# 1. 环境搭建

gradle方式:

Step 1. Add the JitPack repository to your build file  
  
gradle  
Add it in your root build.gradle at the end of repositories:  
  
 allprojects {  
 repositories {  
 ...  
 maven { url 'https://www.jitpack.io' }  
 }  
 }  
Step 2. Add the dependency  
  
 dependencies {  
 implementation 'com.github.TokenTM.tks\_android:lib\_crop:0.1.3\_test'  
 }

demo地址:<https://github.com/TokenTM/tks_android> [点击查看](https://github.com/TokenTM/tks_android)

# 2. 架构说明

统一使用XXF框架,点击->[文档地址](https://github.com/NBXXF/xxf_android)

client对象：TokenTmClient

使用方式:

TokenTmClient.getService(RepoService.class)  
 .xxxFunction()

.subscribe();

已经提供的RepoService

## 2.1. RepoService 服务

//did服务

DIDService;

//私有数据存储

StoreService;

//认证服务

CertService;

//文件服务

FileService;

//身份密码服务

IdentityPwdService;

//上链服务

ChainService;

//node 节点服务

NodeSrvice;

//node 在线加解密服务

NodeEncryptSrvice;

...更多参考sdk源码

## 2.2. 具体使用方式

TokenTmClient.getService(DIDService.class)...

TokenTmClient.getService(StoreService.class)...

......

# 3. 身份密码UI模组

身份份密码UI模组,是基于tokentm 样式UI和IdentityPwdService实现的管理身份密码的界面,如果不喜欢这种界面设计,可以自行设计,然后调用IdentityPwdService实现核心逻辑;

注意:身份密码是RepoService的基础,相关服务调用都需要身份密码,确保数据的安全性,所以在调用这些服务之前一定要先设置身份密码,否则将会报错,推荐在中心化系统注册之后立即设置身份密码,当然也可以在具体业务之前设置身份密码;

## 3.1. 设置身份密码



ComponentUtils.*launchUserIdentityPwdActivity*(  
 DidDemoActivity.**this**,  
 **"17611639080"**,  
 **new** Consumer<String>() {  
 @Override  
 **public void** accept(String uDID) **throws** Exception {  
 *//****TODO 中心化系统和userId进行绑定*** DemoSp.*getInstance*().putString(**"did"**, uDID);  
 **binding**.**didText**.setText(**"did:"** + uDID);  
 }  
 });

## 3.2 重置身份密码

ComponentUtils  
 .*launchUserIdentityPwdAReSetctivity*(  
 DidDemoActivity.**this**,  
 **did**,  
 **"17611639080"**,  
 **new** Consumer<Boolean>() {  
 @Override  
 **public void** accept(Boolean reseted) **throws** Exception {  
 ToastUtils.*showToast*(**"重置成功?"** + String.*valueOf*(reseted));  
 }  
 });

## 3.3 身份密码验证



ComponentUtils.*showIdentityPwdDialog*(  
 DidDemoActivity.**this**,  
 **did**,  
 **new** BiConsumer<DialogInterface, String>() {  
 @Override  
 **public void** accept(DialogInterface dialogInterface, String identityPwd) **throws** Exception {  
 dialogInterface.dismiss();  
 *//****TODO 下一步*** }  
 });

## 3.4 身份密码解开(已有身份密码,需要解开)



ComponentUtils.decryptUDID();

# 4. 认证UI模组

认证UI模组,是基于tokentm 样式UI和CertService实现的用户实名认证和公司法人认证的界面,如果不喜欢这种界面设计,可以自行设计,然后调用CertService实现核心逻辑;

## 4.1. 实名认证:



ComponentUtils.*launchUserCertActivity*(  
 MainActivity.**this**,  
 **new** UserCertByIDCardParams.Builder(did)  
 .setUserName(**"小炫风"**)  
 .setUserIDCard(**"511324198901090148"**)  
 .build(),  
 **new** Consumer<String>() {  
 @Override  
 **public void** accept(String txHash) **throws** Exception {  
 *//****TODO 中心化系统进行记录*** ToastUtils.*showToast*(**"实名认证成功:"** + txHash);  
 }  
 });

## 4.2. 公司认证:



注意: 公司认证前一定要先完成个人身份认证！

ComponentUtils.*launchCompanyCertActivity*(  
 MainActivity.**this**,  
 **new** CompanyCertParams.Builder(did, **"北京百度科技有限公司"**, **"李彦宏"**).build(),  
 **new** Consumer<CompanyCertResult>() {  
 @Override  
 **public void** accept(CompanyCertResult companyCertResult) **throws** Exception {  
 *//****TODO 中心化系统进行记录*** ToastUtils.*showToast*(**"公司认证成功:"** + companyCertResult);  
 }  
 });

## 4.3. 链信认证



*ComponentUtils.launchCompanyChainCertificationActivity(*

*CertificationDetailsActivity.this,*

*new Consumer<CompanyCertResult>() {*

*@Override*

*public void accept(CompanyCertResult companyCertResult) throws Exception {*

*//TODO 处理开启链信认证回值*

*}*

*});*

## 4.4 开启身份认证弹窗



*ComponentUtils.showUserIdentityAuthenticationAlertDialog(  
 CertificationDetailsActivity.this,  
 new BiConsumer<DialogInterface, Boolean>() {  
 @Override  
 public void accept(DialogInterface dialogInterface, Boolean identityPwd) throws Exception {  
 dialogInterface.dismiss();  
 //TODO 点击回调  
 }  
 });*

## 4.5 开启企业认证弹窗



*ComponentUtils.showCompanyCompanyEnterpriseCertificationAlertDialog(  
 CertificationDetailsActivity.this,  
 new BiConsumer<DialogInterface, Boolean>() {  
 @Override  
 public void accept(DialogInterface dialogInterface, Boolean identityPwd) throws Exception {  
 dialogInterface.dismiss();  
 //TODO 点击回调  
 }  
 });*

## 4.6 认证详情



*ComponentUtils.launchUserCertificationDetailsActivity(  
 CertificationDetailsActivity.this,  
 new Consumer<CompanyCertResult>() {  
 @Override  
 public void accept(CompanyCertResult companyCertResult) throws Exception {  
 //TODO 处理认证详情返回值  
 }  
 });*

## 4.7 认证说明



*ComponentUtils.launchCompanyCertificationInstructionsActivity(  
 CertificationDetailsActivity.this,  
 new Consumer<CompanyCertResult>() {  
 @Override  
 public void accept(CompanyCertResult companyCertResult) throws Exception {  
 //TODO 处理认证说明返回值  
 }  
 });*

## 4.8 物权转移记录



*ComponentUtils.launchUserPropertyRightsTransferRecordsActivity(  
 CertificationDetailsActivity.this,  
 new Consumer<CompanyCertResult>() {  
 @Override  
 public void accept(CompanyCertResult companyCertResult) throws Exception {  
 //TODO 处理物权转移记录返回值  
 }  
 });*