Seoul Software ACademy

웹 개발자 부트캠프 과정

SeSAC x CODINGOn

With. 팀 뤼쳐드

API (Get, Post)

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}

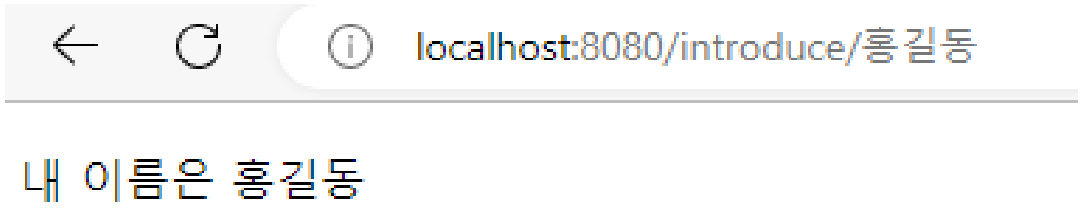
- 만약, 이름을 다르게 설정하고 싶다면?

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

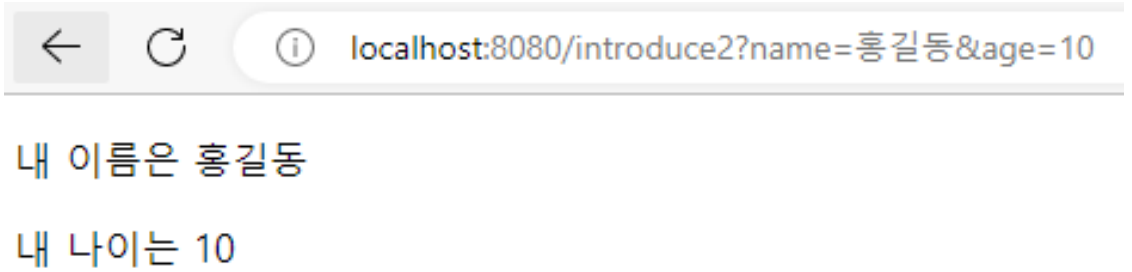
- value2 위치에 넘어온 값이 abc라는 변수에 담기게 된다.
  - Ex) “url주소/1/2” 라는 URL 일 대, abc 변수에 2가 담긴다.

실습. 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.

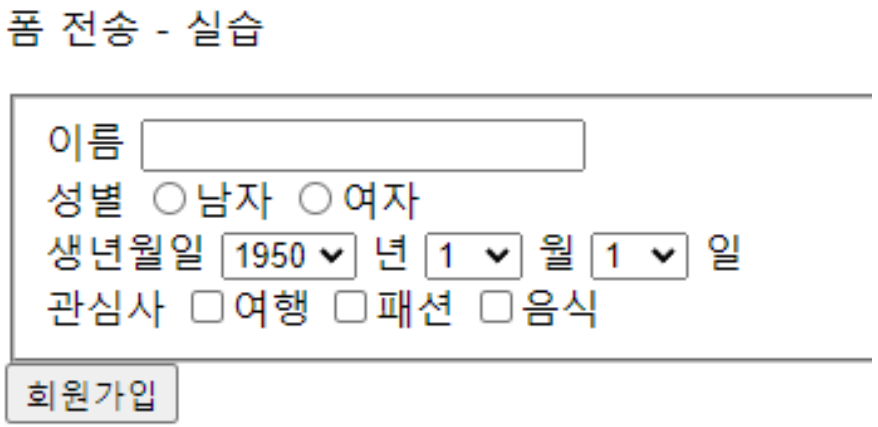
```
@PostMapping(url주소)
public String 함수이름() {
    return 템플릿 파일 명;
}
```

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. post으로 정보 받기



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!

DTO 이용하기

```
<form action="/dto/response1" method="get" style="margin-right: 20px;">
    <h3>Get - DTO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">GET 전송</button>
</form>
<form action="/dto/response2" method="post" style="margin-right: 20px;">
    <h3>Post - DTO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">Post 전송 ( RequestBody X )</button>
</form>
<form action="/dto/response3" method="post" style="margin-right: 20px;">
    <h3>Post - DTO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">Post 전송 ( RequestBody O )</button>
</form>
```

```
public class UserDTO {
    private String name;
    private String age;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAge() {
        return age;
    }

    public void setAge(String age) {
        this.age = age;
    }
}
```

실습. DTO 이용하기 – 일반 폼

```java
@GetMapping("/dto/response1")
@ResponseBody
public String dtoAPI1(UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
@PostMapping("/dto/response2")
@ResponseBody
public String dtoAPI2(UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
@PostMapping("/dto/response3")
@ResponseBody
public String dtoAPI3(@RequestBody  UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
```

Controller 코드가 좌측과 같을 때 정보가 return 되는지 확인하기

Slack 에 return 여부로 과제 제출!

