

Video++互动层Android SDK对接指南

Video++ 互动层目前支持所支持的业务为：

- * 直播互动（请重点参照步骤1.X,2.X,3.X,8.X)
- * 点播互动（请重点参照步骤1.X,2.X,4.X,8.X)
- * OTT互动（请重点参照步骤1.X,2.X,5.X,8.X)
- * Video++商城（请重点参照步骤1.X,2.X,6.X,8.X)
- * 互动娱乐（请重点参照步骤1.X,2.X,7.X,8.X)

1. 对接方式说明：

1.1 SDK配置的demo代码[点击此处](#)，该Demo上面有对接的详细代码可供参考使用

- * 直播互动 Live0sActivity.java
- * 点播互动 Video0sActivity.java
- * OTT互动 VideoOTTActivity.java
- * Video++商城 Mall0sActivity.java
- * 互动娱乐 Huyu0sActivity.java

1.2.除了参考上面的Demo中的配置外，还可以参考下文“对接步骤说明”

2. 对接步骤说明：

2.1 在appliaction的onCreate方法里进行如下配置：

```
Application {  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
        VenvyUIUtil.runOnUIThreadDelay(new Runnable() {  
            @Override  
            public void run() {  
                // VideoType 中标识本次接入业务类型，具体接入请根据实际的业务需求  
  
                // VideoType.BOTH 表示接入点播和直播  
                // VideoType.LIVE0S 表示只接入直播  
                // VideoType.VIDE00S 表示只接入点播  
                // VideoType.OTT 表示只接入OTT  
                // VideoType.MALL 表示只接入子商城  
                // VideoType.HUYU 表示只接入互动娱乐  
                VideoPlus.appCreate(MyApp.this, VideoPlus.VideoType.BOTH);  
            }  
        }, 3000);  
    }  
}
```

```
}
```

2.2 在项目 build.gradle 中配置相关的compile.

如果您使用远程依赖

```
dependencies {
    compile fileTree(dir: x'libs', include: ['*.jar'])
    //Video++ 互动层依赖:(版本号只是举例, 具体版本请咨询对接技术同学)
    compile 'com.videoli:video_pub:1.5.0.4'
    compile "com.github.bumptech.glide:glide:YOU_VERSION" // glide
    compile "com.squareup.okhttp3:okhttp:YOU_VERSION" // okhttp
    compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.0.2' //
mqtt 长链接
    //接入互娱以及子商城时需要另外添加依赖
    compile 'com.videoli:venvy_lua:1.0.0'
    //接入子商城(横屏)时需要另外添加依赖
    compile 'com.android.support:recyclerview-v7:25+'
}
```

如果您选择aar包方式依赖

```
dependencies {
    compile fileTree(dir: x'libs', include: ['*.jar'])
    //Video++ 互动层依赖:(版本号只是举例, 具体版本以及aar包请咨询对接技术同学)
    compile(name: 'venvy_pub-release-2.0.0', ext: 'aar')
    compile "com.github.bumptech.glide:glide:YOU_VERSION" // glide
    compile "com.squareup.okhttp3:okhttp:YOU_VERSION" // okhttp
    compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.0.2' //
mqtt 长链接
    //接入互娱以及子商城时需要另外添加依赖
    compile 'com.videoli:venvy_lua:1.0.0'
    //接入子商城(横屏)时需要另外添加依赖
    compile 'com.android.support:recyclerview-v7:25+'
}
```

2.3 在AndroidManifest.xml中配置如下权限:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

3.直播互动配置步骤:

3.1 直播互动层创建

在xml中配置互动层的VideoLiveView:

```
<cn.com.videopls.pub.live.VideoLiveView
    android:id="@+id/venvyLive"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

在代码中找到控件

```
VideoLiveView videoLiveView = (VideoLiveView) findViewById(R.id.venvyLive);
```

如果您希望在代码中手动创建直播的互动层

```
VideoLiveView videoLiveView= new VideoLiveView(context);
videoLiveView.setLayoutParams(new LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT, ViewGroup.LayoutParams.MATCH_PARENT));
addview(videoLiveView);
```

3.2 初始化 Adapter

创建adapter

