**[#](https://pig4cloud.com/" \l "swagger%E9%85%8D%E7%BD%AE%E6%96%87%E6%A1%A3) Swagger配置文档**

[**#**](https://pig4cloud.com/#%E5%86%99%E5%9C%A8%E6%9C%80%E5%89%8D) **写在最前**

pigX集成了Swagger作为API生成与测试框架,pigX提供了自动化的配置，让您从繁琐的编码工作中解放出来，快速生成自己定制化的Swagger API文档。

[**#**](https://pig4cloud.com/#%E5%BF%AB%E9%80%9F%E4%BD%BF%E7%94%A8) **快速使用**

您可以轻松地在pigX中引入Swagger:

* 在pom.xml中引入以下依赖:

<dependency>

<groupId>com.pig4cloud</groupId>

<artifactId>pigx-common-swagger</artifactId>

</dependency>

* 在应用主类中增加@EnablePigxSwagger2注解

@EnablePigxSwagger2

@EnableFeignClients

@SpringCloudApplication

public class PigxAdminApplication {

public static void main(String[] args) {

SpringApplication.run(PigxAdminApplication.class, args);

}

}

只需以上两步，就能产生当前工程中Spring MVC加载的请求映射所形成的文档。如需要个性化的定制，请看下文。

[**#**](https://pig4cloud.com/#%E9%85%8D%E7%BD%AE%E7%A4%BA%E4%BE%8B%E4%B8%8E%E8%AF%B4%E6%98%8E) **配置示例与说明**

# Swagger相关的配置

swagger:

# 标题,默认空

title: 'PigX Swagger API'

# 描述,默认空

description: '全宇宙最牛逼的Spring Cloud微服务开发脚手架'

# 版本,默认空

version: '1.4.0'

# 许可证,默认空

license: 'Powered By PigX'

# 许可证URL,默认空

licenseUrl: 'https://gitee.com/log4j/pig/wikis'

# 服务条款URL,默认空

terms-of-service-url: 'https://gitee.wang/pig/pigx'

# 文档的host信息，默认：空

host: 'https://gitee.wang/pig/pigx'

# swagger会解析的包路径,默认为空，扫描所有包

base-package: ''

# swagger会解析的url规则

base-path: /\*\*

# 在basePath基础上需要排除的url规则

exclude-path:

- /actuator/\*\*

- /error

# 联系人相关配置

contact:

# 联系人姓名，默认空

name: '冷冷'

# 联系人Email，默认空

email: 'wangiegie@gmail.com'

# 联系人URL，默认空

url: 'https://gitee.wang/pig/pigx'

# 统一鉴权相关配置

authorization:

# 鉴权策略名称，默认空

name: 'pigX OAuth'

# 需要开启鉴权URL的正则，默认匹配所有

auth-regex: '^.\*$'

# 鉴权作用域列表配置,默认空

authorization-scope-list:

# 鉴权作用域名称,默认空

- scope: 'server'

# 鉴权作用域描述,默认空

description: 'server all'

# 校验token的地址列表,默认空

token-url-list:

- 'http://localhost:9999/auth/oauth/token'

**注意:**

* 配置中的鉴权作用域scope必须是数据库sys\_oauth\_client\_details表的scope字段里的内容的一个子集，否则发起Oauth2.0请求时会直接失败。
* 默认情况下Swagger映射Spring MVC中所有的请求,这样的请求包含了排除了Spring Boot默认的监控和异常信息处理路径,通常不是我们想要的。因此提供两种解决方案，任选其一即可。
* 我们可以使用swagger.base-path来指定所有需要生成文档的请求路径基础规则，然后再利用swagger.exclude-path来剔除部分我们不需要的。 我们可以这样设置：

swagger:

base-path: /\*\*

exclude-path:

- /actuator/\*\*

- /error

上面的配置将解析所有除了/actuator开始以及spring boot自带/error请求路径，这样，就排除了Spring Boot默认的监控和异常信息处理路径。

* 除了以上的方法,我们同样可以通过配置包扫描的方式，扫描指定包下的类生成API文档。 我们可以这样设置：

swagger:

base-package: com.pig4cloud.pigx.admin.controller

这样，Swagger只会生成对应包下的API文档，这样，自然也就排除了Spring Boot默认的监控和异常信息处理路径。

[**#**](https://pig4cloud.com/#%E5%A6%82%E4%BD%95%E5%9C%A8pigx-swagger%E4%B8%ADoauth2.0-%E6%8E%88%E6%9D%83) **如何在pigx Swagger中OAuth2.0 授权**

