



TVU Anywhere SDK Integration Guide - Android Platform

Version 2.0
Nov 17, 2022

[Getting Started](#)

[Overview](#)

[Release changes](#)

[V1.0.62](#)

[v1.0.51](#)

[v1.0.83](#)

[Note](#)

[App Key & Secret](#)

[Basic Live Topology and Flow](#)

[Integrate the SDK into your app](#)

[Download and import the SDK](#)

[Add project permissions](#)

[Add library dependencies](#)

[Remove third-party libraries containing lower version of openssl\(1.1.1l\)](#)

[Prevent code obfuscation](#)

[Initialize the SDK](#)

[Essential Guides](#)

[Use Standard Live UI](#)

[1. Start standard UI in the entrance of your live](#)

[2. Standard UI in Demo app](#)

[Add or Remove Token Pair](#)

[1. Use APIs to add or remove token pair](#)

[2. Flow chart of token pair in TVUSDKDemo](#)

[3. Token Pair UI in demo app](#)

[Enable SDK Run in Background](#)

[Support Partyline Only Mode](#)

[Advanced Guides](#)

[Listen for callback events](#)

[Customize Preview UI and Live Control](#)

[1. Custom Preview UI](#)

[2. Custom Live Control](#)

[Call TVU Voice](#)

[1. Make sure add dependency into your module gradle file](#)

[2. Use SDK APIs to reject, accept or hang up the call only in your preview activity](#)

[3. TVU Voice Note](#)

[API Index](#)

[Enum](#)

[Callback](#)

[Singleton](#)

[Life cycle](#)
[Video capture](#)
[Render](#)
[Video output](#)
[Log level](#)
[Live](#)
[Token](#)
[Environment](#)
[VoIP](#)
[PartyLine](#)
[Error Codes](#)

[Notes](#)

[Error: Multiple substitutions specified in non-positional format: did you mean to add the formatted="false" attribute?](#)

[Process "command xxx\sdk\build-tools\26.0.2\aapt.exe" finished with non- zero exit value 1](#)

[Video encoding bitrate abnormal, always 172kbps.](#)

Getting Started

Overview

Welcome to the TVU Anywhere SDK for Android!

This SDK is designed to provide you with a simple guide for integrating TVU Anywhere functionality into your Android application. TVU Anywhere is a powerful, high-quality, low-latency and robust live video transmission solution that is designed to be used with the TVU Receiver (decoder). It is ideal for many live video applications including professional / citizen journalism, remote monitoring / preview, and more. Via the TVU Receiver, it is also possible to utilize TVU Anywhere live streams with other solutions in the TVU eco system including:

- TVU Grid (Point-to-point and point-to-multipoint IP delivery)
- TVU Producer (Cloud based live production)
- TVU MediaMind (Automated, AI based metadata creation and search tool)
- TVU Command Center (Cloud based control and monitoring)

It is also possible to control general smart device camera operation (such as photo, video, flash operation, switch camera, etc.)

Release changes

V1.0.62

1. Support partyline in standard live interface
2. Support external camera

v1.0.51

1. Support scan QR code to pair with receiver in standard live interface

v1.0.83

1. Support partyline only mode

v1.0.102

1. Upgrade rtm sdk to 2.1.8
2. Upgrade libjpeg-turbo to 3.0.2

v1.0.103

1. change data from external storage to internal storage

Note

Please follow [Download and import the SDK](#) to update all to your project when you upgrade to above versions

App Key & Secret

Before integrating the SDK, you will need to apply for an app key and secret for authentication. Please contact the TVU Networks support team for assistance.

Basic Live Topology and Flow

After integration with the SDK, it will be necessary to pair your devices with a TVU Receiver in order to decode your live transmission. This is done using the Token function which is described later in this document.

Integrate the SDK into your app

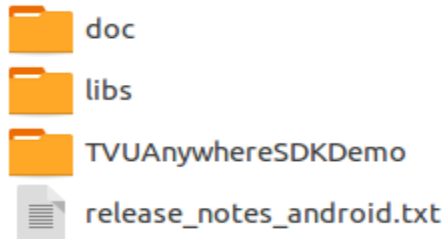
We assume that you have some experience in the development of Android Apps.

