Video++互动层Android SDK对接指南

Video++ 互动层目前支持直播和点播功能的同时集成。其中具体集成直播、点播、或者直播点播融合的版本请先和商务同学确认。

对接方式说明：

1.SDK配置的demo代码点击此处，该Demo上面有直播(demo中LiveActivity.java)和点播(MainActivity.java)对接的详细代码可供参考使用
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”

对接步骤说明：

1. 在项目 build.gradle 中配置相关的compile.

```
dependencies {
    compile fileTree(dir: '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'

}
```

2.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" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

3. 互动层初始化

   在Android的Application的onCrate方法里面配置如下代码：

```
 @Override
    public void onCreate() {
        super.onCreate();
        VenvyUIUtil.runOnUIThreadDelay(new Runnable() {
            @Override
            public void run() {
                VideoPlus.appCreate(MyApp.this, VideoPlus.VideoType.BOTH);
            }
        }, 3000);
    }
```

其中VideoType是枚举类型，具体种类如下表所示

类别	含义
LIVEOS	表示只对接了直播互动功能
VIDEOOS	表示只对接了点播互动功能
OTT	表示只对接了OTT互动功能
BOTH	表示对接了LIVEOS和VIDEOOS(不包括OTT)

4.直播互动配置步骤：

4.1 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" />
```

4.2 初始化直播互动层VideoLiveView对象的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()
                    .setUserId(mRoomId)//roomId 或者userId
                    .setPlatformId(getPlatformId())//videojj直播后台平台Id
                    .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)//视频全屏视屏的宽
                    .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
                    .setIsMango()
                    .build();
            return provider;
        }

    }

```

4.3 将4.2的Adapter对象设置给VideoLiveView：

```
        videoLiveView = (VideoLiveView) findViewById(R.id.venvyLive);
        liveAdatper = new LiveAdapter();
        videoLiveView.setVideoOSAdapter(liveAdatper);
```

4.4 直播互动开启：

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

4.5 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();
  }
```

5. 点播互动配置步骤

5.1 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" />
```

5.2 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
                    .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 IMediaControlListener buildMediaController() {
            return new MyMediaController();
        }
}

```

与直播配置Adapter不用的是，在点播构造Adapter的时候buildMediaController（ ）方法必须重载，此方法返回接口为控制播放器行为。其中getCurrentPosition方法为获取当前播放器时间，单位为毫秒级。

5.3 将5.2的Adapter对象设置给VideoOsView：

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

5.4 在页面Activity的生命周期中调用点播Adapter对应的生命周期：

```
@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();
        //销毁调拨互动
        adapter.onDestroy();
    }

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

6 互动的展示、点击、关闭监听：
如果客户端想要对互动层的展示、点击、关闭做监听可以复写Adapter的buildWidgeShowListener、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> buildWidgetCloseListener() {
            return new IWidgetCloseListener<WidgetInfo>() {
                @Override
                public void onClose(WidgetInfo widgetInfo) {
                    //关闭监听
                }
            };
        }
```

另外直播互动层根据个别对接方需求，新增了监听直播间没有互动层的回调接口：IWidgetEmptyListener,可以复写Adapter的buildWidgetEmptyListener:

```
  @Override
        public IWidgetEmptyListener buildWidgetEmptyListener() {
            return new IWidgetEmptyListener() {
                @Override
                public void onEmpty() {
                    //表示刚进入直播间的时候，该直播间没有互动广告
                }
            };
        }
```

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.

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

v1.3.2.0版本升级内容

11. 对于V1.2.0.x版本接入方，外部三方引入依赖方式有稍微变化，不需要再build.gradle中写除了vido++依赖以外的三方complie依赖，需要在gradle.properties文件中指定三方依赖版本（非常重要）
12. 增强android广告缓存，加载速度优化

v1.3.4.20 版本升级内容（2017.07.06）

13. 将默认图片框架改为Glide，如果对接上需要Fresco 作为图片框架，请联系技术同学。
14. 因为图片框架变更的影响，低于v1.3.4.x以下版本的对接方请留意gradle.properties文件里面的配置。具体可参考demo项目的配置写法（非常重要）。
15. 请在项目application里调用VideoPlus.appCreate()方法。（非常重要，V1.3.4.x及以上版本必须需要调用)
16. fix 曝光成功率问题
17. 修复中插bug

v1.3.6.0 版本升级内容（2017.07.14）

18. 增加热点点击事件回调

19. fix bug

    v1.5.0.6 版本升级内容（2017.08.04）
20. 新增热点展示、点击、关闭监听。
21. 直播新版投票样式。
22. 直播商品列表
23. fix bug

    v1.6.0.2版本升级内容(2017.08.07)

    1.修改返回首页后中插倒计时从头开始的bug
    2.直播点播互动层添加“广告”和“关闭”按钮标识
    3.解决部分机型卡顿问题
    4.直播新增IWidgetEmptyListener回调