Android点播SDK集成文档

简介1111111

百家云Android点播SDK是一个支持视频在线播放和离线播放，视频缓存，可自定义UI界面的播放器。播放器继承自Framelayout, 底层采用ijkplayer，您可以方便地添加到您自己的app当中。推荐使用Android Studio集成。

功能描述

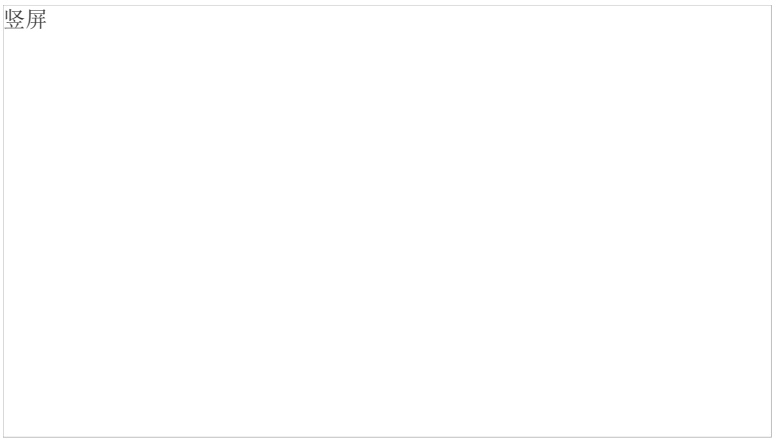
SDK支持Android4.0（api level 14）及以上

功能	描述
在线播放	支持百家云后台配置视频播放以及http(s)协议视频url播放
离线播放	支持本地视频绝对路径(不加密)以及file协议(加密)视频播放，加密和不加密详见视频缓存模块
片头片尾	支持视频片头片尾播放，可在百家云后台配置
自定义界面	sdk提供标准界面，用户可自行修改
视频缓存	支持百家云后台配置视频的缓存功能

示例工程

竖屏

竖屏



github链接：https://github.com/baijia/BJVideoPlayerDemo-Android(apk在apk_bin目录下)

集成SDK

集成前的准备

1) 推荐使用最新版Android studio集成SDK 点击下载（需科学上网）

2) 在工程根目录下build.gradle添加远程仓库

```
maven { url 'https://raw.github.com/baijia/maven/master/' }
```

3) 在需要集成的module添加如下依赖

```
compile 'com.baijia.player:videoplayer:1.6.7'
```

推荐使用最新版本 4）在build.gradle中添加ndk过滤

```
ndk {
    abiFilters 'armeabi-v7a', 'armeabi', 'x86' //x86虚拟机测试用，发版可去掉
}
```

快速集成

1）在布局文件中添加播放器控件

```
<com.baijiahulian.player.BJPlayerView
    android:id="@+id/pv_ac_quiz_play_demo"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:aspect_ratio="fit_parent">
</com.baijiahulian.player.BJPlayerView>
```

2）初始化播放器

application初始化

```
public class App extends Application {
    @Override
    public void onCreate() {
        super.onCreate();
        BJVideoPlayerSDK.getInstance().init(this);
    }
}
```

manifest.xml文件配置

```
<application
    android:name=".App"
    ...其他配置
```

BJPlayerView初始化

```
//变量声明
private BJPlayerView playerView;
//根据id查找播放器
playerView = (BJPlayerView) findViewById(R.id.pv_ac_quiz_play_demo);
//以下三个方法分别设置底部、顶部和中部界面
playerView.setBottomPresenter(new BJBottomViewPresenter(playerView.getBottomView()));
playerView.setTopPresenter(new BJTopViewPresenter(playerView.getTopView()));
playerView.setCenterPresenter(new BJCenterViewPresenter(playerView.getCenterView()));
//初始化partnerId，第一个参数换成您的partnerId
playerView.initPartner(123456L, BJPlayerView.PLAYER_DEPLOY_ONLINE);
```

3) 播放

在合适的时机播放视频，比如点击事件

```
//第一个参数为百家云后台配置的视频id，第二个参数为视频token
playerView.setVideoId(Long.valueOf(videoId), videoToken);
//播放
playerView.playVideo();
```

