弹幕 SDK 及 DEMO 使用说明

一 demo

流程

1. 首先需要用自己的账户密码登录
2. 选择测试接口, 则进入接口测试界面
3. 选择界面测试, 进入弹幕界面测试
    - 弹幕界面测试仅包含收弹幕, 不包含发弹幕
    - 在操作过程中暂无操作提示
    - 点播channelId可以选择 25535757, 直播可以尝试300451.
    - 只有点击了"开始播放" 才会开始接收弹幕
    - 20170613 增加了透明度调整和全屏/半屏功能
    - 20170614 增加了弹幕显示半屏, 以及无遮挡功能

二 SDK

0. 接口返回代码

类型	解释
httpStatus	200 正确，201 创建成功,401 权限验证失败, 422 :请求非法, 404 未找到，500系统错误
body	err有0,401,404,422,500.分别表示成功、权限失败、NotFound、请求非法、系统

1. 初始化

所有与弹幕相关操作的类为`DanmakuSDK`, 而其初始化工具是`DanmakuInitializer`类. 初始化方法示例如下. 集成时, 按照实际情况更改:

```
//初始化, context, 当前平台枚举, 当前软件版本号
```

```
DanmakuInitializer.begin(this, DanmakuSDK.Platform.Phone, "5.0.0")
                .complete();
```

弹幕 SDK 初始化建议放在`Application.onCreate`中.

将`push-sdk-v1.0.2.aar`和`danmaku-1.0.aar`放入相应目录后, 在`build.gradle`中, 加入如下依赖:

```
compile(name: 'pppush-sdk-v1.0.2', ext: 'aar')
compile(name: 'danmaku-1.0', ext: 'aar')
```

2. 调用方法

2.1 发送

接口声明如下:

```
/**
* 发送弹幕
* @param token
* @param username
* @param isLive
* @param channelId 视频或直播流id
* @param content 内容, 50字以内
* @param positionInMilliseconds 位置的毫秒, 精确到100ms即可
* @param callback 成功时, 会返回对应的弹幕id
*/
public static void send(Activity activity,
                        @NonNull String token,
                        @NonNull String username,
                        boolean isLive,
                        @NonNull String channelId,
                        @NonNull String content,
                        long positionInMilliseconds,
                        Callback<String> callback);
```

2.2 点播拉取弹幕列表

声明如下

```
/**
* 获取列表
* 对于点播弹幕：第一次和用户跳转请求的pos必须是1000个单位的整数倍，往前最近的一个位置。第二次请求按照我给的end作为pos。
* 对于直播弹幕：
* 第一次请求直播弹幕，传入参数f=1和pos=自然时间(play返回)。后台返回（自然时间-10s）到（自然时间）的弹幕，并告诉end和ts=10s。
* 前端根据上一次请求的时间，在ts之后做第二次请求，传入pos={end}。我返回（pos）到(pos+10s)的弹幕，并告诉新的end和ts。
* 反复请求。如果出现end时间大于某个产品不能接受的值（一定大于我给的ts）。做第一次请求逻辑->第二次->….
* 直播回看时，保证pos是1000个单位的倍数。
*
* @param isLive 是否是直播
* @param channelId 视频或直播流id
* @param positionInMilliseconds 默认为0，从头开始，单位毫秒，播放位置
* @param f 第一次请求传入 0 或者不传，1表示后面的直播请求 2表示直播回看
*/
public static void getList(Activity activity,
                           boolean isLive,
                           String channelId,
                           long positionInMilliseconds,
                           String f,
                           final Callback<DanmakuListResponse> callback)
```

获取列表功能 **仅用于点播, 直播回放暂不确定, 不能用于直播** .

2.3 直播收取弹幕

直播弹幕通过TCP Socket和服务器进行通讯. 因此可以通过声明或者继承 `DanmakuReceiver` 接收弹幕, 或者通过初始化时传入的 `IDanmakuDeliverListener` 类来接收.

