YiCameraSdk Introduction

Introduction

In this sdk, we provide six main function about YiCamera

1. Process of guiding user to bind device
2. Component of displaying user's device list
3. Component of displaying devices alerts
4. Component of playing device video stream
5. Component of configuring device
6. Component of displaying device cloud

Integration

copy yicamerasdk.aar to your project's libs dir, add the lines below to project's build.gradle

```
implementation fileTree (include: ['* .jar', '*. aar'], dir: 'libs')
implementation "androidx.cardview:cardview:1.0.0"
implementation 'com.google.android.material:material:1.0.0'
implementation 'androidx.multidex:multidex:2.0.1'
implementation 'com.squareup.retrofit2:retrofit:2.5.0'
implementation 'com.squareup.retrofit2:adapter-rxjava2:2.5.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.12.0'
implementation 'com.google.code.gson:gson:2.8.5'
implementation 'com.google.zxing:core:3.3.0'
implementation "androidx.core:core-ktx:+"
implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"

implementation 'androidx.multidex:multidex:2.0.1'
api 'com.squareup.retrofit2:retrofit:2.5.0'
api 'com.squareup.retrofit2:adapter-rxjava2:2.5.0'
api 'com.squareup.retrofit2:converter-gson:2.4.0'
implementation 'io.reactivex.rxjava2:rxjava:2.2.8'
implementation 'io.reactivex.rxjava2:rxandroid:2.1.1'
implementation 'com.squareup.okhttp3:okhttp:3.12.0'
implementation 'com.iheartradio.m3u8:open-m3u8:0.2.4'
implementation 'androidx.appcompat:appcompat:1.1.0'
implementation 'androidx.recyclerview:recyclerview:1.1.0'
implementation "androidx.cardview:cardview:1.0.0"

implementation 'com.google.dagger:dagger:2.24'

implementation 'androidx.paging:paging-rxjava2-ktx:2.1.0'
implementation 'androidx.paging:paging-common-ktx:2.1.0'
implementation 'androidx.paging:paging-runtime-ktx:2.1.0'
implementation 'com.squareup.okhttp3:logging-interceptor:3.12.0'

implementation 'com.uber.autodispose:autodispose-android:1.1.0'
implementation 'com.uber.autodispose:autodispose-ktx:1.1.0'
implementation 'com.uber.autodispose:autodispose-android-ktx:1.1.0'
implementation 'com.uber.autodispose:autodispose-android-archcomponents-ktx:1.1.0'

implementation 'androidx.room:room-runtime:2.1.0-alpha06'
implementation 'androidx.room:room-ktx:2.1.0-alpha06'
annotationProcessor 'androidx.room:room-compiler:2.1.0-alpha06'
kapt 'androidx.room:room-compiler:2.1.0-alpha06'
implementation 'androidx.room:room-rxjava2:2.1.0-alpha06'
implementation 'androidx.room:room-testing:2.1.0-alpha06'
implementation 'androidx.lifecycle:lifecycle-viewmodel:2.0.0'
implementation 'androidx.lifecycle:lifecycle-extensions:2.0.0'

implementation 'com.github.bumptech.glide:glide:4.9.0'
kapt 'com.github.bumptech.glide:compiler:4.9.0'
implementation 'com.alibaba:arouter-api:1.5.0'
kapt 'com.alibaba:arouter-compiler:1.2.2'
```

YiCameraSdk introducion

At first, user infomation must be provided, this can be get from Yi. user info includes user open id, user token.