```
private class LiveAdapter extends VideoPlusAdapter {

    @Override
    public Provider createProvider() {
        final int width = VenvyUIUtil.getScreenWidth(LiveActivity.this);
        final int height =VenvyUIUtil.getScreenHeight(LiveActivity.this);

        Provider provider = new Provider.Builder()
            .setHorVideoHeight(height)//横屏视频的高
            .setHorVideoWidth(width)//横屏视频的宽
            .setVerVideoHeight(screenHeightSmall)//small 视频小屏
            视频的高
            .setVerVideoWidth(width)//small视频小屏视频的宽
            .setVerticalFullVideoWidth(height)//Full 视频全屏视频
```


的高

```
        .setVerticalFullVideoHeight(width)//视频全屏视频的宽  
        .setVerticalType(1)//1 竖屏小屏, 0竖屏全屏  
        .setDirection(2) //2横竖屏, 0竖屏, 1是横屏  
        .build();  
        return provider;  
    }  
  
}
```

将该步骤中的Adapter对象设置给VideoLiveView:

```
liveAdatper = new LiveAdapter();  
videoLiveView.setVideo0SAdapter(liveAdatper);
```

3.3 直播互动开启

```
videoLiveView.start();
```

3.4 直播互动关闭

当页面(Activity)销毁的时候, 需要在onDestory()方法里面销毁直播互动

```
protected void onDestroy() {  
    super.onDestroy();  
    //销毁直播互动  
    videoLiveView.destroy();  
}
```

3.5 传入用户信息

```
@(文档)Override  
public IPlatformLoginInterface buildLoginInterface() {  
    return new PlatFormUserInfoImpl() {  
        @Override  
        public PlatformUserInfo getLoginUser() {  
            PlatformUserInfo userInfo = new PlatformUserInfo();  
            // 房间号 (必要)  
            userInfo.roomId = "";  
            // platformId (必要)  
            userInfo.platformId = "";  
            // 设置用户ID
```

```

        userInfo.uid = "";
        // 设置分区, e.g. lol, hearthstone, dota1 ...
        userInfo.cate = "";
        // 设置用户名称
        userInfo.userName = "";
        // 是否为主播端
        userInfo.isAnchor = true;
        // 是否正在推流
        userInfo.isPush = true;
        return userInfo;
    }
};
}

```

3.6 中插视频广告事件监听

在 adapter 中重写 buildWedgeListener(), 返回 WedgeListener 对象, 选择需要的方法重写

```

/**
 * 中插开始播放
 */
void onStart();

/**
 * 中插播放完毕通知, 播放完毕分3种状态:
 * 1: 中插视频自然播放到指定时间长度
 * 2: 中插在播放期间由后台广告管理撤销当前中插的播放, 导致当前中插页面关闭
 * 3: 中插在播放时间, 配置的展示时间段过期
 */
void onFinish();

/**
 * 没有中插广告
 */
void onEmpty();

/**
 * 点击中插返回按钮
 */
void goBack();

/**
 * 恢复播放
 */
void onResume();

/**
 * 暂停播放
 */

```

```

void onPause();

/**
 *
 * @return 从后台返回后是否需要显示中插
 */
boolean needShowVideo();

/**
 *
 * @return 设置中插缓存的最大值 单位 : M
 */
boolean downloadSwitch();

/**
 *
 * @return 移动网络是否下载视频 true:开启下载, false 关闭下载
 */
int maxCacheSize();

```

以上是直播接入步骤，如果有不清楚的地方可以咨询对接的同学

4.点播互动配置步骤:

4.1 在xml中配置互动层的VideoOsView:

```

<cn.com.videopls.pub.os.VideoOsView
    android:id="@+id/video"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

```

4.2 初始化点播互动层VideoOsView对象的Adapter:

```

private class MyAdapter extends VideoPlusAdapter {

    @Override
    public Provider createProvider() {

        final int width = VenvyUIUtil.getScreenWidth(MainActivity.this);
        final int height = VenvyUIUtil.getScreenHeight(MainActivity.this);