Download and import the SDK

1. Download the SDK package

Please contact TVU Support to get the latest TVU Anywhere SDK.

1.1 The package contains the following files after unzipping:



2. Import the SDK module

2.1 Open your project in **Android Studio** and Click **New->Import Module**;

2.2 In the **Import Module** panel, select the **libtvuanywhere** folder under the **TVUAnywhereSDKDemo** folder as source location and click **Finish**;

Add project permissions

Add the following permissions in the /app/src/main/AndroidManifest.xml file for device access according to your needs:

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
```

Add library dependencies

Add the following lines in the **build.gradle** file of your project:

```
...
allprojects {
    repositories {
        ...
        maven { url 'https://www.jitpack.io' }
        mavenCentral()
    }
}
...
```

Add the following lines in the **/app/build.gradle** file of your project:

```
implementation project(':libtvuanywhere')
implementation fileTree(dir: 'libs', include: ['*.jar'])
implementation 'androidx.appcompat:appcompat:1.3.0'
implementation "com.google.code.gson:gson:2.8.4"
implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
implementation("io.socket:socket.io-client:0.8.3") {
    // excluding org.json which is provided by Android
    exclude group: 'org.json', module: 'json'
}
implementation 'com.google.mlkit:barcode-scanning:16.2.0'
implementation 'org.greenrobot:eventbus:3.1.1'
implementation 'io.agora rtc:full-sdk:4.2.2'
implementation 'io.agora:agora-rtm:2.1.8'
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
implementation 'com.squareup.retrofit2:adapter-rxjava2:2.9.0'
implementation "com.squareup.okhttp3:logging-interceptor:3.11.0"
implementation 'io.reactivex.rxjava2:rxjava:2.2.6'
implementation 'io.reactivex.rxjava2:rxandroid:2.1.1'
implementation 'androidx.recyclerview:recyclerview:1.2.1'
implementation 'com.github.CymChad:BaseRecyclerViewAdapterHelper:3.0.11'
```

Remove third-party libraries containing old version of openssl(1.1.1l)

Add the following lines in the **/app/build.gradle** file of your project:

```

...
android {
    ...

    packagingOptions {
        ...
        exclude 'lib/*/libagora_drm_loader_extension.so'
        exclude 'lib/*/libagora_udrm3_extension.so'
        exclude 'lib/*/libagora_full_audio_format_extension.so'
        ...
    }
}
...

```

Prevent code obfuscation

Add the following lines in the **app/proguard-rules.pro** file to prevent obfuscating the code of the SDK:

```

# proguard for gson
-dontwarn com.google.gson.**

#webrtc proguard
-dontwarn org.webrtc.**
-keep class org.webrtc.** { *, }

# okhttp3 proguard
-dontwarn okhttp3.**
-dontwarn okio.**
-dontwarn javax.annotation.**
-dontwarn org.conscrypt.**

-keep class okhttp3.** { *; }
-keep class okio.** { *; }

# A resource is loaded with a relative path so the package of this class must be preserved.
-keepnames class okhttp3.internal.publicsuffix.PublicSuffixDatabase

# Otto proguard
-keepattributes *Annotation*
-keepclassmembers class ** {
    @com.squareup.otto.Subscribe public *;
    @com.squareup.otto.Produce public *;
}

```



```

}

# EventBus
-keepattributes *Annotation*
-keepclassmembers class * {
    @org.greenrobot.eventbus.Subscribe <methods>;
}
-keep enum org.greenrobot.eventbus.ThreadMode { *; }
-keepclassmembers class * extends org.greenrobot.eventbus.util.ThrowableFailureEvent {
    <init>(java.lang.Throwable);
}

#Agora
-keep class io.agora.**{*;}
-keep class com.tvunetworks.android.anywhere.** { *; }
-keep public class com.tvunetworks.android.engine.TVUEngine {
    public protected private *;
}

```

Initialize the SDK

The SDK Initialization is required before calling any other functions of the SDK. To initialize the SDK, you should call this API in your main activity after necessary library permissions are granted.

Following is sample code to illustrate the init flow. you can reference our demo app.

