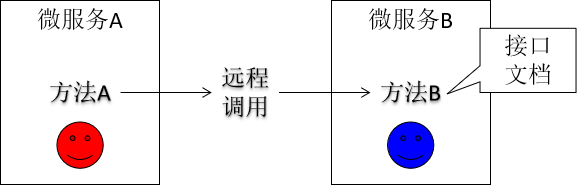
swagger

接口文档生成组件。



## 加入依赖

|  |
| --- |
| <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> |

## 创建Swagger配置类

|  |
| --- |
| @Configuration  @EnableSwagger2  public class AppSwaggerConfig {  @Value("${swagger2.enable:true}")  private boolean enable;  @Bean("用户模块")  public Docket userApis() {  return new Docket(DocumentationType.SWAGGER\_2).groupName("用户模块").select()  .apis(RequestHandlerSelectors.withClassAnnotation(Api.class)).paths(PathSelectors.regex("/user.\*"))  .build().apiInfo(apiInfo()).enable(enable);  }  @Bean("项目模块")  public Docket projectApis() {  return new Docket(DocumentationType.SWAGGER\_2).groupName("项目模块").select()  .apis(RequestHandlerSelectors.withClassAnnotation(Api.class)).paths(PathSelectors.regex("/project.\*"))  .build().apiInfo(apiInfo()).enable(enable);  }  @Bean("审核模块")  public Docket authApis() {  return new Docket(DocumentationType.SWAGGER\_2).groupName("审核模块").select()  .apis(RequestHandlerSelectors.withClassAnnotation(Api.class)).paths(PathSelectors.regex("/auth.\*"))  .build().apiInfo(apiInfo()).enable(enable);  }  @Bean("订单/支付模块")  public Docket payApis() {  return new Docket(DocumentationType.SWAGGER\_2).groupName("订单/支付模块").select()  .apis(RequestHandlerSelectors.withClassAnnotation(Api.class)).paths(PathSelectors.regex("/order.\*"))  .build().apiInfo(apiInfo()).enable(enable);  }  private ApiInfo apiInfo() {  return new ApiInfoBuilder().title("尚筹网-系统平台接口文档").description("提供用户模块/审核模块/项目模块/支付模块的文档")  .termsOfServiceUrl("http://www.atguigu.com/").version("1.0").build();  }  } |

## 在需要生成接口文档的方法上使用注解

### @Api(tags="")

用在请求的类上，表示对类的说明

tags属性：说明类的作用，可以在UI界面上看到

### @ApiOperation(value="")

用在请求的方法上，说明方法的用途、作用

httpMethod属性：指定当前方法访问时使用的请求方式。注意：如果没有设置这个属性，swagger为当前方法的每一个HTTP请求方式都生成对应的文档条目。如果没有极特殊情况，那么这个属性一定要设置。

### @ApiImplicitParams

用在请求的方法上，对参数进行说明

### @ApiImplicitParam

说明请求参数的各个方面。

|  |
| --- |
| @ApiImplicitParams(  {  @ApiImplicitParam(name="username",value="用户名",required=true),  @ApiImplicitParam(name="password",value="密码",required=true)  }  )  @PostMapping("/login")  public AppResponse<UserRespVo> login(String username,String password){  //...  } |

value：参数含义说明

required：是否必须

paramType：参数位置

header –> 请求头的获取：@RequestHeader

query –> 请求参数的获取：@RequestParam

path（用于restful接口）–> 请求路径变量的获取：@PathVariable

body（不常用）

form（不常用）

dataType：参数类型，默认String，其它值dataType="Integer"

defaultValue：参数的默认值

### @ApiResponses

用在请求的方法上，表示一组响应

### @ApiResponse

用在@ApiResponses中，一般用于表达一个错误的响应信息

code：数字，例如400

message：信息，例如"请求参数没填好"

response：抛出异常的类

### @ApiModel

用途1：说明响应数据类型。

具体指使用@ResponseBody修饰方法后，返回JSON数据格式。

用途2：说明入参数据类型

具体指使用@RequestBody修饰入参后，把JSON转换为入参类型，此时无法使用@ApiImplicitParam注解进行描述

### @ApiModelProperty

描述模型属性

## guava版本问题

java.lang.NoSuchMethodError: com.google.common.collect.FluentIterable.concat

原因：guava.18.0.jar中FluentIterable类中没有concat方法。

解决：使用版本更高guava的jar包。

具体做法：

找到distribution-crowd-1-parent

在dependencyManagement中加入

|  |
| --- |
| <dependency>  <groupId>com.google.guava</groupId>  <artifactId>guava</artifactId>  </dependency> |

这样项目中使用的guava版本就是20.0版本。

## 打开swagger界面

将带有swagger注解的微服务启动起来，访问swagger-ui.html

