Spring Framework

25. Spring RESTful 웹서비스 어플리케이션 작성(1)

Tacademy

CONTENTS

학습 목표

- 사용자 관리 RESTful 웹서비스 개요에 대하여 이해할 수 있습니다.
- 사용자 정보 조회 및 등록 기능 구현에 대하여 이해할 수 있습니다.
- 사용자 정보 수정 및 삭제 기능 구현에 대해 이해할 수 있습니다.

1. 사용자 관리 RESTful 웹서비스 개요

사용자 관리 RESTful 웹서비스 URI와 Method

Action	Resource URI	HTTP Method
사용자 목록	/users	GET
사용자 보기	/users/{id}	GET
사용자 등록	/users	POST
사용자 수정	/users	PUT
사용자 삭제	/users/{id}	DELETE

RESTful Controller를 위한 핵심 어노테이션(Annotation)

- Spring MVC에서는 클라이언트에서 전송한 XML이나 JSON 데이터를 Controller에서 Java객체로 변환해서 받을 수 있는 기능(수신)을 제공하고 있다.
- Java객체를 XML이나 JSON으로 변환해서 전송할 수 있는 기능(송신)을 제공하고 있다.

어노테이션	설명
@RequestBody	HTTP Request Body(요청 몸체)를 Java객체로 전달받을 수 있다.
@ResponseBody	Java 객체를 HTTP Response Body (응답 몸체)로 전송할 수 있다.

RESTful Controller를 위한 @ResponseBody가 있는 경우

```
@Controller
@RequestMapping("/user")
public class UserController {
	@RequestMapping( value="/json/{id}", method = RequestMethod.GET)
	@ResponseBody
	public UserModel getByIdInJSON( @PathVariable String id){
		UserModel user = new UserModel();
		user.setId( id);
		user.setName( "ellie");
		return user;
	}
```

- @ResponseBody가 있는 getByIdInJSON 메서드의 경우
  : MappingJacksonHttpMessageConverter가 리턴값인 UserModel 객체를 JSON으로 변환하는 작업을 처리한다.

RESTful Controller를 위한 @ResponseBody가 없는 경우

```
@Controller
@RequestMapping("/user")
public class UserController {
	@RequestMapping( value="/html/{id}", method = RequestMethod.GET)
	public String getByIdInHTML( @PathVariable String id, ModelMap model){
		UserModel user = new UserModel();
		user.setId( id);
		user.setName( "ellie");
		model.addAttribute( "user", user);
		return "user";
	}
}
```

- @ResponseBody가 없는 getByIdInHTML 메서드의 경우

  : ViewResolver에 의해 선택된 /user.jsp가 포워드 되어지고, user.jsp에서 UserModel 객체를 참조한다.

