
}
```

# (get) ?key=value

- `?key=value` 형태로 url이 넘어올 때 Controller에서 받는 방법은 `@RequestParam` 이용하기!

```
@GetMapping(url주소)
public String 함수이름(@RequestParam(value="key") String key) {
    return 템플릿 파일 명;
}
```

- 이때, key라는 변수에 ?key=value 로 넘어온 value 값이 들어간다.
- @RequestParam에 required 값을 설정하면 없어도 실행된다.

# (get) ~/{value}

- ~/{value} 형태로 url 이 넘어올 때 Controller에서 받는 방법은 **@PathVariable** 이용하기!

```
@GetMapping(url주소/{value})  
public String 함수이름(@PathVariable String value) {  
    return 템플릿 파일 명;  
}
```

- 이때, value 라는 변수에 url로 넘어온 값이 담긴다.

# (get) ~/{value}

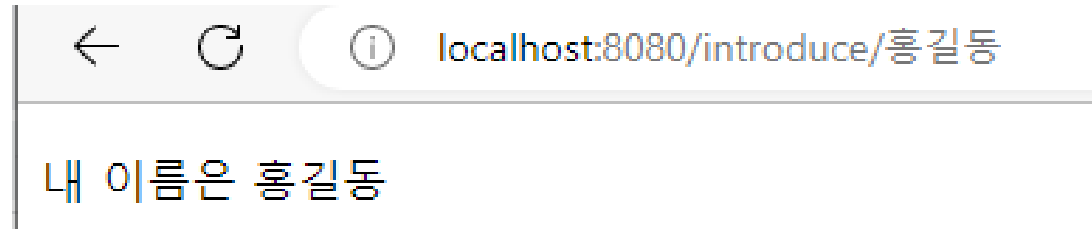
- 만약, 이름을 다르게 설정하고 싶다면?

```
@GetMapping(url주소/{value},{value2})  
public String 함수이름(@PathVariable String value, @PathVariable("value2") String abc) {  
    return 템플릿 파일 명;  
}
```

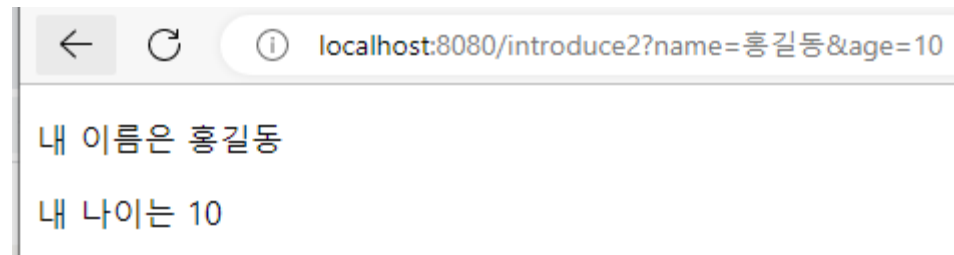
- value2 위치에 넘어온 값이 abc라는 변수에 담기게 된다.
  - Ex) “url주소/1/2” 라는 URL 일 때, abc 변수에 2가 담긴다.

# 실습 3. API-GET

1. “<http://localhost:8080/introduce/홍길동>” 으로 접속했을 때 아래와 같이 나오게 만들기



2. “<http://localhost:8080/introduce2?name=홍길동&age=10>”으로 접속했을 때 아래와 같이 나온다 할 때, 본인 이름으로 변경해서 실습



# POST 방식

- Post method 의 URL을 받을 때는 Controller에서 `@PostMapping(url)`을 이용해야 한다.

```
@PostMapping(url주소)
public String 함수이름() {
    return 템플릿 파일 명;
}
```

# POST 방식

```
// API-post.html
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
  <form action="mvc-post" method="post">
    <input type="text" name="name">
    <input type="number" name="age">
    <button>전송</button>
  </form>
</body>
</html>
```

```
@PostMapping("mvc-post")
public String postMVC(
    @RequestParam(value = "name", required = false) String name,
    @RequestParam(value = "age", required = false) String age,
    Model model) {
    model.addAttribute("name", name);
    model.addAttribute("age", age);
    return "API";
}
```



# 실습 4. post으로 정보 받기

폼 전송 - 실습

이름

성별 ☐ 남자 ☐ 여자

생년월일  년  월  일

관심사 ☐ 여행 ☐ 패션 ☐ 음식

1. 이름, 성별, 생년월일, 관심사 를 포함하여 4개 이상의 정보를 입력받고, post로 form 전송하기

이름 : 홍길동  
성별 : 남자  
생년월일 : 1976-5-15  
관심사 : 여행, 패션, 음식

# API로 이용하기

- 앞에서 배운 방식은 return을 이용해 Template view를 무조건 불러왔다.
- API로만, 데이터만 전달하고 싶을 때는?
- **@ResponseBody** 이용하기

```
@GetMapping("response-string")
@ResponseBody
public String getString(@RequestParam("name") String name) { return "hello " + name; }
```

- Node.js에서의 res.send() 와 동일하다고 생각해도 무방!