

Swagger

Swagger는 REST API를 설계, 빌드, 문서화 및 사용하는 데 도움이 될 수 있는 OpenAPI 사양을 기반으로 구축된 오픈 소스 도구 세트이며, 이 swagger를 프로젝트에 적용 방법을 설명한다.

Configuration

의존성 설정 (gradle)

- ~/build.gradle 파일에 아래 내용 추가

```
dependencies {  
    compile 'io.springfox:springfox-swagger2:2.9.2'  
    compile 'io.springfox:springfox-swagger-ui:2.9.2'  
}
```

설정 클래스 작성

- Application 클래스가 있는 패키지에(예, com.gsc.process.integration) config 패키지를 추가하고(com.gsc.process.integration.config가 된다), SwaggerConfiguration.java 파일을 생성하여 아래 내용을 입력한다.

```
package com.gsc.process.integration.config;  
  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import springfox.documentation.builders.ApiInfoBuilder;  
import springfox.documentation.builders.PathSelectors;  
import springfox.documentation.builders.RequestHandlerSelectors;  
import springfox.documentation.service.ApiInfo;  
import springfox.documentation.spi.DocumentationType;  
import springfox.documentation.spring.web.plugins.Docket;  
import springfox.documentation.swagger2.annotations.EnableSwagger2;  
  
@Configuration  
@EnableSwagger2  
public class SwaggerConfiguration {  
  
    private ApiInfo swaggerInfo() {  
  
        return new ApiInfoBuilder()  
            .title("GSC Process Integration - Swagger Examples")
```

```

        .description("GSC Process Integration Sample API Document")
        .version("0.0.1")
        .build();

    }

    @Bean
    public Docket swaggerApi() {
        return new Docket(DocumentationType.SWAGGER_2)
            .useDefaultResponseMessages(false)
            .select()
            .apis(RequestHandlerSelectors.basePackage("com.gsc.process.integration"))
            .paths(PathSelectors.ant("/api/**"))
            //.paths(PathSelectors.ant("/**"))
            .build()
            .apiInfo(swaggerInfo());
    }
}

```

- @Configuration: 설정 클래스임을 나타내는 annotation
- @EnableSwagger2: Swagger2 버전 활성화 annotation
- ApiInfo swaggerInfo() : Swagger UI에 표시되는 설정 정보. title(), description(), version()
- Docket swaggerApi() : Swagger 설정의 핵심이 되는 Bean, swagger 스프링 MVC 프레임워크를 사용해 스웨거 문서 생성을 구성하는 빌더 클래스
 - useDefaultResponseMessages()
 - false로 설정하면, Swagger에서 제공하는 응답 코드(200, 401, 403, 404)에 대한 기본 메시지를 제거한다.
 - 불필요한 응답 코드와 메시지 제거하기 위함이며, 컨트롤러에 명시한다.
- groupName()
 - Docket Bean이 한 개일 경우 기본 값은 default이므로, 생략 가능하다.
 - 여러 Docket Bean을 생성했을 경우 groupName이 충돌하지 않아야 하므로, 버전을 명시한다.
- select() : ApiSelectorBuilder를 생성한다.
- apis() :
 - **api** 스펙이 작성되어 있는 패키지를 지정한다.
 - **RequestMapping(GetMapping, PostMapping)**이 선언된 **API**를 문서화 한다.
- paths() : apis()로 선택되어진 API 중 특정 path 조건에 맞는 API들을 다시 필터링하여 문서화한다.
- apiInfo()
 - 제목, 설명 등 문서에 대한 정보들을 보여주기 위해 호출한다.
 - 파라미터 : title, description, version, termsOfServiceUrl, contact, license, licenseUrl, vendorExtensions

Model 작성 및 주요 annotation

- 어플리케이션의 Model을 작성할 때 swagger 어노테이션을 이용한다.

```
package com.gsc.process.integration.sample;
```

```
import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.EqualsAndHashCode;
import lombok.NoArgsConstructor;
import lombok.experimental.SuperBuilder;

@SuperBuilder
@Data
@EqualsAndHashCode(callSuper = false)
@NoArgsConstructor
@AllArgsConstructor
@ApiModel(description = "공통 코드 모델")
public class SampleApiModel {
    @ApiModelProperty("그룹 코드")
    private String grpCd;

    @ApiModelProperty("상세 코드")
    private String dtlCd;

    @ApiModelProperty("코드명")
    private String cdNm;

    @Data
    public static class Criteria {
        @ApiModelProperty("사용여부")
        private String useYn;
    }
}
```