2. 사용자 정보 조회 및 등록 기능 구현

사용자 정보 조회 : 사용자 목록 조회 메서드 구현

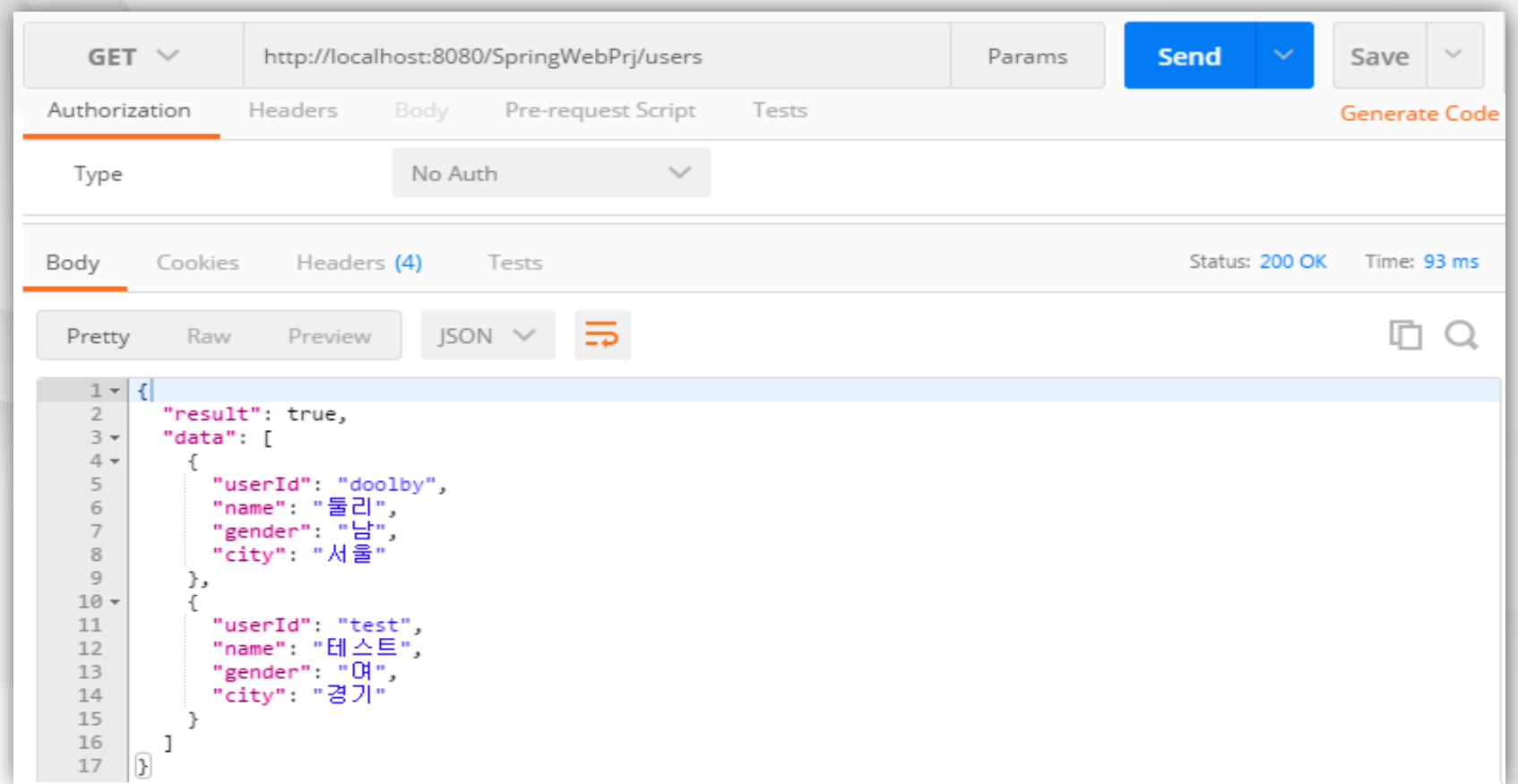
RestfulUserController.java

```
@Controller
public class RestfulUserController {
    @Autowired
    private UserService userService;

    @RequestMapping(value="/users",method=RequestMethod.GET)
    @ResponseBody
    public Map getUserList() {
        List<UserVO> userList = userService.getUserList();
        Map result = new HashMap();
        result.put("result", Boolean.TRUE);
        result.put("data", userList);
        return result;
    }
```