```

import com.tvunetworks.android.sdk.TVUAnywhereSDK;

public class MainActivity extends Activity {

    private static final String[] REQUIRED_PERMISSION_LIST = new
String[]{
        Manifest.permission.CAMERA,
        Manifest.permission.RECORD_AUDIO,
        Manifest.permission.READ_PHONE_STATE,
        Manifest.permission.WRITE_EXTERNAL_STORAGE,
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.ACCESS_COARSE_LOCATION,
        Manifest.permission.ACCESS_FINE_LOCATION,
    };
}

```

```

private List<String> missingPermission = new ArrayList<>();
private static final int REQUEST_PERMISSION_CODE = 0x01;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    if (!hasPermissionsGranted()) {
        checkAndRequestPermissions();
    } else {
        initTVUSDKAfterPermissionsGranted();
    }
}

private void initTVUSDKAfterPermissionsGranted() {
TVUAnywhereSDK.getInstance().tvuInitWhenAppStarted(MainActivity.this)
;
}

/**
 * Check if permissions granted
 *
 * @return true or false
 */
private boolean hasPermissionsGranted() {
    for (String permission : REQUIRED_PERMISSION_LIST) {
        if (ActivityCompat.checkSelfPermission(this, permission)
            != PackageManager.PERMISSION_GRANTED) {
            return false;
        }
    }
    return true;
}

/**
 * Checks if there is any missing permissions, and
 * requests runtime permission if needed.
 */
private void checkAndRequestPermissions() {
    // Check for permissions

```

```

        for (String eachPermission : REQUIRED_PERMISSION_LIST) {
            if (ContextCompat.checkSelfPermission(this,
eachPermission) != PackageManager.PERMISSION_GRANTED) {
                missingPermission.add(eachPermission);
            }
        }

        // Request for missing permissions
        if (!missingPermission.isEmpty() && Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
            ActivityCompat.requestPermissions(this,
                missingPermission.toArray(new
String[missingPermission.size()]),
                REQUEST_PERMISSION_CODE);
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull
String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        // Check for granted permission and remove from missing list
        if (requestCode == REQUEST_PERMISSION_CODE) {
            for (int i = grantResults.length - 1; i >= 0; i--) {
                if (grantResults[i] ==
PackageManager.PERMISSION_GRANTED) {
                    missingPermission.remove(permissions[i]);
                }
            }
        }

        // If there is enough permission, we will init sdk
        if (missingPermission.isEmpty()) {
            initTVUSDKAfterPermissionsGranted();
        } else {
            Toast.makeText(this, "Missing permissions!!!",
Toast.LENGTH_SHORT).show();
        }
    }
}

```

Optionally, you may also want to set the SDK service environment after init SDK.

Note: if you are chinese customers and use a customized live interface, may need to change the server environments to CN as shown in the following, please contact TVU support to confirm the setting.

```
TVUAnywhereSDK.getInstance().tvuSetTVUAnywhereSDKEnvironment(TVUEnvironment.  
ENVIRONMENT_CN);
```

Then, register the SDK with the App key, secret, username and password provided by TVU Networks.

```
/**  
 * Authenticate the user with the system and obtain the auth_token.  
 */  
 * @param appKey TVU service appKey which can get from TVU support with user  
account  
 * @param appSecret TVU service appSecret which can get from TVU support with user  
account  
 * @param username user account username  
 * @param password user account password  
 */  
public void registerTVUAnywhereSDKWithAppKey(String appKey, String appSecret, String  
username, String password)
```

At last, you need to release the SDK resource only once in your main activity delegate method onStop().

```
protected void onStop() {  
    super.onStop();  
    ...  
    TVUAnywhereSDK.getInstance().tvuReleaseWhenAppDestroyed(MainActivity.  
this);  
    ...  
}
```

Essential Guides

Use Standard Live UI

Within TVUAnywhereSDK, there is a standard version of the live broadcast UI that you can call and add into your project. We recommend you use this interface.

1. Start standard UI in the entrance of your live

