一、功能说明

该工具包在springfox-swagger2基础上做了简单封装提升文档的配置化能力。

背景

- 作为敏捷团队后端开发的每一份子, 可能都没有足够时间对所负责的接口进行完整的文档输出;
- 为了节省文档输出的时间成本而又不影响前后端对接工作, 特对swagger在线文档进行规范要求。

二、工具使用

- 添加依赖: pom.xml

```
<dependency>
    <groupId>cn.gmlee</groupId>
    <artifactId>tools-swagger</artifactId>
    <version>3.1.0</version>
</dependency>
```

- 启用工具

> 在启动类上或其他配置类上扫描包: cn.gmlee.tools

```
@SpringBootApplication
@ComponentScan("cn.gmlee.tools")
public class XxxApp {
    public static void main(String[] args) {
        SpringApplication.run(XxxApp.class, args);
    }
}
```

- 环境排除

> 由于近期安全组多次提示stg环境有接口泄露的相关警告, 固有此篇幅, 有同样担心的同学可以继续阅读。

> 默认开启的环境有: local, loc, dev, test, stg, 假设在stg环境不需要开启swagger, 那么请在`application-stg.properties`中添加以下配置

```
# 在当前环境关闭swagger文档
spring.autoconfigure.exclude=cn.gmlee.tools.swagger.config.SwaggerAutoConfiguration
```

- 分组配置

> 当多人同时开发或同时进行多个模块的开发时, 可对接口进行分组交付

```
# swagger配置
tools.webapp.swagger.groups=a.档案组:/archives*/**,b.标签组:/tag*/**,c.报表组:/report*/**,d.检索组:/search*/**,e.默认组:/**
```

- 全局配置

> 全局配置目前支持: 全局请求参数等

```
# 启用参数配置集合: 该配置仅作参考,更改配置需要全量配置
tools.webapp.swagger.global.parametersIndex=0,1
#
================================================================================
tools.webapp.swagger.global.parameters[0].name=token
tools.webapp.swagger.global.parameters[0].description=登陆令牌
tools.webapp.swagger.global.parameters[0].defaultValue=
tools.webapp.swagger.global.parameters[0].required=false
tools.webapp.swagger.global.parameters[0].paramType=header
tools.webapp.swagger.global.parameters[0].javaType=string
tools.webapp.swagger.global.parameters[0].hidden=false
tools.webapp.swagger.global.parameters[1].name=version
tools.webapp.swagger.global.parameters[1].description=版本号
tools.webapp.swagger.global.parameters[1].defaultValue=1.0.0
tools.webapp.swagger.global.parameters[1].required=false
tools.webapp.swagger.global.parameters[1].paramType=header
tools.webapp.swagger.global.parameters[1].javaType=string
tools.webapp.swagger.global.parameters[1].hidden=false
#
================================================================================
tools.webapp.swagger.global.parameters[2].name=appId
tools.webapp.swagger.global.parameters[2].description=应用编号
tools.webapp.swagger.global.parameters[2].defaultValue=
tools.webapp.swagger.global.parameters[2].required=false
tools.webapp.swagger.global.parameters[2].paramType=header
tools.webapp.swagger.global.parameters[2].javaType=string
tools.webapp.swagger.global.parameters[2].hidden=false
tools.webapp.swagger.global.parameters[3].name=timestamp
tools.webapp.swagger.global.parameters[3].description=时间戳
tools.webapp.swagger.global.parameters[3].defaultValue=
tools.webapp.swagger.global.parameters[3].required=false
tools.webapp.swagger.global.parameters[3].paramType=header
tools.webapp.swagger.global.parameters[3].javaType=long
tools.webapp.swagger.global.parameters[3].hidden=false
tools.webapp.swagger.global.parameters[4].name=nonce
tools.webapp.swagger.global.parameters[4].description=随机码
tools.webapp.swagger.global.parameters[4].defaultValue=
tools.webapp.swagger.global.parameters[4].required=false
tools.webapp.swagger.global.parameters[4].paramType=header
tools.webapp.swagger.global.parameters[4].javaType=string
tools.webapp.swagger.global.parameters[4].hidden=false
tools.webapp.swagger.global.parameters[5].name=signature
tools.webapp.swagger.global.parameters[5].description=签名
tools.webapp.swagger.global.parameters[5].defaultValue=
tools.webapp.swagger.global.parameters[5].required=false
tools.webapp.swagger.global.parameters[5].paramType=header
tools.webapp.swagger.global.parameters[5].javaType=string
tools.webapp.swagger.global.parameters[5].hidden=false
#
================================================================================
```

- 其他配置

```
# 配置接口前缀：当文档隐藏在多层网关之后需要用到该前缀（解决接口请求地址不完整问题）
tools.webapp.swagger.prefix=/api/user-archives/
# 配置忽略的参数集：忽略后该类不会显示在文档中
tools.webapp.swagger.ignoredParameterTypes=com.hll.base.bean.Result
```