2. 사용자 정보 조회 및 등록 기능 구현

사용자 정보 조회 : 사용자 목록 조회 메서드 테스트



2. 사용자 정보 조회 및 등록 기능 구현

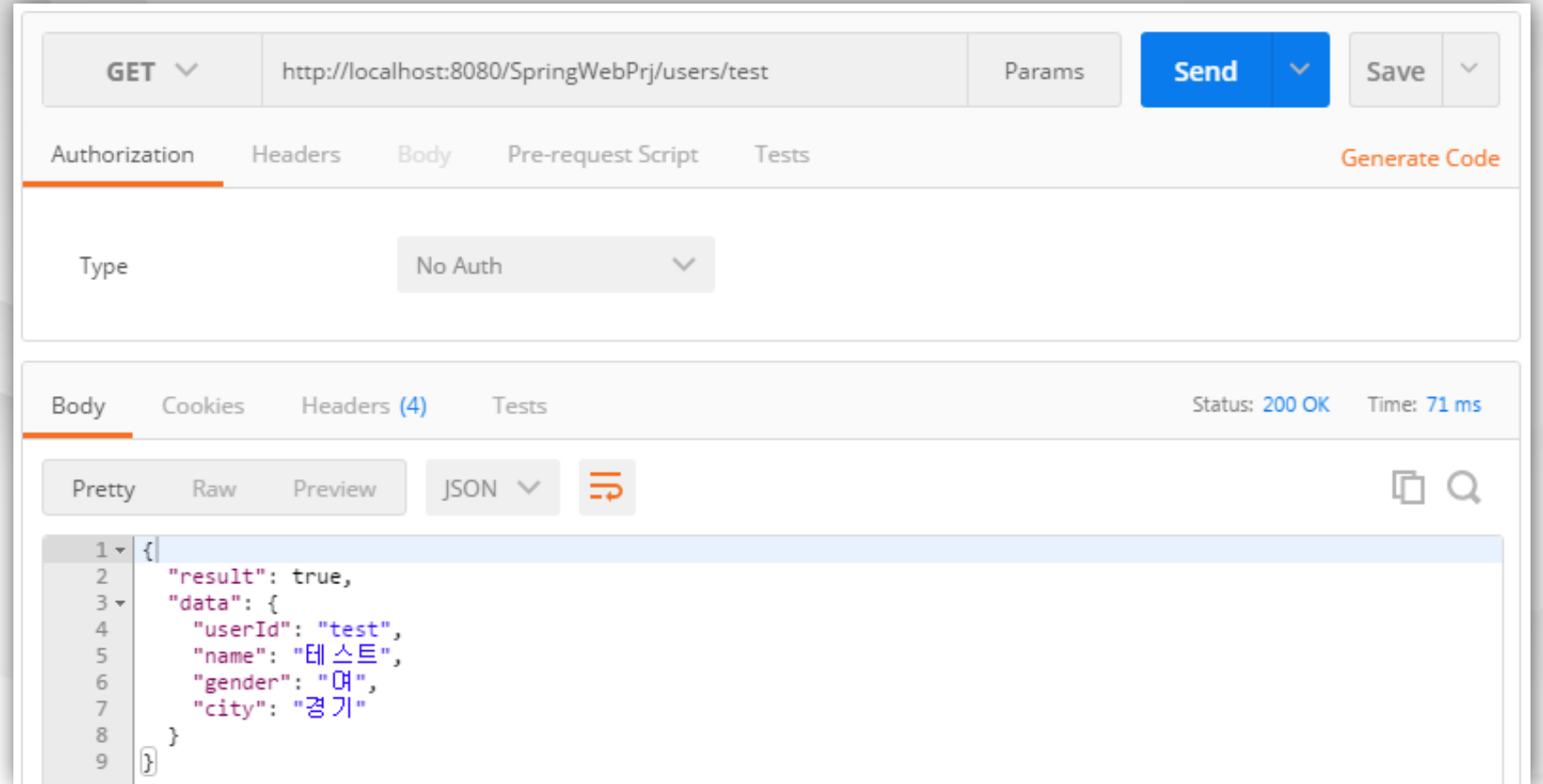
사용자 정보 조회 : 특정 사용자 조회 메서드 구현

RestfulUserController.java

```
@Controller
public class RestfulUserController {
    @Autowired
    private UserService userService;

    @RequestMapping(value="/users/{id}",method=RequestMethod.GET)
    @ResponseBody
    public Map getUser(@PathVariable String id) {
        UserVO user = userService.getUser(id);
        Map result = new HashMap();
        result.put("result", Boolean.TRUE);
        result.put("data", user);
        return result;
    }
```

