# Swagger

## 1.介绍

在项目前后端分离的背景下，如何保证前后端人员能够正常的联调接口呢？Swagger提供了

一种便利的方式，它可以在后端人员修改完接口以后，自动生成一个接口文档且实时更

新，这样，前端人员就可以使用实时更新的文档来使用接口。

优点：非常流行的Api框架；RestFul Api格式文档实时更新；支持多种语言(java、php等)。

官网：<https://swagger.io/>

需要的依赖：swagger2和ui

## 2.springboot集成swagger

(1)新建一个springboot项目。

(2)在pom文件中添加两个依赖。

<!--swagger依赖-->  
<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>

(3)配置swagger开启

方式：创建一个springboot配置类，然后加上注解@EnableSwagger2。例如：

@Configuration  
@EnableSwagger2  
public class SwaggerConfig {  
  
}

以上只是开启了swagger，使用的是默认的配置，详细配置请看下面。

(4)创建一个controller类并写一个方法

@RequestMapping("hello")  
public String hello(){  
 return "hello";  
}

该方法用来测试swagger的默认文档显示。

(5)查看文档显示页面

首先启动项目，然后访问 项目域名/swagger-ui.html。



(6)springboot和swagger存在版本兼容问题

当出现兼容问题时，项目启动会报：Failed to start bean 'documentationPluginsBootstrapper';

原因：Springfox 使用的路径匹配是基于AntPathMatcher的，而Spring Boot 2.6.X使用的是

PathPatternMatcher。

解决方式：使用版本兼容的组合。

比如：springboot2.5.6和swagger2.9.2、springboot2.6.5和swagger3.0.0。

## 3.自定义配置Swagger

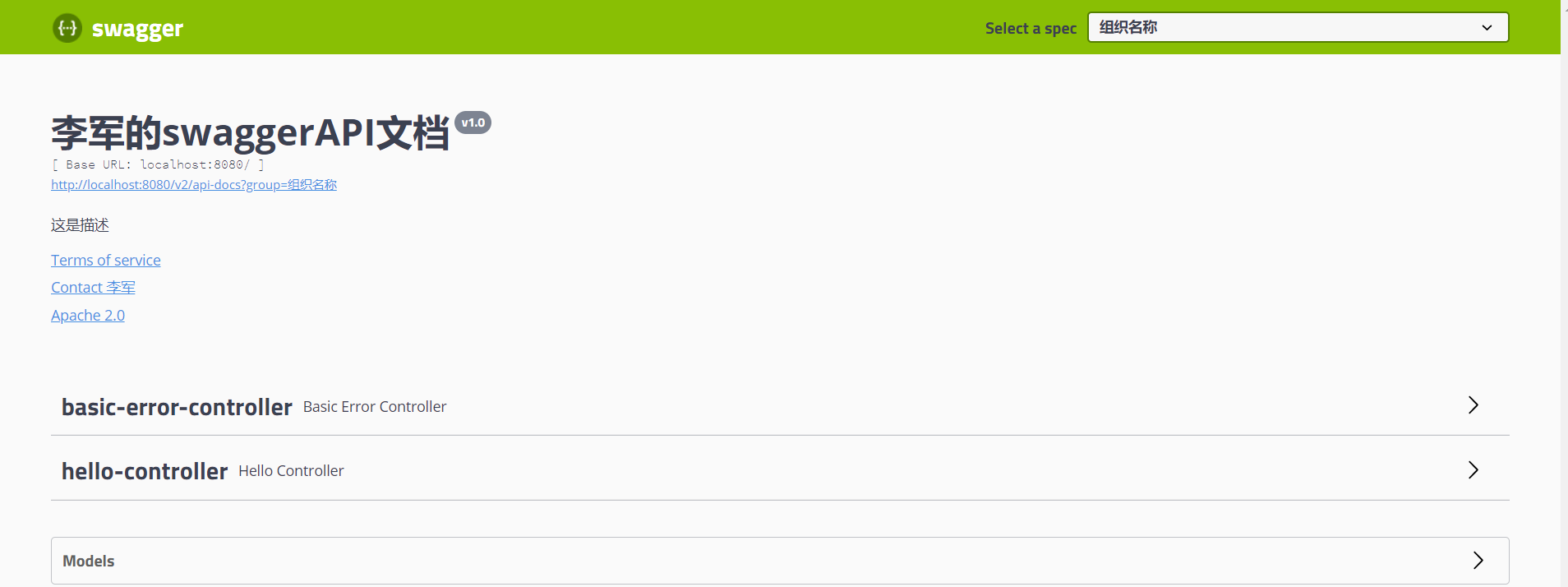
在springboot集成swagger中，已经可以使用swagger了，但是使用的是swagger的默认配

