Spring Boot 集成swagger2构建RESTful API文档

1.构建maven工程，引入相关依赖

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.5.6.RELEASE</version>
</parent>

<dependencies>
    <dependency>
        <groupId>io.springfox</groupId>
        <artifactId>springfox-swagger2</artifactId>
        <version>2.6.1</version>
    </dependency>
    <dependency>
        <groupId>io.springfox</groupId>
        <artifactId>springfox-swagger-ui</artifactId>
        <version>2.6.1</version>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
```

2.配置类Swagger2.java

```
@Bean
public Docket createRestApi() {
    return new Docket(DocumentationType.SWAGGER_2).apiInfo(ApiInfo()).select()
            .apis(RequestHandlerSelectors.basePackage("com.fu.controller")).paths(PathSelectors.any()).build();
}

private ApiInfo ApiInfo() {
    return new ApiInfoBuilder().title("Spring Boot中使用Swagger2构建RESTful APIs")
            .description("更多Spring Boot相关文章请关注：http://blog.didispace.com/")
            .termsOfServiceUrl("http://blog.didispace.com/").version("1.0").build();
}
```

3.Controller类

```
@RestController
```

@RequestMapping("/users") public class UserController {

```
static Map<Long, User> users = new ConcurrentHashMap<Long, User>();

@ApiOperation(value = "获取用户列表", notes = "")
@RequestMapping(value = { "" }, method = RequestMethod.GET)
public List<User> getUserList() {
    List<User> list = new ArrayList<User>(users.values());
    return list;
}

@ApiOperation(value = "创建用户", notes = "根据User对象创建用户")
@ApiImplicitParam(name = "user", value = "用户详细实体user", required = true, dataType = "User")
@RequestMapping(value = { "" }, method = RequestMethod.POST)
public String postUser(@RequestBody User user) {
    users.put(user.getId(), user);
    return "success";
}
```

```
@ApiOperation(value = "更新用户详细信息", notes = "根据url的id来指定更新对象，并根据传过来的user信息来更新用户详细信息")
@ApiImplicitParams({ @ApiImplicitParam(name = "id", value = "用户ID", required = true, dataType = "Long"),
        @ApiImplicitParam(name = "user", value = "用户详细实体user", required = true, dataType = "User") })
@RequestMapping(value = "/{id}", method = RequestMethod.PUT)
public String putUser(@PathVariable Long id, @RequestBody User user) {
    User user2 = users.get(id);
    user2.setName(user.getName());
    user2.setAge(user.getAge());
    users.put(id, user2);
    return "success";
}

@ApiOperation(value = "删除用户", notes = "根据url的id来指定删除对象")
@ApiImplicitParam(name = "id", value = "用户ID", required = true, dataType = "Long")
@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
public String deleteUser(@PathVariable Long id) {
    users.remove(id);
    return "success";
}
```

4.访问地址

```
http://localhost:8080/swagger-ui.html
```