Atitit 接口文档法 swagger法 v2 t88

目录

[1. Javadoc法 1](#_Toc26405)

[2. (9+条消息)swagger2常用注解说明 - 兴国-为梦想而战 - CSDN博客.html 1](#_Toc14931)

[2.1. 首先需要在pom.xml进行增加我们需要Swagger2所需要的依赖。 1](#_Toc10331)

[2.2. 接下来需要在 Application中进行声明@EnableSwagger2如图1.1 2](#_Toc13179)

[2.3. 声明配置内容的配置 如图1.2  swagger cfg file 2](#_Toc11936)

[2.4. 进行代码配置 3](#_Toc21269)

[2.5. 参数注解 4](#_Toc8635)

[2.6. 运行api ui 5](#_Toc18601)

[3. 参数注解 5](#_Toc1644)

[3.1. 方法内部使用@ApiParam 5](#_Toc13294)

[3.2. Req参数等外不参数使用@ApiImplicitParam，表示单独的请求参数，比较通用的参数比如uid等。。。 5](#_Toc15291)

[3.3. Atitit 参数传递的几种模式 ApiParam RequestParam PathVariable 7](#_Toc25928)

[4. 问题：：显示不出来方法 7](#_Toc4083)

[5. ref 8](#_Toc7022)

# Javadoc法

# (9+条消息)swagger2常用注解说明 - 兴国-为梦想而战 - CSDN博客.html

## 首先需要在pom.xml进行增加我们需要Swagger2所需要的依赖。

<!-- Swagger 接口文档 -->

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger2</artifactId>

<version>2.9.2</version>

</dependency>

<dependency>

<groupId>io.springfox</groupId>

<artifactId>springfox-swagger-ui</artifactId>

<version>2.9.2</version>

</dependency>

---------------------

## 接下来需要在 Application中进行声明@EnableSwagger2如图1.1

图1.1

图1.2

代码如下:

8

9

10

11

12

13

14

15

16

17

18

19

20

## 声明配置内容的配置 如图1.2  swagger cfg file

@Configuration

public class Swagger {

@Bean

public Docket controllerApi() {

return new Docket(DocumentationType.SWAGGER\_2)

.apiInfo(new ApiInfoBuilder()

.title("xx公司\_xx项目\_接口文档")

.description("描述内容")

// .contact("候帅")

.version("2.0.0")

.build())

.select()

.apis(RequestHandlerSelectors.basePackage("controller所在的包"))

.paths(PathSelectors.any())

.build();

}

}

1

2

3

4

5

6

7

## 进行代码配置

@Api("业务流程操作api")  
@Controller  
@Component

必须配置为@Controller 才能显示出来。。否则only @Component cant show

方法上的注解@ApiOperation(value = "执行业务流程操作 通过接口", notes = "说明note")

## 参数注解

方法内部使用@ApiParam

String startProcess(@ApiParam(name = "process\_name"

Req参数等外不参数使用@ApiImplicitParam，表示单独的请求参数，比较通用的参数比如uid等。。。

@ApiOperation(value = "执行业务流程操作 通过接口", notes = "说明note")  
@RequestMapping(value = "/startProcess")  
@ResponseBody  
@ApiParam(name = "tkid", value = "用户标识id", required = *true*)  
  
@ApiImplicitParams({  
 @ApiImplicitParam(name = "tkid", value = "用户标识的id", dataType = "string", paramType = "query", required=*true*),  
 @ApiImplicitParam(name = "uname", value = "用户名", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "Descript ", value = "描述", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "type ", value = "类型", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "ProblemReporter ", value = "问题上报人", dataType = "string", paramType = "query"),

## 运行api ui

在编写完后，我们进行运行我们的项目后可以访问[http://localhost:8088/swagger-ui.html](http://localhost:8080/swagger-ui.html) 进行访问Swagger2的相关内容

# 参数注解

## 方法内部使用@ApiParam

String startProcess(@ApiParam(name = "process\_name"

## Req参数等外不参数使用@ApiImplicitParam，表示单独的请求参数，比较通用的参数比如uid等。。。

@ApiOperation(value = "执行业务流程操作 通过接口", notes = "说明note")  
@RequestMapping(value = "/startProcess")  
@ResponseBody  
@ApiParam(name = "tkid", value = "用户标识id", required = *true*)  
  