        Provider provider = new Provider.Builder()
            .setAppKey(getAppKey())//appkey
            .setHorVideoHeight(Math.min(width, height))//横屏视频的高
            .setHorVideoWidth(Math.max(width, height))//横屏视频的宽
    }
}

```

```

        的宽
        .setVerVideoWidth(Math.min(width, height))//small视
        频小屏视频的宽
        .setVerVideoHeight(screenHeightSmall)//small 视频小屏
        视频的高
        .setVideoPath(getVideoPath())//视频地址
        .setVideoType(3)//
        .setVideoTitle("ttt")//
        .build();
        return provider;
    }

    @Override
    public IMediaControllListener buildMediaController() {
        return new MyMediaController();
    }
}

```

由于点播互动需要获得视频的播放时间，所以在接入点播互动构造Adapter的时候 buildMediaController () 方法必须重载，其中getCurrentPosition方法为获取当前播放器时间，单位为毫秒级。其他方法建议实现（比如start(),pause()等等，方便以后我们SDK做扩展，我们目前没有对接入方的播放器做控制。

4.3 将4.2的Adapter对象设置给VideoOsView:

```

videoOsView = (VideoOsView) findViewById(R.id.video);
adapter = new MyAdapter();
videoOsView.setVideoOAdapter(adapter);
adapter.onCreate();

```

4.4 在页面Activity的生命周期中调用点播Adapter对应的生命周期: (如果您有查看我们的demo，请关注下点播页面VideoOsActivity的父类BasePlayerActivity的生命周期调用)

```

@Override
protected void onStart() {
    super.onStart();
    //开启点播互动
    adapter.onStart();
}

@Override
protected void onPause() {
    super.onPause();
    //暂停点播互动
}

```



```

        adapter.onPause();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        //销毁点播互动
        adapter.onDestroy();
    }

    @Override
    protected void onResume() {
        super.onResume();
        //点播互动resume
        adapter.onResume();
    }
}

```

5.OTT互动配置步骤

OTT互动的配置与点播基本相同（请详细参照步骤4.X以及DEMO代码VideoOTTActivity），请将VideoOsView替换为VideoOTTView

6.添加子商城互动

6.1 横屏幕状态下子商城与app交互接口

6.1.1 在xml布局文件中添加子商城布局

```

<cn.com.videopls.pub.mall.VideoMallView
    android:id="@+id/venvyMall"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

```

6.1.2 初始化VideoMallView并配置adapter

```

videoMallView = (VideoMallView) findViewById(R.id.venvyMall);
mallAdapter = new MallAdapter();
videoMallView.setVideoOSAdapter(mallAdapter);
videoMallView.start();

```

6.1.3 初始化adapter

```

private class MallAdapter extends VideoPlusAdapter {
    @Override

```



```

        public Provider createProvider() {
            Provider provider = new Provider.Builder()
                .setUserId("34")//roomId 或者userId
                .setPlatformId("56dd27a8b311dff60073e645")//videojj
直播后台平台Id : 575e6e087c395e0501980c89
                .build();
            return provider;
        }

        @Override
        public IPlatformLoginInterface buildLoginInterface() {
            return iPlatformLoginInterface;
        }

        //横屏下关闭子商城页面回调
        @Override
        public IWidgetCloseListener<WidgetInfo>
        buildWidgetCloseListener() {
            return new IWidgetCloseListener<WidgetInfo>() {
                @Override
                public void onClose(WidgetInfo widgetInfo) {
                    WidgetInfo.WidgetType widgetType = widgetInfo.getWidgetType();

                    if (widgetType == WidgetInfo.WidgetType.MAILL) {
                        mallBtn.setVisibility(View.VISIBLE);
                    }
                }
            };
        }
    }
}

```

6.1.4 上一步骤中代码中iPlatformLoginInterface对象的具体实现说明

子商城需要与第三方app交互来交换各自所需信息：子商城需要第三方app用户信息、子商城h5登陆时通知第三方app、横屏子商城点击付款需要转竖屏通知第三方app等。这些交互信息通过在直播里面Adapter重写buildLoginInterface接口来完成

```

private IPlatformLoginInterface mPlatformLoginInterface = new IPlatformLoginInterface() {