```
private Button startDefaultPreviewBtn;  
...  
startDefaultPreviewBtn.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        TVUAnywhereAPI.getInstance().startDefaultActivity(MainActivity.this);  
    }  
});
```

2. Standard UI in Demo app



Add or Remove Token Pair

Every TVU transmission and receiving device has a unique and automatically generated PairID.

Before going live, it is necessary to add a token pair with an active TVU Receiver for receiving and decoding your stream. The token is a temporary or permanent authorization for a given transmitter (e.g. TVU Anywhere device) to be used with a specific Receiver for decoding. In order to successfully pair a TVU Anywhere transmitter and a TVU Receiver, it is necessary for acceptance on both sides by authorized operators. If a token pair is removed, it is not possible to use TVU Anywhere with the selected Receiver any longer. Please contact TVU support to assist with setup of a TVU Receiver.

We recommend scan QR code to pair with receiver in standard live interface, you can also call following API to pair with receiver manually.

Note: Below all pair ID will use unified format. (for example:c9bf391a288f2c8b)

1. Use APIs to add or remove token pair

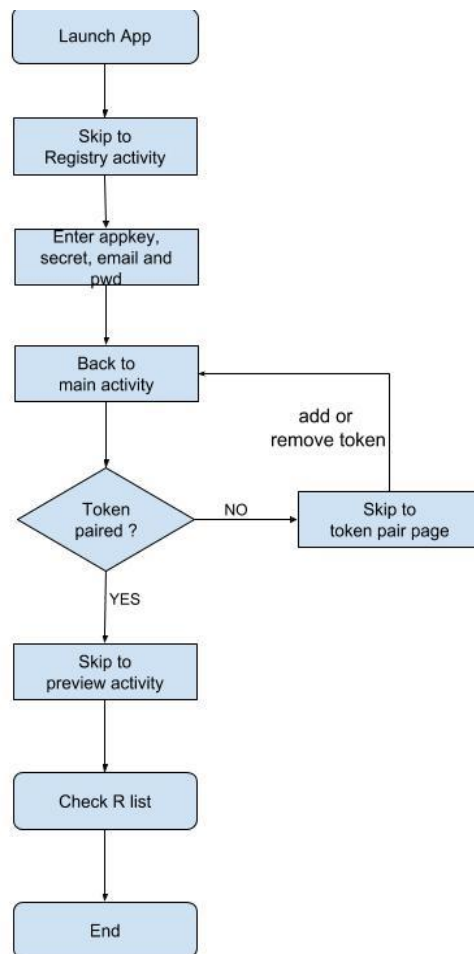
1.1 Add Token Pair

```
/**
 * Pair the device with the receivers.
 *
 * @param receivers receivers Array of the receiver ID you want to pair.
 * @param callback user service callback
 */
public void tvuAddTokenPairingWithPeerId(String[] receivers, TVUServiceCallback callback)
```

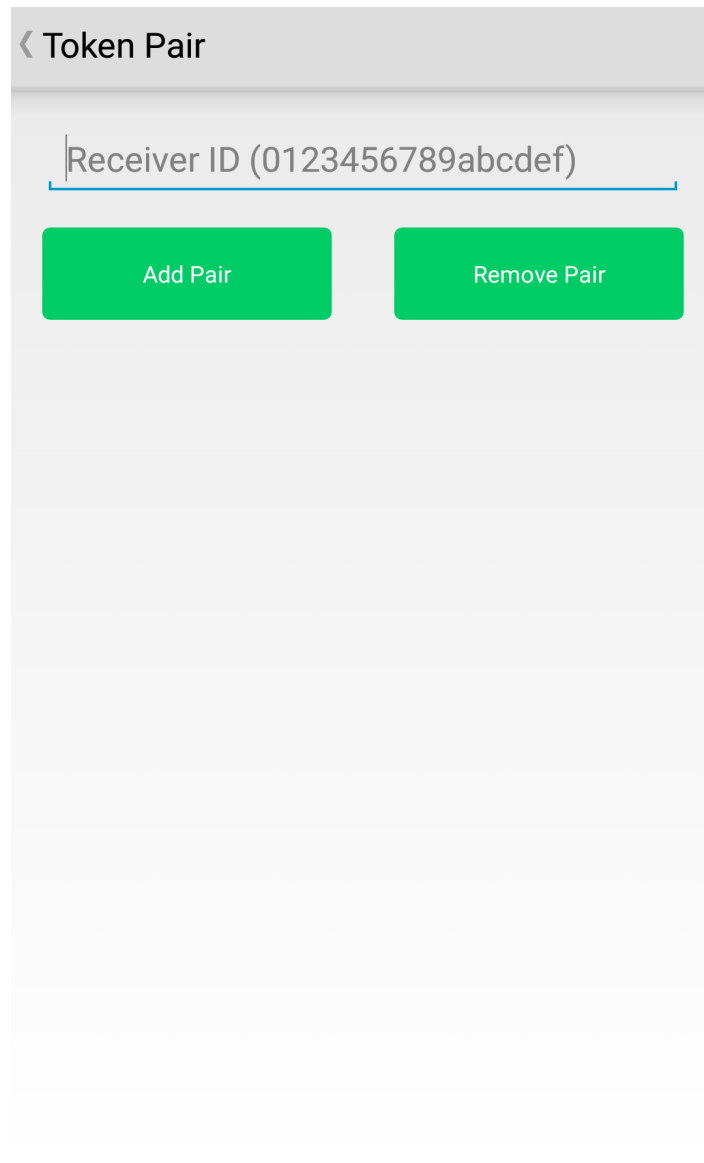
1.2 Remove Token Pair

