5.1 swagger介绍

相信无论是前端还是后端开发，都或多或少地被接口文档折磨过。前端经常抱怨后端给的接口文档与实际情况不一致。后端又觉得编写及维护接口文档会耗费不少精力，经常来不及更新。其实无论是前端调用后端，还是后端调用后端，都期望有一个好的接口文档。但是这个接口文档对于程序员来说，就跟注释一样，经常会抱怨别人写的代码没有写注释，然而自己写起代码起来，最讨厌的，也是写注释。所以仅仅只通过强制来规范大家是不够的，随着时间推移，版本迭代，接口文档往往很容易就跟不上代码了。

使用Swagger你只需要按照它的规范去定义接口及接口相关的信息。再通过Swagger衍生出来的一系列项目和工具，就可以做到生成各种格式的接口文档，生成多种语言的客户端和服务端的代码，以及在线接口调试页面等等。这样，如果按照新的开发模式，在开发新版本或者迭代版本的时候，只需要更新Swagger描述文件，就可以自动生成接口文档和客户端服务端代码，做到调用端代码、服务端代码以及接口文档的一致性。

为了简化swagger的使用，Spring框架对swagger进行了整合，建立了Spring-swagger项目，后面改成了现在的Springfox。通过在项目中引入Springfox，可以扫描相关的代码，生成描述文件，进而生成与代码一致的接口文档和客户端代码。

Springfox对应的maven坐标如下：

```
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>2.9.2</version>
</dependency>
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>2.9.2</version>
</dependency>
```

5.2 swagger常用注解

注解	说明
@Api	用在请求的类上，例如Controller，表示对类的说明
@ApiModel	用在类上，通常是实体类，表示一个返回响应数据的信息
@ApiModelProperty	用在属性上，描述响应类的属性
@ApiOperation	用在请求的方法上，说明方法的用途、作用
@ApiImplicitParams	用在请求的方法上，表示一组参数说明
@ApiImplicitParam	用在@ApiImplicitParams注解中，指定一个请求参数的各个方面

5.3 swagger入门案例

第一步：创建maven工程swagger_demo并配置pom.xml文件

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.2.2.RELEASE</version>
        <relativePath/>
    </parent>
    <groupId>cn.itcast</groupId>
    <artifactId>swagger_demo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>swagger_demo</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger-ui</artifactId>
            <version>2.9.2</version>
        </dependency>
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger2</artifactId>
            <version>2.9.2</version>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
        </dependency>
    </dependencies>
</project>
```

第二步：创建application.yml文件

```yaml
server:
  port: 9000
```

第三步： 创建实体类User和Menu

```
package cn.itcast.entity;

import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

@Data
@ApiModel(description = "用户实体")
public class User {
    @ApiModelProperty(value = "主键")
    private int id;
    @ApiModelProperty(value = "姓名")
    private String name;
    @ApiModelProperty(value = "年龄")
    private int age;
    @ApiModelProperty(value = "地址")
    private String address;
}
```

```
package cn.itcast.entity;

import io.swagger.annotations.ApiModel;
import io.swagger.annotations.ApiModelProperty;
import lombok.Data;

@Data
@ApiModel(description = "菜单实体")
public class Menu {
    @ApiModelProperty(value = "主键")
    private int id;
    @ApiModelProperty(value = "菜单名称")
    private String name;
}
```

第四步：创建UserController和MenuController

```java
package cn.itcast.controller.user;

import cn.itcast.entity.User;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiImplicitParam;
import io.swagger.annotations.ApiImplicitParams;
import io.swagger.annotations.ApiOperation;
import org.springframework.web.bind.annotation.*;
import java.util.ArrayList;
import java.util.List;

@RestController
@RequestMapping("/user")
@Api(tags = "用户控制器")
public class UserController {
    @GetMapping("/getUsers")
    @ApiOperation(value = "查询所有用户", notes = "查询所有用户信息")
    public List<User> getAllUsers(){
        User user = new User();
        user.setId(100);
        user.setName("itcast");
        user.setAge(20);
        user.setAddress("bj");
        List<User> list = new ArrayList<>();
        list.add(user);
        return list;
    }