@ApiImplicitParams({  
 @ApiImplicitParam(name = "tkid", value = "用户标识的id", dataType = "string", paramType = "query", required=*true*),  
 @ApiImplicitParam(name = "uname", value = "用户名", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "Descript ", value = "描述", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "type ", value = "类型", dataType = "string", paramType = "query"),  
 @ApiImplicitParam(name = "ProblemReporter ", value = "问题上报人", dataType = "string", paramType = "query"),

Atitit swagger 参数类型 ApiImplicitParams

@ApiImplicitParams({

@ApiImplicitParam(name = "@name", value = "脚本名称", dataType = "string", paramType = "query"),

})

paramType = "query" header

## Atitit 参数传递的几种模式 ApiParam RequestParam PathVariable

目录

1.1. ApiImplicitParam Req模式 1

2. @ApiParam， body json模式 1

3. @RequestParam,是获取前端传递给后端的参数，可以是get方式，也可以是post方式。 1

4. @PathVariable，是获取get方式，url后面参数，进行参数绑定 2

RequestParam没有descpt 功能

\* 从报错信息上来看name应该是value的别名，它们两个中只有一个能被允许存在。name其实就是我们在前台页面的表单中input中的name，用来解决前后台传入参数不一致的问题。比如前台通过ajax传入后台的参数名是phone，然而后台想接受的phoneNum并且后台代码都已经写好了（全部改的话比较麻烦），那么我们就可以设置name="phone"，后台代码不需要改动。不过，name的功能是和alue一样的，name完全等价于value，使用哪一个，只是个人习惯问题（切记不可同时使用，推荐使用value）

\*/

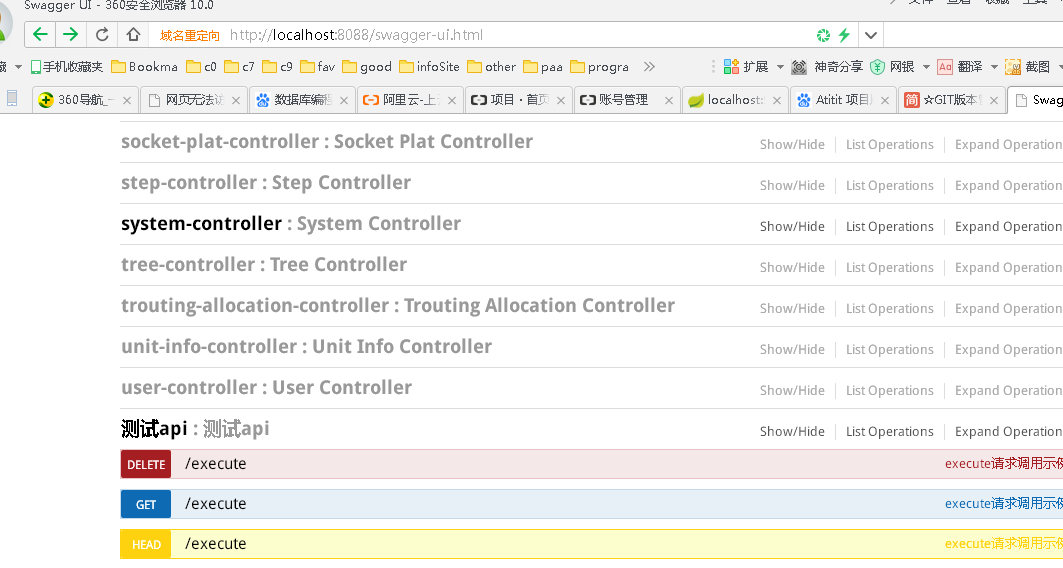
@GetMapping("/queryDataStruts")

@ApiOperation(value = "查询数据源表 字段结构 query table struts", notes = "翻页查询的说明note")

**public** Object queryDataStruts(@RequestParam( name="from",defaultValue = "sys\_area\_t")String from) **throws** Exception {

# 问题：：显示不出来方法

必须要配置@api在类上，才能显示出类来然后是方法



## 参数req query以及desc模式

ApiParam是body模式

RequestParam没有desc，而且是spriboot的anno

可以使用ApiImplicitParam ，这是swagger的接口anno

# ref

(9+条消息)Swagger2--Springboot使用 - 细心程度决定你的成败 - CSDN博客.html