사용자 정보 조회 : 특정 사용자 조회 메서드 테스트



사용자 정보 등록 : 사용자 등록 메서드 구현

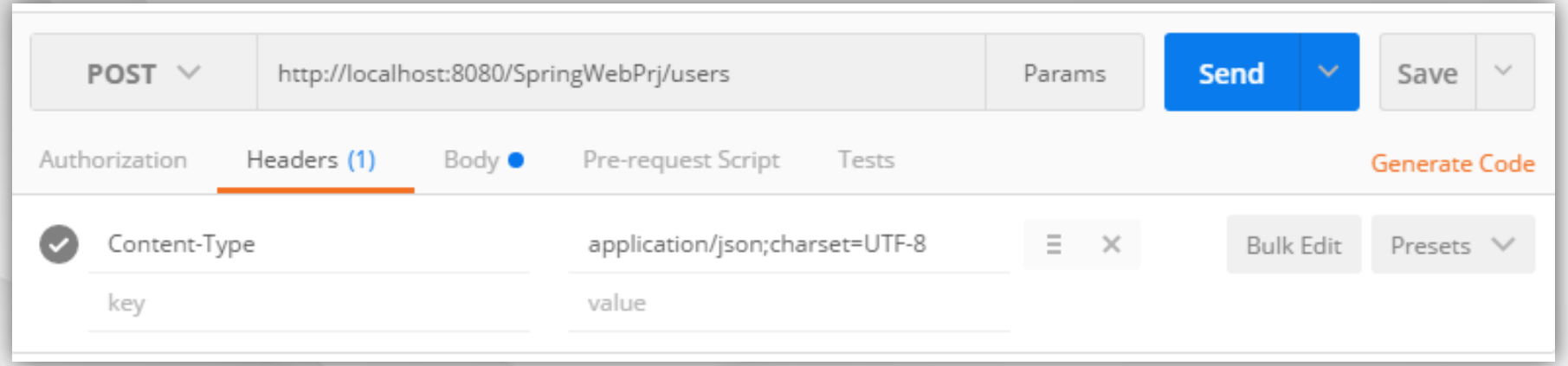
RestfulUserController.java

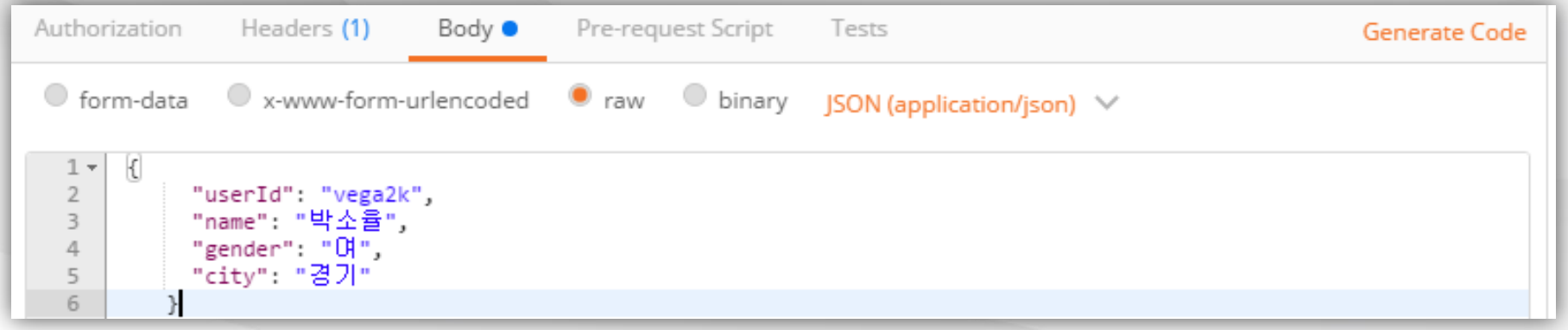
```
@Controller
public class RestfulUserController {
    @Autowired
    private UserService userService;

    @RequestMapping(value="/users",
                    method=RequestMethod.POST,
                    headers = {"Content-type=application/json"})
    @ResponseBody
    public Map insertUser(@RequestBody UserVO user) {
        if (user != null)
            userService.insertUser(user);

        Map result = new HashMap();
        result.put("result", Boolean.TRUE);
        return result;
    }
```