4) 回调接口

回调接口为播放器状态改变之后向上层app的通知，可以在每个回调方法中实现自己的业务逻辑

```
playerView.setOnPlayerViewListener(new OnPlayerViewListener() {
    @Override
    public void onVideoInfoInitialized(BJPlayerView playerView, HttpException exception) {
        //TODO: 视频信息初始化结束
    }
    @Override
    public void onPause(BJPlayerView playerView) {
        //TODO: video暂停
    }
    @Override
    public void onPlay(BJPlayerView playerView) {
        //TODO: 开始播放
    }
    @Override
    public void onError(BJPlayerView playerView, int code) {
        //TODO: 播放出错
    }
    @Override
    public void onUpdatePosition(BJPlayerView playerView, int position) {
        //TODO: 播放过程中更新播放位置
    }
    @Override
    public void onSeekComplete(BJPlayerView playerView, int position) {
        //TODO: 拖动进度条
    }
    @Override
    public void onSpeedUp(BJPlayerView playerView, float speedUp) {
        //TODO: 设置倍速播放
    }
    @Override
    public void onVideoDefinition(BJPlayerView playerView, int definition) {
        //TODO: 设置清晰度完成
    }
    @Override
    public void onPlayCompleted(BJPlayerView playerView, VideoItem item, SectionItem nextSection) {
        //TODO: 当前视频播放完成 [nextSection已被废弃，请勿使用]
    }
    @Override
    public void onVideoPrepared(BJPlayerView playerView) {
        //TODO: 准备好了，马上要播放
        // 可以在这时获取视频时长
        playerView.getDuration();
    }
});
```

5) 生命周期绑定

为防止转屏导致界面重绘，在manifest.xml中相应Activity加入如下配置

```
android:configChanges="keyboardHidden|orientation|screenSize"
```

在Activity或者fragment生命周期回调中加入对应的播放器调用：

```
@Override
protected void onResume() {
    //如果要实现后台播放则不用调用onResume()
    super.onResume();
    if (playerView != null) {
        playerView.onResume();
    }
}
@Override
protected void onPause() {
    super.onPause();
    //如果要实现后台播放则不用调用onPause()
    if (playerView != null) {
        playerView.onPause();
    }
}
@Override
protected void onDestroy() {
    super.onDestroy();
    if (playerView != null) {
        playerView.onDestroy();
    }
}
```

至此，点播sdk快速集成完成。

自定义播放器样式

如果您不想使用SDK自带界面，您可以自己开发自己的界面，代替自带界面

1.新建xml布局

在xml播放器控件中添加自定义布局文件，如果不想使用某一块界面可以将其设置为空白的布局

```
//顶部布局
app:top_controller="@layout/bjplayer_layout_top_controller"
//底部布局
app:bottom_controller="@layout/bjplayer_layout_bottom_controller"
//中部布局
app:center_controller="@layout/bjplayer_layout_center_controller"
```

