GMS_SDK Access File

Contents

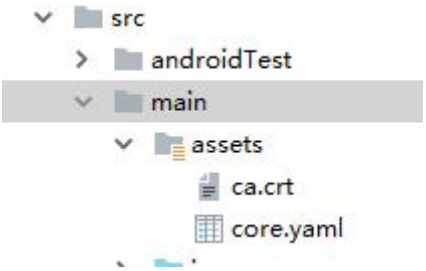
1. IDE Configuration:

After decompression of the access file package provided by us, you can see the following assets certificate folder, libs resource package file, sdkDemo, aar package, GMS_SDK access file, remote dependency library file, and sdk upgrade log under the root directory:

Steps:

1.1 Import assets holder

Import the assets folder into the Android Studio project at the following location:



1.2 Import aar package

Place two aar packages directly under the libs package in the Android Studio project and add the following in the build gradle of the project

```
Repositories{
 flatDir{
dirs 'libs'}}
```

Add two lines (if your Android Studio is a version earlier than 3.0, please modify the following "Implementation" to "Compile")

```
    Implementation(name: 'dg_libproject', ext: 'aar')
    Implementation(name: 'gamesdklib', ext: 'aar')
```

1.3 Import remote dependence library (Note: Either directly copy the libs file to the corresponding location of the project. Only one of them (remote dependence library and libs resource package) can be selected, and the remote dependence library is used below)

Copy from a remote dependence library file and import it to the remote dependence library: As shown in the figure below:

```
dependencies{

    implementation fileTree(include:['*.jar'],dir:'libs')
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation(name: 'dg_libproject', ext: 'aar')
  implementation(name: 'gamesdklib', ext: 'aar')
  implementation('org.web3j:core:3.3.1-android')
  implementation 'com.journeyapps:zxing-android-embedded:3.3.0@aar'
  implementation 'com.google.zxing:core:3.3.0'
    implementation 'com.squareup.okhttp3:okhttp:3.10.0'
  implementation 'com.android.support:multider:1.0.3'
  implementation 'com.android.support.constraint:constraint-layout:1.1.2'

}
```

1.4 so file adaptation

Add the following code to build. gradle under the app to adapt to the model:

```
ndk{
        abiFilters"armeabi-v7a","x86","armeabi"


        }
```

As shown in the figure below:

```
android{
  compileSdkVersion 26
  defaultConfig{
      applicationId "com.GMS.sdkdemo"
      minSdkVersion 16
      targetSdkVersion 26
      versionCode 1
      versionName "1.0"
      testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"

      ndk {
        abiFilters "armeabi-v7a",   "x86" , "armeabi"
          }
      }
```

1.5 Rebuild project

From the toolbar, select "Build"-->"rebuild project"

Tips: Due to different versions of gradle, there may be some slight differences in configuration, and Baidu may be searched to refer to the relevant online configuration files to introduce the aar package to the project.

Note: If the file configuration is completed, the project will report an error: The number of referenced Dex files exceeds the method number limit of 65536, please refer to the website:

https://www.aliyun.com/jiaocheng/804289.html

Start the client configuration after the above IDE configuration

2. Client Configuration: Current version: v1.0.0

2.1 Permission and initialization sdk (access required)

*Config.LOGIN.....Logged in
*Config.UNLOGIN.....Unlogged in
*Config.LOGINOUT.....Logged out
*config.setLandscap.....Horizontal and vertical screen setting, true for horizontal screen and false for vertical screen
*config.setAppId.....appId passed by cp
*config.setApibrowse.....Official browser address passed by cp
*config.setDebugApibrowse.....Demo browser address passed by cp
*config.setFloatGravity.....Initial display location of the float
*config.setDecimals.....Accuracy
*config.setErc721.....Display ERC_721 equipment transaction record or not, true for display and false for no display
*config.setGMSDebug.....Determine whether the environment is official (false) or demo (true)
*config.setLanguage.....en English cn Chinese
*@param amount.....Payment amount
*@param orderId.....Order number
*@param toAddress.....Custom payment address
*@param comment+System.currentTimeMillis //Payment information (Special note: Payment information **must** be followed by a timestamp to avoid repeated transactions)
*@param tokenId.....Transaction ID (unique ID of item)
*@param equip_info..Details of game equipment (including id, name, price, pictures, notes, etc., spliced in json format)
The pictures can be in url address or base64 string format (Prefix required: data:img/jpg;base64,)
*@param comment_erc..Game appid, server category id, game equipment id (**string connected by comma**)
PermissionUtils //Permission utility (Tips: Firstly instantiate a utility object)
setPermission(); //Permission access method
getConfig(); //Get the config object for sdk initialization to call
QueryGameAmount(this,new ICallback);//Method for querying current user's sub-chain currency balance (refer to SDKDEMO)
QueryDgasAmount(new ICallback); //Method for querying current user's DGAS balance (refer to SDKDEMO)
***transErc*(MainActivity.this, orderId, tokenId, equip_info,comment)** //ERC_721 transfer
***transErcAddress*(MainActivity.this,orderId,toAddress,tokenId,equip_info,comment)** //ERC_721 Enter transfer address for transfer
gameQueryAssetErc(tokenId) //ERC_721 Query about asset attribution function
subStringRecharge(); //Method for directly activating dgas or GMS recharge page
fetchSubChain(); //Method of directly activating sub-chain currency transfer page