Ex) O O O

O X O

VO 이용하기

```
<form action="/vo/response1" method="get" style="margin-right: 20px;">
    <h3>Get - VO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">GET 전송</button>
</form>
<form action="/vo/response2" method="post" style="margin-right: 20px;">
    <h3>Post - VO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">Post 전송 ( RequestBody x )</button>
</form>
<form action="/vo/response3" method="post" style="margin-right: 20px;">
    <h3>Post - VO</h3>
    이름 : <input type="text" name="name">
    <br>
    나이 : <input type="text" name="age">
    <br>
    <button type="submit">Post 전송 ( RequestBody O )</button>
</form>
```

```
public class UserVO {
  private String name;
  private String age;

  public String getName() {
    return name;
  }

  public String getAge() {
    return age;
  }
}
```

실습. VO 이용하기 – 일반 폼

```java
@GetMapping("/vo/response1")
@ResponseBody
public String voAPII1(UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
@PostMapping("/vo/response2")
@ResponseBody
public String voAPI2(UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
@PostMapping("/vo/response3")
@ResponseBody
public String voAPI3(@RequestBody  UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
```

Controller 코드가 좌측과 같을 때 정보가 return 되는지 확인하기

Slack 에 return 여부로 과제 제출!

Ex) O O O

O X O

Axios 이용하기

Axios - DTO

```
<h2>DTO 이용하기 - axios</h2>
<div style="display: flex">
    <form action="/axios/response1" method="get" style="margin-right: 20px;" id="form_dto1">
        이름 : <input type="text" name="name">
        <br>
        나이 : <input type="text" name="age">
        <br>
        <button type="button" onclick="dtoResponse1();">GET ( 일반 ) - axios</button>
        <button type="button" onclick="dtoResponse2();">Get ( DTO ) - axios</button>
    </form>
    <form action="/axios/response3" method="post" style="margin-right: 20px;" id="form_dto2">
        <h3>Post ( 일반 ) - axios</h3>
        이름 : <input type="text" name="name">
        <br>
        나이 : <input type="text" name="age">
        <br>
        <button type="button" onclick="dtoResponse3();">Post ( 일반 ) - axios</button>
        <button type="button" onclick="dtoResponse4();">Post ( DTO ) - axios ( RequestBody X )</button>
        <button type="button" onclick="dtoResponse5();">Post ( DTO ) - axios ( RequestBody O )</button>
    </form>
</div>
```

```
<script>
    function dtoResponse1() {
        var form = document.getElementById('form_dto1');
        axios.get(`/axios/response1?name=${form.name.value}&age=${form.age.value}`)
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse1 : ', res.data );
        });
    }
    function dtoResponse2() {
        var form = document.getElementById('form_dto1');
        axios.get(`/axios/response2?name=${form.name.value}&age=${form.age.value}`)
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse2 : ', res.data );
        });
    }
    function dtoResponse3() {
        var form = document.getElementById('form_dto2');
        axios.post(`/axios/response3`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse3 : ', res.data );
        });
    }
    function dtoResponse4() {
        var form = document.getElementById('form_dto2');
        axios.post(`/axios/response4`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse1 : ', res.data );
        });
    }
    function dtoResponse5() {
        var form = document.getElementById('form_dto2');
        axios.post(`/axios/response5`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse5 : ', res.data );
        });
    }
</script>
```