사용자 정보 등록 : 사용자 등록 메서드 테스트





3. 사용자 정보 수정 및 삭제 기능 구현

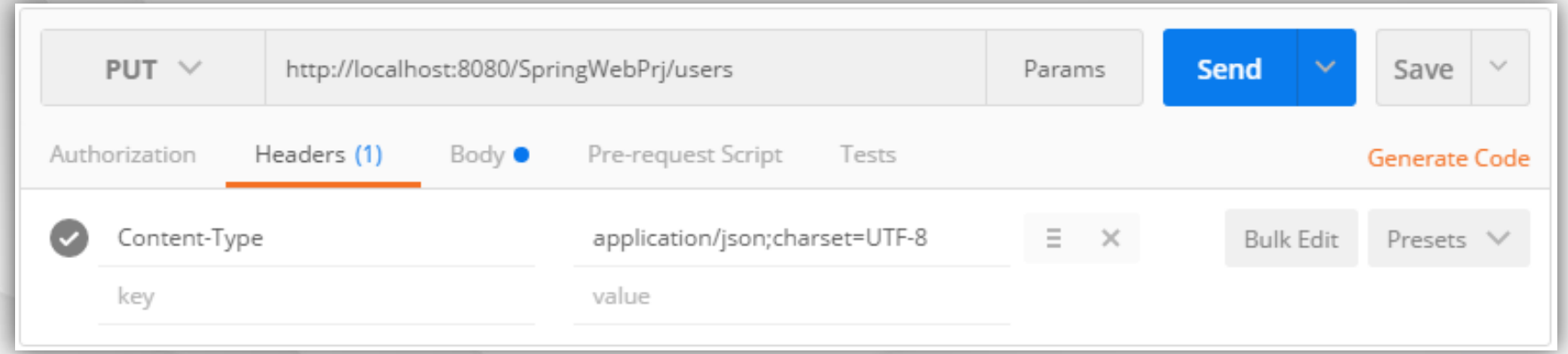
3. 사용자 정보 수정 및 삭제 기능 구현

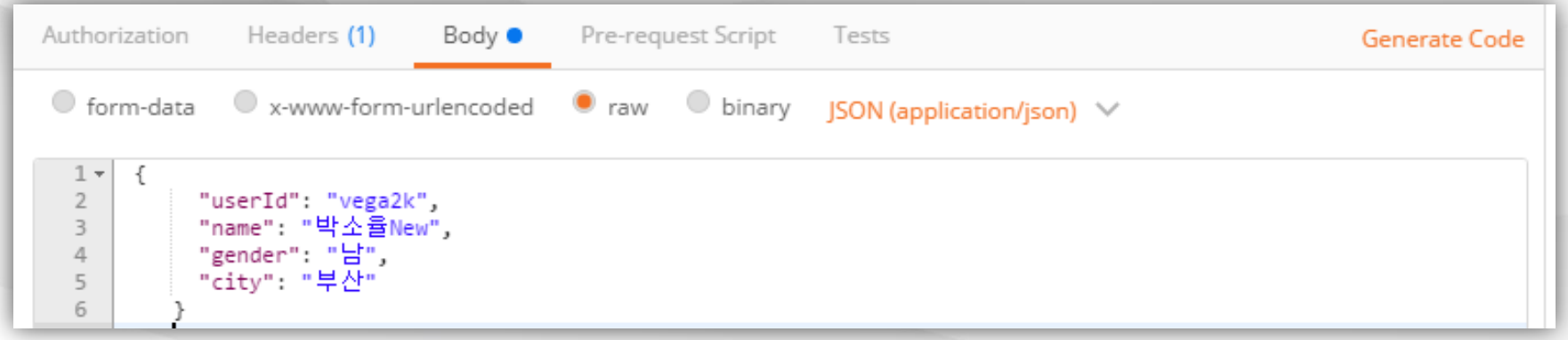
사용자 정보 수정 : 사용자 수정 메서드 구현

RestfulUserController.java

```
@Controller
public class RestfulUserController {
    @Autowired
    private UserService userService;
    @RequestMapping(value="/users",
                    method=RequestMethod.PUT,
                    headers = {"Content-type=application/json"})
    @ResponseBody
    public Map updateUser(@RequestBody UserVO user) {
        if(user != null)
            userService.updateUser(user);
        Map result = new HashMap();
        result.put("result", Boolean.TRUE);
        return result;
    }
```