[**#**](https://pig4cloud.com/#%E5%A2%9E%E5%8A%A0%E5%AE%A2%E6%88%B7%E7%AB%AF) **增加客户端**

默认对所有终端进行验证码校验，但是swagger 模拟的时候不需要。

* 通过界面的形式



* 直接操作**sys\_oauth\_client\_details**表

INSERT INTO `pigx`.`sys\_oauth\_client\_details` (

`authorities`,

`authorized\_grant\_types`,

`web\_server\_redirect\_uri`,

`scope`,

`additional\_information`,

`autoapprove`,

`resource\_ids`,

`refresh\_token\_validity`,

`client\_secret`,

`client\_id`,

`access\_token\_validity`

)

VALUES

(

NULL,

'password,refresh\_token',

NULL,

'server',

NULL,

'true',

NULL,

NULL,

'test',

'test',

NULL

);

[**#**](https://pig4cloud.com/#%E8%BF%87%E6%BB%A4%E6%8C%87%E5%AE%9A%E5%AE%A2%E6%88%B7%E7%AB%AF) **过滤指定客户端**

pigx-gateway-dev.yml

# 不校验验证码终端

ignore:

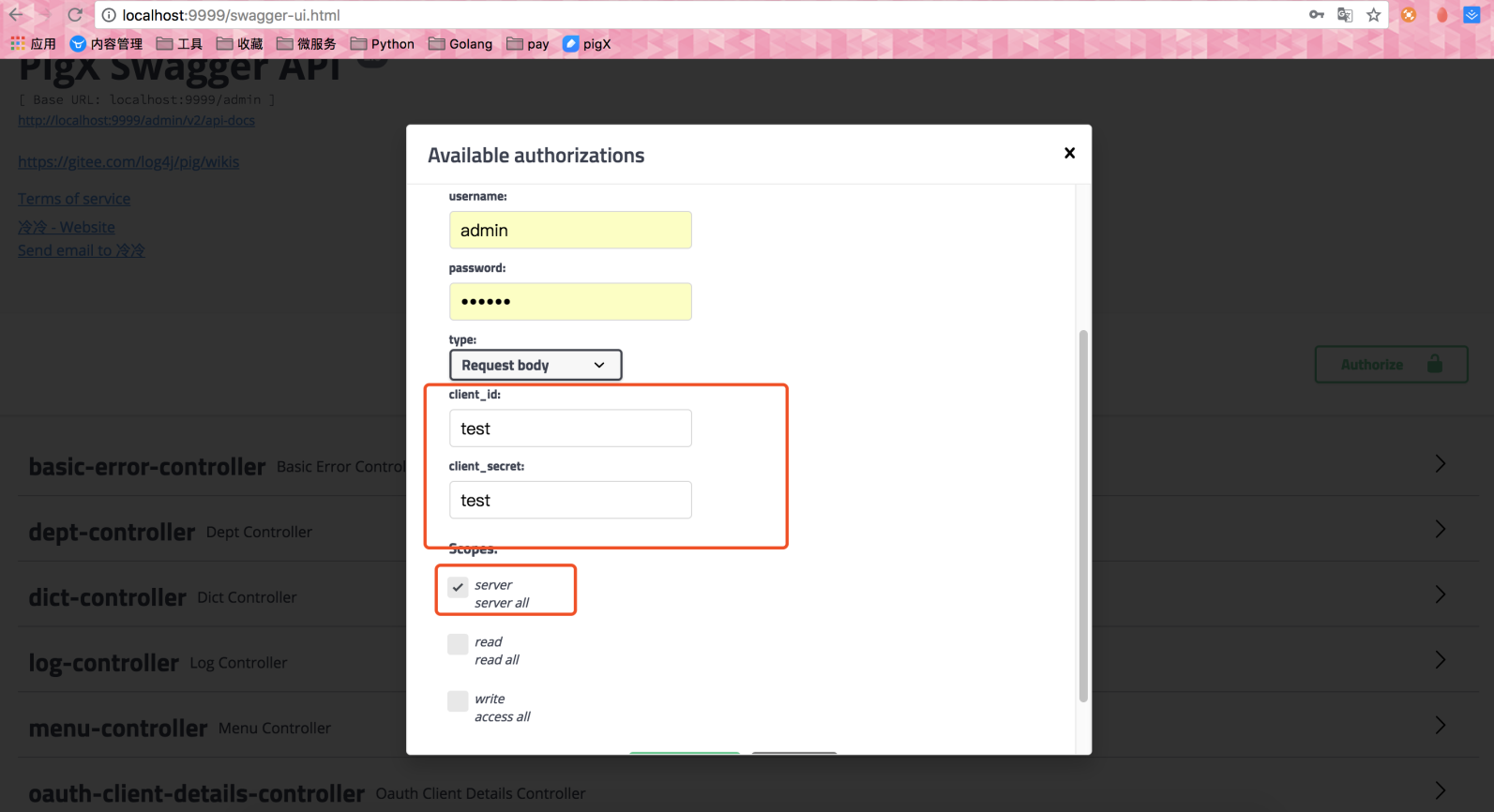
clients:

