13장 RESTful 웹 서비스

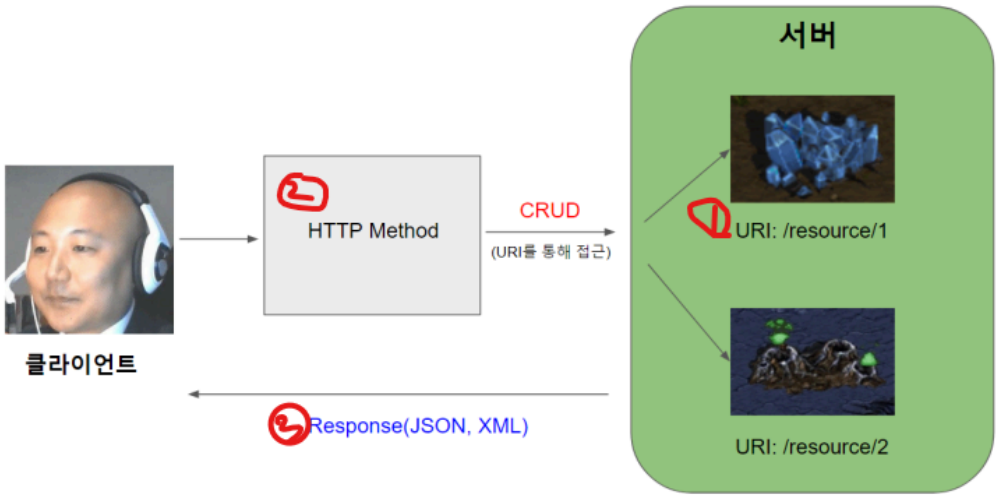
1. RESTful 웹 서비스

REST 원리 사용한 것

REST

HTTP에서 어떤 자원에 대한 CRUD 요청을 리소스와 메서드로 표현하여 특정한 형태로 전달하는 방식

1. CURD 연산을 수행하기 위해서 uri로 자원을 명시
2. HTTP 방식을 사용하여 요청을 보냄
3. 응답은 JSON, XML, TEXT, RSS로 보냄



https://tibetsandfox.tistory.com/19

구성 요소 : 리소스, 메서드, 리소스 형태

1. **리소스** : uri을 말함. 클라이언트는 고유의 아이디가 있는 리소스에 요청을 보냄

2. **메서드** : 서버에 요청을 보내는 방식. GET, POST, PUT, PATCH, DELETE. CURD 연산에 맞는 메서드 사용
3. **리소스 형태** : 서버 클라이언트가 주고 받는 데이터 형태. JSON, XML, TEXT, RSS

---

2. RESTful 방식 애너테이션

@RequestBody, @ResponseBody, @RestController

@RequestBody

- 요청본문
- 이 애너테이션이 붙은 파라미터는 http 요청 본문이 그대로 전달
- http 요청 본문을 자바 객체로 매핑

- name = value 방식은 @RequestParam / @ModelAttribute 사용

  XML/JSON 방식은 @RequestBody 사용

```
@Controller
@RequestMapping("/exam01")
public class Example01Controller {

    @GetMapping
    public String showForm() {
        return "viewPage01";
    }
    @PostMapping
    public String submit(@RequestBody String param, Model model) {

        model.addAttribute("data1", "@RequestBody로 정보 받기");
        model.addAttribute("data2", param);
        return "viewPage01_result";
    }
```

```
}
```

- **name=Hong&age=22&email=hong@naver.com** 으로 응답

- HashMap

```
@Controller
@RequestMapping("/exam02")
public class Example02Controller {

    @GetMapping
    public String showForm() {
        return "viewPage02";
    }

    @PostMapping
    public String submit(@RequestBody HashMap<String, String> map) {
        System.out.println(map);

        return "viewPage02";
    }
}
```

```
<html>
<head>
<title>Chap13</title>
</head>
<body>
    <h3>RESTful 웹 서비스</h3>
    <script src="https://code.jquery.com/jquery-latest.min.js"></script>

    <script type="text/javascript">
    function myMethod() {
```

```
        var get_form = document.form;

        var obj = new Object();
        obj.name= get_form.name.value;    //폼의 name 입력 값
        obj.age = get_form.age.value;      //폼의 age 입력 값
        obj.email = get_form.email.value;   //폼의 email 입력 값
      var json_data = JSON.stringify(obj); //값이나 객체를 JSON 문자열로 변환

        $.ajax({
        type : "POST",   //POST 방식
           url : "exam02",  //요청 URL
           data : json_data, //전송 데이터
        contentType : "application/json", //json 형식
        success : function(data) {
              alert("성공" );
           },
        error : function(jqXHR, textStatus, errorThrown) {
              alert("실패 : " + textStatus);
           }
        });
   }
   </script>

<body>
   <form name="form" method = "post" >
      <p>이름 : <input name="name" />
      <p>나이 : <input name="age" />
      <p>이메일 : <input name="email" />
      <p><input onclick="myMethod()" type="button" value="전송하기"/>
  </form>
</body>
<html>
```

- **{name=Hong, age=22, email=hong@naver.com}** 으로 응답

@ResponseBody

- 응답본문
- 자바 객체를 http요청 바디 내용으로 매핑 → 클라이언트에게 전달

```
@Controller
@RequestMapping("/exam03")
public class Example03Controller {
    @ResponseBody
    @GetMapping
    public Person submit() {
        Person person = new Person();
        person.setName("HongGilSon");
        person.setAge("20");
        person.setEmail("hong@naver.com");
        System.out.println(person);
        return person;
    }
}
```

```
@Data
public class Person {
    private String name;
    private String age;
    private String email;
}
```

- **Person {"name":"Hong", "age":"22", "email":"hong@naver.com"}** 으로 응답

@RestController

- @ResponseBody + @Controller
- 자바 객체가 http 응답 바디 내용에 매핑되어 전달

```
@RestController
@RequestMapping("/exam04")
public class Example04Controller {
    @GetMapping
    public Person submit() {
        Person person = new Person();
        person.setName("HongGilSon");
        person.setAge("20");
        person.setEmail("hong@naver.com");
        System.out.println(person);
        return person;
    }
}
```

- @ResponseBody와 동일하게 응답

---

3. RESTful 웹 서비스 CRUD

- CRUD는 CREATE, READ, UPDATE, DELETE의 줄임말
- uri를 통해 리소스, 리소스 위치 명확하게 식별 가능

표 13-3 HTTP 요청 방식 유형

유형	설명	CRUD 작업
POST	기존 리소스를 갱신하거나 새로운 리소스를 생성	Create
GET	리소스를 조회하여 읽어 오기	Read
PUT	리소스를 변경	Update
DELETE	기존 리소스를 삭제	Delete
OPTION	기존 리소스에 대한 리소스 작업 얻기	

```
@Controller
@RequestMapping("/exam05")
public class Example05Controller {
   @GetMapping
   public String showForm(@ModelAttribute Person person) {
      return "viewPage05";
   }

   @PutMapping
   public String submit(@ModelAttribute Person person, Model model) {
      model.addAttribute("data1", "@PutMapping 적용하기");
      model.addAttribute("data2", person);
      System.out.println(person);
       return "viewPage05_result";
   }
}
```

```
<body>
   <input type="hidden" name="_method" value="put"/> //put으로 처리
</body>
```

- **Person {”name”:”Hong”, “age”:”22”, “email”:”hong@naver.com”}** 으로 응답