2.实现接口

新建class分别实现TopView、CenterView、BottomView回调接口

接口说明如下

```
interface TopView {
    /**
     * 绑定presenter和view
     */
    void onBind(IPlayer player);
    /**
     * 设置视频标题
     *
     * @param title 视频标题
     */
    void setTitle(String title);
    /**
     * 设置了播放器方向
     */
    void setOrientation(int orientation);
    /**
     * 返回键监听
     */
    void setOnBackClickListener(View.OnClickListener listener);
}
```

```
interface BottomView {
    /**
     * 绑定presenter和view
     */
    void onBind(IPlayer player);
    /**
     * 设置视频长度
     *
     * @param duration 播放长度
     */
    void setDuration(int duration);
    /**
     * 设置播放位置
     *
     * @param position 播放位置
     */
    void setCurrentPosition(int position);
    /**
     * 设置是否在播放
     *
     * @param isPlaying true在播放，false 不在播放
     */
    void setIsPlaying(boolean isPlaying);
    /**
     * 设置方向
     *
     * @param orientation 参考android类Configuration的常量
     *                    Configuration.ORIENTATION_LANDSCAPE
     *                    Configuration.ORIENTATION_PORTRAIT
     */
    void setOrientation(int orientation);
    /**
     * 更新进度
     *
     * @param percent 缓冲进度百分比
     */
    void onBufferingUpdate(int percent);
    /**
     * 设置进度条是否可以拖动
     *
     * @param canDrag true可以拖动，false不可以拖动
     */
    void setSeekBarDraggable(boolean canDrag);
}
```

```
interface CenterView {
    /**
     * 绑定presenter和view
     */
    void onBind(IPlayer player);
    /**
     * 点击返回
     *
```

```
8.
9.    * @deprecated
10.   */
11.  boolean onBackTouch();
12.  /**
13.   * 设置播放器方向
14.   *
15.   * @param orientation 方向int值，同Android系统Confi
16.   */
17.  void setOrientation(int orientation);
18.  /**
19.   * 滑动fling手势进度
20.   */
21.  void showProgressSlide(int delta);
22.  /**
23.   * 显示loading
24.   *
25.   * @param message loading信息
26.   */
27.  void showLoading(String message);
28.  /**
29.   * 隐藏loading
30.   */
31.  void dismissLoading();
32.  /**
33.   * 音量手势显示
34.   *
35.   * @param volume    当前音量
36.   * @param maxVolume 最大音量
37.   */
38.  void showVolumeSlide(int volume, int maxVolu
39.  /**
40.   * 亮度显示
41.   *
42.   * @param brightness 亮度
43.   */
44.  void showBrightnessSlide(int brightness);
45.  /**
46.   * 显示错误
47.   */
48.  void showError(int what, int extra);
49.  void showError(int code, String message);
50.  /**
51.   * 显示警告
52.   */
53.  void showWarning(String warn);
54.  /**
55.   * 显示视图
56.   */
57.  void onShow();
58.  /**
59.   * 隐藏视图
60.   */
```

```
    void onHide();
    /**
     * 视频信息加载完成
     *
     * @param videoItem 视频信息model
     */
    void onVideoInfoLoaded(VideoItem videoItem);
    /**
     * @deprecated
     */
    boolean isDialogShowing();
    /**
     * 清晰度调整
     */
    void updateDefinition();
}
```

3.传入接口的实现

将实现类传给BJPlayerView（初始化时）：

```
playerView.setBottomPresenter(BottomViewImpl);
playerView.setTopPresenter(TopViewImpl);
playerView.setCenterPresenter(CenterViewImpl);
```

4.开源了

考虑到自定义播放器样式有一定难度，BJPlayerView的自身实现已经在demo中开源，大家可以参考。 4）需要注意是，TopView、CenterView、BottomView是为了自定义播放器样式而抽象的一套接口，配合1）中描述的自定义布局文件可以实现定制化的回调效果。集成时开发者可以选择性实现部分回调接口，如全部空实现则播放器无以上的回调提示。

下载模块

视频缓冲(旧版)

视频缓存仅支持下载百家云后台配置的视频，不支持任意视频文件下载。

1.初始化数据库

```
/**
 * 设置用户数据库名称，比如123.db  如果app不使用sdk下载模块或者下载不区分用户则不必调用
 * 请务必在VideoDownloadManager初始化之前调用
 * */
BJVideoPlayerSDK.getInstance().setCurUserDBName(dbName);
```

2.初始化下载管理器

可在Activity、fragment或者application中初始化，传入Context实例

```
VideoDownloadManager downloadManager = VideoDownloadService.getDownloadManager(this);
//第一个参数为您申请的partnerId
downloadManager.initDownloadPartner(32975272, BJPlayerView.PLAYER_DEPLOY_ONLINE);
//设置下载目标路径
downloadManager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
"/aa_video_downloaded/");
//设置最大下载并发数，默认3个
downloadManager.getThreadPool().setCorePoolSize(4);
```

3.获取某个视频的清晰度

```
/**
 * 获取当前视频的所有清晰度
 *
 * @param videoId    vid
 * @param videoToken token
 * @param listener   监听
 */
public void getVideoDefinitionById(int videoId, String videoToken, OnVideoDefinitionListener listener)
```

