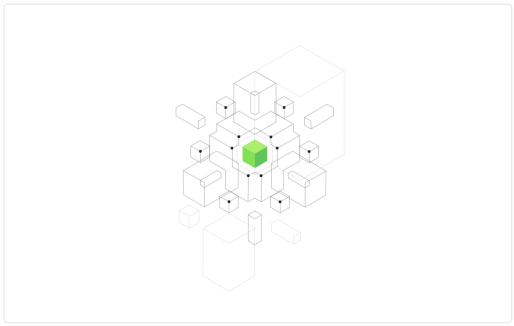
Swagger与Kenife4j-接口文档管理

标签　　空

新增属性

Swagger官网地址：https://swagger.io/

Kenife4j官网地址：https://doc.xiaominfo.com/



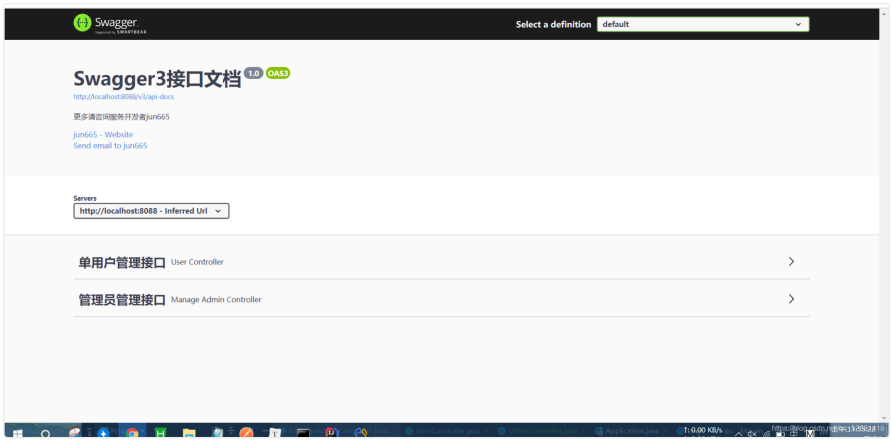
文档基于swagger3.

在前后端分离开发的项目中，后端开发者将接口开发完成后，需要将接口信息提供给前端开发者使用，因此需要有一个接口信息的文档。

如果接口数量庞大，手动去编写文档完全是费时费力的，因此可以在项目中引用一些工具来自动生成接口文档。

Swagger-UI

页面效果如下：



官网地址：https://swagger.io/

maven 依赖

XML

```xml
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-boot-starter</artifactId>
  <version>3.0.0</version>
</dependency>
```

配置及启动

Java

```java
@EnableOpenApi //开启Swagger3，也可以在启动类上
@Configuration
public class Swagger3Config {
    // 可以定义多个bean来实现分组
    @Bean
    public Docket createRestApi() {
        return new Docket(DocumentationType.OAS_30)
                .apiInfo(apiInfo())
                // 指定发布的接口
                .select()
                .apis(RequestHandlerSelectors.withMethodAnnotation(ApiOperation.class))
                .paths(PathSelectors.any())
                .build();
    }

    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
                .title("标题")
                .description("描述")
                .contact(new Contact("作者", "#", "作者邮箱"))
                .version("1.0")
                .build();
    }
}
```

发布接口可指定的几种方式:

Java

```java
// 将所有接口发布
RequestHandlerSelectors.any()
// 将类上含指定注解的接口发布
RequestHandlerSelectors.withClassAnnotation(Api.class)
// 将方法上含指定注解的接口发布
RequestHandlerSelectors.withMethodAnnotation(ApiOperation.class)
// 将指定包的接口发布
RequestHandlerSelectors.basePackage("com.demo.www")
```

标记接口

Java

```java
@Api(tags = "管理员管理接口")
@RestController
public class ManageAdminController {

    // 接口描述
    @ApiOperation("用户修改接口")
    // 请求参数
    @ApiImplicitParams({
            @ApiImplicitParam(name = "a", value = "参数a"),
            @ApiImplicitParam(name = "b", value = "参数b"),
            @ApiImplicitParam(name = "c", value = "参数c"),
            @ApiImplicitParam(name = "d", value = "d为必选参数 ", required = true),
    })
    // 响应参数
    @ApiResponses({
            @ApiResponse(code = 200, message = "操作成功状态码"),
            @ApiResponse(msg, message = "响应信息")
    })
    //修改用户
    @PostMapping("/test")
    public Map test(@RequestParam Map<String, Object> remap) {
      Map map=new HashMap();
      map.put("status",200);
      map.put("msg","操作成功!");
      return map;
    }
}
```

如果请求参数或响应参数是实体类对象的形式，也可以直接在实体类属性添加注解：

请求：

Java

```java
@Data
public class ValidateSignDTO implements Serializable {

    @ApiModelProperty(value = "昵称 ",required=true)
```

```
    private String name;

    @ApiModelProperty(value = "密码",required=true)
    private String password;

    @ApiModelProperty(value = "邮箱",required=true)
    private String email;
}
```

响应：

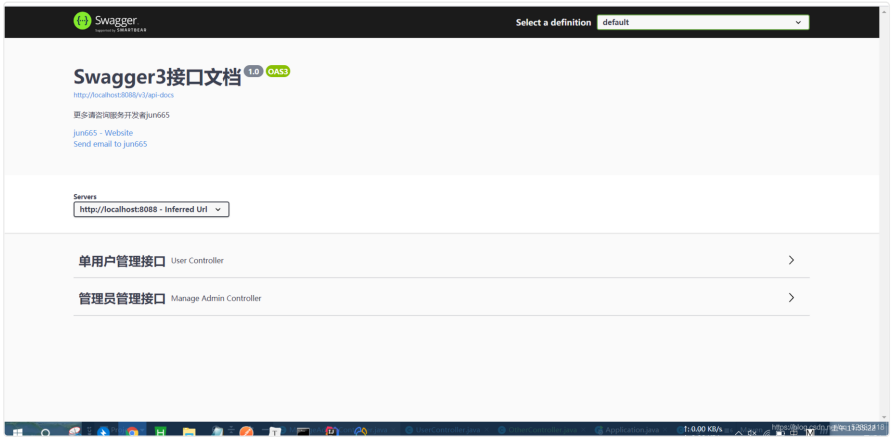
Java

```
@Getter
public class CommonResult<T>{

    @ApiModelProperty(value = "状态码" )
    private long returnCode;

    @ApiModelProperty(value = "提示信息")
    private String returnMessage;

    @ApiModelProperty(value = "响应数据")
    private T bean;
}
```

注意：实体类对象必须含有 getter 方法。

访问地址

启动后端项目，并访问 http://localhost:8088/swagger-ui/ 即可。



Knife4j

Knife4j 又名 swagger-bootstrap-ui，是 swagger 的增强方案，它拥有更加出色的 ui 界面并且支持文档离线为 markdown。

页面效果如下：

官方文档地址：https://xiaoym.gitee.io/knife4j/

基于swagger2.x：https://blog.csdn.net/qq_52681418/article/details/112602084

maven 依赖

基于openAPI2版本使用2.x，基于openAPI3版本使用3.x。

XML

```xml
<dependency>
    <groupId>com.github.xiaoymin</groupId>
    <artifactId>knife4j-spring-boot-starter</artifactId>
    <version>3.0.3</version>
</dependency>

```

配置、注解基本上与swagger一致。

启动

在swagger的基础上添加以下注解启动即可：

Java

```java
@EnableKnife4j
```

访问地址

启动后端项目，并访问 http://localhost:8088doc.html 即可。

Swagger是一款强大的API文档生成器，并且支持API调试。