YoukuPlayerOpenSDK 的接入和使用说明

一、说明

此文档主要介绍 YoukuPlayerOpenSDK 的接入和使用注意事项，各功能点详细使用请参考 Demo 工程。使用过程中如有问题请及时进行沟通。SDK 以 libray 包的形式提供。

二、准备

1：YoukuPlayerOpenSDK 工程

第三方接入需要引用该 library。

2：YoukuLoginSDK 工程

该工程提供登陆相关功能，作为 YoukuPlayerOpenSDK 的一项功能使用，第三方应用无需直接使用该 library 工程，如需要做登陆相关功能，请使用 YoukuPlayerOpenSDK 提供的相关接口。

三、使用方式

第三方应用以 library 包的形式引用 YoukuPlayerOpenSDK

四、主要功能

该版本 OpenSDK 主要提供了以下相关功能，具体可以参考接口介绍部分。

1：播放相关

2：视频缓存相关

3：优酷帐号登陆相关

4：播放本地缓存的视频

五、主要类及接口

1. 主要类

ApiManager

提供了登陆、登出等相关功能

YoukuPlayerView

播放器控件类

YoukuBasePlayerManager

进行视频播放的 Activity 中实例化该类，对播放器进行一些初始化工作。

YoukuPlayer

视频播放主要接口类，成功初始化后可以调用此接口的方法进行相应视频播放。

DownloadManager

视频缓存功能类。

2. 主要接口

1. ApiManager 类

主要功能接口有：

```
public static void doLogin(Context context)
```

用户登录接口。

```
public static void doLogout(Context context)
```

用户登出接口。

```
public static String getLoginUser()
```

获取当前登陆用户的用户名，如果没有登陆则返回空。

```
public List<VideoQuality> getSupportedVideoQuality(IBasePlayerManager basePlayerManager)
```

获取所播放视频支持的清晰度列表。

VideoQuality 代表清晰度值，类型为 enum，取值有：

SUPER： 超清

HIGHT： 高清

STANDARD： 标清

P1080： 1080P

```
public int changeVideoQuality(VideoQuality quality,IBasePlayerManager basePlayerManager)
```

更改所播放视频的清晰度。

返回值：

1：成功

0：不支持此清晰度

2. DownloadManager 类

提供的相关接口，默认缓存路径为：包名/videocache 目录，接入方可以通过在自定义的 Application 中覆写相关方法进行更改。

```
public void createDownload(String videoId, String videoName,
        OnCreateDownloadListener listener)
```
创建视频下载任务

```
public HashMap<String, DownloadInfo> getDownloadedData()
```
获取已经缓存的视频信息

```
public HashMap<String, DownloadInfo> getDownloadingData()
```
获取正在缓存的视频信息

```
public boolean deleteDownloading(String taskId)
```
取消（删除）相应的下载任务

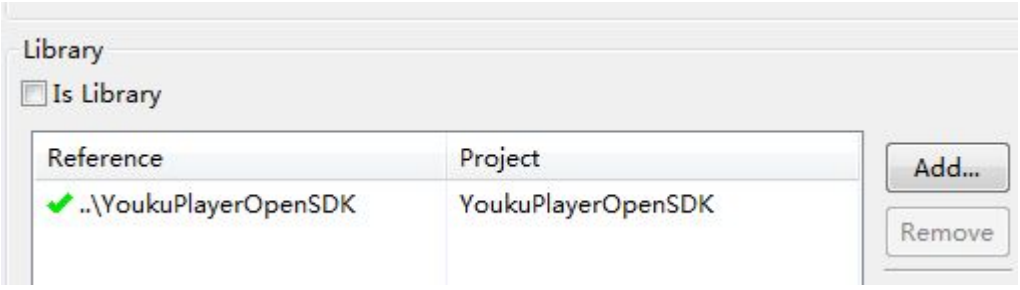
六、参数配置

第三方应用需要在 AndroidMenifest.xml 文件中配置分配的 client_id 和 client_secret 参数：

```
<!-- client_id及client_secret配置 -->
<meta-data android:name="client_id" android:value="85dmcobwswoz4rr6"/>
<meta-data android:name="client_secret" android:value="354849f847e468ee503f8f3f7d84c21d"/>
```