4.添加下载任务

```
/**
 * @param fileName    文件名，用户可用于展示，真实文件名为 “视频id_清晰度”
 * @param serialId    已废弃 传0即可
 * @param videoId     视频id
 * @param videoToken  视频token
 * @param type        视频清晰度 0普清 1高清 2超清 3 720p 4 1080p
 * @param encryptType 加密类型 0 不加密，1加密
 * @param extraInfo   额外信息，客户自己定义，这里只是转存
 */
private void addDownloadVideoTask(final String fileName, long serialId, final int videoId, final String
videoToken, final int type,final int encryptType, final String extraInfo, final OnVideoInfoGetListener
listener)
```

5.在合适位置释放下载器，比如Activity的onDestroy方法：

```
BJVideoPlayerSDK.getInstance().releaseDownloadClient();
```

6.视频缓存(VideoDownloadManager类)

获取所有下载任务

```
/**
 * 获取所有下载任务
 */
public List<DownloadInfo> getAllTask()
```

DownloadInfo字段说明如下：

```
private int id;                 //id自增长
private String taskKey;         //下载的标识键
private String url;             //文件URL
private String targetFolder;    //保存文件夹
private String targetPath;      //保存文件地址
private String fileName;        //保存的文件名
private float progress;         //下载进度
private long totalLength;       //总大小
private long downloadLength;    //已下载大小
private long networkSpeed;      //下载速度
private int state = 0;          //当前状态
private BaseRequest request;    //当前任务的网络请求
@Deprecated
private Serializable data;      //额外的数据 deprecated
private int videoId;            //视频id
private int videoType;          //清晰度
private int encryptType;        //加密类型 0不加密，1加密
private String videoToken;      //视频token，过期逻辑上层处理
private String extraInfo;       //额外信息，用户想存啥就存啥
```

添加所有任务监听：

```
public void addOnAllTaskEndListener(OnAllTaskEndListener allTaskEndListener)
```

移除所有任务监听：

```
public void removeOnAllTaskEndListener(OnAllTaskEndListener allTaskEndListener)
```

根据taskKey（下载标识键）获取下载信息

```
public DownloadInfo getDownloadInfo(String taskKey)
```

开始所有下载任务

```
public void startAllTask()
```

暂停所有下载任务

```
public void pauseAllTask()
```

移除所有下载任务

```
//只删除任务，不删文件
private void removeAllTask()
//删除所有任务和对应的文件
public void removeAllTaskAndFiles()
```

重新下载某个任务

```
public void restartTask(String taskKey)
```

暂停某个任务

```
public void pauseTask(String taskKey)
```

删除某个任务

```
//false只删除任务，不删文件, true删除任务和已下载文件
public void removeTask(String taskKey, boolean isDeleteFile)
```

获取下载目录

```
public String getTargetFolder()
```

视频缓存（新版）

重构后的视频下载模块逻辑更加精简，数据缓存改用磁盘缓存，下载模块不再提供线程管理功能，只提供下载的基本功能，集成者可以根据需求在此基础上自行实现并发控制。

1.manifest.xml中配置DownloadService

```
<service
        android:name="com.baijiayun.download.DownloadService"
        android:enabled="true"
        android:exported="false"></service>
```

2.初始化DownloadManager

```
    //初始化下载
    manager = DownloadService.getDownloadManager(this);
    //设置缓存文件路径
    manager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
"/bb_video_downloaded/");
    //读取磁盘缓存的下载任务
    manager.loadDownloadInfo(32975272);
```

