

给Swagger升级了新版本，没想到居然有这么多坑！

原创 梦想de星空 macrozheng 2020-07-15 09:02

收录于合集
#mall学习教程（技术要点篇）


17个

看着 mall 项目中古老的Swagger API文档样式，这次我终于下定决心要给它升个级了。
升级过程中遇到了好多坑，不过只要用好Maven，这些都不是个事！

选择升级版本

首先我们选择下需要升级的版本，直接去Maven仓库看下，哪个版本使用的比较多。虽然有最新版本 2.10.x，但是几乎没什么人用，而上一个版本 2.9.x 使用的人却很多，看样子还是 2.9.x 版本比较稳定，我们选择升级到 2.9.2 版本。

Home » io.springfox » springfox-swagger2

 **Springfox Swagger2**
JSON API documentation for spring based applications

License	Apache 2.0
Tags	io api swagger
Used By	889 artifacts

Central (23) Spring Plugins (3) ICM (6)

Version		Repository	Usages	Date
2.10.x	2.10.5	Central	0	Jun, 2020
	2.10.4	Central	0	Jun, 2020
	2.10.3	Central	0	Jun, 2020
	2.10.2	Central	0	Jun, 2020
	2.10.1	Central	0	Jun, 2020
	2.10.0	Central	1	Jun, 2020
2.9.x	2.9.2	Central	467	Jun, 2018
	2.9.1	Central	4	Jun, 2018
2.8.x	2.8.0	Central	158	Jan, 2018
2.7.x	2.7.0	Central	132	May, 2017

升级Swagger

接下来我们就可以开始升级Swagger版本了，原来项目里用的是 **2.7.0** 版本。

- 由于 **mall** 项目使用父项目来统一管理依赖，所以只要修改父项目中的Swagger依赖版本即可，父项目的pom.xml在项目根目录下；

```
<properties>
    <swagger2.version>2.9.2</swagger2.version>
</properties>
```

- 运行 **mall-admin** 项目发现无法启动，报错信息如下，有个依赖里面的某个方法找不到了，一看是 **guava** 里面的，估计是版本的问题；

```
*****
APPLICATION FAILED TO START
*****
```

Description:

An attempt was made to call a method that does not exist. The attempt was made from the following

```
springfox.documentation.schema.DefaultModelDependencyProvider.dependentModels(DefaultModelDepen
```

The following method did not exist:

```
com.google.common.collect.FluentIterable.concat(Ljava/lang/Iterable;Ljava/lang/Iterable;)Lcom,
```

The method's class, com.google.common.collect.FluentIterable, is available from the following loca

```
jar:file:/C:/Users/macrozheng/.m2/repository/com/google/guava/guava/18.0/guava-18.0.jar!/com/{
```

It was loaded from the following location:

```
file:/C:/Users/macrozheng/.m2/repository/com/google/guava/guava/18.0/guava-18.0.jar
```

Action:

Correct the classpath of your application so that it contains a single, compatible version of com

Process finished with exit code 1

- 当有好几个依赖都使用了不同版本的 **guava** 包时，**Maven**是如何选择的呢？**Maven**是按照就近原则选择的，层级越是浅的依赖越会被选择；



- 此时推荐使用 **Maven Helper** 这款IDEA插件，直接查看 **mall-admin** 项目是否存在依赖冲突，**guava**版本果然冲突了；



- 通过观察可以发现 **minio** 这个依赖层级最浅，所以使用的是它的**guava**版本，直接排除掉即可；

```
<dependency>
  <groupId>io.minio</groupId>
  <artifactId>minio</artifactId>
  <exclusions>
    <exclusion>
      <artifactId>guava</artifactId>
      <groupId>com.google.guava</groupId>
    </exclusion>
  </exclusions>
</dependency>
```

```
</exclusions>
</dependency>
```

- 排除完成后发现guava的依赖冲突已经不见了，再次运行 `mall-admin` 项目，发现已经可以正常运行了；



- 当我们访问Swagger文档时，又发现了一个问题，会报`NumberFormatException`异常；

```
java.lang.NumberFormatException: For input string: ""
    at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)
    at java.lang.Long.parseLong(Long.java:601)
    at java.lang.Long.valueOf(Long.java:803)
    at io.swagger.models.parameters.AbstractSerializableParameter.getExample(AbstractSerializableParameter.java:100)
```

- 原因是当我们使用`@ApiModelProperty`注解时，作为Long数据类型，如果你不添加 `example` 属性，默认值是空字符串，空字符串转型自然就会报`NumberFormatException`异常；

```
/**
 * 修改订单费用信息参数
 * Created by macro on 2018/10/29.
 */
@Getter
@Setter
public class OmsMoneyInfoParam {
    @ApiModelProperty(value = "订单ID", example = "1")
    private Long orderId;
}
```

- 我们已经使用了很多`@ApiModelProperty`注解，要一个个添加那是不可能的，不过使用新版本的 `swagger-annotations` 和 `swagger-models` 依赖包就可以解决了，于是我们的

