使用文档

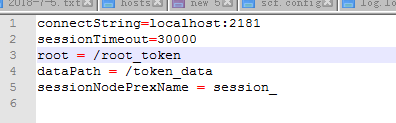
# 过程

名词解释：

具体项目：即客户方的项目、调用sdk来进行接口访问的项目。

* 1. 将HugOpenapiSdk-xxx.jar包引入到具体项目中
  2. 具体的各个功能一一对应的调用接口参考TestClient.java类
  3. 具体项目需要依赖lib下的一些包，如果通过tokenserver获取token,则需要依赖netty, 而直接本地获取token则不需要依赖netty。
* 附加：关于netty版本问题：如果具体项目中有netty3,netty4那么client, tokenserver默认优先用netty4,没有netty4则加载netty3。Client 和tokenserver依赖的netty版本不必相同，即3-4，3-3，4-3，4-4都可以。
  1. 如果不用tokenserver，而用分布式锁的方式来在集群环境下获取锁，目前实现了3种分布式锁：zk, redis, mysql的分布式锁。使用时，只需要在具体项目里添加zk或redis,或mysql的依赖、初始化代码里指定各自的配置文件。
* zk方式：
* 添加依赖zookeeper-3.4.13.jar
* 初始化：其中第二个参数是配置文件：



* 配置文件格式:

connectString ：zk集群地址

sessionTimeout: 连接到zk集群超时的时间

root: 在zk集群上建立的根节点，包括client的临时会话节点和数据节点

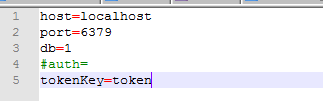
dataPath:数据节点路径，在root下创建

sessionNodePrexName: 临时会话节点的前缀名。

* redis方式：
* 添加依赖jedis-2.9.0.jar
* 初始化：



* 配置文件格式：



host: redis地址

port: 端口

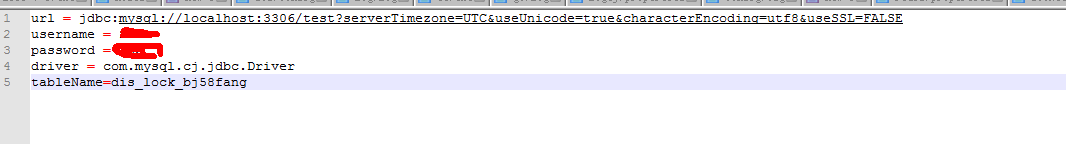
db: 创建保存token的数据库

tokeKey: 保存token使用的键名

* mysql方式：
* 添加依赖mysql-connector-java-8.0.12.jar
* 初始化：



* 配置文件格式：



url: 数据库连接地址

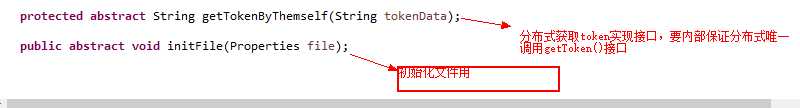
username:用户名

password: 用户名

driver:数据库驱动类

tableName: 保存token的表

* 其他数据库方式：
* 继承com.bj58.fang.hugopenapi.client.provider. CommonProvider抽象类，实现其中两个抽象方法。另外，抽象类提供一个方法String getToken()来从开放后台获取token，这个方法可以在实现getTokenByThemself()中使用。



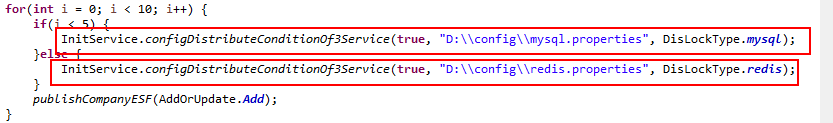
* 初始化：第三个参数就是自己的类



* 配置文件格式，java.util.Properties能识别的格式。
* 组合方式：
* 必须条件：所有客户端在尽可能短的时间内切换完成，比如1min内，否则会因为token存在多个数据库而频繁刷新token。

如果想多种方式结合，而不是使用一种，来保证更高的高可用性，可以在运行时重新初始化的方式切换token获取方式。比如运行中发现redis挂了而zookeeper还可以用，那么可以重新初始化来切换token获取方式，因为内部共享一个本地缓存，所以切换时不会重新请求token。

例子：



切换之后，token就存在切换后的数据库里。

# 从功能找接口

网站上所有列出的接口都在HugOpenapiSdk-xxx.jar 中的几个service中实现：

经纪人:

经纪人公共接口：BrokerPublicService.java

经纪人二手房操作接口：ESFService.java

经纪人租房操作接口：ZFService.java

公司:

公司公共接口：CompanyPublicService.java

公司二手房操作接口：ESFService.java

公司租房操作接口：ZFService.java

公司网络门店操作接口：CompanyPublicService.java

小区操作接口：XQService.java

城市列表查询接口：DispLocalService.java

来电通服务接口：LdtService.java

# 说明

* 1. 如果token的获取从不是从本地获取，而且存在多个客户端同时共用一套clientId，clientSecret那么此时token的获取方式应该改为从一个独立的服务上获取。这个独立的服务就是TokenServer-xxx.jar，可以通过java -Djava.ext.dirs=./lib/ -jar TokenServer-1.0.1-SNAPSHOT.jar来启动，其中./lib/下是依赖的jar。此时需要配置为从独立服务上获取token, 即补充配置该独立服务所在的ip, 在TestClient.java里有说明。

另外，如果独立服务的端口不希望为默认的16778, 那么可以java -Djava.ext.dirs=./lib/ –jar TokenServer-xxx.jar 16779 这样的方式改变端口为16779

* 如果想slf + log4j的方式管理日志、记录日志，那么执行命令为：

java -Djava.ext.dirs=./lib -Dlog4j.configuration=file:./config/log4j.properties -jar TokenServer-1.0.1-SNAPSHOT.jar

相应log4j.properties配置文件可以自行配置。lib目录下有相应的log4j依赖。

* 1. 关于调用方法中的参数问题：
     1. 有的实体型参数的某些属性可能是枚举类型的，且这个枚举的名称就是 “属性名-且头字母大写”，如果可以搜索到，说明这个属性值是枚举的，可以清晰地使用枚举来赋值。比如：新建成交记录接口；这个的接口的参数是DealRecordEntity实体，其中有个属性deptIdType，赋值的时候就可以使用它对应的枚举类：DeptIdType
  2. token获取服务如果想引入到自己的项目中作为一个单独的串行服务，形成自己的服务而不是一个专门的server来启动，那么可以使用TokenServer-xxx.jar中的TokenProvider的getTokenData方法，来制作自己的串行中心服务。

例如：



------------注意：如果直接并行的调用，会不断在token失效后刷新token，可能会因频繁的调用token刷新服务，而被限流。

