

转载请标明出处：

<http://blog.csdn.net/forezp/article/details/71023536>

本文出自[方志朋的博客](#)

swagger,中文“拽”的意思。它是一个功能强大的api框架，它的集成非常简单，不仅提供了在线文档的查阅，而且还提供了在线文档的测试。另外swagger很容易构建restful风格的api，简单优雅帅气，正如它的名字。

一、引入依赖

```
1      <dependency>
2          <groupId>io.springfox</groupId>
3          <artifactId>springfox-swagger2</artifactId>
4          <version>2.6.1</version>
5      </dependency>
6
7      <dependency>
8          <groupId>io.springfox</groupId>
9          <artifactId>springfox-swagger-ui</artifactId>
10         <version>2.6.1</version>
11     </dependency>
12 1
13 2
14 3
15 4
16 5
17 6
18 7
19 8
20 9
21 10
22 11
23 12
```

二、写配置类

```
1 @Configuration
2 @EnableSwagger2
3 public class Swagger2 {
4
5     @Bean
6     public Docket createRestApi() {
7         return new Docket(DocumentationType.SWAGGER_2)
8             .apiInfo(apiInfo())
9             .select()
10
11         .apis(RequestHandlerSelectors.basePackage("com.forezp.controller"))
12             .paths(PathSelectors.any())
13             .build();
14     }
15     private ApiInfo apiInfo() {
16         return new ApiInfoBuilder()
17             .title("springboot利用swagger构建api文档")
18             .description("简单优雅的restfun风格，  
http://blog.csdn.net/forezp")
19             .termsOfServiceUrl("http://blog.csdn.net/forezp")
20             .version("1.0")
21             .build();
22     }
23 }
```

```
24 1
25 2
26 3
27 4
28 5
29 6
30 7
31 8
32 9
33 10
34 11
35 12
36 13
37 14
38 15
39 16
40 17
41 18
42 19
43 20
```

```
44 21
45 22
46 23
47 24
```

通过@Configuration注解，表明它是一个配置类，@EnableSwagger2开启swagger2。apiNfo()配置一些基本的信息。apis()指定扫描的包会生成文档。

三、写生产文档的注解

swagger通过注解表明该接口会生成文档，包括接口名、请求方法、参数、返回信息的等等。

- @Api: 修饰整个类，描述Controller的作用
- @ApiOperation: 描述一个类的一个方法，或者说一个接口
- @ApiParam: 单个参数描述
- @ApiModel: 用对象来接收参数
- @ApiModelProperty: 用对象接收参数时，描述对象的一个字段
- @ApiResponse: HTTP响应其中1个描述
- @ApiResponses: HTTP响应整体描述
- @ApiIgnore: 使用该注解忽略这个API
- @ApiError : 发生错误返回的信息
- @ApiParamImplicitL: 一个请求参数
- @ApiParamsImplicit 多个请求参数