사용자 정보 수정 : 사용자 수정 메서드 테스트





3. 사용자 정보 수정 및 삭제 기능 구현

사용자 정보 삭제 : 사용자 삭제 메서드 구현

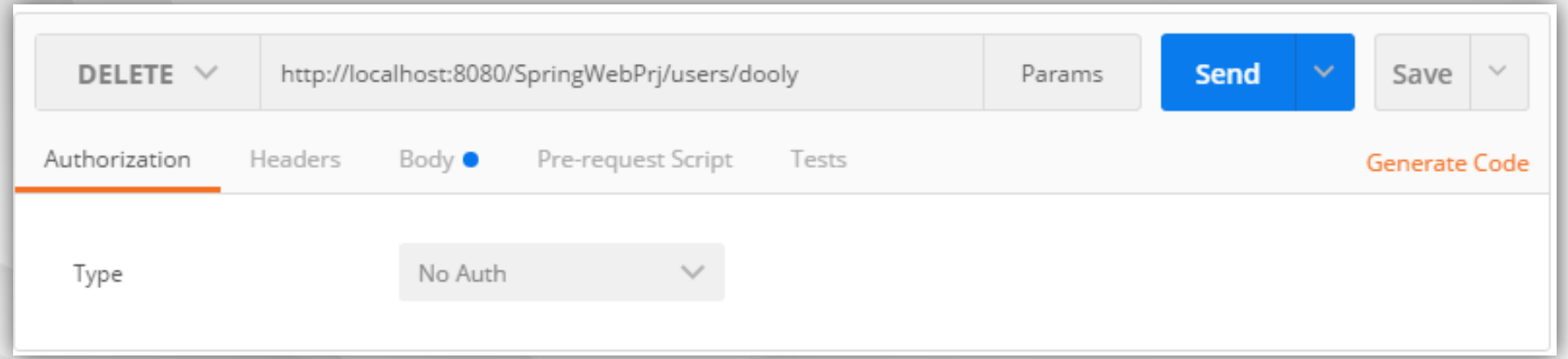
RestfulUserController.java

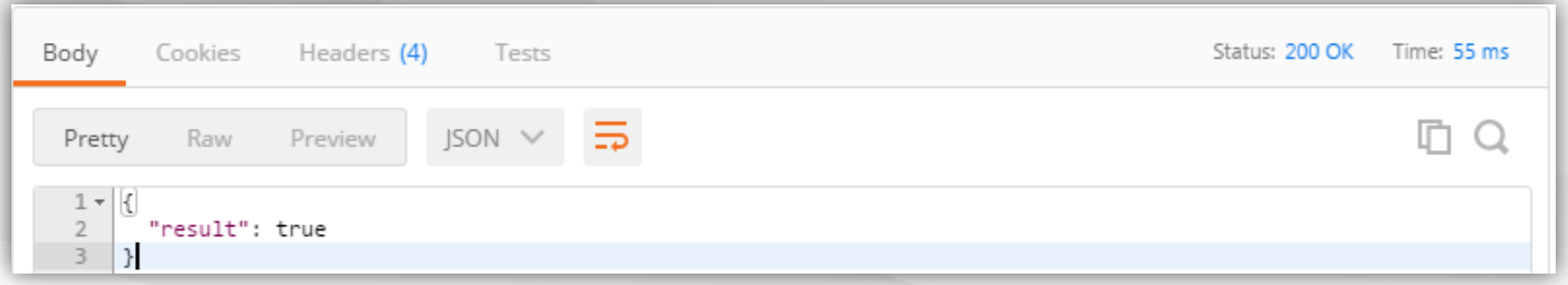
```
@Controller
public class RestfulUserController {
   @Autowired
   private UserService userService;

   @RequestMapping(value="/users/{id}",method=RequestMethod.DELETE)
   @ResponseBody
   public Map deleteUser(@PathVariable String id) {
      userService.deleteUser(id);

      Map result = new HashMap();
      result.put("result", Boolean.TRUE);
      return result;
   }
```

사용자 정보 삭제 : 사용자 삭제 메서드 테스트





학습정리

지금까지 [Spring RESTful 웹서비스 어플리케이션 작성(1)]에 대해서 살펴보았습니다.

사용자 관리 RESTful 웹서비스 개요

- Resource URI와 Http Method 규칙
- @RequestBody, @ResponseBody 어노테이션

사용자 조회 및 등록 기능 구현

- RESTful 방식의 사용자 조회 및 등록하는 기능 구현
- Postman을 이용한 조회, 등록 테스트

사용자 수정 및 삭제 기능 구현

- RESTful 방식의 사용자 수정 및 삭제하는 기능 구현
- Postman을 이용한 수정, 삭제 테스트