Swagger依赖变成了下面这样的：

```
<dependencies>
  <dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <exclusions>
      <exclusion>
        <groupId>io.swagger</groupId>
        <artifactId>swagger-annotations</artifactId>
      </exclusion>
      <exclusion>
        <groupId>io.swagger</groupId>
        <artifactId>swagger-models</artifactId>
      </exclusion>
    </exclusions>
  </dependency>
  <dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
  </dependency>
  <!--解决Swagger 2.9.2版本NumberFormatException-->
  <dependency>
    <groupId>io.swagger</groupId>
    <artifactId>swagger-models</artifactId>
    <version>1.6.0</version>
  </dependency>
  <dependency>
    <groupId>io.swagger</groupId>
    <artifactId>swagger-annotations</artifactId>
    <version>1.6.0</version>
  </dependency>
</dependencies>
```

- 再次运行 `mall-admin` 发现该问题已经解决了，我们在maven中一发现不合适的依赖就排除掉，然后引入合适版本的依赖，这样做真的好么？
- 其实我们可以利用Maven项目的继承特性，直接在父项目中规定好依赖的版本，这样子项目的依赖版本就能统一了；

- 先把原来 `pom.xml` 中排除guava和swagger的配置给去除了，然后修改根目录下的pom.xml文件，指定版本号；

```
<properties>
    <swagger2.version>2.9.2</swagger2.version>
    <swagger-models.version>1.6.0</swagger-models.version>
    <swagger-annotations.version>1.6.0</swagger-annotations.version>
    <guava.version>20.0</guava.version>
</properties>
```

- 在父项目的依赖管理节点下添加需要统一管理的相关依赖，至此Swagger版本升级完成；

```
<dependencyManagement>
    <dependencies>
        <!--Swagger-UI API文档生产工具-->
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger2</artifactId>
            <version>${swagger2.version}</version>
        </dependency>
        <dependency>
            <groupId>io.springfox</groupId>
            <artifactId>springfox-swagger-ui</artifactId>
            <version>${swagger2.version}</version>
        </dependency>
        <!--解决Swagger 2.9.2版本NumberFormatException-->
        <dependency>
            <groupId>io.swagger</groupId>
            <artifactId>swagger-models</artifactId>
            <version>${swagger-models.version}</version>
        </dependency>
        <dependency>
            <groupId>io.swagger</groupId>
            <artifactId>swagger-annotations</artifactId>
            <version>${swagger-annotations.version}</version>
        </dependency>
        <!--统一Guava版本防止冲突-->
        <dependency>
            <groupId>com.google.guava</groupId>
            <artifactId>guava</artifactId>
```

```

        <version>${guava.version}</version>
    </dependency>
</dependencies>
</dependencyManagement>

```

- 当我们配置好Token访问需要权限的接口时，会发现品牌、商品、商品分类下的接口有权限访问，其他提示无权限，那是因为我们使用了如下配置来配置需要登录认证的路径：

```

@Configuration
@EnableSwagger2
public class Swagger2Config {

    private List<SecurityContext> securityContexts() {
        //设置需要登录认证的路径
        List<SecurityContext> result = new ArrayList<>();
        result.add(getContextByPath("/brand/.*"));
        result.add(getContextByPath("/product/.*"));
        result.add(getContextByPath("/productCategory/.*"));
        return result;
    }
}

```

- 修改为全部路径即可，这个和旧版有点不同，旧版访问所有接口都会在头信息中带Token，而新版只会对配置的路径带Token。

```

@Configuration
@EnableSwagger2
public class Swagger2Config {

    private List<SecurityContext> securityContexts() {
        //设置需要登录认证的路径
        List<SecurityContext> result = new ArrayList<>();
        result.add(getContextByPath("/*/.*"));
        return result;
    }
}

```

新老版本界面对比

Swagger升级到2.9.2版本后界面瞬间变得美观了，让我们对新老界面来个对比。

老版本



新版本



项目源码地址

<https://github.com/macrozheng/mall>

推荐阅读

- [fastjson到底做错了什么？为什么会被频繁爆出漏洞？](#)
- [微服务权限终极解决方案，Spring Cloud Gateway + Oauth2 实现统一认证和鉴权！](#)
- [听说你的JWT库用起来特别扭，推荐这款贼好用的！](#)
- [线上项目出BUG没法调试？推荐这款阿里开源的诊断神器！](#)
- [Spring Boot 把 Maven 干掉了，正式拥抱 Gradle！](#)
- [性能优越的轻量级日志收集工具，微软、亚马逊都在用！](#)
- [15个Github使用技巧，你肯定有不知道的！](#)
- [写了100多篇原创文章，我常用的在线工具网站推荐给大家！](#)
- [一个不容错过的Spring Cloud实战项目！](#)
- [我的Github开源项目，从0到20000 Star！](#)



欢迎关注，点个在看

收录于合集 #mall学习教程（技术要点篇） 17

上一篇

RabbitMQ实现延迟消息居然如此简单，整个插件就完事了！

下一篇

Elasticsearch 升级 7.x 版本后，我感觉掉坑里了！

阅读原文

喜欢此内容的人还喜欢

项目中到底该不该用Lombok?
macrozheng