    /**
     *第三方app提供给v++ SDK的登陆信息，其中UID为必填写
     */
    @Override
    public PlatformUserInfo getLoginUser() {
        PlatformUserInfo platformUserInfo = new PlatformUserInfo();
        //用户唯一标志
    }
}

```

```

        platformUserInfo.setUid("必填");
        //用户昵称
        platformUserInfo.setNickName("nullable");
        //用户名
        platformUserInfo.setUserName("nullable");
        //电话号码
        platformUserInfo.setPhoneNum("nullable");
        //token信息
        platformUserInfo.setUserToken("nullable");
        return platformUserInfo;
    }

    //商城h5页面用户登陆，回调给第三方app
    @Override
    public void userLoggedIn(PlatformUserInfo userInfo) {
        //userInfo对象提供来getXX来获取对应的信息
    }

    //h5 点击立即购买等需要转竖屏交互时通知app
    @Override
    public void screenChanged(changedInfo changedInfo) {
        //url为小屏幕URL
    }

};

/**
 * adapter
 */
@Override
public IPlatformLoginInterface buildLoginInterface() {
    return mPlatformLoginInterface;
}

```

6.1.5 打开子商城页面的方式

点击子商城入口（子商城入口需要第三方app自己配置),当点击入口按钮呼出子商城之后，入口需要隐藏。

```

doorBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //点击响应路由，呼出子商城
        Router.route(Scheme.MAILL);

        //隐藏商城入口按钮
        doorBtn.setVisibility(View.GONE);
    }
}

```

```
    }
    });
```

6.1.6 关闭子商城回调

当子商城页面关闭的时候需要通知第三方，第三方在接到回调通知后将入口显示，具体的回调在Adapter里面配置，其代码如下：

```
@Override
    public IWidgetCloseListener<WidgetInfo> buildWidgetCloseListener() {
        return new IWidgetCloseListener<WidgetInfo>() {
            @Override
            public void onClose(WidgetInfo widgetInfo) {
                WidgetInfo.WidgetType widgetType = widgetInfo.getWidgetType();
                if(widgetType == WidgetInfo.WidgetType.MAILL) { //子商城页面关闭
                    //显示子商城入口
                    doorBtn.setVisibility(View.VISIBLE);
                }
            }
        }
    }
```

6.2 竖屏小屏幕子商城配置：

6.2.1在竖屏小屏幕状态下，第三方需要在播放器下面配置SDK提供的MallWebView布局页面(其具体类路径为cn.com.venvy.mall.view.MallWebView) ,用来打开子商城h5页面，另外初始化WebView完毕后需要设置如下配置：

```
//也可以在xml里配置
MallWebView mallWebView = new MallWebView(context);
//步骤1创建的IPlatformLoginInterface对象
mallWebView.setPlatformLoginInterface(mPlatformLoginInterface);
```

6.2.2 注意在竖屏状态下响应screenChanged里面添加如下代码：

```
public void screenChanged(changedInfo changedInfo) {
    //url为小屏幕URL
    mallWebView.setSsId(changedInfo.getSsId());
    mallWebView.setVisibility(View.VISIBLE);
}
```



```
mallWebView.loadUrl(changedInfo.getUrl());  
}
```

6.2.3 在竖屏点击入口，打开MallWebView调用loadUrl方法来获取商品列表：

```
//获取商品货架的URL  
String mallUrl=MallConfig.getMallUrl();  
mallWebView.setSsId(System.currentTimeMillis()+"");  
//拼接第三方房间号  
mallWebView.loadUrl(mallUrl+"?video="+ 第三方app的房间ID)
```

6.3 我的订单

根据产品定义，需要在第三方app的用户信息页面配置“我的订单”选项，当用户点击此选项的时候业务逻辑如下：

- 1、判断第三方app的用户是否登录，如果没有登录则跳转到第三方的登录页面，进行用户的登录操作。
- 2、跳转独立的activity，该activity的contentView为全屏竖屏的VenvyWebView，该activity的配置示例代码请参考Demo中MyOrderActivity。

6.4 Webview的初始化工作

因为sdk的webView用的是腾讯的x5webView,所以对其初始化需在Appcation中做如下配置：所以需要libs文件配置x5WebView的jar，同时需要在src/jniLibs文件放x5的so库，x5jar包[点击此处](#)