现在通过一个栗子来说明：

```
1 package com.forezp.controller;
2
3 import com.forezp.entity.Book;
4 import io.swagger.annotations.ApiImplicitParam;
5 import io.swagger.annotations.ApiImplicitParams;
6 import io.swagger.annotations.ApiOperation;
7 import org.springframework.ui.ModelMap;
8 import org.springframework.web.bind.annotation.*;
9 import springfox.documentation.annotations.ApiIgnore;
```

```
10
11 import java.util.*;
12
13 /**
14  * 用户创建某本图书 POST    /books/
15  * 用户修改对某本图书    PUT /books/:id/
16  * 用户删除对某本图书    DELETE /books/:id/
17  * 用户获取所有的图书 GET /books
18  * 用户获取某一图书 GET /Books/:id
19  * Created by fangzhipeng on 2017/4/17.
20  * 官方文档: http://swagger.io/docs/specification/api-host-and-base-path/
21  */
22 @RestController
23 @RequestMapping(value = "/books")
24 public class BookContrller {
25
26     Map<Long, Book> books = Collections.synchronizedMap(new HashMap<Long,
27 Book>());
28
29     @ApiOperation(value="获取图书列表", notes="获取图书列表")
30     @RequestMapping(value={}, method= RequestMethod.GET)
31     public List<Book> getBook() {
32         List<Book> book = new ArrayList<>(books.values());
33         return book;
34     }
35
36     @ApiOperation(value="创建图书", notes="创建图书")
37     @ApiImplicitParam(name = "book", value = "图书详细实体", required = true,
38 dataType = "Book")
39     @RequestMapping(value="", method=RequestMethod.POST)
40     public String postBook(@RequestBody Book book) {
41         books.put(book.getId(), book);
42         return "success";
43     }
44
45     @ApiOperation(value="获图书细信息", notes="根据url的id来获取详细信息")
46     @ApiImplicitParam(name = "id", value = "ID", required = true, dataType =
47 "Long",paramType = "path")
48     @RequestMapping(value="/{id}", method=RequestMethod.GET)
49     public Book getBook(@PathVariable Long id) {
50         return books.get(id);
51     }
52
53     @ApiOperation(value="更新信息", notes="根据url的id来指定更新图书信息")
54     @ApiImplicitParams({
55         @ApiImplicitParam(name = "id", value = "图书ID", required = true,
56 dataType = "Long",paramType = "path"),
```


```
52         @ApiImplicitParam(name = "book", value = "图书实体book", required
= true, dataType = "Book")
53     })
54     @RequestMapping(value="/{id}", method= RequestMethod.PUT)
55     public String putUser(@PathVariable Long id, @RequestBody Book book) {
56         Book book1 = books.get(id);
57         book1.setName(book.getName());
58         book1.setPrice(book.getPrice());
59         books.put(id, book1);
60         return "success";
61     }
62     @ApiOperation(value="删除图书", notes="根据url的id来指定删除图书")
63     @ApiImplicitParam(name = "id", value = "图书ID", required = true, dataType
= "Long",paramType = "path")
64     @RequestMapping(value="/{id}", method=RequestMethod.DELETE)
65     public String deleteUser(@PathVariable Long id) {
66         books.remove(id);
67         return "success";
68     }
69
70     @ApiIgnore//使用该注解忽略这个API
71     @RequestMapping(value = "/hi", method = RequestMethod.GET)
72     public String jsonTest() {
73         return " hi you!";
74     }
75 }
76 1
77 2
78 3
79 4
80 5
81 6
82 7
83 8
84 9
85 10
86 11
87 12
88 13
89 14
90 15
91 16
92 17
93 18
94 19
95 20
```


96	21
97	22
98	23
99	24
100	25
101	26
102	27
103	28
104	29
105	30
106	31
107	32
108	33
109	34
110	35
111	36
112	37
113	38
114	39
115	40
116	41
117	42
118	43
119	44
120	45
121	46
122	47
123	48
124	49
125	50
126	51
127	52
128	53
129	54
130	55
131	56
132	57
133	58
134	59
135	60
136	61
137	62
138	63
139	64
140	65
141	66

```
142 67
143 68
144 69
145 70
146 71
147 72
148 73
149 74
150 75
151 76
```

通过相关注解，就可以让swagger2生成相应的文档。如果你不需要某接口生成文档，只需要在加@ApiIgnore注解即可。需要说明的是，如果请求参数在url上，@ApiImplicitParam 上加paramType = “path” 。

启动工程，访问：<http://localhost:8080/swagger-ui.html>，就看到swagger-ui:

 **swagger**

default (/v2/api-docs) 

Explore

springboot利用swagger构建api文档

简单优雅的restfun风格， <http://blog.csdn.net/forezp>

book-contrller : Book Contrller

Show/Hide | List Operations | Expand Operations

GET /books 获取图书列表


POST /books 创建图书

Implementation Notes

创建图书


Response Class (Status 200)

string

Response Content Type 

Parameters

Parameter	Value	Description	Parameter Type	Data Type
book	(required) <div></div>	图书详细实体	body	Model <div>Example Value</div>

Parameter content type: application/json 

```
{
  "id": 0,
  "name": "string",
  "price": 0
}
```

整个集成过程非常简单，但是我看了相关的资料，swagger没有做安全方面的防护，可能需要我们自己做相关的工作。

四、参考资料

swagger.io

[Spring Boot中使用Swagger2构建强大的RESTful API文档](#)