三、工具原理

逻辑

- 通过利用SpringMvc以及Swagger自制的注解对接口进行描述
- 获取描述信息并存储为json串
- 根据json串生成swagger在线接口文档

代码

1. 自动装配及核心代码: SwaggerAutoConfiguration
2. 配置装配代码: SwaggerGlobalProperties
3. 静态资源映射代码: SwaggerWebMvcAutoConfiguration

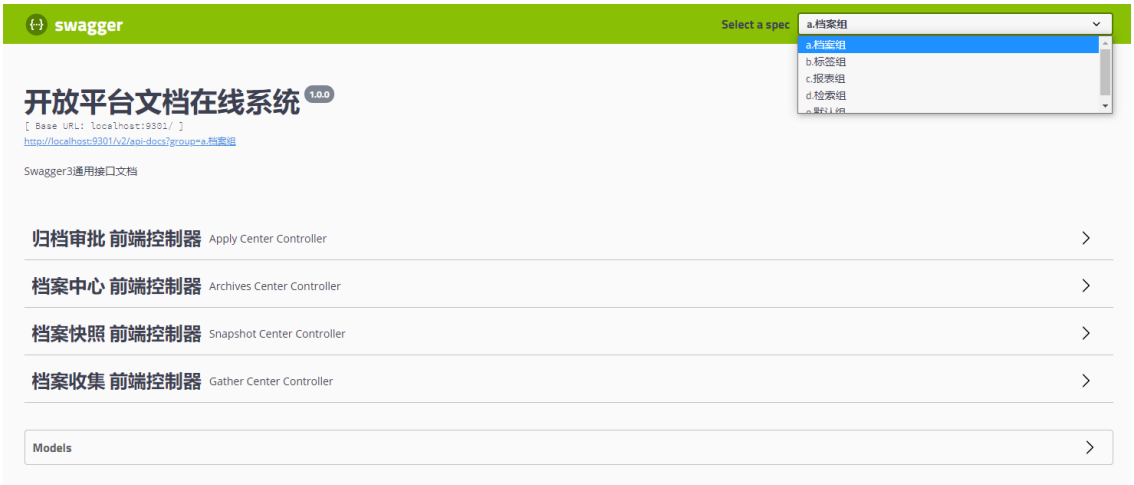
数据

- 存储在运行内存中

四、使用示例

- application.properties

```
# swagger配置
tools.webapp.swagger.groups=a.档案组:/archives*/**,b.标签组:/tag*/**,c.报表组:/report*/**,d.检索组:/search*/**,e.默认组:/**
```



五、附录

5.1 接口规范

5.1.1 数据格式

- 原则上一律以application/json请求及响应

5.1.2 方法声明

- 原则上一律以@PostMapper声明增删改接口
- 另@GetMapper可以使用@PostMapper代替

5.1.3 模型修饰

- 传输到前端或接收前端所封装的对象类名以Vo结尾 (以下称为: 模型)
- 每个模型必须使用@ApiModel注解并填充value, description等内容
- 每个模型须实现java.io.Serializable接口以支持序列化与反序列化
- 每个模型的每个字段必须使用@ApiModelProperty注解填充value

5.1.4 单一职责

- 一个模型一个接口: 不可多个接口使用同一个模型接收数据
- 展示到接口文档中的字段必须是对接方(前端)所 **需要的** 字段

> 前端不需要的字段不要展示, 若后端需要保留请使用@ApiModelProperty的hidden=true隐藏

5.1.5 数据效验

- 每个接口接收的参数如果需要对参数内容进行效验请采用JSR 303规范实现

> JSR-303 是JAVA EE 6 中的一项子规范，叫做Bean Validation，Hibernate Validator 是 Bean Validation 的参考实现 .
>
> Hibernate Validator 提供了 JSR 303 规范中所有内置 constraint 的实现，除此之外还有一些附加的 constraint。

Constraint	详细信息
`@Null`	被注释的元素必须为 `null`
`@NotNull`	被注释的元素必须不为 `null`
`@AssertTrue`	被注释的元素必须为 `true`
`@AssertFalse`	被注释的元素必须为 `false`
`@Min(value)`	被注释的元素必须是一个数字，其值必须大于等于指定的最小值
`@Max(value)`	被注释的元素必须是一个数字，其值必须小于等于指定的最大值
`@DecimalMin(value)`	被注释的元素必须是一个数字，其值必须大于等于指定的最小值
`@DecimalMax(value)`	被注释的元素必须是一个数字，其值必须小于等于指定的最大值
`@Size(max, min)`	被注释的元素的大小必须在指定的范围内
`@Digits (integer, fraction)`	被注释的元素必须是一个数字，其值必须在可接受的范围内
`@Past`	被注释的元素必须是一个过去的日期
`@Future`	被注释的元素必须是一个将来的日期
`@Pattern(value)`	被注释的元素必须符合指定的正则表达式