七、OpenSDK 接入简要流程

1：新建项目并引用 SDK 包



2：实例化 YoukuPlayerBaseConfiguration

所建工程的 Application 类中实例化 YoukuPlayerBaseConfiguration。详细使用请参考 demo。

```
public class MyApplication extends Application {

    public static YoukuPlayerBaseConfiguration configuration;

    @Override
    public void onCreate() {
        // TODO Auto-generated method stub
        super.onCreate();
        configuration = new YoukuPlayerBaseConfiguration(this) {
```

3：在布局文件中添加 YoukuPlayerView 控件

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <com.youku.player.base.YoukuPlayerView
        android:id="@+id/full_holder"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
    </com.youku.player.base.YoukuPlayerView>


</LinearLayout>
```

4：实例化 YoukuBasePlayerManager

需要播放视频的 Activity 中实例化 YoukuBasePlayerManager，在 onCreate 函数中对 YoukuPlayerView 实例进行初始化。详细使用请参考 demo。

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    // TODO Auto-generated method stub
    super.onCreate(savedInstanceState);

    setContentView(R.layout.second);
    iniView();
    basePlayerManager = new YoukuBasePlayerManager(this) {
```

```
// 播放器控件
mYoukuPlayerView = (YoukuPlayerView) this
        .findViewById(R.id.full_holder);
//控制竖屏和全屏时候的布局参数。这两句必填。
mYoukuPlayerView
        .setSmallScreenLayoutParams(new LinearLayout.LayoutParams(
                LinearLayout.LayoutParams.MATCH_PARENT,
                LinearLayout.LayoutParams.WRAP_CONTENT));
mYoukuPlayerView
        .setFullScreenLayoutParams(new LinearLayout.LayoutParams(
                LinearLayout.LayoutParams.MATCH_PARENT,
                LinearLayout.LayoutParams.MATCH_PARENT));
// 初始化播放器相关数据
mYoukuPlayerView.initialize(basePlayerManager);
```

5：传入视频 id 进行播放

在 onInitializationSuccess()函数中获取 YoukuPlayer 实例，通过传入视频 id 进行播放。

```
@Override
public void onInitializationSuccess(YoukuPlayer player) {
    // TODO Auto-generated method stub
    // 初始化成功后需要添加该行代码
    addPlugins();

    // 实例化YoukuPlayer实例
    youkuPlayer = player;

    // 进行播放
    goPlay();
}
```

八、缓存视频

详细使用请参考 demo 中的：CachedAcitivity 和 CachingActivity

1：配置缓存路径

默认缓存路径为：包名/videocache，在 Application 的定义中覆写如下方法更改缓存路径

```
/**
 * 配置视频的缓存路径，格式举例：/appname/videocache/
 * 如果返回空，则视频默认缓存路径为：/应用程序包名/videocache/
 *
 */
@Override
public String configDownloadPath() {
    // TODO Auto-generated method stub

    //return "/myapp/videocache/";          //举例
    return null;
}
```

2：对播放视频进行缓存

可以通过如下方式，如何获取本地缓存视频并进行播放具体可参考 Demo。

```
// 通过DownloadManager类实现视频下载
DownloadManager d = DownloadManager.getInstance();
/**
 * 第一个参数为需要下载的视频id 第二个参数为该视频的标题title 第三个对下载视频结束的监听，可以为空null
 */
d.createDownload("XNzgyODExNDY4", "魔女范冰冰扑倒黄晓明",
        new OnCreateDownloadListener() {

            @Override
            public void onfinish(boolean isNeedRefresh) {
                // TODO Auto-generated method stub

            }
        });
```

3：获取正在下载视频列表

```
//通过DownloadManager类的getDownloadingData()接口函数获取已经下载的视频信息
downloadManager = DownloadManager.getInstance();
Iterator iter = downloadManager.getDownloadingData().entrySet().iterator();
```

4：获取已经下载视频列表

```
//通过DownloadManager类的getDownloadedData()接口函数获取已经下载的视频信息
downloadManager = DownloadManager.getInstance();
Iterator iter = downloadManager.getDownloadedData().entrySet().iterator();
```

5: 删除一个下载任务

```
downloadManager.deleteDownloading(info.taskId);
```