    @PostMapping("/save")
    @ApiOperation(value = "新增用户", notes = "新增用户信息")
    public String save(@RequestBody User user){
        return "OK";
    }

    @PutMapping("/update")
    @ApiOperation(value = "修改用户", notes = "修改用户信息")
    public String update(@RequestBody User user){
        return "OK";
    }

    @DeleteMapping("/delete")
    @ApiOperation(value = "删除用户", notes = "删除用户信息")
    public String delete(int id){
        return "OK";
    }

    @ApiImplicitParams({
           @ApiImplicitParam(name = "pageNum", value = "页码",
                        required = true, type = "Integer"),
        @ApiImplicitParam(name = "pageSize", value = "每页条数",
```

```
                        required = true, type = "Integer"),
    })
    @ApiOperation(value = "分页查询用户信息")
    @GetMapping(value = "page/{pageNum}/{pageSize}")
    public String findByPage(@PathVariable Integer pageNum,
                             @PathVariable Integer pageSize) {
        return "OK";
    }
}
```

```java
package cn.itcast.controller.menu;

import cn.itcast.entity.Menu;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiImplicitParam;
import io.swagger.annotations.ApiImplicitParams;
import io.swagger.annotations.ApiOperation;
import org.springframework.web.bind.annotation.*;
import java.util.ArrayList;
import java.util.List;

@RestController
@RequestMapping("/menu")
@Api(tags = "菜单控制器")
public class MenuController {
    @GetMapping("/getMenus")
    @ApiOperation(value = "查询所有菜单", notes = "查询所有菜单信息")
    public List<Menu> getMenus(){
        Menu menu = new Menu();
        menu.setId(100);
        menu.setName("itcast");
        List<Menu> list = new ArrayList<>();
        list.add(menu);
        return list;
    }

    @PostMapping("/save")
    @ApiOperation(value = "新增菜单", notes = "新增菜单信息")
    public String save(@RequestBody Menu menu){
        return "OK";
    }

    @PutMapping("/update")
    @ApiOperation(value = "修改菜单", notes = "修改菜单信息")
    public String update(@RequestBody Menu menu){
        return "OK";
    }

    @DeleteMapping("/delete")
    @ApiOperation(value = "删除菜单", notes = "删除菜单信息")
    public String delete(int id){
        return "OK";
    }

    @ApiImplicitParams({
        @ApiImplicitParam(name = "pageNum", value = "页码",
                          required = true, type = "Integer"),
        @ApiImplicitParam(name = "pageSize", value = "每页条数",
                          required = true, type = "Integer"),
    })
```

```
    @ApiOperation(value = "分页查询菜单信息")
    @GetMapping(value = "page/{pageNum}/{pageSize}")
    public String findByPage(@PathVariable Integer pageNum,
                             @PathVariable Integer pageSize) {
        return "OK";
    }
}
```

第五步：创建配置类SwaggerAutoConfiguration

```java
package cn.itcast.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import springfox.documentation.builders.ApiInfoBuilder;
import springfox.documentation.builders.RequestHandlerSelectors;
import springfox.documentation.service.ApiInfo;
import springfox.documentation.service.Contact;
import springfox.documentation.spi.DocumentationType;
import springfox.documentation.spring.web.plugins.Docket;
import springfox.documentation.swagger2.annotations.EnableSwagger2;

@Configuration
@EnableSwagger2
public class SwaggerAutoConfiguration {
    @Bean
    public Docket createRestApi1() {
        Docket docket = new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(apiInfo()).groupName("用户接口组")
                .select()
                //为当前包路径

.apis(RequestHandlerSelectors.basePackage("cn.itcast.controller.user"))
                .build();
        return docket;
    }

    @Bean
    public Docket createRestApi2() {
        Docket docket = new Docket(DocumentationType.SWAGGER_2)
                .apiInfo(apiInfo()).groupName("菜单接口组")
                .select()
                //为当前包路径

.apis(RequestHandlerSelectors.basePackage("cn.itcast.controller.menu"))
                .build();
        return docket;
    }