- @ApiModel: 스웨거 모델의 추가 정보를 제공
- @ApiModelProperty(name = "", example = ""): 모델 속성의 데이터를 추가하고 조작
 - name : 이름
 - example : 설명

Controller 작성 및 주요 annotation

- 어플리케이션의 Controller를 작성할 때 swagger 어노테이션을 이용한다.

```
package com.gsc.process.integration.sample;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```

import io.swagger.annotations.Api;
import io.swagger.annotations.ApiOperation;
import io.swagger.annotations.ApiParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@Api("OpenAPI for GSC Process Integration Project")
@RestController
public class SampleApiController {

    @Autowired SampleApiService sampleApiService;

    @GetMapping("/api/sample/return-map-api")
    @ApiOperation(value = "Map 반환 API", notes = "Map을 반환하는 API")
    public Map<String, Object> returnMapApi() {
        Map<String, Object> map = new HashMap<>();
        map.put("test1", 1);
        map.put("test2", 2);

        return map;
    }

    @GetMapping("/api/sample/cmm-code")
    @ApiOperation(value = "공통 코드 반환", notes = "공통 코드 전체 리스트 반환")
    public List<SampleApiModel> getAllCommonCode(@ApiParam(value = "사용 여부", required = false, example = "Y") @RequestParam String useYn) {

        return sampleApiService.getAll(null);
    }
}

```

- `@Api(value = "", tags = "")` : 해당 클래스가 Swagger 리소스라는 것을 명시한다.
 - `value` : 태그를 작성한다.
 - `tags` : tags를 사용하여 여러 개의 태그를 정의한다.
- `@ApiOperation(value = "", notes = "")` : 한 개의 operation(즉 API URL과 Method)을 선언한다.
 - `value` : API에 대한 간략한 설명(제목 같은 느낌으로)으로 작성한다.
 - `notes` : 자세한 설명을 작성한다.
- `@ApiResponse(code = number, message = "")` : operation의 가능한 response 명시
 - `code` : 응답 코드를 작성한다.
 - `message` : 응답에 대한 설명을 작성한다.
 - `responseHeaders` : 헤더를 추가한다.(Optional)
- `@ApiResponses` : 여러 `ApiResponse` 객체 리스트를 허용하는 래퍼
- `@ApiParam(value = "", required = boolean, example = "")` : 파라미터에 대한 정보 명시
 - `value` : 파라미터 정보를 작성한다.
 - `required` : 필수 파라미터이면 `true`, 아니면 `false`를 작성한다.
 - `example` : 테스트를 할 때, 보여줄 예시를 작성한다.
- `@ApiImplicitParam(name = "", value = "", required = true/false, dataType = "", paramType =`

"" , defaultValue = "") : Request Parameter 설명

- @ApiIgnore : @ApiImplicitParam으로 선언하지 않은 parameter 정보들을 swagger UI에서 제외.

접근 방법

- 웹 브라우저의 주소창에 <http://localhost:8080/swagger-ui.html> 입력하여 접속

기타

기타 annotation

- @Authorization: 리소스나 오퍼레이션에 사용되는 권한 부여 체계를 선언
- @AuthorizationScope: OAuth2 인증 범위를 설명
- @ResponseHeader: 응답의 일부로 제공될 수 있는 헤더를 나타낸다
- @SwaggerDefinition: 생성된 스웨거 정의에 추가할 정의 레벨 속성
- @Info: 스웨거 정의의 일반 메타 데이터
- @Contact: 스웨거 정의를 위해 연락할 사람을 설명하는 속성
- @License: 스웨거 정의의 라이선스를 설명하는 속성

official swagger site

- main : <https://swagger.io>
- swagger2 document : <https://swagger.io/docs/specification/2-0/basic-structure/>

a

- [Spring Boot Swagger 3.x 적용](#)
- [java-swagger-ui-3.0-사용하기](#)

From:
<https://theta5912.net/> - reth

Permanent link:
<https://theta5912.net/doku.php?id=public:computer:swagger>

Last update: 2022/07/11 14:19