Call from the main Activity:

```
PermissionUtils permissionUtils=new PermissionUtils();
Public void getConfig(){
config=new Config();
```

```
    config.setAppId("V431rSWOMpq3xGGJYSQTGH5ox1MBiXjJRw");
    config.setApibrowse("http://192.168.2.32:801");
    config.setDebugApibrowse("https://queryfb.GMS.org.4490");
    config.setFloatGravity(FloatGravity.TOP_CENTER);
    config.setDecimals("100000000");
    config.setLanguage("cn");
    config.setErc721(true);
    config.setGMSDebug(true);
    }
    perms = new String[]{
                          Manifest.permission.CAMERA,
                         Manifest.permission.WRITE_EXTERNAL_STORAGE,
                         Manifest.permission.READ_PHONE_STATE,
                         Manifest.permission.ACCESS_FINE_LOCATION
                          };
    private void setPermission(){
            if(android.os.Build.VERSION.SDK_INT>=android.os.Build.VERSION.CODES.M){

            permissionUtils.requestRunPermission(perms,new PermissionListener()){
            @Override
            public void onGranted(){
                        GMSManager.init(this,config);
                        }
            @Override
            public void onDenied(List<String> deniedPermission){
                  for(int i=0;i<deniedPermission.size();i++){
            showRequestRunPermission=ActivityCompat.shouldsShowRequestRunPermissionRationale
(this,.get(i));
            }if(showRequestRunPermission){
            setPermission();
                        }else{
                        showMissingPermissionDialog();
                        }
                        }
                  });
            }else{
                        GMSManager.init(this,config);
            }
    //Permissions callback method
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]permissions,
@NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);

        permissionUtils.onRequestPermissionsResult(requestCode, permissions, grantResults);
          }
```

```
Private void showMissingPermissionDialog(){

AlterDialog.Builder builder=new AlterDialog.Builder(this);

   builder.setTitle("prompt");

   builder.setMessage("For your normal use of SDK,please open the permissions!");

   builder.setPositiveButton("go Set!",new DialogInterface.OnClickListener(){
  Public void onClick(DialogInterface dialog,int which){
                              startAppSettings();
                }
});
}
       Builder.setCancelable(false);
       Builder.show();
}
        Private void startAppSettings(){
        Intent intent=new Intent(Setting.ACTION_APPLICATION_DETAILS_SETTINGS);
        intent.setData(Uri.parse("package:"+getPackageName()));
        startActivityForResult(intent,REQUEST_CODE_SDK_RESULT_PERMISSIONS);
        }
   @Override
   protected void onActivityResult(int requestCode, int resultCode, Intent data){
   super.onActivityResult(requestCode, resultCode, data);
   if (requestCode==REQUEST_CODE_SDK_RESULT_PERMISSIONS){
            setPermission();

   }
```

2.2 Add a life cycle callback (access required)

Add in onStart of main Activity

```
GMSManager.onStart();
```

Add in onResume of main Activity

```
GMSManager.onResume();
```

Add in onPause of main Activity

```
GMSManager.hideFloatintView();
```

Add in onStop of main Activity

```
GMSManager.onStop();
```

Add in onDestroy of main Activity

```
GMSManager.onDestory();
```

2.3 api provided by sdk

In the following table, the following call methods are: GMSManager. function name (...). For the specific use method, please refer to the sdkDemo provided.

Function name	Role	Parameter & type	Description
Login(context)	Login	Context (context)	Just call it directly
setSDKLoginCallback(new ILoginCallBack())	Login callback (Please refer to demo)	ILoginCallBack (callback interface)	OnSuccess（message） Callback success, Cp end obtains: signdata (Login validation parameter) uname (Player name) address (Equipment address) OnFailed（message） Callback failed
Pay(context,orderId,amount， comment)	Payment	Context (context) orderId (payment order) Amount (payment amount) Comment (payment information)	Just call it directly
setSDKPayCallback(new IpayCallback)	Payment callback	IPayCallback (callback interface)	OnPaySuccess(jsonStr) Callback success. Cp end obtains: payOrderId (payment order number) txid (transaction number) onPayFail(message)
PayAddress（context,orderId,toAddress,amount, comment）	Custom payment address	Context (context) orderId (payment order) toAddress (payment address) Amount (payment amount) Comment (payment information)	Just call it directly
openUserCenter()	Open the	None	Just call it directly

Function name	Role	Parameter & type	Description
	User Center		
isLogin()	Judge login or not	None	Just call it directly

transErc(context,orderId,tokenId,equip_info,comment)	ERC_721 Asset transfer	Contex Context) orderId (payment order) tokenId (unique ID of item) equip_info (game equipment information) Comment (transfer information)	Just call it directly

Note: Refer to the server payment callback interface file for the payment callback.

3. Operation of Game Server

Equipment (contract ERC721) blockchain launch

Preparation for development requirements:
Launch to the blockchain through the SDK operating equipment of Web3 (contract ERC721)
SDKs of Web 3 include:
java
https://github.com/web3j/web3j

nodejs
https://github.com/ChainSafe/web3.js

Guide for operating the blockchain through Web3:
https://web3.tryblockchain.org/Web3.js-api-refrence.html

SDK features:

Server validation:
Log in to verify the validity of the user account (ETH address), which can be operated through the SDK of Web3
Reference for validation: https://blog.csdn.net/zgf1991/article/details/113247362

Launch to blockchain:
Identify the equipment to be launched to the blockchain according to the characteristics of the game service and the blockchain delay, and plan the TokenId resource ID (uint256 integer) of the equipment to be launched to the blockchain.

Transfer:
Transfer Ethernet resources such as ERC20 and ERC721 by operating the SDK of Web3. Transfer requires the game server to provide its own account of transferable resources, which can be operated manually or automatically according to requirements.

Query:
Query through the API interface of Ethscan, please refer to:
Chinese website:
https://learnblockchain.cn/docs/etherscan/index.html
Official website:
https://docs.etherscan.io/