```
MyApp extends Application {  
    @Override  
    public void onCreate() {  
        super.onCreate();  
        VideoPlus.appCreate(MyApp.this,VideoPlus.VideoType.MALL);  
    }  
}
```

7.互娱配置步骤

相关 View 的设置同直播，区别在 Adapter 上

```
class MyAdapter extends EnjoyAdapter {
```



```

@Override
public IPlatformLoginInterface buildLoginInterface() {
    return new PlatFormUserInfoImpl() {
        @Override
        public PlatformUserInfo getLoginUser() {
            PlatformUserInfo platformUserInfo = new PlatformUserInf
o();

            // 是否为主播
            platformUserInfo.isAnchor = true;
            // 是否正在推流
            platformUserInfo.isPush = true;
            // 用户 id
            platformUserInfo.uid = "";
            // 房间号
            platformUserInfo.roomId = "";
            // 平台 id
            platformUserInfo.platformId = "";
            return platformUserInfo;
        }
    };
}
}

```

8.互动的相关监听方法:

如果您想要对互动层上面所展示的广告行为做监听，比如展示、点击、关闭，则可以按照需求复写Adapter的buildWidgetShowListener、buildWidgetClickListener、buildWidgetCloseListener方法，如下所示：

```

public IWidgetClickListener<WidgetInfo> buildWidgetClickListener() {
    return new IWidgetClickListener<WidgetInfo>() {

        @Override
        public void onClick(@Nullable WidgetInfo widgetInfo) {
            //点击监听
        }
    };
}

public IWidgetShowListener<WidgetInfo> buildWidgetShowListener(
) {
    return new IWidgetShowListener<WidgetInfo>() {
        @Override
        public void onShow(WidgetInfo widgetInfo) {
            //展示监听
        }
    };
}
}

```

```

        public IWidgetCloseListener<WidgetInfo> buildWidgetCloseListene
r() {
            return new IWidgetCloseListener<WidgetInfo>() {
                @Override
                public void onClose(WidgetInfo widgetInfo) {
                    //关闭监听
                }
            };
        }
    }
}

```

通过 `widgetInfo.getWidgetType()` 获取广告的类型

```

public enum WidgetType {
    TAG, //热点类型
    CLOUD,
    INFO, //link, 或者信息层
    VIDEOCLIP, //中插
    NEWS, //新闻
    LOTTERY, //抽奖
    BUBBLE, //气泡
    EASYSHOP, //轻松购
    BASICWIKI, //百科
    BASICMIX, //轮播广告
    ADINFO, //海报
    VOTE, //投票
    COUPONGIFT, //优惠券红包
    CODEGIFT, //口令红包
    ADGIFT, //视频红包 (二维码红包)
    CARDGAME, //卡牌
    GIFT, //红包
    TEXT, //文字链
    GOODS, //商品
    PRAISE, //点赞
    PIC, //图片广告
    MAILL, //子商城
}

```

注意事项:

Video++互动层对于三方的引入，为了避免冲突和重复引入，导入三方功能开放给最外层调用，如项目中已配置，可按照项目中原有配置来，如没配置相关依赖，需要在项目中添加。具体依赖可看build.gradle文件注释说明。

SDK 必须的依赖配置

```

compile "com.github.bumptech.glide:glide:YOU_VERSION" // glide

```

```
compile "com.squareup.okhttp3:okhttp:YOU_VERSION" // okhttp
compile 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.0.2' // mqtt
t 长链接
```

V1.2.0 版本升级内容

1. android支持远程依赖接入
2. android直播优化内存占用
3. android点播优化内存占用
4. android点播横竖屏切换曝光投递优化

V1.2.0.3 版本升级内容

5. 增加中间层Provider动态更新功能
6. 增加图片下载成功监控

v1.2.0.4 版本升级内容

7. 将SDK编译版本下调，同时将依赖项目暴露给调用方
8. Fix 点播读取文件BUG

V1.2.0.6 版本升级内容

9. fix 部分机型上闪退的bug

v1.2.0.7版本内容

10. 修复config接口中返回的对外链处理