3.新建一个下载任务

```
downloadManager.newDownloadTask("video_" + videoId, Long.parseLong(videoId), token, definitionList,
0, "haha")
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe(new Action1<DownloadTask>() {
                    @Override
                    public void call(DownloadTask task) {
                        //直接开始下载
                        task.start();
                    }
                }, new Action1<Throwable>() {
                    @Override
                    public void call(Throwable throwable) {
                        throwable.printStackTrace();
                    }
                });
```

newDownloadTask的参数说明如下

```
/**
 * 创建点播下载
 * @param fileName 保存的下载的文件名称
 * @param videoId  视频id
 * @param token    视频token
 * @param definitions  下载的清晰度list,从前往后匹配
 * @param encryptType 加密类型 0 不加密，1加密
 * @param extraInfo  额外信息，客户自己定义，这里只是转存
 * @return
 */
public Observable<DownloadTask> newDownloadTask(final String fileName, final long videoId, final String token,
                                                final List<VideoDefinition> definitions, final int encryptType, final String extraInfo)
```

4.DownloadTask接口说明

```
public interface DownloadTask {

    /**
     * 开始任务
     */
    void start();

    /**
     * 暂停任务
     */
    void pause();

    /**
     * 重新开始任务
     */
    void restart();

    /**
     * 取消任务
     */
    void cancel();

    /**
     * 删除任务和文件
     */
    void deleteFiles();

    /**
     * 设置下载状态的监听
     *
     * @param listener
     */
    void setDownloadListener(DownloadListener listener);

    /**
     * 获取视频相关信息，数据结构和DownloadInfo类似
```

```
     *
     * @return
     */
    DownloadModel getDownloadInfo();

    DownloadModel getVideoDownloadInfo();

    /**
     * 获取下载状态
     * New(0),          // new -> downloading
     * Downloading(1),  // downloading -> pause, error, finish
     * Pause(2),        // pause -> downloading
     * Error(3),        // error -> new
     * Finish(4),       //
     * Cancel(5);
     * @return
     */
    TaskStatus getTaskStatus();

    /**
     * 获取下载速度，单位为byte
     * @return
     */
    long getSpeed();

    /**
     * 获取下载进度
     * @return
     */
    float getProgress();

    /**
     * 获取要下载的总字节数
     * @return
     */
    long getTotalLength();

    /**
     * 获取已下载字节数
     * @return
     */
    long getDownloadedLength();

    /**
     * 获取下载类别
     * @return
     */
    DownloadType getDownloadType();

    /**获取文件名称
     * @return
     */
```

```
89.     String getFileName();
90.
91.     /**
92.      * 获取视频时长
93.      * @return
94.      */
95.     long getVideoDuration();
96. }
```

5.下载状态监听

给newDownloadTask()获得的DownloadTask设置DownloadListener

```
task.setDownloadListener(new DownloadListener() {
            @Override
            public void onProgress(DownloadTask task) {
                //TODO 更新下载进度
            }

            @Override
            public void onError(final DownloadTask task, HttpException e) {
                holder.download.setText("出错");
                holder.download.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        task.start();
                    }
                });
                e.printStackTrace();
                //下载地址已失效，5103(token已失效)
                if(e.getCode() == 403 || e.getCode() == 5103){
                    //TODO 需要用户传入新的token，重新获取下载链接
                    newDownloadTask(String.valueOf(task.getDownloadInfo().videoId), "test12345678");
                }
            }

            @Override
            public void onPaused(final DownloadTask task) {

            }

            @Override
            public void onStarted(final DownloadTask task) {

            }

            @Override
            public void onFinish(DownloadTask task) {
                //TODO 更新UI
            }

            @Override
            public void onDeleted(long videoid) {
                //TODO 更新UI
            }
        });
```

这里要特别注意的一点是OnError回调中，code==403的情况，视频的下载链接存在一个有效期，如果视频下载中断后视频链接过期则会返回403。上面的示例代码中onError已经给出了处理方案，重新调用`newDownloadTask(String videoId, String token)`获取视频信息，但这里的token需要集成者重新传入有效的token。

6.demo中的SimpleVideoDownloadActivity简单集成了下载功能和UI效果，大家可以参考。

数据迁移

从旧版本下载改用到新版本下载模块，需要兼容用户旧的下载记录，因为旧版本下载记录保存在数据库中，新版本下载记录保存在本地文件中。

1.RecoverDbHelper初始化

```
/**
 * 初始化
 * @param context
 * @param recoveryPath 旧的视频保存的文件夹
 * @param recoveryCallback callback
 */
public void init(Context context, String recoveryPath, IRecoveryCallback recoveryCallback){
}
```

