

스프링부트로 RestFulAPI 구현하기

# 9장 스웨거 문서

- SwaggerUI
- Schema
- Operation
- Tag
- RequestDto
- ResponseDto

박명회

## 9장 Swagger 문서란...

Swagger는 웹 서비스 명세를 문서화 해주는 오픈 소스 소프트웨어 프레임워크이다.

즉 웹 서비스가 어떤 로직을 수행하고, 이 로직을 수행하기 위해서는 어떤 값을 요청하며, 이에 따른 응답값은 무엇인지 정리해서 문서화해주는 프로젝트이다.

보통 웹 어플리케이션을 개발할 때 프론트와 백엔드로 팀을 나누어서 개발을 하는데, 이 때, 백엔드 팀이 만든 서비스를 swagger로 문서화해서 프론트 팀으로 넘겨 로직의 이해도를 높이고, 소통한다. swagger를 사용하면 개발과정 속에서 계속 변경되는 명세 문서를 알아서 주기적으로 업데이트해 주기 때문에 번거로움을 없애고, 시간을 절약할 수 있다.

## 9장 Swagger 문서란...

springboot 3.0.0 이상부터는 springfox가 아닌 springdoc-openapi-ui 라이브러리 사용 권장.

spring에는 2가지 스웨거가 존재한다. springfox와 springdoc이 있는데 과거 springfox를 주로 사용했다.

하지만 springfox는 2020년 7월 14일 기준으로 더 이상 업데이트가 안되고 있다. 반면 springdoc은

주기적으로 업데이트 중이다. springboot 버전이 올라갈수록 더더욱 springdoc-openapi를 사용해야 할 듯 하다.

```
implementation 'org.springdoc:springdoc-openapi-starter-webmvc-ui:2.6.0'
```