실습. Axios - DTO

```java
// DTO - axios
@GetMapping("/axios/response1")
@ResponseBody
public String axiosAPI1(@RequestParam(value="name") String name, @RequestParam(value="age") String age) {
    String msg = "이름 : " + name + "\n나이 : " + age;
    return msg;
}
@GetMapping("/axios/response2")
@ResponseBody
public String axiosAPI2(UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
@PostMapping("/axios/response3")
@ResponseBody
public String axiosAPI3(@RequestParam(value="name") String name, @RequestParam(value="age") String age) {
    String msg = "이름 : " + name + "\n나이 : " + age;
    return msg;
}
@PostMapping("/axios/response4")
@ResponseBody
public String axiosAPI4(UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
@PostMapping("/axios/response5")
@ResponseBody
public String axiosAPI5(@RequestBody  UserDTO userDTO) {
    String msg = "이름 : " + userDTO.getName() + "\n나이 : " + userDTO.getAge();
    return msg;
}
```

실습. Axios – DTO (2)

Controller 코드가 앞의 실습과 같을 때 정보가 return 되는지 확인하기

Slack 에 return 여부로 과제 제출!

Ex)

- Get 일반 :
- Get DTO :
- Post 일반 :
- Post DTO - RequestBody X :
- Post DTO - RequestBody O :

- Get 일반 : 가능
- Get DTO : Null
- Post 일반 : Null
- Post DTO - RequestBody X : 실패
- Post DTO - RequestBody O : 실패

AI 기능은 '스페이스 키', 명령어는 '/' 입력

Axios - VO

```
<h2>VO 이용하기 - axios</h2>
<div style="display: flex">
    <form id="form_vo1" style="margin-right: 20px;">
        <h3>Get ( 일반 ) - axios</h3>
        이름 : <input type="text" name="name">
        <br>
        나이 : <input type="text" name="age">
        <br>
        <button type="button" onclick="voResponse1();">GET ( 일반 ) - axios</button>
        <button type="button" onclick="voResponse2();">Get ( VO ) - axios</button>
    </form>
    <form id="form_vo2" style="margin-right: 20px;">
        <h3>Post ( 일반 ) - axios</h3>
        이름 : <input type="text" name="name">
        <br>
        나이 : <input type="text" name="age">
        <br>
        <button type="button" onclick="voResponse3();">Post ( 일반 ) - axios</button>
        <button type="button" onclick="voResponse4();">Post ( VO ) - axios ( RequestBody X )</button>
        <button type="button" onclick="voResponse5();">Post ( VO ) - axios ( RequestBody O )</button>
    </form>
</div>
```

```
<script>
    function voResponse1() {
        var form = document.getElementById('form_vo1');
        axios.get(`/axios/vo/response1?name=${form.name.value}&age=${form.age.value}`)
        .then((res)=>{
            console.log( res );
            console.log( 'dtoResponse1 : ', res.data );
        });
    }
    function voResponse2() {
        var form = document.getElementById('form_vo1');
        axios.get(`/axios/vo/response2?name=${form.name.value}&age=${form.age.value}`)
        .then((res)=>{
            console.log( res );
            console.log( 'voResponse2 : ', res.data );
        });
    }
    function voResponse3() {
        var form = document.getElementById('form_vo2');
        axios.post(`/axios/vo/response3`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'voResponse3 : ', res.data );
        });
    }
    function voResponse4() {
        var form = document.getElementById('form_vo2');
        axios.post(`/axios/vo/response4`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'voResponse4 : ', res.data );
        });
    }
    function voResponse5() {
        var form = document.getElementById('form_vo2');
        axios.post(`/axios/vo/response5`, {name: form.name.value, age: form.age.value})
        .then((res)=>{
            console.log( res );
            console.log( 'voResponse5 : ', res.data  );
        });
    }
</script>
```

실습. Axios - VO

```java
@GetMapping("/axios/vo/response1")
@ResponseBody
public String axiosvoAPI1(@RequestParam(value="name") String name, @RequestParam(value="age") String age) {
    String msg = "이름 : " + name + "\n나이 : " + age;
    return msg;
}
@GetMapping("/axios/vo/response2")
@ResponseBody
public String axiosvoAPI2(UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
@PostMapping("/axios/vo/response3")
@ResponseBody
public String axiosvoAPI3(@RequestParam(value="name") String name, @RequestParam(value="age") String age) {
    String msg = "이름 : " + name + "\n나이 : " + age;
    return msg;
}
@PostMapping("/axios/vo/response4")
@ResponseBody
public String axiosvoAPI4(UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
@PostMapping("/axios/vo/response5")
@ResponseBody
public String axiosvoAPI5(@RequestBody UserVO userVO) {
    String msg = "이름 : " + userVO.getName() + "\n나이 : " + userVO.getAge();
    return msg;
}
```

실습. Axios – VO (2)

Controller 코드가 앞의 실습과 같을 때 정보가 return 되는지 확인하기

Slack 에 return 여부로 과제 제출!