```
/**
 * Remove pair between the device with the receivers.
 *
 * @param receivers receivers Array of the receiver ID you want to remove
 * @param callback user service callback
 */
public void tvuRemoveTokenPairingWithPeerId(String[] receivers, TVUServiceCallback callback)
```

2. Flow chart of token pair in TVUSDKDemo



3. Token Pair UI in demo app



Enable SDK Run in Background

Enable or disable SDK running under background even user lock screen or put the SDK into background.

When the flag is enabled, live, partyline will be still ongoing even user lock screen.

Suggest call it after live standard UI is loaded.

```
TVUAnywhereAPI.getInstance().startDefaultActivity(MainActivity.this);
```

```
TVUAnywhereAPI.getInstance().enableBackgroundRunning(true);
```

```
/**
 * Enable or disable SDK running under background when user lock screen or put the SDK
 into background.
 *
 * @param enable
 */
void enableBackgroundRunning(boolean enable)
```

Support Partyline Only Mode

```
/**
 * set whether to support partyline only mode or not
 *
 * @param partylineOnlyMode true: enable partyline only mode, else not
 */
public void tvuSetPartylineOnlyMode(boolean partylineOnlyMode)
```

Please call this API after init sdk to enable or disable partyline only mode in sdk.

If enabled, open the code scanning interface by default in standard live interface and you can scan QR code to join partyline directly.

Advanced Guides

Listen for callback events

The **TVUSDKCallback** allows you to subscribe to callback events that provide status updates on the operations performed in your app that are related to the SDK. For example, you might want to be notified when live operation failed.

```
/**
 * start live state
 *
 * @param liveState TVULiveState(SUCCESS, BUSY or UNKNOWN)
 */
void onStartLive(TVULiveState liveState)
```

```

/**
 * stop live state
 *
 * @param liveState TVULiveState
 */
void onStopLive(TVULiveState liveState)

/**
 * receive remote messages from msgpipe
 * @param content
 * @param length
 * @param type
 */
void onMsgReceived(String content, int length, int type)

```

The **TVUServiceCallback** allows you to subscribe to callback events that return results on the operations performed in your app that are related to the SDK. For example, you might want to be notified when operating token pair and TVU Voice.

```

/**
 * notified when operating token pair or TVU Voice
 * @param result
 */
void onResult(String result)

```

Customize Preview UI and Live Control

We have already provided a standard interface to control start or stop live with paired receivers. Optionally, you may also want to customize your own live UI and control live operation within your own app. We also provide a related API to start or stop live with paired receivers.

Note: Below all pair ID will use unify format. (eg:0123456789abcdef)

1. Custom Preview UI

1.1 Call this API when your preview activity is in the created status to do some initialization.

Note: you should provide a GLSurfaceView in your own preview layout.