    //构建 api文档的详细信息
    private ApiInfo apiInfo() {
        return new ApiInfoBuilder()
                //页面标题
                .title("API接口文档")
                //创建人
                .contact(new Contact("黑马程序员", "http://www.itheima.com", ""))
                //版本号
                .version("1.0")
                //描述
                .description("API 描述")
```

```
                .build();
    }
}
```

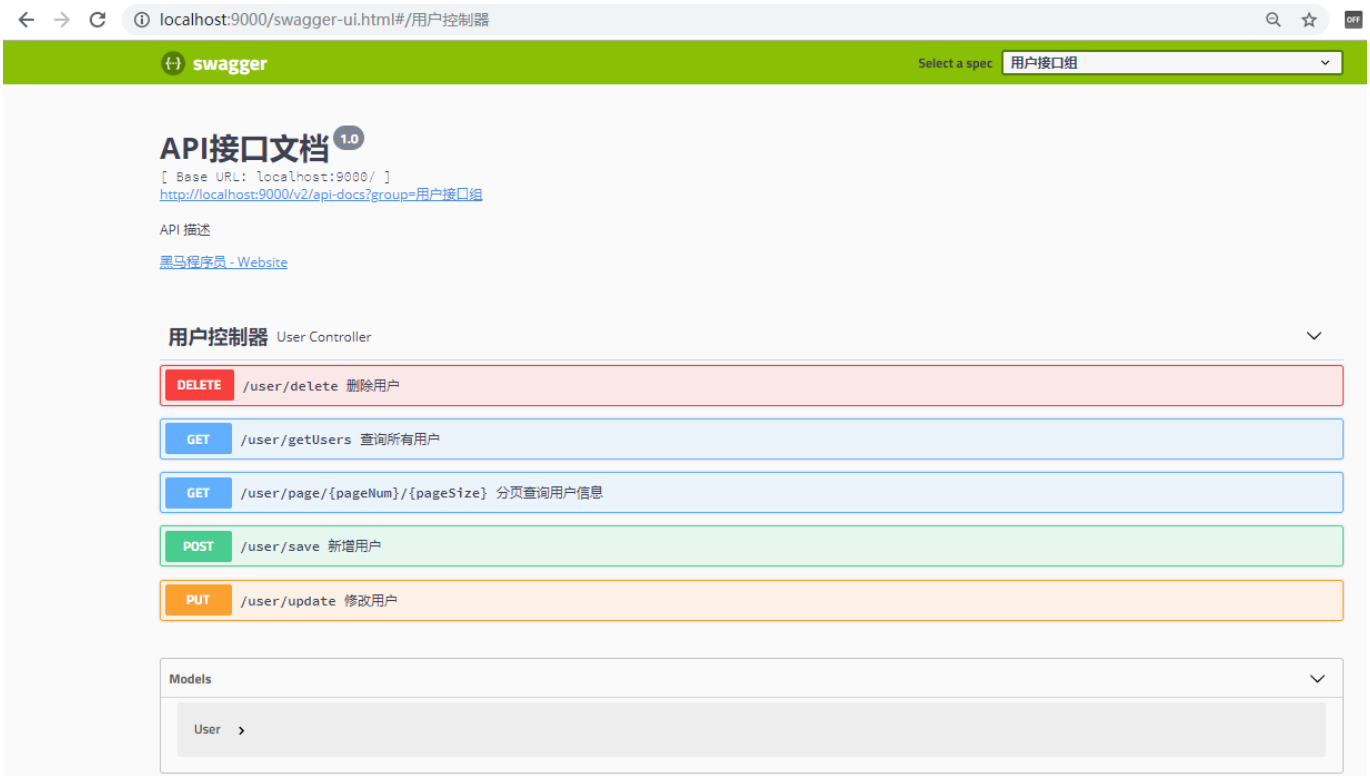
第六步：创建启动类SwaggerDemoApplication

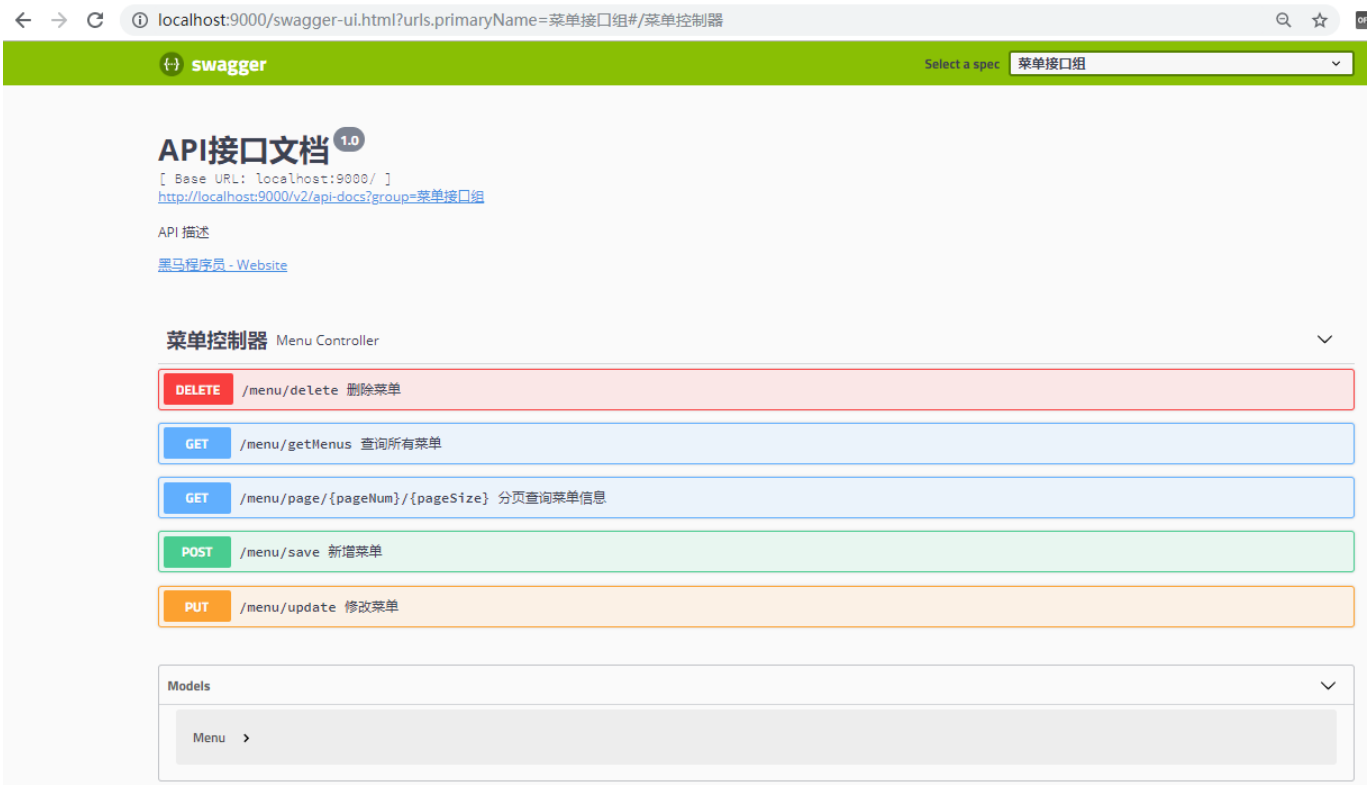
```
package cn.itcast;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SwaggerDemoApplication {
    public static void main(String[] args) {
        SpringApplication.run(SwaggerDemoApplication.class, args);
    }
}
```

执行启动类main方法启动项目，访问地址：http://localhost:9000/swagger-ui.html




localhost:9000/swagger-ui.html?urls.primaryName=菜单接口组#/菜单控制器
swagger
Select a spec
菜单接口组
API接口文档
1.0
[Base URL: localhost:9000/]
http://localhost:9000/v2/api-docs?group=菜单接口组
API 描述
黑马程序员 - Website
菜单控制器 Menu Controller
DELETE /menu/delete 删除菜单
GET /menu/getMenus 查询所有菜单
GET /menu/page/{pageNum}/{pageSize} 分页查询菜单信息
POST /menu/save 新增菜单
PUT /menu/update 修改菜单
Models
Menu