Ex)

- Get 일반 : 가능
- Get VO: Null
- Post 일반 : Null
- Post VO - RequestBody X : 실패
- Post VO - RequestBody O : 실패

총 정리

1.일반 폼 전송 시

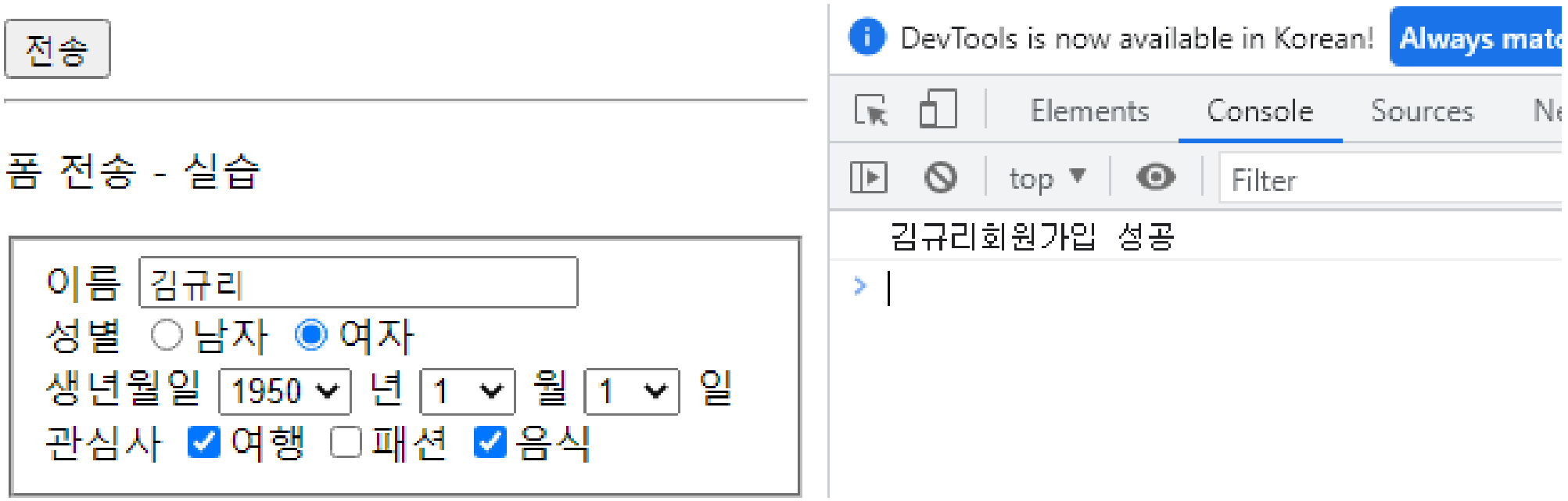
1. RequestParam : Get과 Post 둘 다 가능
2. PathVariable : Get 만 가능

2. DTO 이용 - 일반 폼 전송
   - Get 전송 가능
   - Post 전송 - RequestBody 가 없을 경우 가능
   - Post 전송 - RequestBody 가 있을 경우 실패
3. VO 이용 - 일반 폼 전송
   - Get 전송 Null
   - Post 전송 - RequestBody 가 없을 경우 Null
   - Post 전송 - RequestBody 가 있을 경우 실패

1. Axios 이용 - DTO
   - Get 일반 : 가능
   - Get DTO : 가능
   - Post 일반 : 실패
   - Post DTO - RequestBody X : Null
   - Post DTO - RequestBody O : 가능
2. Axios 이용 - VO
   - Get 일반 : 가능
   - Get VO: Null
   - Post 일반 : 실패
   - Post VO- RequestBody X : Null
   - Post VO- RequestBody O : 가능

실습. 동적폼전송 실습

Axios를 이용해 동적폼전송 구현하기 (VO 이용)



실습. 회원관리 시스템

- C : 회원 가입
- R : 로그인
- U : 회원 정보 수정
- D : 회원 탈퇴

- 위의 기능을 수행할 수 있는 RESTful 한 시스템을 만들어보시오.