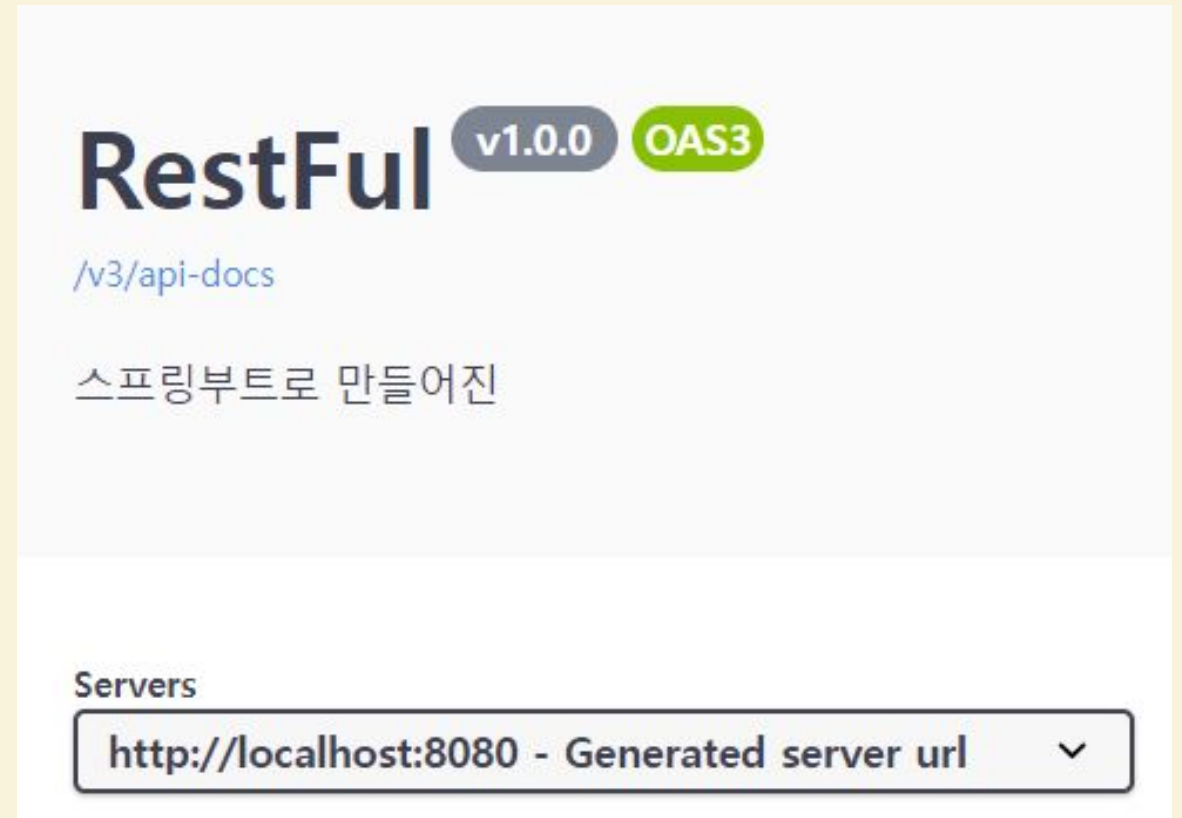
## 9장 Swagger문서

The screenshot displays the Swagger UI interface in a web browser. The address bar shows the URL `localhost:8080/swagger-ui/index.html`. The Swagger logo and "Supported by SMARTBEAR" are in the top left. A search bar contains `/v3/api-docs`, and an "Explore" button is on the right. The main heading is "OpenAPI definition" with version tags `v0` and `OAS3`. Below this, a "Servers" section shows a dropdown menu with the selected value `http://localhost:8080 - Generated server url`. The "user-controller" section is expanded, showing a list of API endpoints with their HTTP methods and a dropdown arrow on the right of each row:

- PUT** `/user/update`
- POST** `/user/insert`
- GET** `/user/selectall`
- GET** `/user/select/{id}`
- DELETE** `/user/delete/{id}`

## 9장 Swagger 전체 문서

```
@Configuration
@OpenAPIDefinition(
    info = @Info(title = "RestFul",
        description = "스프링부트로 만들어진",
        version = "v1.0.0"
    )
)
public class SwaggerConfig {
}
```



## 9장 Controller 설정

```
@Tag(name="UserController", description = "UserController API 입니다.")
@RestController
@RequiredArgsConstructor
@RequestMapping("user")
public class UserController {
    ....
}
```

```
@PutMapping("update")
@Operation(summary = "Update the user information based on the provided UserRequestDto")
@ApiResponses(value = {
    @ApiResponse(responseCode = "200", description = "Successfully updated user"),
    @ApiResponse(responseCode = "400", description = "Invalid input data"),
    @ApiResponse(responseCode = "500", description = "Internal server error")
})
public ResponseEntity<User> update(@Valid @RequestBody UserRequestDto userRequestDto){
    User dbUser = userService.saveUser(userRequestDto);
    return ResponseEntity.ok(dbUser);
}
```



## 9장 UserRequestDto Swagger 문서

@Data

```
public class UserRequestDto {  
    private long id;
```

```
    @NotBlank(message = "이름은 공백일 수 없습니다.")
```

```
    @Schema(description = "사용자 이름입니다.", example = "홍길동")  
    private String name;
```

```
    @NotBlank(message = "이메일은 공백일 수 없습니다.")
```

```
    @Email(message = "이메일 형식이 아닙니다.")
```

```
    @Schema(description = "사용자 email입니다.", example = "eee@naver.com")
```

```
    private String email;
```

```
}
```

Example Value | Schema

```
{  
  "id": 0,  
  "name": "홍길동",  
  "email": "eee@naver.com"  
}
```

## 9장 UserResponseDto Swagger 문서

```
package com.pmh.ex08.user;

import com.fasterxml.jackson.annotation.JsonInclude;
import jakarta.validation.constraints.Email;
import jakarta.validation.constraints.NotBlank;
import lombok.Data;

@Data
@JsonInclude(JsonInclude.Include.NON_NULL)
public class UserResponseDto {
    private long id;

    @NotBlank(message = "이름은 공백일 수 없습니다.")
    private String name;

    @NotBlank(message = "이메일은 공백일 수 없습니다.")
    @Email(message = "이메일 형식이 아닙니다.")
    private String email;
}
```

Code	Details
200	<div><div>Response body</div><div><pre>{   "id": 0 }</pre></div></div> <div><div>Response headers</div><div><pre>connection: keep-alive content-type: application/json date: Sat, 03 Aug 2024 06:56:48 GMT keep-alive: timeout=60 transfer-encoding: chunked</pre></div></div>



## 9장 Swagger 문서 JWT 추가

```
@Configuration
@OpenAPIDefinition(
    info = @Info(title = "MH 가 만든 USER,MAIN",
        description = "유저 등록 수정 삭제 조회 등이",
        version = "v1.0.0"
```

```
@SecurityRequirement(name = "Bearer Authentication")
```

```
)
@SecurityScheme(
    name = "Bearer Authentication",
    type = SecuritySchemeType.HTTP,
    bearerFormat = "JWT",
    scheme = "bearer"
)
```

```
public class SwaggerConfig {
}
```

```
@RestController
@RequestMapping("/test")
@RequiredArgsConstructor
@SecurityRequirement(name = "Bearer Authentication")
public class TestController {
```

감사합니다