```

/**
 * called when preview activity created
 *

```

```

* @param activity Activity
* @param view GLSurfaceView
* @param keepAspectRatio boolean if true, show preview as 16:9, else not
* @param callback tvu sdk callback
*/
public void tvuCreated(Activity activity, GLSurfaceView view, boolean keepAspectRatio,
TVUSDKCallback callback)

```

Optionally, you may change capture resolution and frame rate. If not, default resolution (1280x720) and frame rate (30fps) will be used.

Note: This API must be called before calling tvuResume.

```

/**
 * set preview width, height and frame rate before calling tvuResume
 *
 * @param width preview width
 * @param height preview height
 * @param fps frame rate
 */
public void tvuSetParameters(int width, int height, int fps)

```

1.2 Call this API when your preview activity is in the resumed status to start encoder and open camera.

```

/**
 * start encoder and open camera when activity is in resume status
 *
 * @param cameraId rear:0 front:1
 */
public void tvuResume(int cameraId)

```

1.3 Call this API when your preview activity is in the paused status to stop the encoder and close the camera.

```

/**
 * stop encoder and close camera when activity is in pause status
 */
public void tvuPause()

```

1.4 Call this API when your preview activity is in the destroyed status to release some resource.

```

/**
 * stop sdk running when activity is in destroy status
 */
public void tvuDestroy()

```

2. Custom Live Control

2.1 Before using APIs to start or stop live, you can register a callback to listen for start or stop live status in your onCreate method first.

2.2 Prepare to start live, you may want to use the API to know the available online receivers with which you have paired.

```

/**
 * get online receiver name list which split by comma
 *
 * @return receiver name list
 */
public String tvuGetReceiverNameList()

```

2.3 Check current live status and decide whether to start or stop live.

```

/**
 * get living receiver status
 *
 * @return status 0:standby 1:live
 */
public int tvuGetLivingReceiverStatus()

```

2.4 Now you can use APIs to start or stop live with receiver name.

```

/**
 * start live with receiver
 *
 * @param rName receiver name
 * @return -1: Receiver name is empty, 1: NTP offset is wrong, 0: success
 */
public int tvuStartLive(String rName)

/**

```

```

* stop live with receiver
*
* @param rName receiver name
* @return -1: Receiver name is empty, 1: NTP offset is wrong, 0: success
*/
public int tvuStopLive(String rName)

```

Alternatively, if you'd like to start or stop live with one receiver by its peerId, you may use the APIs below:

```

/**
* start live with receiver peerId
*
* @param peerId receiver peerId
* @return -1: receiver peerId is empty, 1: NTP offset is wrong, 0: success
*/
public int tvuStartLiveWithPeerId(String peerId)

/**
* stop live with receiver peerId
*
* @param peerId receiver peerId
* @return -1: receiver peerId is empty, 1: NTP offset is wrong, 0: success
*/
public int tvuStopLiveWithPeerId(String peerId)

```

2.5 After successfully going live, you may use APIs to get the live receiver name and bitrate info.

```

/**
* get living receiver name
*
* @return rName living receiver name
*/
public String tvuGetLivingReceiverName()

/**
* get live bitrate
*
* @return live bitrate
*/

```

```
public int tvuGetLiveBitrate()
```

Call TVU Voice

TVU Voice is a VoIP call feature developed by TVU. It enables a reliable VoIP call between TVU Anywhere device and a TVU Receiver. You can use the API to accept, reject, or hangup the TVU Voice session.

1. Make sure add dependency into your module gradle file

```
implementation "com.skyfishjy.ripplebackground:library:1.0.1"
implementation("io.socket:socket.io-client:0.8.3") {
    // excluding org.json which is provided by Android
    exclude group: 'org.json', module: 'json'
}
```

2. Use SDK APIs to reject, accept or hang up the call *only in your preview activity*

```
/**
 * set callback to listen for coming TVU Voice
 */
@param callback TVUServiceCallback
*/
public void setTVUVoiceComingCallback(final TVUServiceCallback callback)

/**
 * accept TVU Voice single call
 */
public void tvuAcceptTVUVoice()

/**
 * reject TVU Voice single call
 */
public void tvuRejectTVUVoice()

/**
 * hangup TVU Voice single call
 */
```

```

public void tvuHangupTVUVoice()