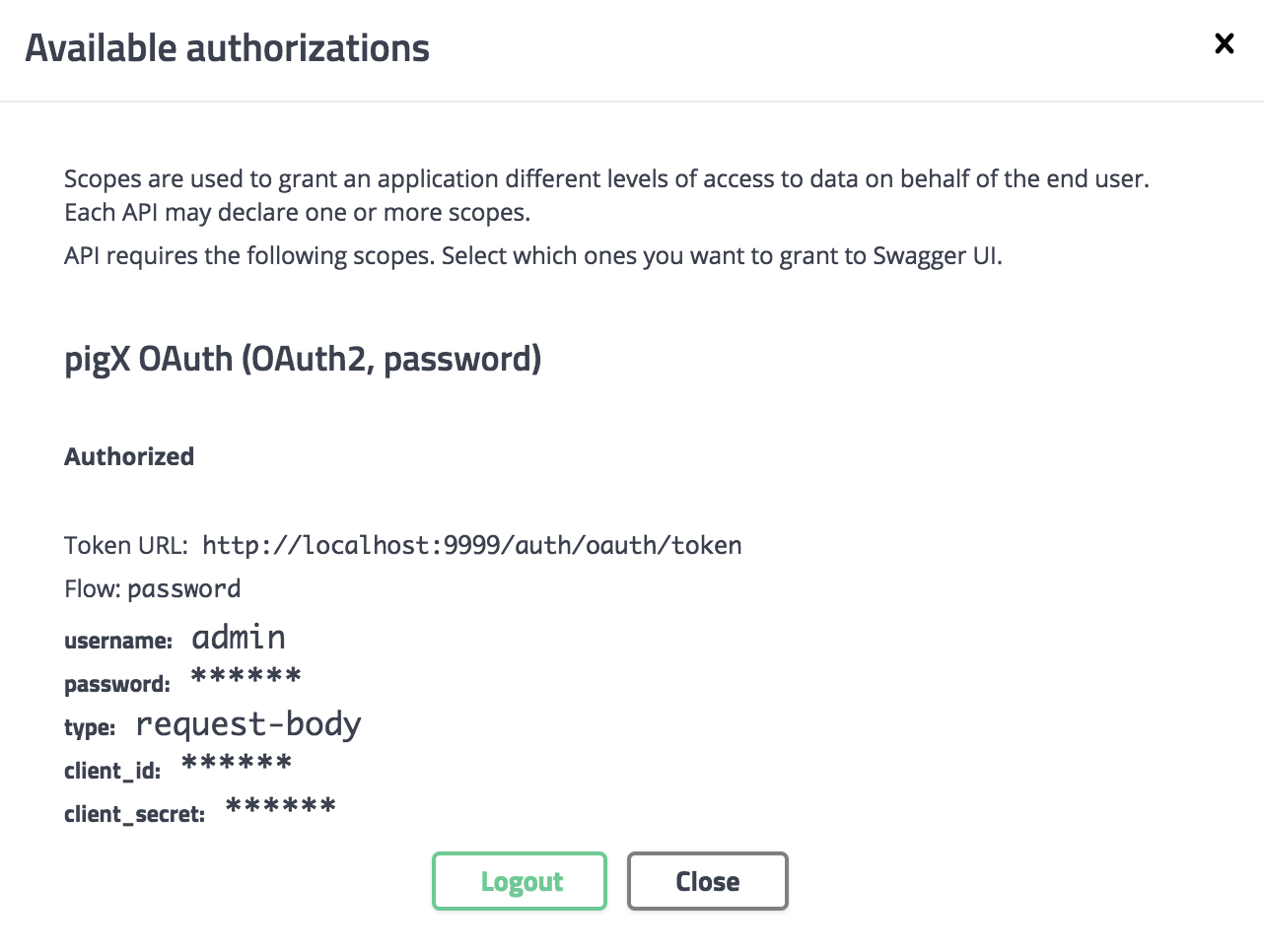
- test

[**#**](https://pig4cloud.com/#%E8%AE%BF%E9%97%AEswagger-ui%E9%A1%B5%E9%9D%A2) **访问swagger-ui页面**