置，那么如何使用swagger的自定义配置呢？

在配置类中加上一些配置信息即可。实例代码：

@Configuration  
@EnableSwagger2  
public class SwaggerConfig {  
  
 @Bean  
 public Docket getDocket(){  
 return new Docket(DocumentationType.*SWAGGER\_2*)  
 .apiInfo(getApiInfo())  
 .groupName("组织名称");  
 }  
  
 private ApiInfo getApiInfo(){  
 //作者信息  
 Contact contact = new Contact("李军","","2369951991@qq.com");  
 return new ApiInfo("李军的swaggerAPI文档", "这是描述", "v1.0",  
 "http://localhost:8080", contact, "Apache 2.0",  
 "http://www.apache.org/licenses/LICENSE-2.0", new ArrayList());  
 }  
  
}

在代码中可以看到，主要是将文档的类型(比如：DocumentationType.SWAGGER\_2)以及文档的基础信息(比如：ApiInfo对象)封装到Docket对象来实现swagger的自定义配置。下面是自定义之后，文档页面的展示：



其他的一些配置可以查看源码来配。

## 4.其他配置

@Bean  
public Docket getDocket(Environment environment){  
 return new Docket(DocumentationType.*SWAGGER\_2*)  
 .apiInfo(getApiInfo())  
 .groupName("组织名称")  
 //flag为false时表示不启用Swagger，默认是启用的  
 .enable(false)  
 .select()  
 //指定扫描哪个包下的Swagger接口  
// .apis(RequestHandlerSelectors.basePackage("com.swagger"))  
 //在任何类上都扫描Swagger接口  
// .apis(RequestHandlerSelectors.any())  
 //在任何类上都不扫描Swagger接口  
// .apis(RequestHandlerSelectors.none())  
 //扫描前缀是/hello请求下的Swagger接口  
 .paths(PathSelectors.*ant*("/hello/\*\*"))  
 .build();  
}

在代码中可以看到，新增了俩配置，一个是是否开启Swagger；一个是如何扫描Swagger接口。

## 5.面试题

(1)如何让项目在测试环境开启Swagger，在生产环境关闭Swagger？

答：可以设置两个子配置文件，application-dev.properties和application-pro.properties。第一

个是测试环境走的配置文件，第二个是生产环境走的配置文件。在主配置文件

application.properties中设置spring.profiles.active=dev时，走application-dev.properties，

此时根据boolean flag = environment.acceptsProfiles(Profiles.of("dev"))来判断项目是否处

于dev环境中，再根据flag的值设置Swagger是否启用。代码实例：

@Bean  
public Docket getDocket(Environment environment){  
 //预设是dev环境  
 Profiles profiles = Profiles.*of*("dev");  
 //判断项目是否处在profiles表示的环境中，是的话，flag就是true，反之就是false  
 boolean flag = environment.acceptsProfiles(profiles);  
 return new Docket(DocumentationType.*SWAGGER\_2*)  
 .apiInfo(getApiInfo())  
 .groupName("组织名称")  
 //flag为false时表示不启用Swagger，默认是启用的  
 .enable(flag);  
}

(2)如何配置多个分组？

答：使用@Bean多注入几个Docket对象即可。代码实例：

@Bean  
public Docket getDocket2(Environment environment){  
 return new Docket(DocumentationType.*SWAGGER\_2*).groupName("第二分组");  
}  
@Bean  
public Docket getDocket3(Environment environment){  
 return new Docket(DocumentationType.*SWAGGER\_2*).groupName("第三分组");  
}

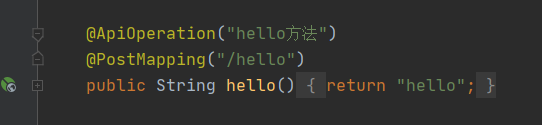
## 6.Swagger的注解

(1) @ApiModel修饰类、接口、注解、枚举，说明其含义，一般用于实体类。

(2) @ApiModelProperty修饰方法、属性，说明其含义，一般用于实体类上的属性。



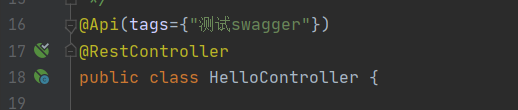
(3) @ApiOperation修饰类、接口、注解、枚举、方法，说明其含义，用在方法上才有效果。



(4) @ApiParam一般用在方法的形参上。



(5) @Api一般用在类上，它的属性tags的值是一个String数组，用来写描述信息的。



## 7.注意事项

在生产环境中一定不能开启Swagger，否则用户可以通过 项目域名/swagger-ui.html来访问接口文档；其次关闭Swagger也可以节省内存，加快项目的启动速度。