Second, set user info to YiCameraSdk and init YiCameraSdk.

init YiCameraSdk

```
// init YiCameraSDK using appid, app package name
YiCameraSdk.init(getApplication(),appid,appsign,appPackage);
// get uuid from YiCameraSdk after YiCameraSdk inited
String uuid = YiCameraSdk.getUUID();
// set RELEASE for release url, SANDBOX for debug url
YiCameraSdk.setEnv(YiCameraSdk.Env.RELEASE);
// request user info with this uuid
// ...
// after getting user info, set to YiCameraSdk
YiCameraSdk.setUserInfo(openid,token);

To listen setUserInfo result:
1.YiCameraSdk.setOnUserVerifyListener(this);
2.Make your activity implements YiCameraSdk.OnUserVerifyListener
3.Add the following method:
@Override
public void OnUserVerifyResult(boolean result,int code) {//result: true means success, false means fail
    //do something
}
```

To start process of binding device

```
OpenDeviceBindFragment fragment = new OpenDeviceBindFragment();
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, fragment);
ft.commitAllowingStateLoss();
```

To display devices alerts, add code below to your activity

```
NewAlertFragment fragment = new NewAlertFragment();
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, fragment);
ft.commitAllowingStateLoss();
```

To display devices list

```
DeviceListFragment deviceListFragment = new DeviceListFragment();
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, deviceListFragment);
ft.commitAllowingStateLoss();
```

To start realtime video streaming

```
StartCameraPlayerFragmentfragment = StartCameraPlayerFragment.newInstance(uid, new YiCameraSdk.
Callback() { //uid is the uniqe id of device
    @Override
    public void onResult(boolean success, String reason) {

    }
});
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, fragment);
ft.commitAllowingStateLoss();
```

To start device configuring activity

```
CameraSettingFragment fragment = CameraSettingFragment.newInstance(uid);
//uid is the uniqe id of device
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, fragment);
ft.commitAllowingStateLoss();
```

To display device cloud

```
CloudVideoFragment fragment = new CloudVideoFragment();
FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
ft.add(containerId, deviceListFragment);
ft.commitAllowingStateLoss();

To listen upgrade button clicked:
1.YiCameraSdk.setOnUpgradeClickedListener(this);
2.Make your activity implements YiCameraSdk.OnUpgradeClickedListener
3.Add the following method:
@Override
public void onUpgradeClicked(String deviceId, String orderCode, int currentStatus) {
    //do something
}
```

To listen button Clicked in fragment:

```
1.YiCameraSdk.setOnFragmentBackBtnListener(this);
2.Make your activity implements YiCameraSdk.OnFragmentBackBtnListener
3.Add the following method:
@Override
public void OnFragmentBackBtn(int type) {
        /**
         * type:
         *         YiCameraSdk.BIND_FRAGMENT:  OpenDeviceBindFragment Back Button Clicked
         *         YiCameraSdk.CAMERA_PLAY_FRAGMENT:  StartCameraPlayerFragment Back Button Clicked
         *         YiCameraSdk.CAMERA_SETTINGS_FRAGMENT:  CameraSettingFragmentBack Button Clicked
         *         YiCameraSdk.CAMERA_SETTINGS_FRAGMENT_DELETE:  Delete device success when clicked
"Delete" Button in CameraSettingFragment
         */

    //do something
}
```

Demo Code Structure

In the demo, we have the files below:

```
 AlertActivity.java
 BaseApplication.java
 CloudActivity.java
 DemoActivity.java
 DeviceActivity.java
 GlideConfiguration.java
 LoginActivity.java
 http
     DemoApi.java
     DemoHttp.java
```

1. BaseApplication.java

   extended from MultiDexApplication to support over 63335 methods
2. LoginActivity.java

   a demo login activity, which implement the process of getting userid and user openid, user token.
3. DemoActvitiy.java

   after get user info, init YiCameraSdk with user info and app info
4. GlideConfiguration.java

   To configure Yi customed decoder for Yi encrypted images
5. DeviceActivity.java

   hold DeviceListFragment to show devices
6. AlertActvitity.java

   hold NewAlertFragment to show alerts
7. CloudActvitity.java

   hold CloudVideoFragment to show device cloud
8. http

   retrofit apis

check the code for more detail

Change List

Change Details	Area	Type	Status	Version	Verified Date
Fix the bug that not able to update select and update firmware in H30.	Android SDK	Bug	Fixed	V1.1.4_20201110	2020/11/10