从1.6.3版本开始，要求通过hosts进行访问，在pigx的默认配置下,可以访问http://pigx-gateway:9999/swagger-ui.html打开swagger页面。

[**#**](https://pig4cloud.com/#%E5%A1%AB%E5%86%99%E5%AE%A2%E6%88%B7%E7%AB%AF%E4%BF%A1%E6%81%AF) **填写客户端信息**





[**#**](https://pig4cloud.com/#swagger-faq) **Swagger FAQ**

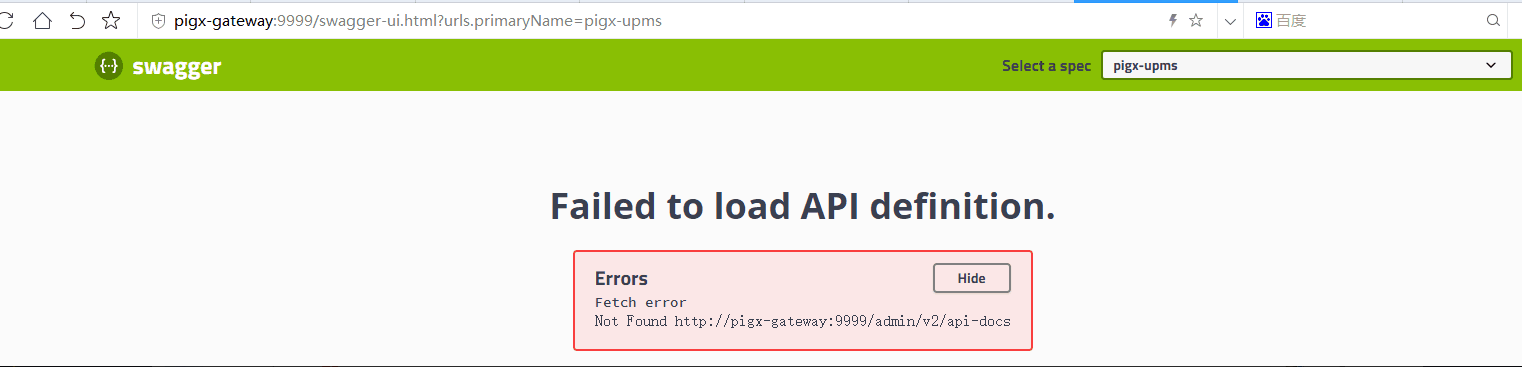
* 为何要进行认证的操作?

认证后Spring Security的上下文对象中才会有值，很多操作如获取当前用户信息都依赖于Spring Security上下文。

* 刷新页面后认证失效?

官方UI比较蠢萌，虽然提供了Oauth2.0的认证功能，但是没有存储的措施，所以刷新页面后相关参数就会丢失。解决办法暴力一点的措施是修改官方UI添加存储措施，但是这个我肯定不会了。目前比较可行的就是修改代码进行swagger全局参数配置。

* 切换swagger分组文档页面报错



出现这个问题要么是你对应的服务没有启动，要么是你访问的服务还没有启动完毕，如果还没启动完毕的话，不妨等个十几二十秒再进行访问，最新的master-mp3分支支持在pigx-gateway-dev.yml中配置ignore.swagger-providers属性来屏蔽掉不希望生成swaager文档的微服务。

* 认证过程中出现Auth ErrorError: Upgrade Required?

这个不用怀疑，原因一般不外乎三个。

一、用户名或密码错误

可以打开谷歌开发者工具，观察request详情和response详情以及返回的状态码，如果是426的话，就证明获取用户信息的时候失败了，可以判断是作为缓存中间件的redis并没有启动，那么只要启动redis，另一个原因是redis中有脏数据，这个时候清空redis即可。清空的具体步骤如下:

windows平台下可以打开redis-cli.exe，然后执行flushdb或者flushall命令即可。

二、使用了需要验证码的客户端

除了上面的原因，还有可能返回428的状态码，而会出现这个问题就是使用了需要验证码的客户端。

三、跨域

排除所有不可能，剩下的那个不管多不可思议，都是事实真相。除开这两个原因，还有可能会出问题的，只有一种情况，那就是出现了跨域问题。

如果本地启动出现了问题，可以观察请求头里是否存在跨域，如果是OPTIONS请求基本就是跨域了。

在目前的项目机制下，开发时解决跨域的最简单的一个方案就是不要通过http://localhost:9999/swagger-ui.html或者http://127.0.0.1:9999/swagger-ui.html去访问网关上的swagger,而是直接通过http://pigx-gateway:9999/swagger-ui.html去访问,这样就能避免跨域的问题。