/**
 * Whether the app is currently in the calling status.
 *
 * @return true: App is voicing; false: App is not voicing.
 */
public boolean tvuIsTVUVoicing()

/**
 * query login or voice call status
 *
 * @return login or voice call status
 */
public String tvuQueryVoiceCallStatus()

```

3. TVU Voice Note

This function is available only using the live view, and can only accept TVU Voice calledcalls from the remote side such as a TVU Receiver or Command Center.

In the following cases, the SDK will reject a new TVU Voice call:

- 1) not on the live view
- 2) TVU Voice connection is already active

API Index

Enum

```

enum TVUEnvironment {
    ENVIRONMENT_US(0), // United States environment
    ENVIRONMENT_CN(1), // China environment
    ENVIRONMENT_RD(2), // research department environment
    ENVIRONMENT_QA(3), // QA department environment
    ENVIRONMENT_ALI(4), // Ali environment
    ENVIRONMENT_CCTV(5); // CCTV environment

```

```

}

enum TVULiveState {
    /**
     * start or stop live success
     */
    SUCCESS(0),
    /**
     * receiver is living with other T
     */
    BUSY(1),
    /**
     * other unknown type
     */
    UNKNOWN(255);
}

enum TVULogLevel {
    LOG_LEVEL_FATAL(0),
    LOG_LEVEL_ERROR(1),
    LOG_LEVEL_WARN(2),
    LOG_LEVEL_INFO(3),
    LOG_LEVEL_DEBUG(4);
}

```

Callback

```

interface TVUSDKCallback {
    /**
     * start live
     *
     * @param liveState TVULiveState
     */
    void onStartLive(TVULiveState liveState);

    /**
     * stop live
     *
     * @param liveState TVULiveState
     */
    void onStopLive(TVULiveState liveState);
}

```



```

/**
 * receive remote messages from msgpipe
 * @param content
 * @param length
 * @param type
 */
void onMsgReceived(String content, int length, int type);
}

interface TVUServiceCallback {
/**
 *
 * @param result
 */
void onResult(String result);
}

```

Singleton

```

/**
 * create a single instance of TVUAnywhereSDK
 *
 * @return TVUAnywhereSDK instance
 */
public static TVUAnywhereSDK getInstance()

```

Life cycle

```

/**
 * init sdk when app process started
 *
 * @param context Context
 */
public void tvuInitWhenAppStarted(Context context)

/**
 * release sdk when app process will destroy
 *
 * @param context
 */

```

```

public void tvuReleaseWhenAppDestroyed(Context context)
/**
 * called when activity created
 *
 * @param activity    Activity
 * @param view        GLSurfaceView
 * @param keepAspectRatio boolean if true, show preview as 16:9, else not
 * @param callback    tvu sdk callback
 */
public void tvuCreated(Activity activity, GLSurfaceView view, boolean keepAspectRatio,
TVUSDKCallback callback)

/**
 * start encoder and open camera when activity is in resume status
 *
 * @param cameraId rear:0 front:1
 */
public void tvuResume(int cameraId)

/**
 * stop encoder and close camera when activity is in pause status
 */
public void tvuPause()

/**
 * stop sdk running when activity is in destroy status
 */
public void tvuDestroy()

```

Video capture

```

/**
 * switch camera status and restart encoder
 *
 * @param cameraId rear:0 front:1
 * @return if not found, return false; else, return true
 */
public boolean tvuSwitchCamera(int cameraId)

/**
 * get camera resolution list

```

```

*
* @return resolution list
*/
public String[] tvuGetResolutionList()

/**
* get camera frame rate list
*
* @return frame rate list
*/
public int[] tvuGetFrameRateList()

/**
* turn flash light on or off
*
* @param isFlashOn current flash status
* @return if not supported, return false; else, return true
*/
public boolean tvuOperateFlash(boolean isFlashOn)

```

Render

```

/**
* take picture and save it to the gallery
*/
public void tvuTakePicture()

```

Video output

```

/**
* set preview width, height and frame rate
*
* @param width preview width
* @param height preview height
* @param fps frame rate
*/
public void tvuSetParameters(int width, int height, int fps)

```

Log level