2.开始迁移

```
RecoverDbHelper.getInstance().recoveryDbData();
```

3.示例代码

完整的代码如下，也可在demo中查看。

```
        manager = DownloadService.getDownloadManager(getApplicationContext());
        //设置缓存文件路径
        manager.setTargetFolder(Environment.getExternalStorageDirectory().getAbsolutePath() +
"/bb_video_downloaded/");
        //TODO RecoverDbHelper为恢复旧版下载记录的工具类。没有从旧版迁移到新版的需求不用调用此工
具类
        RecoverDbHelper.getInstance().init(getApplicationContext(),
Environment.getExternalStorageDirectory().getAbsolutePath() + "/aa_video_downloaded/",
                new IRecoveryCallback() {
                    @Override
                    public void recoverySuccess() {
                        //重新获取taskList，true代表强制刷新
                        manager.loadDownloadInfo(32975272, 1, true);
                        adapter.notifyDataSetChanged();
                    }
                });
        //读取磁盘缓存的下载任务
        manager.loadDownloadInfo(32975272, 1);
        //TODO 这一句必须在manager.loadDownloadInfo()之后，确保DownloadManager已初始化完毕
        RecoverDbHelper.getInstance().recoveryDbData();
```

视频加密

在线播放和离线下载的视频支持自定义ev1格式(百家云自定义的视频格式)和mp4两种格式，集成时可以选择是否使用加密格式。之前通过file协议在java层对视频加解密的逻辑已去掉，如有视频加密需要请仔细阅读以下步骤。

1.在线播放

```
//1代表使用加密视频,0反之
playerView.initPartner(partnerId,BJPlayerView.PLAYER_DEPLOY_ONLINE,1);
```

2.离线下载

```
//1代表使用加密视频,0反之
manager.loadDownloadInfo(partnerId, 1);
``

##常用api说明
###播放器(BJPlayerView类)
设置视频填充颜色
```java
初始化合作方
public void initPartner(long partnerId) {
        //正式服 && 默认获取加密视频
        initPartner(partnerId, PLAYER_DEPLOY_ONLINE, 1);
    }

    public void initPartner(long partnerId, int deploy) {
        //默认获取加密视频
        initPartner(partnerId, deploy, 1);
    }

 /**
     * 设置 合作方 ID。 如果没有设置， 无法使用 SDK
     *
     * @param partnerId 合作方 ID
     * @param deploy    运行环境;
     *                  <ul>
     *                  <li>测试环境: PLAYER_DEPLOY_DEBUG</li>
     *                  <li>Beta 环境: PLAYER_DEPLOY_BETA</li>
     *                  <li>线上环境: PLAYER_DEPLOY_ONLINE</li>
     *                  </ul>
     *@param encryptType 视频是否加密。1加密；0不加密
     */
    public void initPartner(long partnerId, int deploy, int encryptType)
/**
 * 设置视频画面四周补边的颜色<br/>
 *
 * @param color 颜色值，缺省值为Color.argb(255, 0, 0, 0)
 */
public void setVideoEdgePaddingColor(int color)
```

设置片头片尾播放策略

```
/**
 * 设置片头片尾的播放方式，默认是不播
 *
 * @param headTailPlayMethod <ul>
 *                           <li>不播: HEAD_TAIL_PLAY_NONE</li>
 *                           <li>播一次: HEAD_TAIL_PLAY_ONCE</li>
 *                           <li>每次都播: HEAD_TAIL_PLAY_EVERY</li>
 *                           </ul>
 */
public void setHeadTailPlayMethod(int headTailPlayMethod)
```

开始播放视频
```

```
/**
 * <p>播放视频</p>
 * 会检测网络状况
 */
public synchronized void playVideo()
```

```
/**
 * <p>播放视频</p>
 * @param netTypeCheck, true检测网络，false不检测
 */
public synchronized void playVideo(boolean netTypeCheck)
```

```
/**
 * <p>播放视频</p>
 * @param pos 从特定位置播放，单位为秒
 */
public synchronized void playVideo(int pos)
```