当需要接收某个直播弹幕时, 需要注册该直播的channelId

```
DanmakuSDK.getInstance().registerDanmakuSocket(context, true, channelId);
```

当需要停止接收弹幕或者退出activity时, 则调用如下方法:

```
DanmakuSDK.getInstance().unregisterDanmakuSocket(context);
```

或者

```
DanmakuSDK.getInstance().unregisterDanmakuSocket(context, channelId);
```

进行解除绑定

2.4. 退出 APP

由于弹幕 sdk 当中含有线程池, 长连接等资源, 因此在退出app时, 需要调用 `DanmakuSDK.getInstance().onAppTerminal(context);` 进行资源的释放

3. 弹幕 View 的使用. 如果直接使用 `DanmakuView`, 那么可以只关心2.1节, 第2节中的其他的不用关心

3.1 `DanmakuView` 可以直接添加在界面xml当中, 在开始播放视频时, 需要设置视频信息, 调用 `DanmakuView` 的如下方法:

```
/**
* 设置播放信息
* @param channelId 视频id, videoId
* @param isLive
*/
public void setPlayInfo(String channelId, boolean isLive)
```

3.2 在开始播放或者继续播放时需要调用 `DanmakuView.resume` 方法, 声明如下:

```
/**
* 开始播放或暂停后继续
* @param clearDanmakus 是否清理之前已经显示的弹幕
* @param milliSeconds 当前播放到的位置的毫秒. 直播则使用直播到的东八区时间毫秒
*/
public void resume(boolean clearDanmakus, long milliSeconds)
```

3.3 需要暂停时:

```
/**
* 暂停
```

```
*/
public void pause()
```

3.4 由于在播放过程中, 需要定时跟播放进度进行同步, 因此每隔500ms或者其他时间间隔, 或者在卡顿时, 调用如下方法进行时间同步:

```
/**
* 更新时间, 直播则使用直播到的东八区时间毫秒
* @param milliSeconds
*/
public void updatePlayTime(long milliSeconds)
```

3.5 直播回看时:

```
/**
* 定位视频位置
*
* @param milliSeconds
*/
public void seekTo(long milliSeconds)
```

3.6 直播回看后, 返回直播时一定要调用如下:

```
/**
* 如果在直播时调用过 seekTo方法, 则回到直播的时候, 需要调用本方法来重置弹幕状态
* @param clearDanmakus 清理当前屏幕上的弹幕
* @param milliSeconds 当前时间
*/
public void backToLive(boolean clearDanmakus, long milliSeconds)
```

3.7 在Activity/Fragment的生命周期中, 一定要调用如下接口:

```
/**
*  在销毁界面时一定要调用, 否则可能会有资源部被释放问题
*/
public void onDestroy(Context context)
```

```
public void onResume(Context context)

public void onPause(Context context)
```

这里, onPause和onResume中不控制弹幕的暂停与开始, 因此需要手动调用 `resume` 和 `pause` 方法进行控制

3.8 弹幕不覆盖和弹幕半屏/全屏显示

由于个别视频弹幕很多, 因此需要使用弹幕防覆盖的功能时, 调用 `DanmakuView` 的:

```
/**
* 设置弹幕不被覆盖
* @param dontCover
*/
public void setDontCover(boolean dontCover)
```

半屏弹幕显示, 调用:

```
/**
* 是否只展示半屏字幕
* @param half
*/
public void setHalfView(boolean half)
```

__ 注意: 半屏弹幕指的是从右向左飘动的弹幕只显示在屏幕的上半部分, 固定在视频上部和下部的弹幕不受影响 __

4. 弹幕网络接口文档:

http://sharepoint/tech/clouddivision/_layouts/15/WopiFrame2.aspx?sourcedoc=/tech/clouddivision/Shared%20Documents/%E9%95%BF%E8%BF%9E%E6%8E%A5%E4%BB%A5%E5%8F%8A%E5%BC%B9%E5%B9%95%E6%A2%B3%E7%90%86/%E5%BC%B9%E5%B9%951.0%E6%8E%A5%E5%8F%A3%E6%96%87%E6%A1%A3.docx&action=default