```

/**

```

```
* set sdk log level  
*/  
public void tvuSetAppLogLevel(TVULogLevel logLevel)
```

Live

```
/**  
* get peer id of device which is unique  
*  
* @return peer id  
*/  
public String tvuGetMyPeerID()  
  
/**  
* get living receiver name  
*  
* @return rName living receiver name  
*/  
public String tvuGetLivingReceiverName()  
  
/**  
* get living receiver status  
*  
* @return status 0:standby 1:live  
*/  
public int tvuGetLivingReceiverStatus()  
  
/**  
* start live with receiver  
*  
* @param rName receiver name  
* @return -1: Receiver name is empty, 1: NTP offset is wrong, 0: success  
*/  
public int tvuStartLive(String rName)  
  
/**  
* stop live with receiver  
*  
* @param rName receiver name  
* @return -1: Receiver name is empty, 1: NTP offset is wrong, 0: success  
*/
```

```

public int tvuStopLive(String rName)

/**
 * get online receiver list which split by comma
 *
 * @return receiver name list
 */
public String tvuGetReceiverNameList()

```

Token

```

/**
 * Authenticate the user with the system and obtain the auth_token.
 *
 * @param appKey TVU service appKey which can get from TVU support with user
account
 * @param appSecret TVU service appSecret which can get from TVU support with user
account
 * @param username user account username
 * @param password user account password
 */
public void registerTVUAnywhereSDKWithAppKey(String appKey, String appSecret, String
username, String password)

/**
 * Pair the pack with the receivers.
 *
 * @param receivers receivers Array of the receiver ID you want to pair.
 * @param callback user service callback
 */
public void tvuAddTokenPairingWithPeerId(String[] receivers, TVUServiceCallback callback)

/**
 * Remove pair the pack with the receivers.
 *
 * @param receivers receivers Array of the receiver ID you want to remove
 * @param callback user service callback
 */
public void tvuRemoveTokenPairingWithPeerId(String[] receivers, TVUServiceCallback
callback)

```

Environment

```
/**
 * Setting TVUAnywhere SDK environment
 *
 * @param environment SDK environment
 */
public void tvuSetTVUAnywhereSDKEnvironment(TVUEnvironment environment)
```

VoIP

```
/**
 * set callback to listen for coming TVU Voice
 *
 * @param callback TVUServiceCallback
 */
public void setTVUVoiceComingCallback(final TVUServiceCallback callback)

/**
 * Whether the app is currently in the calling status.
 *
 * @return true: App is voicing; false: App is not voicing.
 */
public boolean tvuIsTVUVoicing()

/**
 * accept TVU Voice single call
 */
public void tvuAcceptTVUVoice()

/**
 * reject TVU Voice single call
 */
public void tvuRejectTVUVoice()

/**
 * hangup TVU Voice single call
 */
public void tvuHangupTVUVoice()
```

```

/**
 * query login or voice call status
 *
 * @return login or voice call status
 */
public String tvuQueryVoiceCallStatus()

```

PartyLine

```

/**
 * set whether to support partyline only mode or not
 *
 * @param partylineOnlyMode true: enable partyline only mode, else not
 */
public void tvuSetPartylineOnlyMode(boolean partylineOnlyMode)

/**
 * get current partyline only mode
 *
 * @return partyline only mode, default false
 */
public boolean tvuGetPartylineOnlyMode()

```

Error Codes

N/A

Notes

Error: Multiple substitutions specified in non-positional format; did you mean to add the formatted="false" attribute?

That is because String resource has multiple %s or similar in SDK. To avoid this, you can override them and identify each like this: %1\$s in your app. Example:

```
<string name="sdk_string">First: %1$s - Last: %2$s</string>
```

Where %1\$s is the first substitution and %2\$s is the second.

If you do not mean to perform any substitution, just add the attribute formatted="false".

Example:

```
<string name=sdk_string" formatted="false">Level: 100%</string>
```

Process "command

xxx\sdk\build-tools\26.0.2\aapt.exe" finished with non-zero exit value 1

Please refer to sdk demo and keep up with demo.

Video encoding bitrate abnormal, always 172kbps.

We found the issue on huawei mate 30 and app build with 32bit format.

Solution: build app support both 32bit and 64bit platform.