获取视频时长

```
/**
 * 获取视频时长，需在视频初始化
 */
int getDuration()
```

设置用户自定义信息，用户您的app的自定义统计

```
/**
 * <p>设置用户自定义信息</p>
 *
 * @param userInfo 用户自定义消息
 */
public void setUserInfo(String userName, int userNumber)
```

播放http(s)网络视频或者本地视频

```
/**
 * <p>设置视频源路径</p>
 *
 * @param path 视频 url, 可以是网络视频，也可以是本地视频路径
 */
public void setVideoPath(String path)
```

获取当前视频信息

```
/**
 * 获取视频详细信息
 *
 * @return VideoItem 视频信息model
 */
@Override
public VideoItem getVideoItem()
```

播放器手势开关控制

```
/**
 * 是否允许通过触摸手势控制音量
 */
public void enableVolumeGesture(boolean enable)

/**
 * 是否允许通过触摸手势控制亮度
 */
public void enableBrightnessGesture(boolean enable)

/**
 * 是否允许通过触摸手势拖拽
 */
public void enableSeekGesture(boolean enable)
```

是否响应横竖屏切换

```
//true:横竖屏切换时播放器ui不变化
public void setForbidConfiguration(boolean forbidConfiguration)
```

设置视频清晰度

```
/**
 * <p>设置播放视频清晰度</p>
 *
 * @param definition 清晰度
 *                   <ul>
 *                   <li>标清(0): VIDEO_DEFINITION_STD</li>
 *                   <li>高清(1): VIDEO_DEFINITION_HIGH</li>
 *                   <li>超清(2): VIDEO_DEFINITION_SUPER</li>
 *                   <li>720P(3): VIDEO_DEFINITION_720p</li>
 *                   <li>1080P(4): VIDEO_DEFINITION_1080p</li>
 *                   </ul>
 */
@Override
public void setVideoDefinition(int definition)
```

设置播放倍速

```
/**
 * <p>设置视频播放倍速</p>
 *
 * @param rate 播放速率
 *             <ul>
 *             <li>1 倍正常速率: VIDEO_RATE_1_X</li>
 *             <li>1.1 倍正常速率: VIDEO_RATE_1_1_X</li>
 *             <li>1.2 倍正常速率: VIDEO_RATE_1_2_X</li>
 *             <li>1.5 倍速率: VIDEO_RATE_1_5_X</li>
 *             <li>1.8 倍速率: VIDEO_RATE_1_8_X</li>
 *             <li>2 倍速率: VIDEO_RATE_2_X</li>
 *             </ul>
 */
@Override
public void setVideoRate(int rate)
```

错误码对照

错误码	说明
-10000	ijkplayer内部错误,比如文件异常
-1	无网络连接

注：ijkplayer没有给出具体错误码，详见源码的头文件ijkplayer_ android_def.h

其余错误码可参考AndroidSDK的MediaPlayer（需科学上网）

常见问题

1）为什么播本地视频总是提示-10000错误？

-10000错误码为ijkplayer底层报出的错误，请检查本地文件是否为合法视频文件，尝试使用系统播放器能不能正常播放。如果您使用视频下载模块下载视频，请检查是否是加密下载，加密视频需使用file协议的地址，不加密视频则使用视频文件绝对路径。

2) 为什么我集成之后播放视频一直在loading，没有播放出来？

首先检查网络连接是否正常，再者视频渲染采用surfaceview，初始化需要一定的时间，可在调用播放视频方法的时候尝试加入500ms左右延时。

3）为什么一直提示我token解析失败？

一个视频id对应一个token，token是百家云后台生成的，您可以联系后台开发人员确定token是否是正确的，再者查看初始化播放器的时候部署环境是否为在线环境playerView.initPartner(123456L, BJPlayerView.PLAYER_DEPLOY_ONLINE);

4）我不想在xml中添加控件，可不可以使用java代码生成播放器？

可以的。BJPlayerView继承自FrameLayout, FrameLayout所有的特性都是支持的。